

Proceedings of the Seminar in Computer Science (CS-E4000), Fall 2023

Antti Ylä-Jääski and Sara Ranjbaran

Tutors for seminar topics

Alexander Jung, Anton Debner, Antti Ylä-Jääski , Aziza Zhanabatyrova, Blerta Lindqvist, Daolang Huang, Erik Schultheis, Hannu Flinck, Jacopo Bufalino, Jose Luis Martin Navarro, Katsiaryna Haitsiukevich, Matti Siekkinen, Mikko Kiviharju, Mohit Sethi, Paul E. Chang, Rongjun Ma, Sanna Suoranta, Tianyi Zhou, Vesa Hirvisalo, and Vivienne Wang

Preface

The *Seminar on Network Security*, *Seminar on Internetworking* and *Seminar on Software Technology and Systems Research* were previously separate Master's level courses in computer science at Aalto University. These seminar courses have now merged into one seminar course. These seminar series have been running continuously since 1995. From the beginning, the principle has been that the students take one semester to perform individual research on an advanced technical or scientific topic, write an article on it, and present it on the seminar day at the end of the semester. The articles are printed as a technical report. The topics are provided by researchers, doctoral students, and experienced IT professionals, usually alumni of the university. The tutors take the main responsibility of guiding each student individually through the research and writing process.

The seminar course gives the students an opportunity to learn deeply about one specific topic. Most of the articles are overviews of the latest research or technology. The students can make their own contributions in the form of a synthesis, analysis, experiments, implementation, or even novel research results. The course gives the participants personal contacts in the research groups at the university. Another goal is that the students will form a habit of looking up the latest literature in any area of technology that they may be working on. Every year, some of the seminar articles lead to Master's thesis projects or joint research publications with the tutors.

Starting from the Fall 2015 semester, we have merged the three courses into one seminar that runs on both semesters. Therefore, the theme of the seminar is broader than before. All the articles address timely issues in security and privacy, networking technologies and software technology.

These seminar courses have been a key part of the Master's studies in several computer-science major subjects at Aalto, and a formative experience for many students. We will try to do our best for this to continue. Above all, we hope that you enjoy this semester's seminar and find the proceedings interesting.

Seminar papers

Aaron Campbell , <i>Understanding Adversarial Sample Transferability</i>	9
Tutor: Blerta Lindqvist.	
Arthur Becker , <i>Calibration in Quantized LLMs</i>	19
Tutor: Erik Schultheis.	
Daniel Kaluski , <i>Do cookie consent forms protect your data?</i>	33
Tutor: Sanna Suoranta.	
Danila Mokeev , <i>Investigating temporal Gaussian processes for solar PV forecasting</i>	45
Tutor: Paul E. Chang.	
Doan Tuan Dat , <i>Microservice - Advantages and weaknesses in industry</i>	57
Tutor: Ylä-Jääski Antti.	
Gabriel Gomes Ziegler , <i>Learning Models - A Literature Review</i> ...	69
Tutor: Alex Jung.	
Harry Pham , <i>Certification of Machine Learning Robustness against Evasion Attacks</i>	79
Tutor: Mikko Kiviharju.	
Han Huazhi , <i>Exploring the Practical Application of Distributed Artificial Intelligence: Structures and Tools</i>	93
Tutor: Vesa Hirvisalo, Anton Debner.	
Ilmari Tarpila , <i>Where is the Beef? Extending the Training Data</i> ..	105
Tutor: Alexander Jung.	
Ingi Þór Sigurðsson , <i>Heimbjartur: Testing network policies with eBPF</i>	117
Tutor: Jacopo Bufalino.	
Jaakko Perttula , <i>Prospects of WebAssembly to address performance and energy challenges of mobile clients</i>	129
Tutor: Hannu Flinck.	
Jere Hirviniemi , <i>Adversarial perturbation attacks on CNNs</i>	141
Tutor: Blerta Lindqvist.	
Jonas Bäck , <i>Software supply chain security: the SLSA specification</i>	149
Tutor: Jacopo Bufalino.	
Joonas Savola , <i>Third party cookies and how browser developers plan to deprecate them</i>	159
Tutor: Sanna Suoranta.	
Juho Pellinen , <i>CDN attacks</i>	171
Tutor: Jose Luis Martin Navarro.	
Jussi Impiö , <i>Exploiting HTTP/2: Investigating Attack Strategies against Content Delivery Networks</i>	185

<i>Tutor: Jose Luis Martin Navarro.</i>	
Jussi Vaitomaa , <i>Microservices - when and how to use them</i>	197
<i>Tutor: Antti Ylä-Jääski.</i>	
Lauri Ahlberg , <i>Large language models for protein structure prediction and design</i>	209
<i>Tutor: Katsiaryna Haitsiukevich.</i>	
Leevi Parkkinen , <i>Software supply chain security: the SLSA specification</i>	221
<i>Tutor: Jacopo Bufalino.</i>	
Leevi Rönkkö , <i>Caching is vulnerable: Modern threats facing CDN Edge Servers</i>	233
<i>Tutor: Jose Navarro.</i>	
Markus Granström , <i>Unidirectional gateways for security of OT networks</i>	245
<i>Tutor: Mohit Sethi.</i>	
Max Ulfves , <i>A Survey of Methods for Robust Camera based Pose Estimation</i>	257
<i>Tutor: Aziza Zhanabatyrova.</i>	
Niko Nästi , <i>Advancements in Neural Radiance Fields: A Comparative Study of Instant NeRF and BakedSDF</i>	267
<i>Tutor: Matti Siekkinen.</i>	
Nuutti Henriksson , <i>Password usability and security</i>	277
<i>Tutor: Sanna Suoranta.</i>	
Otso Laasonen , <i>Neural Radiance Fields And Its Extensions And Applications In Computer Graphics</i>	289
<i>Tutor: Matti Siekkinen.</i>	
Reda Bellafqih , <i>Privacy and Dark Patterns</i>	301
<i>Tutor: Suoranta Sanna.</i>	
René Steeman , <i>New challengers in the field of stance detection</i> ...	313
<i>Tutor: Tianyi Zhou.</i>	
Riste Ristov , <i>Testing network policies with eBPF</i>	325
<i>Tutor: Jacopo Bufalino.</i>	
Roni Kirla , <i>Prospects of WebAssembly to address performance and security challenges beyond the browser</i>	337
<i>Tutor: Hannu Flinck.</i>	
Samuel Ashraff , <i>Migrating from monolithic architecture to microservices: A comparative analysis of migration approaches</i>	349
<i>Tutor: Antti Ylä-Jääski.</i>	
Sergio Hernández , <i>A Comprehensive Overview of Goal-Conditioned Hierarchical Reinforcement Learning: Algorithms, Challenges, and Future Directions</i>	359
<i>Tutor: Vivienne Wang.</i>	
Sneha Saj , <i>Query Formation: A Comparative Analysis of Search En-</i>	

<i>gines and AI Models</i>	371
<i>Tutor: Rongjun Ma.</i>	
Sonika Baniya , <i>Pulsing Convex DDoS Attack in CDN Networks</i> ..	383
<i>Tutor: Jose Luis Martin Navarro.</i>	
Venla Kuosa , <i>OT vs IT Cybersecurity Management</i>	389
<i>Tutor: Mikko Kiviharju.</i>	
Xinyu Zhang , <i>Amortized Meta Bayesian Optimization</i>	401
<i>Tutor: Daolang Huang.</i>	
Yu Xu , <i>Neural Radiance Fields (NeRF): Current State of Art and Ongoing Challenges</i>	409
<i>Tutor: Matti Siekkinen.</i>	
Zehui Xu , <i>Testing network policies with eBPF</i>	421
<i>Tutor: Jacopo Bufalino.</i>	
Ziqi Wang , <i>Decoding the Black-Box: A Review of Machine Learning Explainability Methods</i>	433
<i>Tutor: Alex Jung.</i>	

Understanding Adversarial Sample Transferability

Aaron Campbell

aaron.campbell@aalto.fi

Tutor: Blerta Lindqvist

Abstract

This paper conducts a comprehensive literature survey on the transferability of adversarial attacks in the context of image classification models. The survey encompasses classic and state-of-the-art methodologies employed in adversarial perturbation attacks, particularly focusing on their transferability between different machine learning models. The exploration includes factors influencing transferability, such as model architecture and complexity, and discusses defense strategies. In conclusion, the paper emphasizes the evolving landscape of adversarial attacks and underscores the importance of ongoing research and interdisciplinary collaboration to enhance the robustness of machine learning models against transferability challenges.

***KEYWORDS:** adversarial perturbations, transferability, image classification, machine learning security, defense mechanisms, model architecture, symmetry defense, training datasets, white-box attacks, black-box attacks*

1 Introduction

Over the past decade, machine learning has experienced a rise in popularity across a diverse range of fields, from outperforming humans in

various games, to autonomous technologies, and medical analysis. The increased ubiquity of these machine learning methods has caused some to raise concerns about the safety, security, and reliability of these methods, particularly in the context of adversarial attacks.

One such attack is the adversarial perturbation attack, which most commonly attacks computer vision and image classification models. In the attack, an input image is perturbed, modified at the pixel level, by an adversarial algorithm which exploits hidden patterns within the target model, causing a misclassification through changes which are imperceptible to human observers [1, 2]. Although this phenomenon is most commonly associated with image classification, where strategic pixel adjustments can befuddle an image recognition algorithm that humans would easily decipher, it extends its reach into other domains such as evading spam filters through the addition of 'good' words and misspelling of 'bad' words [3].

One particularly interesting property of adversarial perturbation attacks is the transferability of adversarial samples between different machine learning models. Often, adversarial samples created to cause a misclassification by one machine learning model can effectively deceive different models trained for the same task, regardless of the underlying model architecture, dataset, or methodology [4, 5]. This inter-model transferability is an essential consideration for researchers and practitioners of these classification models, as it emphasizes a need for more robust countermeasures against adversarial threats, since model or dataset confidentiality are no longer sufficient protection against these types of attacks.

This paper aims to provide a comprehensive overview of the last decade of research on adversarial attacks, focusing on the past five years to capture recent developments in the field of transferability. By summarizing the existing literature, we seek to shed light on the current state of knowledge and identify avenues for future research. In order to better understand the transferability of adversarial attacks and the way this informs machine learning development, this paper summarizes the recent literature regarding the transferability of adversarial samples in the context of image classification models.

This paper is organized as follows: Section 2 provides a background on adversarial perturbation attacks, introducing the fundamental concepts and various types of attacks, including classic and cutting-edge method-

ologies. Section 3 delves into the intriguing property of attack transferability, exploring how adversarial samples designed for one model often cause misclassification in another model trained for the same task. The section discusses factors influencing transferability and examines both untargeted and targeted attacks. Additionally, it explores the ongoing efforts to defend against transferability, highlighting the incorporation of adversarial samples into training datasets and the impact of model architecture choices. Finally, Section 4 concludes the essay, summarizing the complexity of adversarial perturbation attacks, the challenges they pose to machine learning models, and the evolving landscape of defense strategies. The conclusion emphasizes the need for ongoing research and interdisciplinary collaboration to address emerging threats and enhance the robustness of machine learning models in the face of adversarial challenges.

2 Background

In order to understand the transferability of adversarial attacks, it is useful to first become familiar with the underlying concepts behind adversarial perturbation attacks. This section first briefly introduces adversarial perturbation attacks. Then the different types of adversarial attacks are discussed ranging from classic attacks to some of the most cutting edge. Finally, the topic of attack transferability will be introduced with explanation and discussion of its importance to the field.

2.1 Adversarial Perturbation Attacks

Adversarial perturbation attacks involve the deliberate modification of inputs to a classification model by an adversarial entity, aiming to induce misclassification. First identified by Szegedy et al. in 2014, these attacks exploit the somewhat non-intuitive nature of how many image classification models establish boundaries for their input-output relationships [6]. The primary objective of these attacks is to subtly adjust an input image until it just surpasses the threshold for its correct class label, resulting in an erroneous class assignment in the output.

In order to achieve the maximum effect with minimal changes to the input image, these attacks often utilize the decision boundaries and loss function of the target model. In a classification task, some

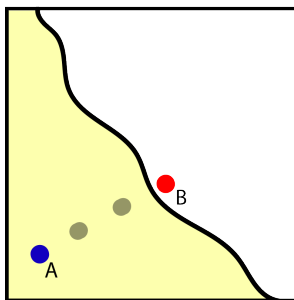


Figure 1. An example illustration of how an adversarial attack might be created. The blue node A represents the original 'true' class of the sample image. The gray trail shows how a gradient-based attack might perturb the image along the gradient towards a decision boundary. Once the point passes the boundary (red B node) the image is successfully perturbed into an adversarial sample.

input is mapped onto an output space delineated by class boundaries. Whichever boundary the output best fits into represents the class label assigned to the input. The effective distance between an input's true class and the class predicted by the model is quantified into the loss function of the model, where a lower loss value represents a more truthful classification. Perturbation attacks typically use the loss function to find the optimal direction for perturbations, efficiently guiding the model across a classification boundary and causing misclassification.

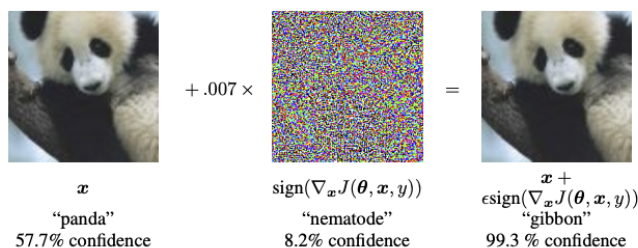


Figure 2. Example FGSM attack, image from Papernot et al. [4]

One early example of perturbation attacks, the Fast Gradient Sign Method (FGSM), employs this strategy [4]. FGSM relies on the gradient of the model's loss function with respect to the input data, providing insights into how the loss changes with small alterations in the input. Utilizing this information, FGSM determines the direction in which changes to the input have the most significant impact on the loss value. Through iterative perturbation in this direction, using a small step size, FGSM efficiently generates adversarial samples that are minimally altered yet highly effective at causing misclassification, as seen in Figure 2. This methodology sheds light on how decision boundaries are located and efficiently crossed in the context of adversarial attacks.

2.2 Types of Attacks

Adversarial perturbation attacks manifest in various forms, categorized primarily as white-box and black-box attacks. These classifications refer to the extent of the attacker's knowledge about the target model, influencing the methodology employed for crafting adversarial examples.

In white-box attacks, the assumption is that the attacker possesses complete access to the target model, including its architecture, parameters, and training data. This high level of access enables a more precise and informed approach, often involving calculations based on underlying aspects of the model. White-box attacks are particularly effective at generating strong and undetected adversarial samples. Examples of such attacks include gradient-based techniques like FGSM and the improved stealthiness demonstrated by L-norm attacks as outlined by Carlini and Wagner [4, 7].

Alternatively, black-box attacks work on the assumption that little to no information is known about the target model. These attacks more closely mimic real-world threats to target models which may be proprietary or trained on confidential or sensitive data. In this scenario, attack methods often resort to trial and error, generating adversarial examples through prompting the model and considering the effect of changes to the response. Notable black box attacks include the Zeroth Order Optimization attack and its successor the HopSkipJump attack [2, 8].

3 Attack Transferability

An intriguing property of adversarial perturbation attacks against classification models is the tendency for an adversarial sample designed to target one model to often cause a misclassification in another model trained for the same task [6]. This property of adversarial sample transferability is critical to the field of adversarial attacks since white-box attacks can be used against a surrogate model with the same task as the target model [4]. The adversarial sample can then be transferred to the target model enabling the application of efficient and effective white-box methods even in a black-box scenario.

3.1 Factors Affecting Transferability

The transferability of adversarial samples is influenced by various factors, one of which is the architecture and complexity of both target and surrogate models. Models with similar architectures tend to exhibit better transferability, likely owing to shared patterns in feature learning, higher gradient alignment, and classification boundaries [9, 10]. Surprisingly, less complex surrogate models demonstrate superior transferability to complex targets, indicating potential over-fitting in more complex models, which diminishes adversarial transferability [10, 7, 11].

An intriguing insight from Dong et al. underscores the high transferability of a translation-invariant attack generated using an ensemble approach [12]. The resulting adversarial samples, fine-tuned for varying levels of translation-invariability, exhibit increased transferability. Another study by Xie et al. in 2018 found that the transferability of adversarial samples could be improved by incorporating small random perturbations in the sample generation process [13]. These studies highlight the adaptability and effectiveness of certain adversarial strategies on sample transferability, and further support the theory that transferability is inversely related to over-fitting [10, 7, 11].

Furthermore, in untargeted attacks where the objective is misclassification without a specific target class, the relationship between surrogate and target models becomes even more nuanced. A 2019 study by Naseer et al. demonstrates that a cross-domain surrogate model, trained on diverse datasets encompassing paintings, cartoons, and medical images, successfully generates adversarial samples for a natural image classification model [5]. This finding suggests an inherent similarity in image classification tasks, a concept further explored by Nakka et al. in 2021 [14]. Nakka et al. delve into the hierarchical learning of features in convolutional neural networks (CNNs), noting that lower layers primarily learn color and edge-related features, middle layers focus on texture, and higher layers specialize in objects and class labels. Nakka et al. use these findings to build an attack around a loss function targeting these mid-level features which they show leads to cutting-edge results [14]. This understanding of the inherent commonality between different models sheds light on the findings of Naseer et al., as alterations to the cross-domain surrogate likely affected the target CNN in a similar way.

3.2 Defending Against Adversarial and Transfer Attacks

While the research community has actively sought to develop defenses against adversarial attacks, achieving robust protection remains a challenge. Defense mechanisms often lag behind the ingenuity of adversarial attacks, making it an ongoing area of research to enhance the security and reliability of machine learning models in the face of adversarial threats. Furthermore, many defenses offer limited protection against transferred adversarial attacks [4]. Nonetheless, there are several base protections which are shown to have a protective effect against basic adversarial samples.

A fundamental defense against adversarial attacks is the incorporation of adversarial samples into the training dataset of the model, especially those demonstrating strong transferability, as highlighted by Carlini et al. [7]. By training on adversarial samples, the model is made stronger against similar attacks in the future.

Architecture choice is another crucial consideration. As previously noted, more intricate models tend to exhibit poorer transferability to simpler ones [7, 11]. Thus, choosing as simple a model as the task permits, provided the classification accuracy remains acceptable, increases the security of the model against potential attacks. Additionally, caution should be exercised regarding skip-connections in CNN structures. A study by Wu et al. in 2020 revealed that skip connections could be exploited to craft stronger adversarial samples by utilizing lower-level node values [15]. The authors argue that avoiding skip connections can enhance the model's security although some performance may be reduced.

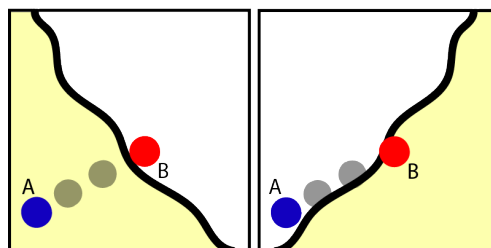


Figure 3. Example of how the recent Symmetry Defense works by utilizing the common asymmetry of decision boundaries.

An intriguing recent development in adversarial defense is the Symmetry Defense proposed by Lindqvist in 2023 [16]. Lindqvist observed that decision boundaries for CNNs rarely exhibit symmetry across translations like reflection or rotation as seen in Figure 3. Building on this

insight, Lindqvist proposes a symmetry defense, where an input image undergoes a sequence of preliminary transformations before processing. Adversarial attacks often prioritize human imperceptibility, generating samples close to the decision boundary. However, the decision area's asymmetry means that translating the image can often neutralize adversarial effects, leading to correct sample classification. This defense was shown to be effective in preventing adversarial attacks to the original model or among models with the same parameters trained on adversarial samples [16].

4 Conclusion

The field of adversarial perturbation attacks is complex and constantly evolving. These attacks pose a significant challenge to the security and reliability of machine learning models. The transferability of adversarial samples between different models adds another layer of complexity, making it a critical area for further research. While factors influencing transferability, such as model architecture and complexity, have been explored, a complete understanding of this phenomenon is yet to be achieved.

Defenses against adversarial attacks are crucial but struggle to keep pace with the ingenuity of attack methodologies. The Symmetry Defense and the incorporation of adversarial samples into training datasets demonstrate the diversity of defense strategies. However, these strategies often rely on a deep understanding of the underlying model dynamics. Ongoing research is needed to deepen our understanding of the principles of convolutional neural networks, adversarial attacks, transferability, and related defense mechanisms. This need is especially pronounced as new developments, such as the use of generative adversarial networks (GANs) for generating adversarial samples, continue to reshape the landscape of this field [17].

In this rapidly evolving domain, collaboration and interdisciplinary efforts are essential to stay ahead of emerging threats and enhance the robustness of machine learning models against adversarial challenges.

References

- [1] I. J. Goodfellow, J. Shlens, and C. Szegedy, “Explaining and harnessing adversarial examples,” 2015.
- [2] P.-Y. Chen, H. Zhang, Y. Sharma, J. Yi, and C.-J. Hsieh, “Zoo: Zeroth order optimization based black-box attacks to deep neural networks without training substitute models,” in *Proceedings of the 10th ACM Workshop on Artificial Intelligence and Security*, CCS ’17, ACM, Nov. 2017.
- [3] J. Gao, J. Lanchantin, M. L. Soffa, and Y. Qi, “Black-box generation of adversarial text sequences to evade deep learning classifiers,” 2018.
- [4] N. Papernot, P. McDaniel, and I. Goodfellow, “Transferability in machine learning: from phenomena to black-box attacks using adversarial samples,” 2016.
- [5] M. Naseer, S. H. Khan, H. Khan, F. S. Khan, and F. Porikli, “Cross-domain transferability of adversarial perturbations,” 2019.
- [6] C. Szegedy, W. Zaremba, I. Sutskever, J. Bruna, D. Erhan, I. Goodfellow, and R. Fergus, “Intriguing properties of neural networks,” 2014.
- [7] N. Carlini and D. Wagner, “Towards evaluating the robustness of neural networks,” 2017.
- [8] J. Chen, M. I. Jordan, and M. J. Wainwright, “Hopskipjumpattack: A query-efficient decision-based attack,” in *2020 IEEE Symposium on Security and Privacy (SP)*, pp. 1277–1294, 2020.
- [9] L. Wu and Z. Zhu, “Towards understanding and improving the transferability of adversarial examples in deep neural networks,” in *Proceedings of The 12th Asian Conference on Machine Learning* (S. J. Pan and M. Sugiyama, eds.), vol. 129 of *Proceedings of Machine Learning Research*, pp. 837–850, PMLR, 18–20 Nov 2020.
- [10] A. Demontis, M. Melis, M. Pintor, M. Jagielski, B. Biggio, A. Oprea, C. Nita-Rotaru, and F. Roli, “Why do adversarial attacks transfer? explaining transferability of evasion and poisoning attacks,” 2019.
- [11] X. Chen, S. Wang, M. Long, and J. Wang, “Transferability vs. discriminability: Batch spectral penalization for adversarial domain adaptation,” in *Proceedings of the 36th International Conference on Machine Learning* (K. Chaudhuri and R. Salakhutdinov, eds.), vol. 97 of *Proceedings of Machine Learning Research*, pp. 1081–1090, PMLR, 09–15 Jun 2019.
- [12] Y. Dong, T. Pang, H. Su, and J. Zhu, “Evading defenses to transferable adversarial examples by translation-invariant attacks,” 2019.
- [13] C. Xie, Z. Zhang, Y. Zhou, S. Bai, J. Wang, Z. Ren, and A. Yuille, “Improving transferability of adversarial examples with input diversity,” 2019.
- [14] K. k. Nakka and M. Salzmann, “Learning transferable adversarial perturbations,” in *Advances in Neural Information Processing Systems* (M. Ranzato, A. Beygelzimer, Y. Dauphin, P. Liang, and J. W. Vaughan, eds.), vol. 34, pp. 13950–13962, Curran Associates, Inc., 2021.

- [15] D. Wu, Y. Wang, S.-T. Xia, J. Bailey, and X. Ma, "Skip connections matter: On the transferability of adversarial examples generated with resnets," 2020.
- [16] B. Lindqvist, "Symmetry defense against cnn adversarial perturbation attacks," 2023.
- [17] S. Hu, X. Liu, Y. Zhang, M. Li, L. Y. Zhang, H. Jin, and L. Wu, "Protecting facial privacy: Generating adversarial identity masks via style-robust makeup transfer," 2022.

Calibration in Quantized LLMs

Arthur Becker

arthur.becker@aalto.fi

Tutor: Erik Schultheis

Abstract

This work deals with calibration of quantized Large Language Models (LLMs). First, this work shows the relation between quantization, calibration expressed as Brier Score and perplexity, general quality metric. Second, this paper proposes an effective method of mitigation of the quality drop caused by quantization. The results may be a step further to the world of accessible LLMs.

KEYWORDS: Large Language Models, Quantization, Calibration, Perplexity, Brier Score

1 Introduction

Not only have increased sizes of Large Language Models (LLMs) led to better model quality, but also to challenges related to required resources. LLMs can be quantized to enable usage of low-capable hardware, however, quantization also causes certain quality decrease.

To evaluate model quality, different metrics can be used such as MMLU [1], GLUE [2], perplexity and others. Thus, the quality drop can be described differently too. It has been already observed, that *perplexity* of a language model becomes worse with increasing degree of compression. [3]

It can be assumed that the model also loses the ability to predict probabilities accurately, i.e. the calibration metrics worsen. *Brier score* [4] can be used to describe and compare calibration of different models. The interest in this metric is justified by the assumption that some simple methods (e.g. Isotonic Regression [], Platt scaling [5]) can probably improve it. Furthermore, improvement of calibration measures can lead to better performance on overall quality metrics like perplexity.

[goals, results, outline](#)

2 Background

2.1 Large Language Model

Large Language Models gained much interest in the last year: especially, after the release of ChatGPT [6]. However, this release was not the first important milestone in the history of LLMs. the basic architecture used by current LLMs was proposed by *Vaswani et al.* [7] in 2017. By that time, the dominant sequence-to-sequence models used architectures based on recurrent neural networks. Recurrent neural networks process data sequentially. The proposed deep neural network architecture was called Transformer and depends on self-attention mechanism without using recurrence or convolutions. The advantage of this model is the ability to process data in parallel that leads to faster inference.

In most of language models, the text is considered to be a sequence of tokens (or pieces of text) that can be for example a single letter, a word or a sub-word, depending on the design choice. Transformer models work with embeddings that are vector representations of tokens. Although the Transformer does not process data sequentially, the sequence position of every token is still an important information, therefore, it is encoded in its embedding. In general, language models aim to model the probabilities distribution of future tokens [6], e.g. given a sequence of tokens t_1, \dots, t_{i-1} , the model predicts the probability p of t_i being the next token:

$$P(t_i | t_1, \dots, t_{i-1}) = p$$

The term *Large Language Model* refers to Transformer-based language models with billions parameters that have been trained on data sets with terabytes of text [8]. Surprisingly, LLMs show *emergent abilities* that are not present in smaller models [6]. For instance, GPT-3 [9] showed

in-context learning ability that refer to the ability to generate expected output by completing the input word sequence after seeing only few task demonstrations or instructions. In contrast, the previous models GPT-2 [10] and GPT-1 [11] cannot perform well on the same tasks [6].

2.2 Quality of Large Language Models

There are different methods to evaluate model performance. MMLU benchmark [1] measures multitask accuracy in tests requiring knowledge and ability to solve problems in fields such as mathematics, law, computer science and others. GLUE benchmark [2] consists of tests evaluating general linguistic abilities of a model such as sentiment classification, paraphrasing and detection of semantic similarity.

Perplexity is a quality metric often used to evaluate the general ability of a language model to predict the next word. A lower perplexity value indicates a better model performance. Given a language model $p(x | x_1, \dots, x_n)$ and a token sequence $t = t_1, \dots, t_N$, perplexity is calculated as:

$$\text{ppl}_p(t_1, \dots, t_n) = \exp\left(\frac{1}{N} - \sum_{i=1}^N \log p(t_i | t_1, \dots, t_{i-1})\right)$$

2.3 Quantization

Increasing LLMs' sizes in order to improving their abilities, researchers face a challenge of increasing demand of computational resources needed for these LLMs. To enable running LLMs on low-cost hardware, *quantization techniques* can be used. The following overview is mainly based on the work of *Gholami et al.* [12].

Quantization methods map real values with floating points of weights and/or activations of a neural network to integer values of lower precision. The *uniform quantization function* maps values of weights and/or activations to uniformly spaced integer values:

$$Q(x) = \text{Int}\left(\frac{x}{S}\right) - Z$$

where S is a scaling factor, Z is an integer zero point and $\text{Int}(x)$ a function rounding x to the nearest integer value. In contrast to the uniform quantization, a *non-uniform quantization function* maps to integer values that are not necessarily uniformly spaced.

After a neural network has been quantized, adjustments of its parameters may be necessary. There are two approaches to do so. The first

approach is *Quantization-Aware Training (QAT)* [13, 14]: model is re-trained after being quantized. This method has a significant disadvantage that re-training's costs are high. A more efficient alternative to it is *Post-Training Quantization (PTQ)* [13, 15] which does not require any re-training. However, the accuracy after PTQ is typically lower than after QAT.

2.4 Calibration

Calibration expresses the ability of a model to predict probabilities accurately. *Well-calibrated* models are essential in the fields with high risks, e.g. in medicine [16]. In context of LLMs, calibration can also play an important role. For instance, GPT-4, a general purpose model not specialized on medical tasks, passed *United States Medical Licensing Examination (USMLE)* [16]. Therefore, there is a discussion about possible medical applications of LLMs in clinics, education and research [17]. Despite showing ability to pass a medical examination, LLMs have not achieved top-scores even in student examinations [16, 18] yet, that is why medical decisions should consider not only the model's output, but also the actual level of confidence.

Calibration can be assessed in different ways. Assume a binary classification problem with two labels: positive and negative. Calibration curves like in Figure 2 visualizes the calibration of a model along the whole range of probabilities. Calibration curves can help understand what probabilities the model predicts better or worse. To plot calibration curve, the probabilities are sorted and divided into bins. The probabilities can be divided either in bins representing uniform intervals or into bins containing equal number of probabilities. The mean predicted probability of a bin is compared to the fraction of positives labels related to the probabilities in the bin.

In order to compare calibration of different models, a single metric can be useful. *Brier Score* [4] is often used for this purpose. Given the predicted probabilities p_1, \dots, p_N and observed probabilities o_1, \dots, o_N , the Brier Score is the average squared prediction error:

Not only can calibration be evaluated differently, but also there are different methods of improving calibration. For instance, *Zadrozny and Elkan* [19] proposed applying *Isotonic Regression* to obtain accurate probability estimates. Training an Isotonic Regressor means finding a step-wise monotonically increasing function $f(x)$ that minimized the predic-

tion error represented by sum of squared differences between and true labels y_1, \dots, y_n and calibrated probabilities $f(p_1), \dots, f(p_n)$:

$$\sum_{i=1}^N (y_i - f(p_i))^2$$

Isotonic regression is a powerful method, but requires a lot of data samples to prevent overfitting [19].

Platt Scaling is another effective technique for calibration of probabilistic models. *Platt* trained parameters m, n of a sigmoid function applied on raw scores of a SVM ¹ in order to output calibrated probabilities [5]:

$$P(y_i = 1 | p_i) = \sigma_{m,n}(p_i) = \frac{1}{1 + \exp(mp_i + n)}$$

Not only is this method effective on SVMs, but also this method showed the ability to calibrate probabilities of other probabilistic models than SVMs, e.g. decision trees, random forests, neural nets. Platt calibration is particularly effective on small calibration sets, but it is less powerful than Isotonic Regression if there is sufficient calibration data [20].

Both methods work on binary classification problems. One-against-all strategy can help calibrate a classifier with more than two labels [19]. In this manner, a calibrator for each class is trained separately. After obtaining a raw probability for a single class, the calibrator specific for this class outputs its calibrated probability.

3 Impact of Quantization on Model Quality

3.1 Perspective on the Calibration Problem

The problem of predicting the next token can be seen from different perspectives. One can view it as a multi-class classification problem with context as an input and the predicted token as a label. The calibration from this point of view would require a one-vs-all calibrator for each of the tokens in the vocabulary that is more resource intensive. This perspective is interesting to investigate, however, it is rational to test perspectives offering simpler solutions first. In this paper, the problem of predicting the next token is seen as a binary classification problem with the context t_1, \dots, t_n and the token hypothesis t_H given as input, and labels $\{0, 1\}$

¹Support Vector Machine

meaning that the token t_H is wrong or correct, respectively. From this point of view, calibration methods require training only one calibrator.

3.2 Data Set

The dataset **WikiText** [21] can be used to compare Brier Scores and perplexity of different models. The used dataset consists of extracts of verified articles on Wikipedia² concatenated to a single text. The data set is divided in three distinct sets: for training, validation and test. In this experiment, the calibration and perplexity will be evaluated based on the test set.

There are two main reasons to use this data set. First, the data set contains text with broad vocabulary covering various topics. This can guarantee to a certain degree that the evaluation results are not limited to a small application domain. Second, worsening of perplexity was already shown on this data set [3], therefore, this experiment can have focus on the calibration perspective.

3.3 Experimental setup

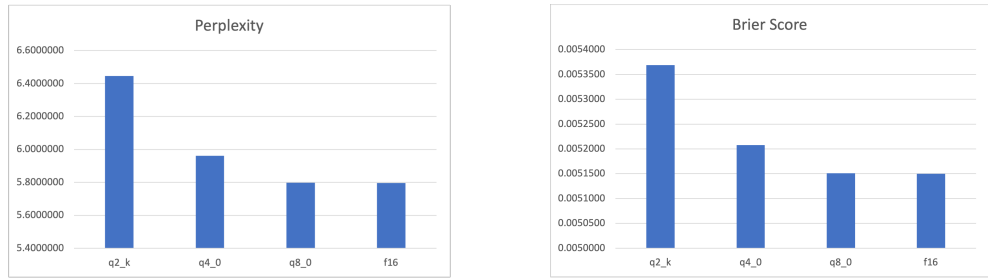
Llama 2 is a family of Large Language Models having from 7B to 70B parameters released by *Touvron et al.* [22] as a model with free access for research and commercial use. This model is one of the strongest free accessible models showing top scores on different benchmarks [1, 2], therefore, the model gained much interest of the AI³ community. Having different ready-for-use implementations of quantization algorithms and inference on different hardware, Llama 2 is a reasonable choice for this experiment. Particularly, an open source project *llama.cpp* [3] was used to quantize the models and perform inference. Versions of Llama 2 (7B) with the following weights resolutions were used: 16-bit floats (f16, original), 8-bit (q8_0), 4-bit (q4_0), and 2-bit (q2_k) integers.

3.4 Considered Probabilities

The model has a hyperparameter *context size* which determines the length of the context window, i.e. the maximum amount of tokens which are considered when predicting the next token. In this experiment, the input is divided into distinct sequences of context length (512 tokens), and only

²<https://wikipedia.org/>

³Artificial Intelligence



(a) Perplexity

(b) Brier Score

Figure 1. Comparison between the original model Llama 2 with 7B parameters (f16) and its quantized versions: 8-bit (q8_0), 4-bit (q4_0), and 2-bit (q2_k)

sequences of full length are used to calculate the investigated metrics. The model becomes these sequences as input, and only the second half is used for the calculations.

This approach does not fully equivalent to the definition of perplexity presented above. The assumption behind this approach is that the probabilities for the first half of a context window are not expressive enough for a model predicting the next token **based on given context**. The practical reason of this implementation is that this implementation allows a fair comparison to the results obtained in [3] using this approach.

The model generates probabilities for all 32000 tokens in vocabulary of Llama 2. In order to investigate the calibration of the models, the probability of the correct token and further 99 highest probabilities are collected and considered to be independent probabilistic experiments. There is 96% probability on average that one of these 100 tokens appears to be the right token, therefore it is a reasonable design choice. Collecting only a fraction of all generated probabilities allows faster evaluation of the experiment results and requires less storage space.

3.5 Results

The results of the comparison are shown in Figure 1. It is noticeable that both perplexity and Brier Score become worse when the model is compressed more. For instance, the 2-bit versions of Llama 2 has the worst results on both metrics. Based on these experiments, the conclusion is that perplexity, Brier Score and quantization are related.

Although the calibration results are worse for the more compressed models, even the most compressed 2-bit quantized model shows results near to perfect calibrated (Figure 2).

4 Mitigation of Quality Drop through Calibration

4.1 Experimental Setup

Because perplexity and Brier Score seem to be related, it is worth to try to impact one metric to improve the other one. That is, the calibration methods can help not only improve Brier Score, but also perplexity.

The experimental setup is similar to the one mention above. The same probabilities and labels are used to calibrate the probabilities with isotonic regression and Platt scaling. The same test set of WikiText [21] is used to compare the results. The validation set of WikiText is used as calibration set. Because the calibration and test sets have the same source, the decision was taken to use additionally another test set to evaluate the general impact of the calibration methods independent from the domain. An extract from the dataset **roleplay_raw_text** [23] consists of conversations between different character in context of a role play. Therefore, this dataset is suitable to challenge the results of calibration methods on different language domains.

There are two important details about the usage of the calibration methods. First, whereas isotonic regression can use probabilities as input, Platt scaling works with the raw outputs of the model [5]. Therefore, it is necessary to apply *an inverse sigmoid function* that can convert the probability to the original scale:

$$\sigma(x) = \ln\left(\frac{x}{1-x}\right)$$

Second, calibrated probabilities do not necessary sum to 1. Moreover, the obtained raw probabilities give a sum of about 0.95 – 0.96 on average. In order to perform a fair comparison, the experiment needs a normalization of calibrated probabilities. As mentioned above, only 100 probabilities

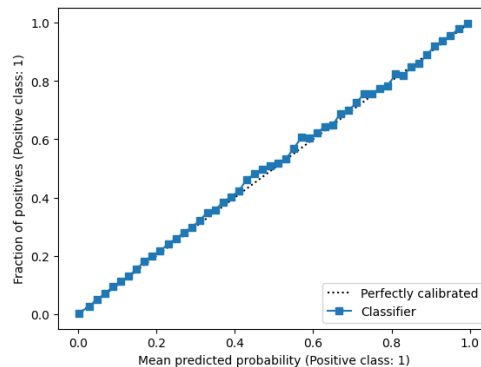


Figure 2. Calibration curve: 2-bit quantized version of Llama 2 before re-calibrating

of the candidates p_i, \dots, p_{100} are obtained for the next token. $c(p_1), \dots, c(p_{100})$ are the probabilities received after the calibration by one of the applied methods. To not increase overall probability average and not impact perplexity by normalization, it is assumed that the sum of the 100 candidates' probabilities does not change after calibration:

$$\sum_{j=1}^{100} c(p_i) = \sum_{j=1}^{100} p_i$$

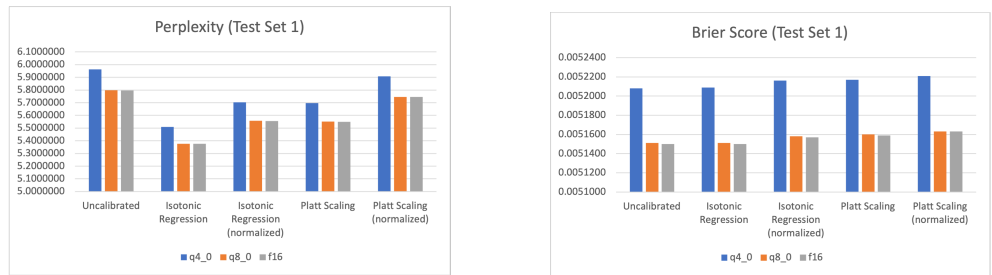
Thus, the formula for a normalized calibrated probability is:

$$\text{norm}(p_{i,j}) = \frac{c(p_{i,1})}{\sum_{j=1}^{100} c(p_i)} \sum_{j=1}^{100} p_i$$

5 Results

In terms of Brier Score, Platt scaling showed improvements neither on the test set 5 nor on the calibration set 4. Isotonic Regression softly improved Brier Score on the calibration set, but not on the test sets.

There noteworthy outcomes regarding perplexity. Both Isotonic Regression and Platt scaling improved perplexity both with normalization and without it. Isotonic Regression seems to be especially effective in improving perplexity. For instance, the 4-bit version outperformed the original version on this metric despite being three times lighter. It is important that the improvements were noticeable also on a out-of-domain test set. This shows that calibration improves perplexity in general



(a) Perplexity

(b) Brier Score

Figure 3. Comparison of calibration methods on the test set 1 [21]. Results for the 2-bit quantized version are not included because its values dominate the scale and make the visual comparison difficult.

6 Conclusion

Not only have experiments shown the impact of quantization on perplexity, but also on the calibration expressed as Brier Score. Isotonic Regression and Platt scaling showed the ability to improve perplexity of quantized models. Isotonic Regression improved perplexity so effectively that more than a three times lighter model outperformed the original model.

The impressive results motivate for further research in this field. However, perplexity is just one of the metrics describing the quality of the model, and this metric is cannot capture all the perspectives of LLMs [24]. Therefore, limits of proposed calibration method should be investigated. If calibration of quantized LLMs shows improvements on further metrics, it will be a ster further in the diretion of accessible LLMs.

References

- [1] D. Hendrycks, C. Burns, S. Basart, A. Zou, M. Mazeika, D. X. Song, and J. Steinhardt, “Measuring massive multitask language understanding,” *ArXiv*, vol. abs/2009.03300, 2020.
- [2] A. Wang, A. Singh, J. Michael, F. Hill, O. Levy, and S. R. Bowman, “GLUE: A multi-task benchmark and analysis platform for natural language understanding,” *CoRR*, vol. abs/1804.07461, 2018.
- [3] “GitHub - ggerganov/llama.cpp: Port of Facebook’s LLaMA model in C/C++ — github.com.” <https://github.com/ggerganov/llama.cpp>. [Accessed 24-11-2023].
- [4] G. W. BRIER, “Verification of forecasts expressed in terms of probability,” *Monthly Weather Review*, vol. 78, p. 1–3, Jan. 1950.
- [5] J. Platt, “Probabilistic outputs for support vector machines and comparisons to regularized likelihood methods,” *Adv. Large Margin Classif.*, vol. 10, 06 2000.
- [6] W. X. Zhao, K. Zhou, J. Li, T. Tang, X. Wang, Y. Hou, Y. Min, B. Zhang, J. Zhang, Z. Dong, Y. Du, C. Yang, Y. Chen, Z. Chen, J. Jiang, R. Ren, Y. Li, X. Tang, Z. Liu, P. Liu, J.-Y. Nie, and J.-R. Wen, “A survey of large language models,” 2023.
- [7] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin, “Attention is all you need,” 2023.
- [8] M. Shanahan, “Talking about large language models,” 2023.
- [9] T. Brown, B. Mann, N. Ryder, M. Subbiah, J. D. Kaplan, P. Dhariwal, A. Nee-lakantan, P. Shyam, G. Sastry, A. Askell, S. Agarwal, A. Herbert-Voss, G. Krueger, T. Henighan, R. Child, A. Ramesh, D. Ziegler, J. Wu, C. Winter, C. Hesse, M. Chen, E. Sigler, M. Litwin, S. Gray, B. Chess, J. Clark, C. Berner, S. McCandlish, A. Radford, I. Sutskever, and D. Amodei, “Language models are few-shot learners,” in *Advances in Neural Information Processing Systems* (H. Larochelle, M. Ranzato, R. Hadsell, M. Balcan, and H. Lin, eds.), vol. 33, pp. 1877–1901, Curran Associates, Inc., 2020.
- [10] A. Radford, J. Wu, R. Child, D. Luan, D. Amodei, and I. Sutskever, “Language models are unsupervised multitask learners,” 2019.
- [11] A. Radford and K. Narasimhan, “Improving language understanding by generative pre-training,” 2018.
- [12] A. Gholami, S. Kim, Z. Dong, Z. Yao, M. W. Mahoney, and K. Keutzer, “A survey of quantization methods for efficient neural network inference,” 2021.
- [13] M. Nagel, M. Fournarakis, R. A. Amjad, Y. Bondarenko, M. van Baalen, and T. Blankevoort, “A white paper on neural network quantization,” 2021.
- [14] S. A. Taylor, J. Fernandez-Marques, and N. D. Lane, “Degree-quant: Quantization-aware training for graph neural networks,” in *International Conference on Learning Representations*, 2021.

- [15] Z. Yao, R. Yazdani Aminabadi, M. Zhang, X. Wu, C. Li, and Y. He, “Zeroquant: Efficient and affordable post-training quantization for large-scale transformers,” in *Advances in Neural Information Processing Systems* (S. Koyejo, S. Mohamed, A. Agarwal, D. Belgrave, K. Cho, and A. Oh, eds.), vol. 35, pp. 27168–27183, Curran Associates, Inc., 2022.
- [16] H. Nori, N. King, S. M. McKinney, D. Carignan, and E. Horvitz, “Capabilities of gpt-4 on medical challenge problems,” 2023.
- [17] A. J. Thirunavukarasu, D. S. J. Ting, K. Elangovan, L. Gutierrez, T. F. Tan, and D. S. W. Ting, “Large language models in medicine,” *Nature Medicine*, vol. 29, no. 8, pp. 1930–1940, 2023.
- [18] A. J. Thirunavukarasu, R. Hassan, S. Mahmood, R. Sanghera, K. Barzangi, M. El Mukashfi, and S. Shah, “Trialling a large language model (chatgpt) in general practice with the applied knowledge test: Observational study demonstrating opportunities and limitations in primary care,” *JMIR Medical Education*, vol. 9, p. e46599, Apr. 2023.
- [19] B. Zadrozny and C. Elkan, “Transforming classifier scores into accurate multiclass probability estimates,” in *Proceedings of the eighth ACM SIGKDD international conference on Knowledge discovery and data mining*, KDD02, ACM, July 2002.
- [20] A. Niculescu-Mizil and R. Caruana, “Predicting good probabilities with supervised learning,” in *Proceedings of the 22nd international conference on Machine learning - ICML '05*, ICML '05, ACM Press, 2005.
- [21] S. Merity, C. Xiong, J. Bradbury, and R. Socher, “Pointer sentinel mixture models,” 2016.
- [22] H. Touvron, L. Martin, K. Stone, P. Albert, A. Almahairi, Y. Babaei, N. Bashlykov, S. Batra, P. Bhargava, S. Bhosale, D. Bikel, L. Blecher, C. C. Ferrer, M. Chen, G. Cucurull, D. Esiobu, J. Fernandes, J. Fu, W. Fu, B. Fuller, C. Gao, V. Goswami, N. Goyal, A. Hartshorn, S. Hosseini, R. Hou, H. Inan, M. Kardas, V. Kerkez, M. Khabsa, I. Kloumann, A. Korenev, P. S. Koura, M.-A. Lachaux, T. Lavril, J. Lee, D. Liskovich, Y. Lu, Y. Mao, X. Martinet, T. Mihaylov, P. Mishra, I. Molybog, Y. Nie, A. Poulton, J. Reizenstein, R. Rungta, K. Saladi, A. Schelten, R. Silva, E. M. Smith, R. Subramanian, X. E. Tan, B. Tang, R. Taylor, A. Williams, J. X. Kuan, P. Xu, Z. Yan, I. Zarov, Y. Zhang, A. Fan, M. Kambadur, S. Narang, A. Rodriguez, R. Stojnic, S. Edunov, and T. Scialom, “Llama 2: Open foundation and fine-tuned chat models,” 2023.
- [23] “TokenBender/roleplay_raw_text · Datasets at Hugging Face — huggingface.co.” https://huggingface.co/datasets/TokenBender/roleplay_raw_text. [Accessed 24-11-2023].
- [24] A. Jaiswal, Z. Gan, X. Du, B. Zhang, Z. Wang, and Y. Yang, “Compressing llms: The truth is rarely pure and never simple,” 2023.

1 Appendix: Experiment Results

Table 1. Perplexity (calibration set)

	q2_k	q4_0	q8_0	f16
Uncalibrated	6.7358950	6.1455070	5.9864830	5.9836660
Isotonic Regression	6.1088260	5.6473340	5.5167100	5.5144290
Isotonic Regression (normalized)	6.3747500	5.846311	5.703004	5.700664
Platt Scaling	6.3883490	5.8637430	5.7211930	5.7185630
Platt Scaling (normalized)	6.6901380	6.0818920	5.9248930	5.9222290

Table 2. Brier Score (calibration set)

	q2_k	q4_0	q8_0	f16
Uncalibrated	0.0054550	0.0052810	0.0052240	0.0052230
Isotonic Regression	0.0054510	0.0052780	0.0052220	0.0052200
Isotonic Regression (normalized)	0.0054620	0.005285	0.005228	0.005227
Platt Scaling	0.0054630	0.0052880	0.0052310	0.0052300
Platt Scaling (normalized)	0.0054720	0.0052920	0.0052350	0.0052340

Table 3. Perplexity (test set 1)

	q2_k	q4_0	q8_0	f16
Uncalibrated	6.4454550	5.9618580	5.7978280	5.7961940
Isotonic Regression	5.8960620	5.5097640	5.3767770	5.3757050
Isotonic Regression (normalized)	6.1521060	5.703024	5.557634	5.55621
Platt Scaling	6.1263060	5.6976950	5.5503730	5.5488220
Platt Scaling (normalized)	6.4114380	5.9081970	5.746216	5.7446900

Table 4. Brier Score (test set 1)

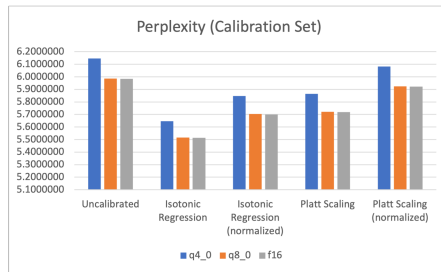
	q2_k	q4_0	q8_0	f16
Uncalibrated	0.0053690	0.0052080	0.0051510	0.0051500
Isotonic Regression	0.0053680	0.0052090	0.0051510	0.0051500
Isotonic Regression (normalized)	0.0053790	0.005216	0.005158	0.005157
Platt Scaling	0.0053770	0.0052170	0.0051600	0.0051590
Platt Scaling (normalized)	0.0053870	0.0052210	0.0051630	0.0051630

Table 5. Perplexity (test set 2)

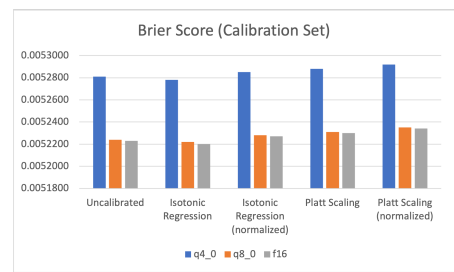
	q2_k	q4_0	q8_0	f16
Uncalibrated	9.7133240	9.0946550	8.9413800	8.9393760
Isotonic Regression	8.6853400	8.2034810	8.0733970	8.0723270
Isotonic Regression (normalized)	9.0257010	8.4846060	8.3414670	8.3397010
Platt Scaling	9.0567010	8.5012700	8.3658830	8.3641420
Platt Scaling (normalized)	9.6299590	8.9639630	8.7985270	8.7965920

Table 6. Brier Score (test set 2)

	q2_k	q4_0	q8_0	f16
Uncalibrated	0.0061740	0.0060540	0.0060270	0.0060270
Isotonic Regression	0.0061720	0.0060540	0.0060280	0.0060270
Isotonic Regression (normalized)	0.0061850	0.0060620	0.0060330	0.0060330
Platt Scaling	0.0061840	0.0060630	0.0060330	0.0060330
Platt Scaling (normalized)	0.0061940	0.0060660	0.0060360	0.0060350

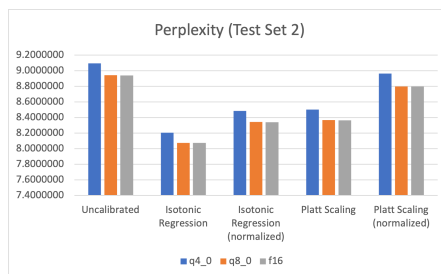


(a) Perplexity

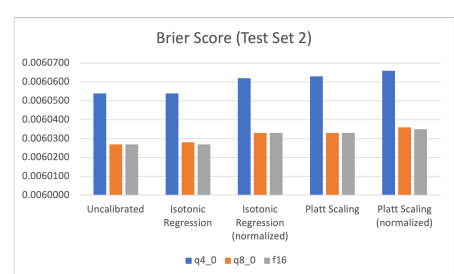


(b) Brier Score

Figure 4. Calibration results (calibration set [21])



(a) Perplexity



(b) Brier Score

Figure 5. Calibration results (test set 2 [23])

Do cookie consent forms protect your data?

Daniel Kaluski

daniel.kaluski@aalto.fi

Tutor: Sanna Suoranta

Abstract

KEYWORDS: cookies, consent, privacy

The introduction of GDPR in 2018 caused websites in the EU to ask for user consent when processing sensitive data. Despite this, websites tend to overlook user preferences and nudge users to give full consent even when they would not normally do that. In this paper we review a few studies to show how websites nudge users to give their consent, whether it has any impact on the data being collected and what data is actually being collected through the cookies. In many cases cookie consent forms give the illusion that the user has a say in whether their data is being collected. Not only do websites ignore which cookies the user has accepted, but websites collect other data that the cookie consent forms don't necessarily cover creating fingerprints where multiple websites can identify the same user even when using different credentials. Accepting all cookies is often made very easy while declining all cookies requires additional steps. Even though many studies claim data from the cookies is being collected without consent, there is little information about what the data most often is. These findings suggest that more studies regarding collection specific data need to be done. Moreover, the compliance of GDPR among websites should be more strict and users should be made more aware, how to spot violations and how to control what data they give away.

1 Introduction

Most websites today use internet cookies or generally known as cookies. Their initial purpose is to store data such as web tokens. Websites use HTTP and HTTPS protocols to load pages and since they are stateless, the TCP connection is closed between each request [1]. Web tokens stored in the cookies are proof that the credentials have been checked and confirmed to be correct, allowing the user to stay signed in and not having to insert credentials again when using the website. There are concerns, however, whether the websites are storing more information than needed. While web tokens or IDs might be beneficial, storing information such as internet history in the cookies does not benefit the user. Instead, it might benefit other entities who can use that data for advertising.

Users should have a say in what data is gathered from them when visiting a website or using an app [2]. Many websites use cookie consent forms to give users the choice of which data to store. In the forms, users can allow or disallow the use of certain cookies. Users might have to consent to some cookies in order to access the websites. Even if the user declines cookies, the website could still save some data that the user did not consent to, despite it being against the General Data Protection Regulation (GDPR).

There are concerns about what gathered data is used for. Today, personal data could be valuable for companies when it comes to advertising, as mentioned earlier. There is also the question of what else they could do with that data. If the company themselves do not use it for advertising, they might be selling it to someone else, all this without the general public knowing about it [3, 4]. People ought to know where and how their own personal data is being used.

The aim of this paper is to analyze whether cookie consent forms actually matter when it comes to identifying users and collecting sensitive data, how websites try to obtain consent from users and what type of data is gathered.

2 GDPR and consent

General data protection regulation or commonly known as GDPR is a set of rules and regulations on how personal data is supposed to be processed and stored in the EU region [2]. The regulations apply to people if they are

in the EU. This means if a company is stationed outside and they want to gather data from people in the EU, they must comply with GDPR. Because of this, companies might not want to make their websites accessible in the EU region.

The GDPR resolution states how the data must be lawfully gathered, kept up to date and how only necessary data is allowed to be gathered to mention some. It also states what security measures must be applied to ensure safe storage of the data. Companies storing the data are responsible and accountable if there is a data leak or GDPR laws have been violated [2].

If companies want to gather more than just the necessary data, user must give consent. GDPR states rules on how consent can be given and how it should be communicated to the user. On gdpr.eu it states the following [2]:

- Consent must be “freely given, specific, informed and unambiguous.”
- Requests for consent must be “clearly distinguishable from the other matters” and presented in “clear and plain language.”
- Previously given consent can be withdrawn whenever and the decision to do so must be honored. The legal basis of the the processing cannot be simply change to one of the other justifications.
- Children under 13 can only give consent with permission from their parent.
- Documentary evidence of consent must be kept.

All websites in the EU region must follow these rules. Yet, Kampanos et al. [5] have shown that from 14000 websites in the United Kingdom and 3000 websites in Greece, only 44% and 48%, respectively, show a cookie banner when accessing the page even when cookies are collected. According to European Commission [6], 74% of the observed 478 websites displayed banners, of which 54% did not ask for any consent. If so many websites have failed to comply with GDPR in some way, one should perhaps question all other cookie consent forms.

3 Gaining consent

One way to easily obtain full consent from the user is to make accepting all the cookies much easier than rejecting them, as shown by several researches [7, 8, 9]. All possible options for choosing the cookies the user wants to give consent to can be hidden behind a few button clicks or an unpleasantly looking form. All the while consenting to all cookies can be just one click away behind a very nice looking button that attracts the users attention .

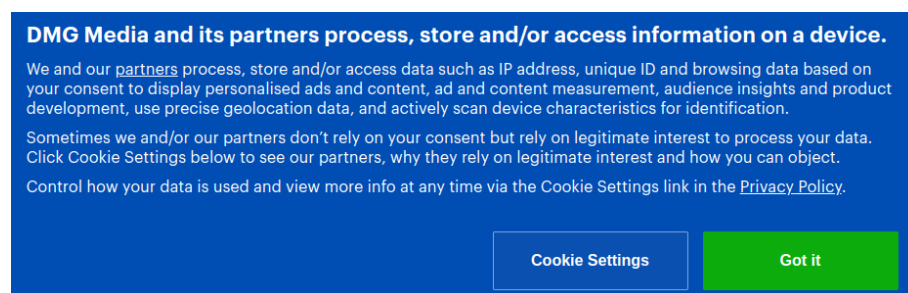


Figure 1. Example of a cookie banner which attempts to get the user to accept all cookies

Figure 1 shows the cookie consent form when access dailymail.co.uk for the first time. There is a very easy option to accept all cookies by pressing the "Got it!" button. The green color is clearly visible and catches user's attention. The "Cookie settings" button, however, was made the same color as the banner blending in with all the other text. It also has a different tone when compared to "Got it!". Most noticeably, there is not button for rejecting all cookies.

Figure 2 shows the cookie settings of the dailymail.co.uk. Similarly to the first banner, it has a button for accepting all cookies which again is clearly visible and separates itself from others. No button for rejecting all cookies is visible forcing the user to go through all the settings to ensure unwanted cookies are not turned on. The "Vendors" tab has over a thousand different vendors to choose and the user can individually choose which ones to allow, with some of the already being enabled by default.

While not GDPR compliant, some websites implement cookie walls. A banner is displayed blocking the usage of the website until a choice has been made [10]. Disallowing cookies can lead to the website showing only part of its content and, in some cases, refuse to show the content completely. Only by accepting all cookies the user can get full access to

How we personalise your experience

Purposes / Features
Vendors

Purposes

Associated Newspapers Ltd. participates in the [IAB Europe Transparency & Consent Framework \(IAB TCF\)](#) and complies with its specifications and policies. This Consent Management Platform is operated by us and has identification number 27.

We work with advertising partners to show you content and advertisements for products and services you might like. Understand how your data may be used and who our partners are below:

1	Store and/or access information on a device	Consent <input type="checkbox"/> ▼
2	Personalised ads and content, ad and content measurement, audience insights and product development	Legitimate interest <input checked="" type="checkbox"/> Consent <input type="checkbox"/> ▼

Special Purposes

- A. Ensure security, prevent fraud, and debug ▼
- B. Technically deliver ads or content ▼

Features

- A. Match and combine offline data sources ▼
- B. Link different devices ▼
- C. Receive and use automatically-sent device characteristics for identification ▼

Privacy & Cookies Policy

Save & Exit
Allow all

Figure 2. Pressing the cookie settings in Figure 1

Welcome to our ad-free, tracking-free version of Healthline

We detect that you are in one of the member countries of the UK/EU/EEA, which is now subject to the General Data Protection Regulation (GDPR). Unfortunately, a tracking-free version of our full website is currently unavailable in these countries. We are engaged on the issue and committed to looking at options that support our full range of digital offerings to this market

While we continue to identify technical compliance solutions that will allow all readers to experience our content, we are providing you with 10 articles that highlight the breadth and quality of our content. You are on this page because you disallowed the purposes listed in the "How we use your data" section of our [Privacy Settings](#) page.

We believe that health information should be free to everyone and we rely on advertising to make this possible on our family of websites [Healthline](#), [Medical News Today](#), [Greatist](#), and [Psych Central](#). Providing the best health information in the world is expensive. We spend up to thousands of dollars per article to ensure it is accurate and precise with quality review by a doctor or other certified, trained medical professional.

The modern digital advertising ecosystem functions on cookies and other data. Cookies and similar tracking technologies allow us to improve your browsing experience, store or access relevant information, customize content and offers, personalize advertising, analyze our traffic, and better understand you. In order to access all of our content and website features, we request you allow all of the purposes listed in the "How we use your data" section of our [Privacy Settings](#) page.

As you access these 10 articles and learn more about Healthline, we hope to earn your trust about our [privacy](#) and [advertising](#) practices.

When you return to this site in the future, you will have the opportunity to revisit and renew your [Privacy Settings](#). Thanks again.

17 Effective Ways to Lower Your Blood Pressure

March 08, 2021

High blood pressure, or hypertension, is called the "silent killer" for good reason. It often has no symptoms, but is [...] from Healthline

7 Foods to Help Your Acid Reflux

March 08, 2021

We include products we think are useful for our readers. If you buy through links on this page, we may [...] from Healthline

Figure 3. Declining cookies on <https://www.healthline.com/> restricts access to the website

the website. Should the cookies be essential for the website to work, for example web tokens, user does not have to give consent. This means the website is deliberately trying to obtain consent for additional unnecessary cookies.

4 Gathered data

There is the big question that many are eager to know when it comes to the topic: What is the actual data gathered from users and for what is its purpose? This section attempts to answer this question with an example. Firstly subsection 4.1 shows an example of website and the cookies it uses is given. Subsection 4.2 explains how other data than cookies can be used in identifying users.

4.1 Cookies in depth

There are studies that have done research about whether websites track users using different methods and how trackers are put into browsers [11]. For example, Matte et al.[12] found that in some cases the website stores a positive consent even when the user had explicitly declined cookies. One example of storing an advertising cookie without the user's consent is also presented by Santos et al. [10]. These studies come to

the conclusion that users are indeed being tracked more than one would anticipate and often without the user's consent. Yet, they do not mention exactly, what is the actual data that is being collected. Often saying it is for advertising, for example.

Since users should be able to know, what data is collected, information about what each cookie does falls to this category. Next, a few examples of the collected cookies and their categories will be presented gathered from www.helsinki.com, the website for the university of Helsinki. There are four classified categories of cookies and their amount. Necessary technical cookies (40), Functional cookies (22), Statistics cookies (47) and Marketing cookies (43). There is also a category for Unclassified cookies (12), however they all have "Pending" as their "Purpose" type.

For the most part there is a good explanation about the purpose and the data collected. Necessary technical cookies are storing the language preference, information about which server-cluster the user has connected to in order to optimize load balancing and for saving consent. Functional cookies are often about allowing third-party services like Twitter, currently known as X, with the website. Statistics cookies collect data such as visiting duration and, according to the form itself, they cannot be used to identify users. Lastly marketing cookies: the form claims they collect data to show more relevant advertisements. There are a few cookies that check the last URL the user was on to detect how they ended up on this website.

While most of the cookie's, especially the necessary cookies, explanations are clear some marketing cookies leave wondering, what is the actual data gathered. One marketing cookie called "muc_ads" that is provided by Twitter Inc. says its purpose is the following: "Collects data on user behaviour and interaction in order to optimize the website and make advertisement on the website more relevant." There is no explanation of what is meant by user behavior or interaction. One can speculate that interaction in this scenario means clicking different buttons and advertisements on the website as the type of the cookies is written as "HTTP". A similar case is with a cookie called "_gcl_au [x3]" provided by Google. Its purpose is the following: "Used by Google AdSense for experimenting with advertisement efficiency across websites using their services." No de-

tails provided about what the experimenting in this context means. It also mentions using other services without providing information what those services are. It should also be noted that there is no additional button that would take the user somewhere where the specific cookies are explained in more detail or with examples. They merely give a link to the provider.

4.2 Techniques to follow users

Usually cookies save data needed for the session like a web token and user options. In addition, they can save data useful for advertisers. What is often not considered is the environment where the websites are rendered. Some information does not need to be stored within the cookies for it to be beneficial for advertisers or identifying users online. Web browsers have access to the hardware in which all websites are rendered. While often needed, this information can be used to create a fingerprint of the user. This is called browser fingerprinting [13].

There also exists first-party ID leaking in which the primary website gathers identifying data of the user and leaks it to third parties [14].

Third-party synchronization takes identifying data from different sources and then tries to match them to identify the user. Let's take a situation where a user visits websites A and B. Both require at least a username to log in to. Even though the user created two completely different usernames, a fingerprint was created allowing the two websites to be able to identify the user as the same person, because he visited the websites on the same machine.

4.3 Obeying cookie consent forms

Papadogiannakis et al.[14] describes how different websites and their ways of creating fingerprints of users were studied. Using a web crawler, they visited websites and automatically either accepted, denied or did not make any choice regarding cookies. They analyzed the collected cookies, looked at the URL of HTTP GET requests as well as the body of POST requests looking for any unique IDs. ID synchronization is detected if network traffic from more than one website contains the same ID. In total 850 000 websites were crawled with disturbing results. When all cookies were

accepted, they found that 65.35% of the websites engaged in first-party leaking and 29.61% in third-party synchronization.

However, Papadogiannakis [14] do not give explicit information about what the user actually agreed to when accepting all cookies was given. That is why it might be safe to assume the user did consent to some sort of ID synchronization in most websites where all cookies were accepted. When the cookies were rejected or no decision has yet been made, the results are even more disturbing. When cookies were rejected, 56.41% engaged in first party leaking and 26.20% in third-party synchronization. When no action was made, the results were 52.38% and 24.03% respectively. One would expect these numbers to be much lower, yet they barely differ when accepting all cookies. In the case of browser fingerprinting, they found that 73.5% of the crawled websites performed fingerprinting no matter the choice. In some cases rejecting cookies made browser fingerprinting even worse.

5 Discussion

It is evident by the studies mentioned in this paper and examples in figures that websites do try to make users accept all cookies to obtain the most data through dark patterns. Dark patterns in this context means influencing the user behavior through nudging and other deceptive tactics to make actions they would not normally do. Yet one big question still remains: What is exactly the data that is gathered by the cookies? Many studies claim that data is being collected but fail to pinpoint what the data actually is. Is it the browser history, shopping history or perhaps something else? There is also rarely the distinction between the cookies gathered when consent is given, when it is not given and when no choice was yet been made. It is easy to say that simply more studies should be done about the gathered data and how it is processed in the backend. The difficulty in here is that it might not be entirely obvious what the cookies inside the browser are as for the most part their values look like random strings to the naked eye. Often the parsed data from cookies also consists of random data in JSON format.

It is also not beneficial for companies to directly say which data they gather and what is exactly done with that data as it might cause more

people to actively reject cookies. Even when a website does inform what each individual cookie does, there is no guarantee that they explained it all or if they actually comply with GDPR.

When it comes to fingerprinting in GDPR, it is debatable whether using browser fingerprinting falls under it. The data that the browser collects includes things such as window size, operating system and browser type [15]. This type of data is in a gray area when arguing whether it is sensitive or not [16]. However, it is still being used to provide very accurate fingerprints of users. Santos et al. [17] argue that using fingerprints to respawn cookies, i.e. inserting cookies in browsers by first fingerprinting, would fall under the GDPR and is thus illegal if the user did not consent to it.

6 Conclusion

This paper examined the different ways websites ask for consent to collect data from the users as well as the data that is gathered depending on the choices the user made. Since GDPR was put into act in 2018, websites operating in the EU region are required to ask for consent whenever they gather data. Despite this companies still neglect GDPR. They often give users the illusion of choice by blocking the content partially or completely if the user not consent to cookies. Option to deny all cookies can be troublesome to find or it does not exist at all the while the "Accept all" -button is made to be clearly visible nudging users to click it. Still, no matter what the choice is, websites gather identifying data at a similar rate as if the user agreed to everything.

This paper examined how websites try to convince the user to accept cookies, whether cookie consent forms respect the choice the user has done and what data is being collected. Since GDPR was put into act in 2018, websites operating in the EU region are required to ask for consent whenever they gather sensitive data that is not mandatory for the website to work. While there is often the choice to deny most if not all cookies, accepting all cookies is made much easier nudging users to accept all cookies. Consent forms also give an illusion that they somehow protect the user even when denying all cookies. Websites take part in collecting sensitive data that is able to identify users which in turn can be used to show advertisements more relevant to the user. GDPR, and in turn consent forms, do not necessarily protect users from their data being collected as some of the data

can be deemed to not be sensitive.

References

- [1] “Hypertext transfer protocol http/1.1.” “<https://datatracker.ietf.org/doc/html/rfc2616>”. Accessed 21.11.2023.
- [2] gdpr.eu, “What is gdpr, the eu’s new data protection law?” “<https://gdpr.eu/what-is-gdpr/>”. Accessed 21.11.2023.
- [3] D. Milmo, “Facebook sued for collecting personal data to target adverts.” “<https://www.theguardian.com/technology/2022/nov/21/woman-launches-high-court-challenge-of-facebook-use-of-personal-data-for-ads>” Accessed 21.11.2023.
- [4] P. Papadopoulos, N. Kourtellis, P. R. Rodriguez, and N. Laoutaris, “If you are not paying for it, you are the product: How much do advertisers pay to reach you?,” in *Proceedings of the 2017 Internet Measurement Conference*, 2017.
- [5] G. Kampanos and S. F. Shahandashti, “Accept all: The landscape of cookie banners in greece and the uk,” in *IFIP International Conference on ICT Systems Security and Privacy Protection*, Springer, 2021.
- [6] Article 29 Working Party, “Cookie sweep combined analysis – report.” WP229, 2 2015. Available from the Article 29 Working Party archives or the European Data Protection Board (EDPB).
- [7] T. H. Soe, O. E. Nordberg, F. Guribye, and M. Slavkovik, “Circumvention by design-dark patterns in cookie consent for online news outlets,” in *Proceedings of the 11th nordic conference on human-computer interaction: Shaping experiences, shaping society*, 2020.
- [8] M. Eryn and B. Eleanor, “Prospective consent: The effect of framing on cookie consent decisions,” in *CHI Conference on Human Factors in Computing Systems Extended Abstracts*, 2022.
- [9] C. Utz, M. Degeling, S. Fahl, F. Schaub, and T. Holz, “(un) informed consent: Studying gdpr consent notices in the field,” in *Proceedings of the 2019 acm sigsac conference on computer and communications security*, pp. 973–990, 2019.
- [10] C. Santos, N. Bielova, and C. Matte, “Are cookie banners indeed compliant with the law? deciphering eu legal requirements on consent and technical means to verify compliance of cookie banners,” *arXiv preprint arXiv:1912.07144*, 2019.
- [11] P. Papadopoulos, N. Kourtellis, and E. Markatos, “Cookie synchronization: Everything you always wanted to know but were afraid to ask,” in *The World Wide Web Conference*, pp. 1432–1442, 2019.
- [12] C. Matte, N. Bielova, and C. Santos, “Do cookie banners respect my choice?: Measuring legal compliance of banners from iab europe’s transparency and consent framework,” in *2020 IEEE Symposium on Security and Privacy (SP)*, pp. 791–809, IEEE, 2020.

- [13] U. Iqbal, S. Englehardt, and Z. Shafiq, “Fingerprinting the fingerprinters: Learning to detect browser fingerprinting behaviors,” in *2021 IEEE Symposium on Security and Privacy (SP)*, pp. 1143–1161, IEEE, 2021.
- [14] E. Papadogiannakis, P. Papadopoulos, N. Kourtellis, and E. P. Markatos, “User tracking in the post-cookie era: How websites bypass gdpr consent to track users,” in *Proceedings of the web conference 2021*, pp. 2130–2141, 2021.
- [15] M. A. Obidat, S. Obeidat, J. Holst, and T. Lee, “Canvas deceiver—a new defense mechanism against canvas fingerprinting,” pp. 66–74, 2020.
- [16] F. Adamsky, S. Schiffner, and T. Engel, “Tracking without traces—fingerprinting in an era of individualism and complexity,” in *Privacy Technologies and Policy: 8th Annual Privacy Forum, APF 2020, Lisbon, Portugal, October 22–23, 2020, Proceedings 8*, pp. 201–212, Springer, 2020.
- [17] I. Fouad, C. Santos, A. Legout, and N. Bielova, “Did i delete my cookies? cookies respawning with browser fingerprinting.,” *arXiv*, 2021.

Investigating temporal Gaussian processes for solar PV forecasting

Danila Mokeev

danila.mokeev@aalto.fi

Tutor: Paul E. Chang

Abstract

Gaussian processes (GPs) are a powerful class of non-parametric machine learning models that can be applied to many types of problems. While they work for small datasets, they suffer from their computational cost of $\mathcal{O}(n^3)$ in the number of data points. In this paper, we show how this problem can be solved in $\mathcal{O}(n)$ for 1-D data, and apply to it a solar PV dataset

KEYWORDS: PV forecasting, Gaussian Processes, efficient temporal GPs

1 Introduction

Photovoltaic (PV) forecasting is a major ongoing challenge, especially with the current transition towards renewables. Given the intermittent nature of solar energy, better predictions reduce the use of last-minute interventions of heavily CO₂-emitting petrol-based power plants.

Temporal forecasting of PV output can be accomplished through a variety of models, ranging from straightforward autoregressive approaches to more sophisticated deep neural network architectures. However, the former often falls short in terms of accuracy, while the latter may exhibit slower computational speeds. Gaussian processes (GPs) emerge as a ju-

delicious middle ground, particularly for scenarios with limited data points (small n), as they enable the generation of reliable predictions in little time. This efficacy is achieved through the application of a periodic prior, specifying the class of functions under consideration for fitting. For big values of n however, the computation becomes intractable due to the $\mathcal{O}(n^3)$ complexity in training (which involves a matrix inversion [1]). In this paper, we will try to investigate the approach introduced in [2], namely State Space Variational Inference (SSVI), for forecasting PV production in $\mathcal{O}(n)$ on 1D temporal data.

2 Current challenges

In recent years, great progress has been made in addressing the scaling issue mentioned above. Most approaches work by approximating the true solution, as it is very difficult to overcome the cubic complexity while remaining exact. We can broadly split the proposed solutions into 2 classes: inducing points, where the goal is to reduce the number of used data points, and other linear algebra techniques, where we approximate the covariance matrix by making assumptions about its structure.

The inducing points method has a rich history, as many variations were proposed ([3],[4], [5], [6], etc). First versions only used a subset of the training points by selecting them according to some criterion, while more modern approaches try to find $m \ll n$ new points through some objective function. This approach allows us to calculate the covariance matrix more efficiently in $\mathcal{O}(m^2n)$.

Covariance matrix approximation is a different direction some papers take. This can for example involve decomposing it into a Kronecker [7] or a Toeplitz [8] product. These approaches are however limited by the training points they use (they need to be regularly spaced), which is why more recent work [9] has shown to allow for any inducing points by approximating the covariance matrix between the training and induced points with cubic interpolation. This allows to bring the complexity down to $\mathcal{O}(n + m \log m)$.

These previous approximations however make no assumptions on the type of data they are operating on. Assuming our data is 1-D, we can convert

our original GP into an equivalent discretized stochastic differential equation (SDE) [2]. The main advantage of using this method is that it gives exact results while having an $\mathcal{O}(n)$ complexity, thanks to a Kalman filter/smoothing operating on a state-space reformulation.

3 A brief overview of GPs

3.1 Definition

Intuitively, Gaussian processes (GPs) are an extension of the multivariate normal distribution, where the domain is continuous (infinite), instead of being finite. It is therefore a stochastic process, where any finite linear combination of its variables is a Gaussian multivariate. Formally speaking, if $\{X_t, t \in T\}$ is a stochastic process (where T can be any set of indices, but usually time for us), it is Gaussian if for every set of indices $t_1 \dots t_n$, the joint distribution $(X_{t_1}, X_{t_2} \dots X_{t_n})$ is a multivariate normal [10].

Analogous to how a normal multivariate is parametrized by a mean vector and covariance matrix, a GP $f(\mathbf{x})$ is parametrized by a **mean function** $\mu(\mathbf{x}) : \mathbb{R}^d \rightarrow \mathbb{R}$ and a **covariance function** (also called **kernel**) $k(\mathbf{x}, \mathbf{x}') : \mathbb{R}^d \times \mathbb{R}^d \rightarrow \mathbb{R}$ with $\mathbf{x} \in \mathbb{R}^d$. They are typically defined as

$$\mu(\mathbf{x}) = \mathbb{E}[f(\mathbf{x})] \tag{1}$$

$$k(\mathbf{x}, \mathbf{x}') = \mathbb{E}[(f(\mathbf{x}) - \mu(\mathbf{x}))(f(\mathbf{x}') - \mu(\mathbf{x}'))] \tag{2}$$

We will assume $\mu(\mathbf{x}) = 0$, as it does not limit our modeling capability and reduces the number of hyperparameters. Playing with these parameters allows us to define a prior on the types of functions that the process will generate (which is the natural extension of vectors when we increase the number of dimensions to ∞). We will usually denote it as

$$f(\mathbf{x}) \sim \mathcal{GP}(\mu(\mathbf{x}), k(\mathbf{x}, \mathbf{x}'))$$

3.2 Learning with GPs

As in any machine learning task, we will start with a dataset $\mathcal{D} = \{\mathbf{X}, \mathbf{y}\}$ with $\mathbf{X} \in \mathbb{R}^{n \times d}$ and $\mathbf{y} \in \mathbb{R}^n$. n is the number of observations and d the dimension of the input. The output is one-dimensional, but this can be generalized.

Now utilizing Bayes' formula, and assuming that all samples are independent, we can write the posterior of the GP as

$$p(\mathbf{f} \mid \mathcal{D}, \theta) = \frac{\mathcal{N}(\mathbf{f} \mid \mathbf{0}, \mathbf{K})}{p(\mathbf{y} \mid \mathbf{X}, \theta)} \prod_{i=1}^N p(y_i \mid f(\mathbf{x}_i)). \quad (3)$$

The numerator corresponds to the GP prior, the denominator is the marginal, and the part to the right is the likelihood, which is simply the product of point likelihoods thanks to independence. We also have that $\mathbf{f} = [f(x_1) \dots f(x_n)]$ and $\mathbf{K}_{i,j} = k(x_i, x_j) \forall i, j = 1 \dots n$ the gram matrix between training points.

Regression with GPs will correspond to calculating the value $f_* = f(\mathbf{x}_*)$ for a test point x_* . We will assume that the measurement process is not perfect, hence $y = f(\mathbf{x}) + \epsilon$, $\epsilon \sim \mathcal{N}(0, \sigma^2 I)$. Because the noise is gaussian, the likelihood is also gaussian and due to our assumption of gaussian prior (by definition of a gaussian process), the posterior can easily be calculated in closed form. One can derive [1]

$$(\mathbf{f}_* \mid \mathcal{D}, \mathbf{X}_*) \sim \mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma}) \quad (4)$$

$$\boldsymbol{\Sigma} = \mathbf{K}_{**} - \mathbf{K}_*^T (\mathbf{K} + \sigma^2 I)^{-1} \mathbf{K}_* \quad (5)$$

$$\boldsymbol{\mu} = \mathbf{K}_*^T (\mathbf{K} + \sigma^2 I)^{-1} \mathbf{y} \quad (6)$$

4 Kernels for GPs

Kernels play a major role in the results one can obtain with a GP model. They encode our assumptions about the data, notably continuity, smoothness, or periodicity, therefore making some kernels more suitable for a particular trend than others. They are also easily combinable through a sum, a product, or exponentiation. In the following sections, we will briefly go over some common ones, with a plot to illustrate their purpose.

4.1 Radial basis function (RBF) kernel

This kernel is written as follows:

$$\mathbf{K}(\mathbf{x}, \mathbf{x}' \mid \mathbf{l}) = \exp - \frac{\|\mathbf{x} - \mathbf{x}'\|^2}{2l^2}$$

with \mathbf{l} the lengthscale, hence controlling how smooth or wiggly the functions are. This kernel is one of the most used ones in practice as it is

continuously differentiable (very smooth) and allows to extrapolate up to l timesteps. Please note that this kernel only depends on the difference of the norm of the inputs, not their values. Such types of kernels are called isotropic, which means they are shift-invariant (which implies they are also stationary).

4.2 Periodic kernel

This kernel is written as follows:

$$\mathbf{K}(\mathbf{x}, \mathbf{x}' | l, p) = \exp\left(-\frac{2 \sin(\pi|x - x'|/p)}{l^2}\right)$$

with p the period, or the distance between 2 neighboring peaks, and l the lengthscale, analogous to the RBF kernel. It is also isotropic, and we can easily model periodic functions.

4.3 Matern kernel

The general form of the kernel is the following:

$$\mathbf{K}(\mathbf{x}, \mathbf{x}' | \nu, l) = \frac{2^{1-\nu}}{\Gamma(\nu)} \left(\frac{\sqrt{2\nu} |x - x'|}{l}\right)^\nu K_\nu \left(\frac{\sqrt{2\nu} |x - x'|}{l}\right)$$

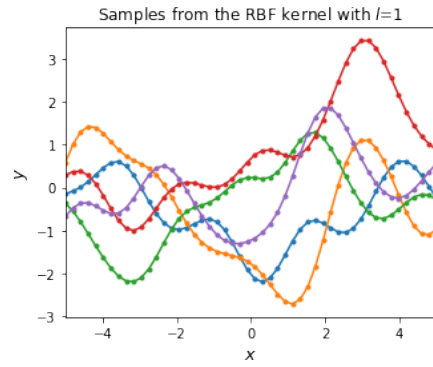
with ν a parameter that controls the smoothness of the functions (the sampled functions will be $\lceil \nu \rceil + 1$ times differentiable), l the lengthscale, K_ν the modified Bessel function and $\Gamma(\nu)$ the Gamma function. This form however is very rarely used, and instead, we prefer to fix ν to $\{1/2, 3/2, 5/2\}$.

4.4 Quasi-periodic kernel

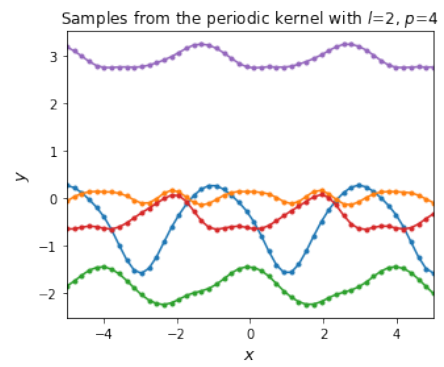
As mentioned at the beginning of this section, kernels can be combined through various operations, for example, a product. The quasi-periodic kernel is defined as the product of a Matern (here Matern52) and a periodic kernel:

$$\mathbf{K}(\mathbf{x}, \mathbf{x}' | \ell_1, \ell_2, p) = \left(1 + \frac{\sqrt{5} |x - x'|}{\ell_1} + \frac{5(x - x')^2}{3\ell_1^2}\right) \exp\left(-\frac{\sqrt{5} |x - x'|}{\ell_1} - \frac{2 \sin(\pi|x - x'|/p)}{\ell_2^2}\right)$$

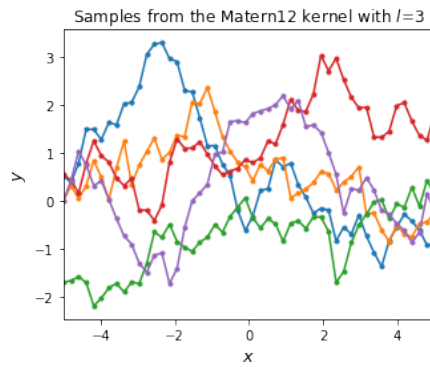
In the following experiments, we will use this kernel as it models well our data patterns, with a constant period but a varying shape/scale.



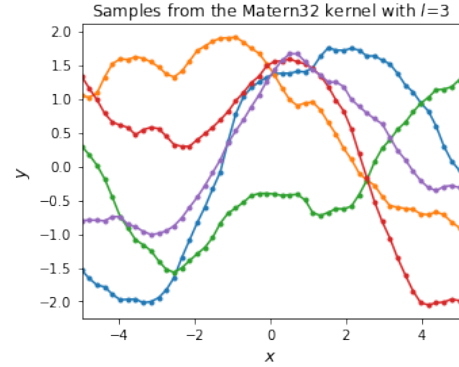
(a) RBF kernel



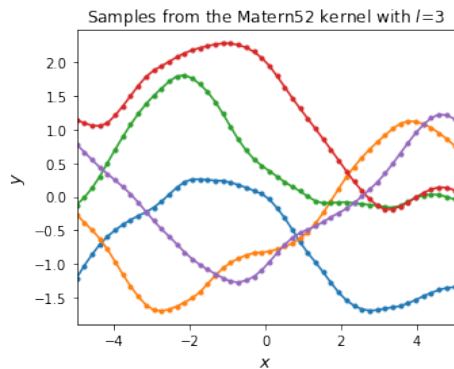
(b) Periodic kernel



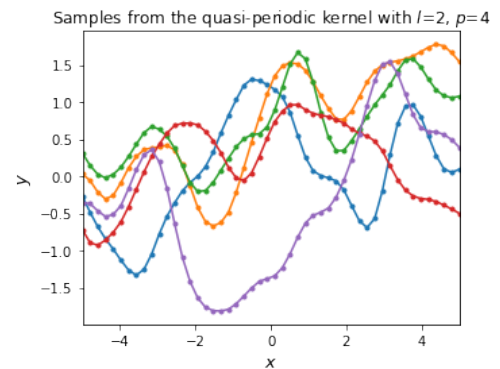
(c) Matern 1/2 kernel



(d) Matern 3/2 kernel



(e) Matern 5/2 kernel



(f) Quasi-periodic kernel

5 State-space Variational Inference (SSVI)

The novel approach introduced in [2] is based on a reformulation of the standard GPs to a Stochastic Differential Equation (SDE).

An SDE is a generalization of the classical differential equation, as it includes one or more terms which are stochastic processes. This means that the solutions are not simple functions anymore, but stochastic processes.

One possible class of continuous SDE is the following:

$$a_0 f(t) + a_1 \frac{df(t)}{dt} + \dots + a_m \frac{d^m f(t)}{dt^m} = w(t) \quad (7)$$

The only stochastic term here is $w(t)$, white noise. Due to the fact that it is Gaussian, and the Gaussian property is preserved under linear transformations, the solution of the equation will also be a GP. One can show that we can rewrite it as a continuous state-space model [2]:

$$\begin{aligned} \frac{d\mathbf{f}(t)}{dt} &= \mathbf{F}\mathbf{f}(t) + \mathbf{L}w(t), \\ y_k &= \mathbf{H}\mathbf{f}(t_k) + \varepsilon_k, \end{aligned}$$

All the higher-order derivatives are now contained in the \mathbf{F} matrix and \mathbf{f} contains all the m components of the solution, each of them a 1-D stochastic process. The ε term is added as we consider the data measurement process to be noisy. As we are only interested in dealing with 1-D data, we use $\mathbf{H} = (1 \ 0 \ \dots \ 0)$ to keep only the first component. \mathbf{L} has the form $(0 \ \dots \ 0 \ 1)^T$.

This previous formulation is however not useful for practical applications, and we must discretize it. This gives us

$$\begin{aligned} \mathbf{f}_k &= \mathbf{A}_{k-1}\mathbf{f}_{k-1} + \mathbf{q}_{k-1}, \text{ where } \mathbf{q}_{k-1} \sim N(\mathbf{0}, \mathbf{Q}_{k-1}), \\ y_k &= \mathbf{H}\mathbf{f}_k + \varepsilon_k, \text{ where } \varepsilon_k \sim N(0, \sigma_{\text{noise}}^2). \end{aligned}$$

This is the final version of the equations and the form that SSVI utilizes (also called dual reformulation). If the kernel in GP space adheres to common conditions, as is the case with typical kernels, it can undergo a reformulation into the state-space domain. These equations are then solved (also called inference in the signal processing world) using first a Kalman filter (forward pass), and then a Kalman smoother (backward pass). Both passes are linear in time, giving it a total $\mathcal{O}(n)$ complexity.

6 Dataset

The dataset is extracted from [11] and only uses one of the given features, the PV production, making it one dimensional. It spans an entire year, where the data is collected every 5 minutes from 8am to 4pm for a certain number of systems located at different locations. Each system corresponds to a different set of solar panels. We create 52 folds of one week each and then split it into training and testing. We also apply some

preprocessing on the data, namely standardization and smoothing with a Savgol filter. This is what the data looks like for one week and system:

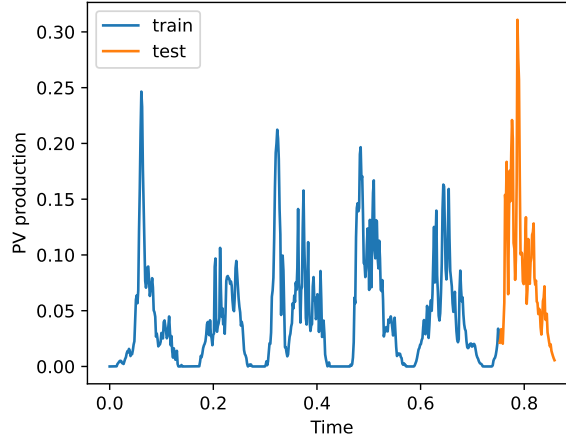


Figure 2. Example of one week from the dataset

7 Experiments

As the goal is to show that SSVI is fast and scales linearly in terms of datapoints, while also staying accurate, we will be conducting 2 experiments in which we will try to validate our claims. For both of them, the gaussian process model comes from the `gpflow` library, and uses a periodic Matern32 kernel. The SSVI model uses a quasi-periodic kernel, which is very similar to the GP one and utilizes a fast and numerically stable implementation with the `jax` framework. The autoregressive model is of order 3, coming from the `statsmodels` library. All work is done using the Python language. The data only uses one fold and one system.

7.1 Experiment 1: training time comparison

In this experiment, we will train our method on folds of increasing length, with a comparison with the classical GP approach.

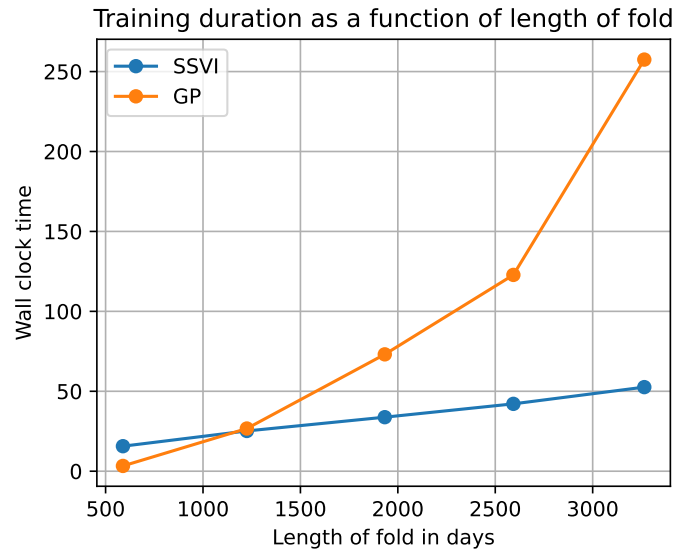


Figure 3. Comparison of training time of SSVI, compared to standard GPs

As we can see, the training times for SSVI scale linearly, and even manage to remain below 1 minute for $n = 3500$. The standard GP approach follows the $\mathcal{O}(n^3)$ curve, validating our initial claims.

7.2 Experiment 2: prediction accuracy comparison on training data

In this experiment, we will again train SSVI on various fold lengths, here providing a comparison in terms of the prediction accuracy with a simple autoregressive model of order 3 (AR3), as well as a classic GP. The little triangles represent the different data points.

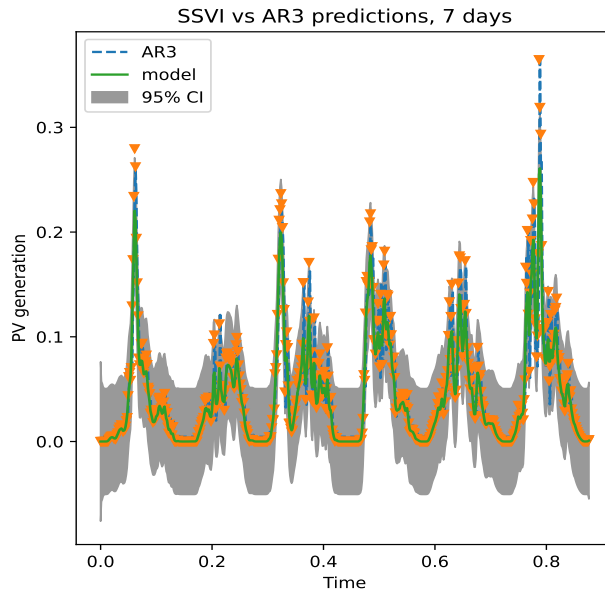


Figure 4. Accuracy of SSVI vs AR3 vs GP on 7 days, training data

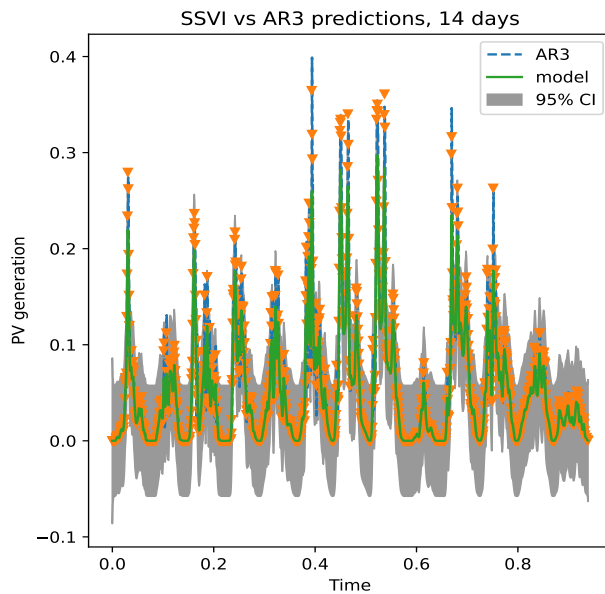


Figure 5. Accuracy of SSVI vs AR3 vs GP on 14 days, training data

As it can be seen, SSVI provides an equivalent prediction accuracy when compared to other models, even when n increases.

8 Conclusion

This paper has evaluated SSVI as an alternative approach to GPs for 1-D temporal data, and concluded that it indeed does provide an efficient and accurate approach when compared to classical GPs.

References

- [1] C. E. Rasmussen and C. K. I. Williams, *Gaussian processes for machine learning*. London, England: MIT Press, 2005. ISBN 9780262182539
- [2] A. Solin, “Stochastic Differential Equation Methods for Spatio-Temporal Gaussian Process Regression,” Doctoral thesis, School of Science. ISBN 978-952-60-6711-7 (electronic), 978-952-60-6710-0 (printed) 2016. [Online]. Available: <http://urn.fi/URN:ISBN:978-952-60-6711-7>
- [3] N. Lawrence, M. Seeger, and R. Herbrich, “Fast sparse gaussian process methods: The informative vector machine,” in *Advances in Neural Information Processing Systems*, S. Becker, S. Thrun, and K. Obermayer, Eds., vol. 15. MIT Press, 2002. [Online]. Available: https://proceedings.neurips.cc/paper_files/paper/2002/file/d4dd111a4fd973394238aca5c05bebe3-Paper.pdf
- [4] E. Snelson and Z. Ghahramani, “Sparse gaussian processes using pseudo-inputs,” in *Advances in Neural Information Processing Systems*, Y. Weiss, B. Schölkopf, and J. Platt, Eds., vol. 18. MIT Press, 2005. [Online]. Available: https://proceedings.neurips.cc/paper_files/paper/2005/file/4491777b1aa8b5b32c2e8666dbe1a495-Paper.pdf
- [5] J. Quiñero-Candela and C. E. Rasmussen, “A unifying view of sparse approximate gaussian process regression,” *Journal of Machine Learning Research*, vol. 6, no. 65, pp. 1939–1959, 2005. [Online]. Available: <http://jmlr.org/papers/v6/quinonero-candela05a.html>
- [6] M. Titsias, “Variational learning of inducing variables in sparse gaussian processes,” in *Proceedings of the Twelfth International Conference on Artificial Intelligence and Statistics*, ser. Proceedings of Machine Learning Research, D. van Dyk and M. Welling, Eds., vol. 5. Hilton Clearwater Beach Resort, Clearwater Beach, Florida USA: PMLR, 16–18 Apr 2009, pp. 567–574. [Online]. Available: <https://proceedings.mlr.press/v5/titsias09a.html>
- [7] Y. Saatchi, “Scalable inference for structured gaussian process models,” Ph.D. dissertation, 11 2011.
- [8] J. P. Cunningham, K. V. Shenoy, and M. Sahani, “Fast gaussian process methods for point process intensity estimation,” in *Proceedings of the 25th International Conference on Machine Learning*, ser. ICML '08. New York, NY, USA: Association for Computing Machinery, 2008. doi: 10.1145/1390156.1390181. ISBN 9781605582054 p. 192–199. [Online]. Available: <https://doi.org/10.1145/1390156.1390181>
- [9] A. G. Wilson and H. Nickisch, “Kernel interpolation for scalable structured gaussian processes (KISS-GP),” *CoRR*, vol. abs/1503.01057, 2015. [Online]. Available: <http://arxiv.org/abs/1503.01057>
- [10] Wikipedia contributors, “Gaussian process — Wikipedia, the free encyclopedia,” 2023, [Online; accessed 16-September-2023]. [Online]. Available: https://en.wikipedia.org/w/index.php?title=Gaussian_process&oldid=1173180554
- [11] “openclimatefix (Open Climate Fix),” Oct. 2023, [Online; accessed 2. Oct. 2023]. [Online]. Available: <https://huggingface.co/openclimatefix>

- [12] “Gaussian processes (1/3) - From scratch,” Jan. 2019, [Online; accessed 17. Sep. 2023]. [Online]. Available: <https://peterroelants.github.io/posts/gaussian-process-tutorial>
- [13] Andy Jones, “The Matérn class of covariance functions,” *Andy Jones*, Jul. 2021. [Online]. Available: <https://andrewcharlesjones.github.io/journal/matern-kernels.html>
- [14] “Kernel Cookbook,” Aug. 2017, [Online; accessed 30. Sep. 2023]. [Online]. Available: <https://www.cs.toronto.edu/~duvenaud/cookbook>
- [15] P. E. Chang, W. J. Wilkinson, M. E. Khan, and A. Solin, “Fast variational learning in state-space gaussian process models,” in *2020 IEEE 30th International Workshop on Machine Learning for Signal Processing (MLSP)*, 2020. doi: 10.1109/MLSP49062.2020.9231560 pp. 1–6. [Online]. Available: <https://doi.org/10.1109/MLSP49062.2020.9231560>
- [16] J. Bradbury, R. Frostig, P. Hawkins, M. J. Johnson, C. Leary, D. Maclaurin, G. Necula, A. Paszke, J. VanderPlas, S. Wanderman-Milne, and Q. Zhang, “JAX: composable transformations of Python+NumPy programs,” 2018. [Online]. Available: <http://github.com/google/jax>

Microservice - Advantages and weaknesses in industry

Doan Tuan Dat

dat.doan@aalto.fi

Tutor: Ylä-Jääski Antti

Abstract

Microservices technique is state-of-the-art in architecture and widely used in Big Tech, the most prosperous and influential technological companies in the IT industry. It not only provides various benefits to the company, from design and development to operation, but also costs resources to maintain and build the architecture. This paper analyses the advantages and drawbacks of microservices for application products. Second, this paper presents challenges in building microservices applications and discusses when and why to use microservices. It is also essential to evaluate the use cases for microservices in modern software demand to show that microservices are not a silver bullet for all cases.

KEYWORDS: Microservices, Monolithic, Service orientation architecture, Enterprise

1 Introduction

One of the most critical factors in software development is software architecture. Software architecture can determine the life cycle, maintenance, performance and cost of a product. During the preceding years, the first and the primary idea that came from practitioners is monolithic

approach. However, this methodology quickly shows weaknesses and demands a new solution in software development [1]. Microservices were then proposed and applied to enterprise products to improve issues caused by the monolithic approach. Microservices architecture (MSA) provides the ability to develop highly maintainable, testable and loosely coupled systems. It also offers availability and scalability for the system [2].

Using different methodologies, including literature review, comparison studies and analysis of industry numbers, the main contributions to this paper are comparing MSA and monolithic approaches, discussing challenges around microservices and evaluating industrial solutions.

The paper is organised as follows: Section 2 comprises background information about different generations of software architecture. Section 3 evaluates the strengths and weaknesses of MSA in different aspects. Section 4 presents challenges in choosing MSA as a central direction. Section 5 reports some use cases in which microservices succeed and fail. Finally, the last section yields a conclusion about when and why to use microservices.

2 Background

2.1 Monolithic architecture and service-oriented architect

Before the introduction of MSA, monolith and service-oriented architecture (SOA) were previous generations. Laigner et al. [3] claim that the monolithic application is software in which different components, such as authorization, business logic and notification module, are combined into a single program from a single platform, and it is deployed as a single standalone program. Dragoni et al. [2] argue that a monolith is a software application whose modules cannot be executed independently. Although the monolith system can run multiple processes, accessing multiple central processing units, it indicates the system can be scaled, but it is required to be completely cloned [1]. The target of changing from object-oriented solution to service-oriented solution is better when controlling software system design, implementation and evolution. This led to the introduction of microservices afterwards. Another advantage of componentization is the ability to adapt business capabilities to the system architecture [4].

2.2 Microservices

There is no exact definition for the term *microservice*. The term *microservices* was first introduced in 2011 at a workshop in Venice and gained popularity in years later. The ideas around it are then presented by James Lewis and Adrian Cockcroft at Netflix as a "fine-grained SOA" [4]. It is a new computing paradigm that decomposes the traditional monolithic system into a set of fine-grain services. These services are independently developed, tested and deployed [5]. Until now, MSA has been defined as a composition of small services, each running its own processes, providing one part of the business logic, and communicating via lightweight mechanisms that are built on top of SOA [2, 6, 4]. In addition, each service has a private database that is inaccessible to other services directly [6].

3 Evaluation

3.1 Advantages

MSA offers considerable benefits to software development. The key benefits that would be discussed in this paper are availability, containerization, and technology heterogeneity [7].

3.1.1 Availability

The first difference between monolithic architecture and MSA is the number of components. As a single unit, monolithic systems are intimidated by single-point failure. All elements in a monolithic design are tightly coupled, and any failure can cause the whole system to crash. On the other hand, all services in MSA are isolated, and any component collapse does not stop the entire system.

3.1.2 Scalability

Scalability is the most vital reason why systems are migrated into MSA [8]. Monolithic components share many of the same resources, including databases and servers. This makes the structure overloaded. It is different from MSA; each service has its own resources, and the service can handle more requests without being collapsed. Scalability helps the system serve more users and provide a positive experience.

3.1.3 Containerisation

Heinrich et al. define containerization as a technology to virtualize applications in a lightweight strategy that has resulted in a significant uptake in cloud application management [7]. For microservices, container association is the perfect support. This reduces the downtime, deployment time and cost of microservices applications because the container does not need to host an extensive application [5]. Moreover, MSA and containerized applications are currently supported by orchestration frameworks, such as Kubernetes and Docker Swarm. These frameworks offer numerous features, including automated scheduling, load balancing, scalability, automated rollouts and rollbacks [9].

3.1.4 Technology Heterogeneity

Different services and different teams can use different technologies for implementation. For example, a service that performs complex calculations might be built in a language known for high performance, while another service that deals with high I/O might use a different technology better suited for that task. The team members can choose technology stacks that are familiar; this allows products to be developed, tested, released and deployed faster, and productivity is enhanced [2].

3.2 Disadvantages

Diverse gains have been introduced. However, MSA is not a silver bullet, and there are problems when engineers use MSA in their products. This paper briefly evaluates the main disadvantages of MSA: complexity, maintenance, performance and testing.

3.2.1 Complexity

A single-point failure is not the main problem with MSA. However, the more complex the application, the more weakness appears. Components in MSA are dependable, and a single crash component can also lead to the collapse of the whole system. Maintaining high availability in MSA is a complicated task since MSA itself does not provide high availability by default [10]. A large system requires complex behaviours, such as sophisticated message chains and disturbing environmental configurations. As a result, this boosts the debugging and development costs and leads to a higher probability of run-time failure [5].

3.2.2 Performance

Opposed to a single notable component, microservices communicate through the network. It has been observed that performance is slightly reduced in MSA [11]. There are multiple reasons for the drop in performance. Firstly, the number of CPU instructions needed to process a single request, such as bare process, docker host and docker bridge, in MSA is double compared to Monolith [11]. Secondly, network communication is a negative factor. Distributed applications are connected through networks, which leads to additional latency and speed loss [5].

3.2.3 Integration testing

Testing can be a challenge in MSA when the system is really large and there are many connections between components. A service may block others; therefore, the testing process of different pieces needs to be done together. There are more interfaces and larger sizes to test because MSA needs to be tested on both single components and the whole system working together. Testing a single service is possible, but the cost may increase, and the collaboration of different services is a problematic process [2].

3.2.4 Maintenance

Maintenance is usually more expensive in MSA systems. Diverse technology stacks and system complexity make MSA systems lack standardisation [11]. More issues will arise if engineers do not understand and do not have the skills to manage the complexity of the system. Furthermore, continuous integration (CI) and continuous delivery (CD) are not as simple compared to the monolithic strategy. Each service would have a different pipeline, and coordinating deployments across multiple services requires accurate planning and automation.

3.3 Challenges

Apart from the advantages and drawbacks, MSA has significant design, operation and maintenance challenges. The following raises some main challenges in those steps.

3.3.1 Design stage

Architecture and design patterns

Microservices must follow different principles: scalability, availability, resiliency, independence, decentralised governance, failure isolation, auto-

provisioning, and continuous delivery through DevOps. A significant topic is applying those principles and building an entirely flawless system. There are different design patterns for microservices, such as Aggregator, API gateway, Chained or Chain of Responsibility [12]. There are even more valuable and complex design patterns for microservices. It can be seen that the first and foremost problem in working with MSA systems is to design and apply complicated theories. The initial step can impact the workload and quality of subsequent steps.

Data consistency

Unlike monolithic applications, data is centralised, and data in MSA systems is distributed in different databases and data warehouses. The classic approach cannot work for MSA applications. There are alternative solutions to handle this issue, such as cloning-based methods and private database methods. In cloning-based methods, database instances are replicated for all different components. Data overwriting needs the broadcast operation of all other nodes to inform about data altering. In the private database method, all components have a private database. However, all choices have different flaws. Designing a perfect MSA is a real challenge [2].

3.3.2 Development stage

In the development stage, practitioners may feel the most complicated solving data consistency, distributed transactions, and query complexity [13]. The following subsection discusses the debugging strategy, which is one of the most challenging.

Debugging

Debugging in an MSA system is problematic because it is distributed, dynamic and concurrent. Compared to a monolithic system, MSA has complexity in both design and implementation. To begin with, MSA has inconsistent communications and environment configurations. It is because of the distribution of instances and heterogeneity in technology. When users call an API, complex asynchronous interactions are engaged. This compound chain can easily lead to runtime failure. Furthermore, microservices instances can be created, changed, and destroyed dynamically, which means there is a lack of correspondence between microservices and system nodes in MSA. All the complexity and inconsistency of components make it rigorous to debug in MSA [5].

3.3.3 Operation stage

During the operation stage, monitoring and management are the most cared subjects. Helpful information about these facets will be covered below.

Monitoring

Monitoring in MSA can be done using application performance management (APM) tools. This tool collects data and detects abnormal behaviour in the system. APM tools could be better and work correctly to support new programming languages. Second, detecting abnormal behavior in the MSA system is also tricky. The system is dynamic and does not have a steady state. There is a possibility of a false alarm using an existing monitoring system [7].

Management

The MSA system appears to be overly complicated at every stage, even for those in management. MSA management has such problems as cascading failures, operational complexity, service coordination, and service location [13]. To reduce the workload of the management process, the project needs to be conducted with clear documentation, service boundaries, and good team collaboration.

4 Case Study

4.1 Netflix

Netflix is one of the most prominent streaming platforms in the digital landscape. At first, Netflix was just a DVD rental startup. They sold and delivered DVDs to customers. In 2008, there was a setback that suspended their data centre and DVD shipping for days. A demand for a new solution forced Netflix to begin a system transformation.

4.1.1 Netflix Architecture

Netflix system includes two main components: Amazon Web Services (AWS) cloud components used for hosting and an in-house network for content delivery [14].

Content delivery

The Netflix content delivery has three critical components: client, back-end, and content delivery network (CDN) [14].

Backend:

Netflix backend consists of the following elements [14]:

- EC2 used as computing instance
- S3 used as storage
- Business logic microservices
- AWS DynamoDB and Cassandra used as scalable database structure
- Big data processing tool: Hadoop, Spark, Flink, and other tools.
- Custom-built video processing and transcoding tools.
- Open connect CDN refers to Open Connect Appliances (OCAS), a fast streaming and storing video network developed by Netflix.

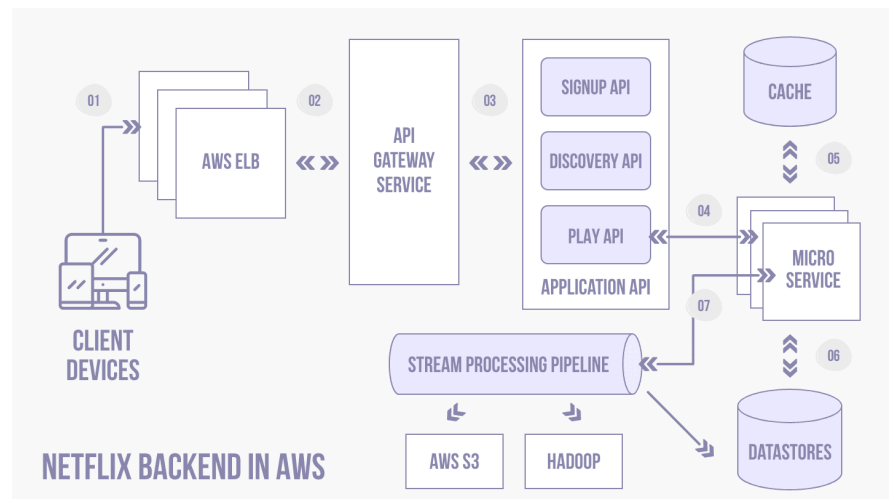


Figure 1. Netflix backend [14]

The backend services are isolated. These components communicate when the client sends a complicated request, such as playing video. The microservices chain works as follows when users press the playback button:

- AWS elastic load balancing (ELB) handles requests from users and forward them to API Gateway Service, which is built with advanced functions, such as traffic monitoring, security and dynamic routing.

- The request is then forwarded to Application API component. There are multiple API, including sign-up, discovery, authentication and play.
- If users click the playback button, Play API will call the microservices chain. For example, Playback Apps, Steering and Cache-Control are called. The services are separated and activated depend on sent requests.
- There are more services, such as Stream Processing Pipeline, which is used to analyse user data for personal recommendation. Data from Stream Processing Pipeline is then operated using big data tools, such as AWS S3, Hadoop and Cassandra.

4.1.2 Benefits of MSA

The result of transforming to cloud-native is impressive to both the company and users. As a result of the elasticity of the AWS cloud, they can distribute billions of workloads with low latency to different geographies and increase availability up to 99.99%. By moving to cloud-native, they do not need to manage the data centres themselves, and they can focus on entertaining content and business. This reduces on-premise operation costs and increases resources for different business tasks [15]. In 2023, Netflix has more than 200 million users, which is the highest rate compared to other platforms [16]. Meanwhile, they can serve 4K video quality, which is a notable output in the streaming industry.

4.2 Discussion

After evaluation and analysis of the case study of Netflix, the perquisite of MSA is evident. It offers many benefits, from costs to performance. In my opinion, MSA is a great and appreciable choice when planning for a new product. However, developers should take into account factors, such as applicable resources, requirements, and time constraints of the project. On occasion, monolith can be a favourable choice for small-scale and low-budget projects, such as start-up ventures and early-stage enterprises. These parties lack the conditions to conduct and operate a complete MSA project. On the other hand, firms that are already successful and want to expand the market and serve more users can pick MSA as a means of gaining access to more customers and raising their revenue.

5 Conclusion

This paper evaluates the advantages and disadvantages of MSA in comparison to the monolithic approach. Although MSA has so many profits, it is not always an ubiquitous option for building enterprise solutions. The paper has also presented challenges in building MSA products, which is a consideration for project conductors. There are so many problems with debugging, monitoring and management. Finally, this paper has shown a scenario in which MSA benefits the company and generates revenue. The MSA approach may reduce operational costs when combined with cloud-native. The concluding decision for the architecture depends on business, the available resources and maybe investors.

References

- [1] B. Götz, D. Schel, D. Bauer, C. Henkel, P. Einberger, and T. Bauernhansl, "Challenges of production microservices," *Procedia CIRP*, vol. 67, pp. 167–172, 2018. 11th CIRP Conference on Intelligent Computation in Manufacturing Engineering, 19-21 July 2017, Gulf of Naples, Italy.
- [2] N. Dragoni, S. Giallorenzo, A. L. Lafuente, M. Mazzara, F. Montesi, R. Mustafin, and L. Safina, "Microservices: yesterday, today, and tomorrow," 2017.
- [3] R. Laigner, Y. Zhou, M. A. V. Salles, Y. Liu, and M. Kalinowski, "Data management in microservices: State of the practice, challenges, and research directions," *Proc. VLDB Endow.*, vol. 14, p. 3348–3361, sep 2021.
- [4] M. Fowler and J. Lewis, "Microservices.," 2014. <https://martinfowler.com/articles/microservices.html>.
- [5] G. Liu, B. Huang, Z. Liang, M. Qin, H. Zhou, and Z. Li, "Microservices: architecture, container, and challenges," in *2020 IEEE 20th International Conference on Software Quality, Reliability and Security Companion (QRS-C)*, pp. 629–635, 2020.
- [6] K. Gos and W. Zabierowski, "The comparison of microservice and monolithic architecture," in *2020 IEEE XVIth International Conference on the Perspective Technologies and Methods in MEMS Design (MEMSTECH)*, pp. 150–153, 2020.
- [7] R. Heinrich, A. van Hoorn, H. Knoche, F. Li, L. E. Lwakatare, C. Pahl, S. Schulte, and J. Wettinger, "Performance engineering for microservices: Research challenges and directions," in *Proceedings of the 8th ACM/SPEC on International Conference on Performance Engineering Companion, ICPE '17 Companion*, (New York, NY, USA), p. 223–226, Association for Computing Machinery, 2017.
- [8] A. Henry and Y. Ridene, *Migrating to Microservices*, pp. 45–72. Cham: Springer International Publishing, 2020.

- [9] Y. Wang, H. Kadiyala, and J. Rubin, “Promises and challenges of microservices: an exploratory study,” *Empirical Software Engineering*, vol. 26, p. 63, May 2021.
- [10] V. Saquicela, G. Campoverde, J. Avila, and M. E. Fajardo, “Building microservices for scalability and availability: Step by step, from beginning to end,” in *New Perspectives in Software Engineering* (J. Mejia, M. Muñoz, Á. Rocha, and Y. Quiñonez, eds.), (Cham), pp. 169–184, Springer International Publishing, 2021.
- [11] F. Auer, V. Lenarduzzi, M. Felderer, and D. Taibi, “From monolithic systems to microservices: An assessment framework,” *Information and Software Technology*, vol. 137, p. 106600, 2021.
- [12] R. Bhojwani, “Design patterns for microservices .,” 2022. <https://dzone.com/articles/design-patterns-for-microservices>.
- [13] J. Soldani, D. A. Tamburri, and W.-J. Van Den Heuvel, “The pains and gains of microservices: A systematic grey literature review,” *Journal of Systems and Software*, vol. 146, pp. 215–232, 2018.
- [14] K. Varshneya, “Understanding design of microservices architecture at netflix,” 2021. <https://www.techaheadcorp.com/blog/design-of-microservices-architecture-at-netflix/>.
- [15] S. Radhakrishnan, “Netflix case study: Analysis,” 2021. <https://www.linkedin.com/pulse/netflix-case-study-analysis-ajeenkya-suryawanshi>.
- [16] R. Binns, “Netflix statistics 2023: Subscriber amount, time watched, and platform growth,” 2023. <https://www.independent.co.uk/advisor/vpn/netflix-statistics>.

Predicting Concept Drift with Explainability Methods in Machine Learning Models - A Literature Review

Gabriel Gomes Ziegler

`gabriel.gomesziegler@aalto.fi`

Tutor: Dr.techn. Alex Jung

Abstract

KEYWORDS: Explainable Machine Learning, Concept Drifts, Model Monitoring, Interpretable Drifts

1 Introduction

After the recent advances in Machine Learning (ML) and particularly Deep Learning (DL) with deep neural networks that often outperform humans with a black-box approach, the need for explainability and interpretability of ML models has become a hot topic in the industry and academia. Explainability in machine learning models holds significant importance, but its necessity varies across different application domains. For instance, sectors such as healthcare, finance, and autonomous driving demand a higher degree of explainability compared to others due to accountability concerns. Additionally, with the usage of ML models in production, another relevant area of study grew in importance: MLOps (Machine Learning Operations) — where the ML lifecycle is managed, from development to deployment and monitoring.

In this context, the usage of interpretable and explainable features gained traction also for the monitoring of ML models in production, where

the prediction of concept drifts and model drifts is a crucial task in order to guarantee the expected performance of a model. Concept drifts are changes in the data distribution that may affect the performance of the ML model, while model drifts are changes in the model itself that may affect its performance. It is important to detect these drifts as soon as possible, so that the ML model can be retrained or updated to avoid performance degradation. Detecting concept drifts and model drifts is a challenging task, and there are several approaches to tackle it, such as the usage of statistical methods, ensemble methods, and explainability methods as we will look in more depth in this study.

This paper reviews the literature on the topic of explainability and interpretability of ML models, with a focus on the detection of concept drifts and model drifts. The paper is organized as follows: Section Context and Foundations, Section 3 Research Methodology, Section 4 Interpretable Drifts — how interpretability and explainability are used to detect drifts and Section 5 Conclusion.

2 Context and Foundations

Many recent studies have been conducted on the topic of explainability and interpretability of ML models, and several approaches have been proposed to tackle the problem of detecting concept drifts and model drifts with methods such as LIME (Local Interpretable Model-Agnostic Explanations), tree ensembles, feature importance and SHAP (SHapley Additive exPlanations) values being the most recurring in the literature [1, 2, 3, 4, 5, 6, 7, 8, 9].

In order to understand the context of this study, we will first look at the definitions of concept drift, model drift, and covariate shift.

- **Model Drift:** Degradation of the model performance. It could be caused by concept drift or data drift.
- **Data Drift:** Represents any change in the statistical properties of the data that a model is processing over time. Data drift can encompass a variety of changes, including but not limited to changes in input features (covariates), changes in the distribution of the target variable, or changes in the relationship between the input features and the target variable.

- **Concept Drift:** When $P(Y|X)$ changes but $P(X)$ remains the same.
- **Label Shift:** When $P(Y)$ changes but $P(Y|X)$ remains the same.
- **Covariate Shift:** When $P(X)$ changes but $P(Y|X)$ remains the same. Covariate shift is a special case of concept drift [10].

2.1 State of the art for explainability and interpretability of ML models

Explainability and interpretability — often used interchangeably — are important concepts in the context of ML models that often lack a clear definition, however, they can be understood as “the ability to explain or to present in understandable terms to a human” [11]. In the context of ML models, explainability and interpretability are used to understand the inner workings of the model, and to understand the reasons behind the predictions of the model. This is useful for several reasons, such as debugging the model, detecting biases, improving the model, increasing trust in the model, and complying with regulations. This topic became even more relevant with the recent advances in ML and particularly DL with deep neural networks that often outperform humans with a black-box approach. The surge of DL models in production has led to the development of several methods for explainability and interpretability of complex black-box models such as SHAP values, LIME, and attention mechanisms [12, 13, 14]. Different from traditional explainability methods such as feature importance, these methods are model-agnostic and can be used to analyse any ML model, including DL models.

2.2 State of the art for concept drift detection

In the dynamic realm of machine learning, the challenge of concept drift is a well-known problem. Strategies for identifying and adapting to concept drift have generally fallen into two categories: altering the internal workings of a classifier or employing ensemble classifiers [15, 16]. While some classifiers such as Diversity for Dealing with Drifts (DDD) and Recurring Concept Drifts (RCD) come equipped with built-in drift detection techniques, these approaches can also function externally, coupled with different classifiers.

A frequent technique involves the use of a "base learner" to monitor shifts in model accuracy as an indicator of concept drift. Several detection methodologies, such as DDM [17], EDDM [18], and Page-Hinkley Test, have gained recognition for their solid statistical foundations. Other methods such as ADWIN [19], Paired Learners [20], EWMA [21], and STEPD [22] bring different dimensions to drift detection.

2.3 The need for explainability and interpretability in drift detection

Concept drift detection is instrumental in maintaining the performance of machine learning models in dynamic settings. However, the mere identification of a drift event is often insufficient for comprehensively addressing the issue. It is crucial to understand why a drift has occurred to tailor the appropriate model adjustments effectively. This is where the need for explainability and interpretability in drift detection comes into play.

Explainability can offer insights into the internal workings of a detection method, revealing how it reached a particular conclusion about the presence of drift. This can be invaluable for practitioners, enabling them to make data-driven decisions rather than relying on black-box mechanisms [11].

Interpretability serves to present these insights in an intuitive manner, allowing both technical and non-technical stakeholders to grasp the significance of a detected drift [23]. This could manifest in various forms, such as visualizations or easily digestible metrics, and is vital for securing stakeholder trust and enabling quick, informed actions.

Incorporating these elements into drift detection methods aligns with the broader shift towards more transparent and accountable machine learning systems. The ability to explain and interpret drift events not only bolsters the robustness of a model but also enriches the ethical standards of automated decision-making processes.

3 Research Methodology

3.1 Objectives

This study aims to delve into the intersection of explainability methods and concept drift detection within the context of machine learning mod-

els. The objective is to explore how explainability and interpretability tools can enhance the efficacy and reliability of drift detection techniques. Understanding how these two aspects synergize can provide a holistic approach to maintaining model robustness and adaptability in dynamic environments.

3.2 Research Questions

This research revolves around the following core questions:

- What are the state-of-the-art methods for explainability and interpretability in machine learning models?
- What current techniques are prevalent for detecting concept drift in machine learning models?
- How can methods of explainability and interpretability be integrated with concept drift detection techniques to create a more robust, understandable, and adaptive model?

3.3 Search Strategy

For conducting this literature review, a targeted search string was utilized in Google Scholar to fetch relevant publications. The search string was:
("concept drift" OR "data drift" OR
"model decay" OR "distribution shift") AND
("feature importance" OR "LIME" OR
"SHAP" OR "interpretability") OR
"interpretable concept drift"

This query was designed to capture a wide range of articles, journals, and papers that discuss concept drift and its detectability, as well as explainability and interpretability in machine learning. By combining keywords related to both subject areas, the search aimed to find resources that specifically discuss the intersection of these topics.

4 Interpretable Drifts

In recent years, the study of concept drift in machine learning has received considerable attention, primarily due to the impacts that the deterioration of an ML model in production can have. Alongside this, there has been a growing trend towards incorporating explainability methods into concept drift detection and understanding.

The traditional focus has primarily been on early detection of concept drift to allow for model adaptation [6]. However, there is also an importance of not just identifying but also explaining the nature and cause of the drift. For instance, using decision trees — a highly interpretable model — to elucidate the principal causes of drift and its severity has been noted as a significant advancement in this regard [6].

Explainability is also becoming a key component in specialized domains such as healthcare. For example, ensemble learning techniques have been applied to detect concept drift in online drug recommendation systems. These systems not only handle the drift but also use interpretable concept drift detection mechanisms to ensure high accuracy and reliability [2]. Similarly, the use of SHAP values in manufacturing processes has shown that explainability can produce economic benefits by improving error prediction [9].

Another method being used to increase interpretability for concept drifts is the usage of visualization of concept drift, which contributes to explainability by allowing humans to better comprehend the nature of the drift. Techniques based on feature importance have been particularly useful in visualizing how models adapt to drifting data streams [3]. Explainable machine learning techniques such as SHAP have also been used to track data drift and identify emergent health risks in emergency departments [4].

In summary, the integration of explainability into concept drift research is not only a trend but increasingly seen as a necessity as companies and researchers try to mitigate concept drifts and predict them. It not only improves model robustness but also enables better decision-making by providing insights into the ‘why’ behind the drift. This aligns well with the broader push for more interpretable and accountable machine learning models, especially in critical applications where understanding the reason for change is as important as detecting the change itself.

5 Conclusion

This literature review has explored the growing intersection of explainability and concept drift detection in ML models. With the rising importance of ML in various domains such as healthcare, finance, and autonomous driving, the need for transparent and understandable models has become more pressing than ever. As ML models are increasingly deployed in dynamic environments where data distributions can change, the ability to detect and understand concept drift becomes vital for maintaining model performance and trust.

From the reviewed literature, it is evident that while traditional methods have primarily focused on early detection of concept drift, there is some trend towards incorporating explainability into this area. Methods such as decision trees, feature importance, and SHAP values have been employed not only to detect but also to explain the nature and cause of drift, which can be particularly valuable in specialized domains such as healthcare and manufacturing [6, 9]

Moreover, the use of visual techniques for enhancing explainability has shown promise in helping both technical and non-technical stakeholders understand the implications of concept drift [3].

The integration of explainability methods into concept drift detection aligns with the broader movement towards more interpretable and accountable ML models. It not only adds a layer of robustness but also contributes to the ethical standards of automated decision-making processes, especially in critical applications.

In summary, the marriage of explainability and concept drift detection offers a holistic approach to maintaining model robustness in dynamic environments. As the field continues to evolve, further research is needed to refine these methods and understand how they can be effectively implemented across various domains and applications.

References

- [1] F. Hinder, V. Vaquet, J. Brinkrolf, and B. Hammer, “Model based explanations of concept drift,” *Neurocomputing*, vol. 555, 3 2023. doi: 10.1016/j.neucom.2023.126640. [Online]. Available: <https://arxiv.org/abs/2303.09331v1>
- [2] Y. Peng, Q. Qiu, D. Zhang, T. Yang, and H. Zhang, “Ensemble learning for interpretable concept drift and its application to drug recommendation,” *International Journal of Computers, Communications and Control*, vol. 18, 1 2023. doi: 10.15837/ijccc.2023.1.5011
- [3] M. Sarnovský, “Concept drift visualization using feature importance on the streaming data,” 2022. doi: 10.1109/SAMI54271.2022.9780841 pp. 449–454.
- [4] C. Duckworth, F. P. Chmiel, D. K. Burns, Z. D. Zlatev, N. M. White, T. W. Daniels, M. Kiuber, and M. J. Boniface, “Using explainable machine learning to characterise data drift and detect emergent health risks for emergency department admissions during covid-19,” *Scientific Reports*, vol. 11, 12 2021. doi: 10.1038/s41598-021-02481-y
- [5] A. Nesvijevskaia, S. Ouillade, P. Guilmin, and J. D. Zucker, “The accuracy versus interpretability trade-off in fraud detection model,” *Data and Policy*, vol. 3, 7 2021. doi: 10.1017/dap.2021.3
- [6] Thuener, L. Hélio, B. A. L. M. J. Guilherme, and Silva, “Interpretable concept drift,” J. Paulo, G. H. M. T. J. M. R. S., and Papa, Eds. Springer International Publishing, 2021. ISBN 978-3-030-93420-0 pp. 271–280.
- [7] A. P. Cassidy and F. A. Deviney, “Calculating feature importance in data streams with concept drift using online random forest,” 2014. doi: 10.1109/BigData.2014.7004352 pp. 23–28.
- [8] G. Stiglic and P. Kokol, “Interpretability of sudden concept drift in medical informatics domain,” 2011. doi: 10.1109/ICDMW.2011.104 pp. 609–613.
- [9] C. Seiffer, H. Ziekow, U. Schreier, and A. Gerling, *Detection of Concept Drift in Manufacturing Data with SHAP Values to Improve Error Prediction*. ISBN 9781612088914
- [10] C. Huyen and an O’Reilly Media Company. Safari, *Designing Machine Learning Systems*. ISBN 9781098107963
- [11] F. Doshi-Velez and B. Kim, “Towards a rigorous science of interpretable machine learning,” 2 2017. [Online]. Available: <http://arxiv.org/abs/1702.08608>
- [12] S. Lundberg and S.-I. Lee, “A unified approach to interpreting model predictions,” 5 2017. [Online]. Available: <http://arxiv.org/abs/1705.07874>
- [13] M. T. Ribeiro, S. Singh, and C. Guestrin, ““why should i trust you?” explaining the predictions of any classifier,” vol. 13-17-August-2016. Association for Computing Machinery, 8 2016. doi: 10.1145/2939672.2939778. ISBN 9781450342322 pp. 1135–1144.

- [14] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. u. Kaiser, and I. Polosukhin, "Attention is all you need," in *Advances in Neural Information Processing Systems*, I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, Eds., vol. 30. Curran Associates, Inc., 2017. [Online]. Available: https://proceedings.neurips.cc/paper_files/paper/2017/file/3f5ee243547dee91fbd053c1c4a844/Paper.pdf
- [15] L. L. Minku and X. Yao, "Ddd: A new ensemble approach for dealing with concept drift," *IEEE Transactions on Knowledge and Data Engineering*, vol. 24, pp. 619–633, 2012. doi: 10.1109/TKDE.2011.58
- [16] P. M. Gonçalves and R. S. Barros, "Rcd: A recurring concept drift framework," *Pattern Recognition Letters*, vol. 34, pp. 1018–1025, 7 2013. doi: 10.1016/J.PATREC.2013.02.005
- [17] J. Gama, P. Medas, G. Castillo, and P. Rodrigues, "Learning with drift detection," *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, vol. 3171, pp. 286–295, 2004. doi: 10.1007/978-3-540-28645-5_29/COVER. [Online]. Available: https://link.springer.com/chapter/10.1007/978-3-540-28645-5_29
- [18] M. Baena-García, J. D. Campo-Avila, R. Fidalgo, A. Bifet, R. Gavaldà, and R. Morales-Bueno, "Early drift detection method."
- [19] A. Bifet and R. Gavaldà, "Learning from time-changing data with adaptive windowing," vol. 7, 04 2007. doi: 10.1137/1.9781611972771.42
- [20] S. H. Bach and M. A. Maloof, "Paired learners for concept drift," *Proceedings - IEEE International Conference on Data Mining, ICDM*, pp. 23–32, 2008. doi: 10.1109/ICDM.2008.119
- [21] G. J. Ross, N. M. Adams, D. K. Tasoulis, and D. J. Hand, "Exponentially weighted moving average charts for detecting concept drift," *Pattern Recognition Letters*, vol. 33, pp. 191–198, 1 2012. doi: 10.1016/j.patrec.2011.08.019. [Online]. Available: <https://ui.adsabs.harvard.edu/abs/2012PaReL..33..191R/abstract>
- [22] K. Nishida and K. Yamauchi, "Detecting concept drift using statistical testing," *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, vol. 4755 LNAI, pp. 264–269, 2007. doi: 10.1007/978-3-540-75488-6_27/COVER. [Online]. Available: https://link.springer.com/chapter/10.1007/978-3-540-75488-6_27
- [23] D. V. Carvalho, E. M. Pereira, and J. S. Cardoso, "Machine learning interpretability: A survey on methods and metrics," 8 2019.

Certification of Machine Learning Robustness against Evasion Attacks

Harry Pham

harry.pham@aalto.fi

Tutor: Mikko Kiviharju

Abstract

Nowadays, machine learning (ML) techniques have been widely adopted for various applications including safety-critical ones such as autonomous driving and security cameras. However, any ML models are vulnerable to evasion attacks to some extent, which can result in fatal consequences during deployment. Evasion attacks involve using adversarial inputs to make an ML model give wrong predictions, so there is an urgent need to keep ML models robust against such attacks. One of the defence approaches is to certify these models' robustness against adversarial outputs. This paper discusses evasion attacks, reviews the state-of-the-art robustness certification approaches and how users may apply these approaches to their ML models, and analyses the challenges that researchers need to tackle to provide better certification frameworks.

KEYWORDS: *machine learning, neural networks, robustness, certification, adversarial attacks, evasion attacks*

1 Introduction

In recent years, Machine Learning (ML), or generally Artificial Intelligence (AI), has been widely used in various applications such as image classification [1], medical imaging [2], malware detection [3], and chatbots[4]. Due to the increasing adoption of ML in different areas, the security and reliability of these systems have gained higher attention, especially when there have been new threat models and attack vectors against ML technology [5]. These threats can be loosely categorised into the following types: evasion attacks (synthesising inputs to distort the model's output, poisoning and backdoor attacks (implanting weaknesses into the model), and extraction attacks (retrieving data from the model or information about the model).

The risk of adversarial attacks significantly increases when ML is deployed for safety-critical applications [6] such as manufacturing robots and self-driving cars. Furthermore, different ML architectures are exposed to different risk levels. For example, the more a model relies on external resources, the higher risk it faces [5]. Additionally, adversarial attacks can be performed on a model in both training and inference modes. Therefore, it is important to defend ML systems, especially to certify their robustness against adversarial attacks.

This paper will discuss the robustness of ML systems against evasion attacks and review the recent approaches to certifying their robustness against such attacks.

This paper is organised as follows. Section 2 describes the evasion attacks and the defending approaches against them. Section 3 reviews the state-of-the-art certification methods to guarantee the robustness of ML systems against evasion attacks. Section 4 discusses robustness certification practices. Section 5 analyses the feasibility of robustness certification and its practical limitations. Finally, Section 6 provides concluding remarks.

2 Evasion Attacks

2.1 Definition

Evasion attacks against machine learning models refer to manipulations of input data carefully crafted to fool the model. Adversaries exploit vul-

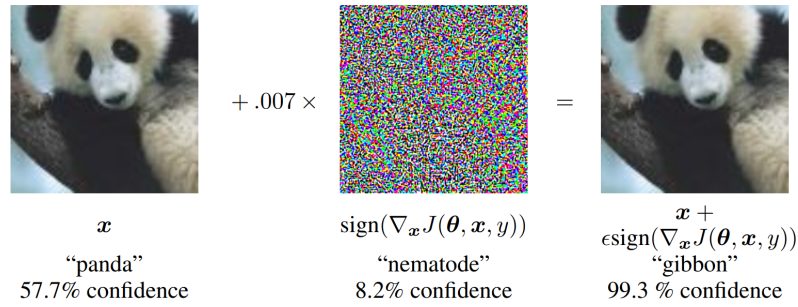


Figure 1. An illustration of how a perturbation vector (centre) added to an image of a panda (left) can create an adversarial input (right) which looks almost identical to the original image to human’s eyes but results in a different prediction. [7]

nerabilities in the model’s decision boundaries by adding subtle perturbations into input data, leading to misclassifications [8, 9, 10]. Figure 1 shows a classic example of evasion attacks against an image classifier [7]. Due to this nature, evasion attacks occur at the inference phase, where adversaries present carefully crafted inputs that exploit the model’s weaknesses to produce inaccurate results [11].

2.2 Impact

The impact of evasion attacks can be severe, leading to potential security and privacy breaches, particularly in critical applications such as autonomous vehicles, healthcare diagnostics, and financial fraud detection. Adversaries can cause misclassification of sensitive data, leading to financial losses, privacy breaches, or even physical harm in safety-critical systems [11]. Moreover, the presence of evasion attacks can erode user trust in machine learning systems, potentially hindering the widespread adoption of these technologies in various domains [12].

2.3 Adversarial Robustness

Thus, there is a need to keep the ML models robust against adversarial examples, meaning a model must be able to maintain its performance in the presence of perturbed inputs crafted by an evasion adversary [5, 13]. In other words, for all inputs x and their ground-truth labels y , a model f is adversarial robust if, for any adversarial input $x_{adv} = x + \epsilon$ in which ϵ is the perturbation, the model gives the correct classifications $f(x) = f(x_{adv}) = y$.

2.4 Defence Approaches

There are a number of general defence approaches that can be used to protect machine learning models from evasion attacks:

- **Adversarial Training:** This involves training the model on a dataset that contains adversarial examples [14], which helps the model learn to identify and resist adversarial examples.
- **Attack Detection:** This is applied during the inference phase to reveal malicious inputs and protect the overall system against malfunctions [15].
- **Certification:** This involves using mathematical techniques to prove that the model is robust to adversarial attacks [5]. This can be done for specific types of adversarial attacks, or for a more general class of adversarial attacks.

This paper focuses on the certification approach which is discussed in Section 3 to protect ML models from evasion attacks.

3 Robustness Certification

Robustness certification is essential for ensuring the safety and security of Neural Networks (NN) by evaluating their resistance to adversarial inputs. This process involves assessing a model's resilience to specific input perturbations, thereby maintaining the original predictions for unperturbed inputs[5]. Robustness certification uses algorithms to estimate robustness bounds and metrics [6], while verification entails formally detecting precise bounds [5]. Its implementation during the AI life cycle helps in early vulnerability identification and adjustments, improving the model's resilience against potential adversarial attacks.

There are multiple certification approaches have been proposed, and they can be categorised according to a taxonomy proposed in [13] as illustrated in Figure 2: *complete*, *deterministic incomplete*, and *probabilistic incomplete* certification methods. Since probabilistic methods only yield incomplete verification, this paper simply refers to methods in the *deterministic*

incomplete category as *incomplete* and those in the *probabilistic incomplete* category as *probabilistic*.

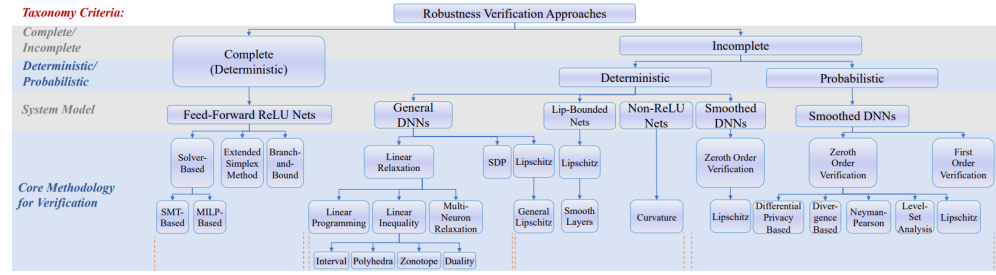


Figure 2. Taxonomy of robustness certification approaches proposed in [13]. They are divided into *complete* and *incomplete* approaches in the first level and into *deterministic* and *probabilistic* approaches in the second level. The *System Model* is the ML model type for which a certification approach is applicable.

3.1 Complete Approaches

To achieve the most accurate robustness assessment, the preferred solution is to use methods from the complete verification category. They are particularly favourable for ensuring that crafting adversarial examples within a defined perturbation space is not feasible [13], and the goal is to establish precise bounds for a model’s robustness, offering a guarantee against potential vulnerabilities [5]. There are two primary techniques for complete verification. The first relies on solvers for mathematical problems, such as Satisfiability Modulo Theories (SMT) [16] and Mixed-Integer Linear Programming (MILP) [17]. Since many NN are based on affine transformations and ReLU activation functions, expressed through linear inequalities, logical solvers can determine their satisfiability, providing assertions about the model’s robustness. The second technique involves Branch-and-Bound (BaB) algorithms [18, 19] which leverage the piecewise-linear property of NN with ReLU activation [5]. They iteratively traverse the model, applying branching and bounding steps. Branching involves splitting a neuron’s ReLU activation function into two linear constraints while bounding performs an incomplete verification, yielding lower and upper bounds for the deviation between original and manipulated predictions [20]. If the lower bound is sufficiently high, indicating a minor difference between original and manipulated predictions, the model is verified. On the contrary, if the upper bound is too low, implying a major difference between the two predictions, the model is not verified. In ambiguous cases, the branching step continues, recursively splitting neurons and applying bounding until each branch leads to

a verification decision or is entirely transformed into linear constraints, which can then be solved using linear programming [13, 5]. Despite delivering complete robustness verification, these techniques are computationally costly and thus only suitable for less complex models [21, 5, 13].

3.2 Incomplete Approaches

Incomplete approaches have emerged as a response to the scalability challenges posed by complete approaches. The fundamental hurdle in ensuring the robustness of NN lies in their inherent non-linearity. Incomplete methods tackle this issue by overapproximating (or relaxing) non-linear layers and deriving bounds for their robustness [6]. However, a notable drawback is that they cannot guarantee that a non-robust result from the certification process implies the existence of an adversarial example [13]. Compared to complete verification methods, incomplete methods offer less stringent bounds. They fall into distinct categories, such as linear relaxations, semidefinite programming (SDP), and Lipschitz-based certification. Linear relaxations, exemplified by techniques like Interval Bound Propagation (IBP) [22] and Polytope-based Overapproximation [23, 24], aim to approximate non-linear layers, relying heavily on piecewise-linear activation functions like ReLU. SDP approaches [25, 26] encode piecewise-linear activations as quadratic constraints, transforming the robustness bounding problem into a convex one. Lipschitz-based certification approaches [27, 28] focus on calculating a global Lipschitz constant to bind model robustness. Some methods enhance the tightness of this constant by combining it with other incomplete techniques, such as IBP [27], or by incorporating local Lipschitz bounds [29]. There are also hybrid approaches merging concepts from both complete and incomplete methods and seeking a balance between tightness and computational efficiency for a more scalable and accurate approximation of robustness bounds [5].

3.3 Probabilistic Approaches

Probabilistic certification algorithms, also known as randomised smoothing methods [30, 31, 32], are the alternative to complete verification methods in Section 3.1. These methods involve adding random noise into the input data [30], creating a smoothed version of the original model as demonstrated in Figure 3 [31]. The smoothed model's decisions are then analysed to determine the probability that they align with the original

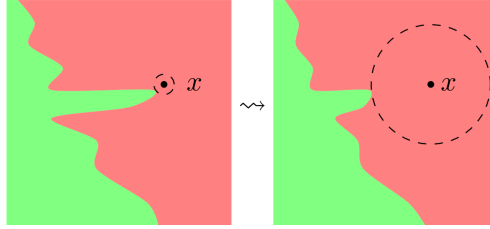
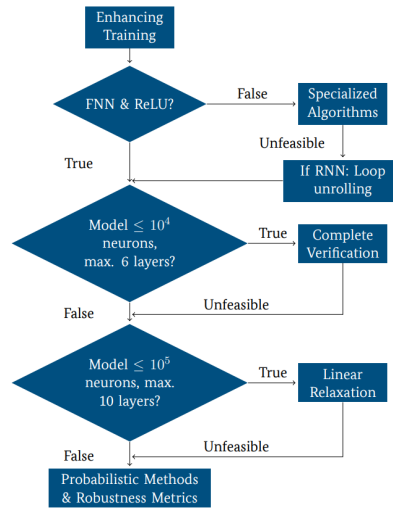


Figure 3. Randomised smoothing eliminates pointed areas of a classifier’s decision boundary (left) and gives extra robust radius surrounding inputs (right), making adversarial inputs within such radius (dotted area) fail to alter the model’s predictions [6].

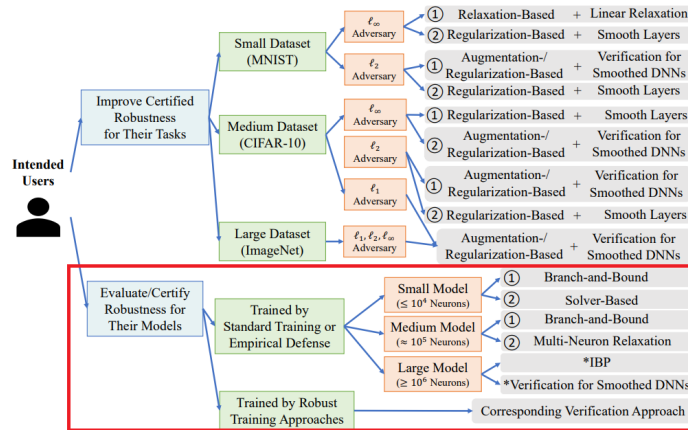
model’s predictions for original inputs [32]. By estimating this probability, probabilistic certification methods provide robustness guarantees for the smoothed model [6]. The choice of noise distribution used in the smoothing process significantly impacts the model’s performance and the tightness of the robustness bounds obtained [5]. A notable advantage of probabilistic certification methods is that they are independent of the model’s architecture, activation function, and size [21], which makes them not only applicable to a wide range of NN configurations but also scalable to certify large and complex NN such as ImageNet [30, 6]. However, it is important to note that these methods only provide probabilistic certification results which are valid for the smoothed models.

4 Certifying Robustness in Practice

With multiple certification approaches mentioned in Section 3, ML developers need to determine the suitable approaches to apply to their ML models. Based on the size of training parameters and architecture of the models, Germany’s Federal Office for Information Security (BSI) [5] and Li et al. [13] provide guidelines for practising robustness certification demonstrated in Figure 4a and Figure 4b respectively. In both guidelines, small feed-forward ReLU NN models (i.e. having 10^4 neurons or fewer) can be certified with complete verification methods, especially BaB algorithms being the most capable technique [13]. For medium NN models (i.e. having around 10^5 neurons), Li et al.’s guideline [13] claims that BaB algorithms are possible as the most suitable technique, while the BSI’s guideline [5] suggests using Linear Relaxation which gives looser verification results. Li et al. also mentioned Multi-Neuron Relaxation, an incomplete verification approach, as a runner-up suitable approach for medium models [13], suggesting that the computational cost of complete



(a) Best practices of certifying robustness proposed in [5]



(b) Suggested certification approaches for different model sizes (red box) [13]

Figure 4. Suggested guidelines of applying robustness certification.

verification approaches makes them unfeasible for some medium models. When it comes to large models (i.e. having more than 10^6 neurons), complete verification methods are pushed to their limits such that none can finish in practical time, so ML developers must use incomplete and probabilistic methods such as IBP and Randomised Smoothing. As advanced and high-performance NN models are deeper and larger, there should be more efforts to make deterministic and complete verification methods more scalable to larger models.

Additionally, both papers highlighted that enhancing training (or specifically robust training) should be used if applicable because it makes a model achieve a certain level of robustness after the training procedure and improves the efficiency of some certification approaches [5, 13]. Both guidelines cover primarily feed-forward ReLU NN models, but there is a wide variety of model architectures which also need certifying. The BSI's guideline mentions Specialised Algorithms which are for other architec-

tures such as convolutional NN (CNN), but we expect more general certification guidelines in the future covering various model architectures. This and other limitations are analysed further in Section 5.

5 Limitations & Outlook

The two main challenges to robustness certification are that (1) most certification methods are applicable for feed-forward ReLU NN only and (2) deterministic methods are not scalable to large models of depth and width.

There are many model architectures have been deployed such as residual NN (RNN), transformers, and graph NN (GNN), and they are all vulnerable to adversarial examples leading to potential security issues [33]. Therefore, we need more effort into certification approaches for various ML model architectures individually and more expansive robustness certification frameworks in the future. On the bright side, there have been proposed certification methods for GNN [34], transformers [35], and CNN [36], so future studies on robustness certification practice may include these specialised approaches to form more general guidelines.

State-of-the-art NN models are getting deeper and wider in order to gain better performance, while complete certification methods are only scalable to certify small and medium models. There exists a trade-off between scalability and certification tightness as larger models make deterministic approaches too inefficient that we have to rely on probabilistic approaches giving looser verification. Users wishing to certify their large models need to identify certain goals when using probabilistic methods such as randomised smoothing and be aware of the levels of robustness that their models may reach when putting them into deployment. In the meantime, future research needs to improve the efficiency of deterministic approaches. For example, LiResNet [37] is a newly proposed model architecture allowing efficient calculations of Lipschitz bounds and thus enabling scalable deterministic robustness guarantee. Therefore, it is possible to fill up this research gap.

Beyond evasion attacks, ML models' robustness also faces other attack types including backdoor and poisoning attacks. Such attacks happen during the training phase when malicious inputs are added to the training set which can degrade the model's performance or trigger a malicious behaviour during the inference phase. Their impact is almost as seri-

ous as evasion attacks, so there should be certification methods to defend ML models against them. Subsequently, future robustness certification practice guidelines should also consider safeguarding ML models against backdoor and poisoning attacks.

6 Conclusion

This paper underscores the critical importance of ensuring the robustness of machine learning (ML) models, particularly in the face of evasion attacks that exploit vulnerabilities in decision boundaries. As ML continues to permeate safety-critical applications like autonomous driving and healthcare diagnostics, the potential consequences of adversarial attacks become increasingly severe, ranging from financial losses to physical harm. The examination of evasion attacks, along with a comprehensive review of state-of-the-art robustness certification approaches, elucidates the multifaceted challenges in safeguarding ML models.

The paper describes certification approaches as a defence strategy and categorises them as complete, incomplete, and probabilistic. While complete methods offer precise bounds, their computational cost limits their applicability to smaller models. Incomplete methods and probabilistic approaches address scalability concerns but at the expense of stricter bounds. The practical guidelines presented for certification based on model size provide valuable insights for ML developers.

However, challenges persist, particularly in extending certification methods to diverse ML architectures beyond feed-forward ReLU networks. The trade-off between scalability and certification tightness, especially for larger models, necessitates ongoing research efforts to enhance the efficiency of deterministic approaches. Furthermore, future robustness certification practices should evolve to encompass various attack types, including backdoor and poisoning attacks, ensuring a comprehensive defence against adversarial threats. As ML continues to advance, the development of more inclusive certification frameworks and guidelines becomes imperative for fortifying the resilience of these systems in real-world applications.

References

- [1] J. Yu, Z. Wang, V. Vasudevan, L. Yeung, M. Seyedhosseini, and Y. Wu, “Coca: Contrastive captioners are image-text foundation models,” 2022. [Online]. Available: <https://doi.org/10.48550/arXiv.2205.01917>
- [2] R.-G. Dumitru and D. Peteleaza, “Using duck-net for polyp image segmentation,” *Scientific Reports*, vol. 13, 06 2023. doi: 10.1038/s41598-023-36940-5. [Online]. Available: <https://doi.org/10.1038/s41598-023-36940-5>
- [3] M. H. Faruk, H. Shahriar, M. Valero, F. Barsha, S. Sobhan, M. Khan, M. Whitman, A. Cuzzocrea, D. Lo, A. Rahman, and F. Wu, “Malware detection and prevention using artificial intelligence techniques,” in *2021 IEEE International Conference on Big Data (Big Data)*. Los Alamitos, CA, USA: IEEE Computer Society, dec 2021. doi: 10.1109/BigData52589.2021.9671434 pp. 5369–5377. [Online]. Available: <https://doi.ieeecomputersociety.org/10.1109/BigData52589.2021.9671434>
- [4] T. B. Brown, B. Mann, N. Ryder, M. Subbiah, J. Kaplan, P. Dhariwal, A. Neelakantan, P. Shyam, G. Sastry, A. Askell, S. Agarwal, A. Herbert-Voss, G. Krueger, T. Henighan, R. Child, A. Ramesh, D. M. Ziegler, J. Wu, C. Winter, C. Hesse, M. Chen, E. Sigler, M. Litwin, S. Gray, B. Chess, J. Clark, C. Berner, S. McCandlish, A. Radford, I. Sutskever, and D. Amodei, “Language models are few-shot learners,” *CoRR*, vol. abs/2005.14165, 2020. [Online]. Available: <https://arxiv.org/abs/2005.14165>
- [5] L. Adilova, K. Böttinger, V. Danos, S. Jacob, F. Langer, T. Markert, M. Poretschkin, J. Rosenzweig, J.-P. Schulze, and P. Sperl, “Security of ai-systems: Fundamentals,” 2022. [Online]. Available: <https://publica.fraunhofer.de/handle/publica/443025>
- [6] B. G. Anderson, T. Gautam, and S. Sojoudi, “An overview and prospective outlook on robust training and certification of machine learning models,” 2022. [Online]. Available: <https://doi.org/10.48550/arXiv.2208.07464>
- [7] I. J. Goodfellow, J. Shlens, and C. Szegedy, “Explaining and harnessing adversarial examples,” 2015. [Online]. Available: <https://doi.org/10.48550/arXiv.1412.6572>
- [8] A. Athalye, N. Carlini, and D. Wagner, “Obfuscated gradients give a false sense of security: Circumventing defenses to adversarial examples,” 2018. [Online]. Available: <https://doi.org/10.48550/arXiv.1802.00420>
- [9] J. Yi, Y. Chen, J. Li, S. Sett, and T. W. Yan, “Predictive model performance: Offline and online evaluations,” in *Proceedings of the 19th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, ser. KDD ’13. New York, NY, USA: Association for Computing Machinery, 2013. doi: 10.1145/2487575.2488215. ISBN 9781450321747 p. 1294–1302. [Online]. Available: <https://doi.org/10.1145/2487575.2488215>
- [10] W. Brendel, J. Rauber, and M. Bethge, “Decision-based adversarial attacks: Reliable attacks against black-box machine learning models,” 2018. [Online]. Available: <https://doi.org/10.48550/arXiv.1712.04248>
- [11] H. Xu, Y. Ma, H. Liu, D. Deb, H. Liu, J. Tang, and A. K. Jain, “Adversarial attacks and defenses in images, graphs and text: A review,” *Int. J. Autom.*

Comput., vol. 17, no. 2, pp. 151–178, 2020. doi: 10.1007/s11633-019-1211-x. [Online]. Available: <https://doi.org/10.1007/s11633-019-1211-x>

- [12] A. Ilyas, S. Santurkar, D. Tsipras, L. Engstrom, B. Tran, and A. Madry, “Adversarial examples are not bugs, they are features,” 2019. [Online]. Available: <https://doi.org/10.48550/arXiv.1905.02175>
- [13] L. Li, T. Xie, and B. Li, “Sok: Certified robustness for deep neural networks,” 2023. [Online]. Available: <https://doi.org/10.48550/arXiv.2009.04131>
- [14] E. Wong, L. Rice, and J. Z. Kolter, “Fast is better than free: Revisiting adversarial training,” 2020. [Online]. Available: <https://doi.org/10.48550/arXiv.2001.03994>
- [15] G. Vacanti and A. V. Looveren, “Adversarial detection and correction by matching prediction distributions,” 2020. [Online]. Available: <https://doi.org/10.48550/arXiv.2002.09364>
- [16] G. Katz, C. Barrett, D. L. Dill, K. Julian, and M. J. Kochenderfer, “Reluplex: An efficient smt solver for verifying deep neural networks,” in *Computer Aided Verification*, R. Majumdar and V. Kunčak, Eds. Cham: Springer International Publishing, 2017. ISBN 978-3-319-63387-9 pp. 97–117. [Online]. Available: https://doi.org/10.1007/978-3-319-63387-9_5
- [17] V. Tjeng and R. Tedrake, “Verifying neural networks with mixed integer programming,” *CoRR*, vol. abs/1711.07356, 2017. [Online]. Available: <http://arxiv.org/abs/1711.07356>
- [18] R. Bunel, J. Lu, I. Turkaslan, P. H. Torr, P. Kohli, and M. P. Kumar, “Branch and bound for piecewise linear neural network verification,” *Journal of Machine Learning Research*, vol. 21, no. 42, pp. 1–39, 2020. [Online]. Available: <http://jmlr.org/papers/v21/19-468.html>
- [19] C. Ferrari, M. N. Mueller, N. Jovanović, and M. Vechev, “Complete verification via multi-neuron relaxation guided branch-and-bound,” in *International Conference on Learning Representations*, 2022. [Online]. Available: https://openreview.net/forum?id=l_amHf1oaK
- [20] J. Lu and M. P. Kumar, “Neural network branching for neural network verification,” *CoRR*, vol. abs/1912.01329, 2019. [Online]. Available: <http://arxiv.org/abs/1912.01329>
- [21] A. Oprea and A. Vassilev, “Adversarial machine learning: A taxonomy and terminology of attacks and mitigations (draft),” National Institute of Standards and Technology, Tech. Rep., 2023. [Online]. Available: <https://doi.org/10.6028/NIST.AI.100-2e2023.ipd>
- [22] S. Gowal, K. Dvijotham, R. Stanforth, R. Bunel, C. Qin, J. Uesato, R. Arandjelovic, T. A. Mann, and P. Kohli, “Scalable verified training for provably robust image classification,” in *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*, 2019. doi: 10.1109/ICCV.2019.00494 pp. 4841–4850. [Online]. Available: <https://doi.org/10.1109/ICCV.2019.00494>
- [23] H. Zhang, T.-W. Weng, P.-Y. Chen, C.-J. Hsieh, and L. Daniel, “Efficient neural network robustness certification with general activation functions,” in *Advances in Neural Information Processing Systems*,

- S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett, Eds., vol. 31. Curran Associates, Inc., 2018. [Online]. Available: https://proceedings.neurips.cc/paper_files/paper/2018/file/d04863f100d59b3eb688a11f95b0ae60-Paper.pdf
- [24] T.-W. Weng, H. Zhang, H. Chen, Z. Song, C.-J. Hsieh, D. Boning, I. S. Dhillon, and L. Daniel, “Towards fast computation of certified robustness for relu networks,” 2018. [Online]. Available: <https://doi.org/10.48550/arXiv.1804.09699>
- [25] A. Raghunathan, J. Steinhardt, and P. S. Liang, “Semidefinite relaxations for certifying robustness to adversarial examples,” in *Advances in Neural Information Processing Systems*, S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett, Eds., vol. 31. Curran Associates, Inc., 2018. [Online]. Available: https://proceedings.neurips.cc/paper_files/paper/2018/file/29c0605a3bab4229e46723f89cf59d83-Paper.pdf
- [26] M. Fazlyab, M. Morari, and G. J. Pappas, “Safety verification and robustness analysis of neural networks via quadratic constraints and semidefinite programming,” *IEEE Transactions on Automatic Control*, vol. 67, no. 1, pp. 1–15, 2022. doi: 10.1109/TAC.2020.3046193. [Online]. Available: <https://doi.org/10.1109/TAC.2020.3046193>
- [27] S. Lee, J. Lee, and S. Park, “Lipschitz-certifiable training with a tight outer bound,” in *Advances in Neural Information Processing Systems*, H. Larochelle, M. Ranzato, R. Hadsell, M. Balcan, and H. Lin, Eds., vol. 33. Curran Associates, Inc., 2020, pp. 16 891–16 902. [Online]. Available: https://proceedings.neurips.cc/paper_files/paper/2020/file/c46482dd5d39742f0bfd417b492d0e8e-Paper.pdf
- [28] M. Wu, M. Wicker, W. Ruan, X. Huang, and M. Kwiatkowska, “A game-based approximate verification of deep neural networks with provable guarantees,” *Theoretical Computer Science*, vol. 807, pp. 298–329, 2020. doi: <https://doi.org/10.1016/j.tcs.2019.05.046> In memory of Maurice Nivat, a founding father of Theoretical Computer Science - Part II. [Online]. Available: <https://www.sciencedirect.com/science/article/pii/S0304397519304426>
- [29] H. Zhang, P. Zhang, and C.-J. Hsieh, “Recurjac: An efficient recursive algorithm for bounding jacobian matrix of neural networks and its applications,” *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 33, no. 01, pp. 5757–5764, Jul. 2019. doi: 10.1609/aaai.v33i01.33015757. [Online]. Available: <https://ojs.aaai.org/index.php/AAAI/article/view/4522>
- [30] M. Lecuyer, V. Atlidakis, R. Geambasu, D. Hsu, and S. Jana, “Certified robustness to adversarial examples with differential privacy,” 2019. [Online]. Available: <https://doi.org/10.48550/arXiv.1802.03471>
- [31] J. Cohen, E. Rosenfeld, and Z. Kolter, “Certified adversarial robustness via randomized smoothing,” in *Proceedings of the 36th International Conference on Machine Learning*, ser. Proceedings of Machine Learning Research, K. Chaudhuri and R. Salakhutdinov, Eds., vol. 97. PMLR, 09–15 Jun 2019, pp. 1310–1320. [Online]. Available: <https://proceedings.mlr.press/v97/cohen19c.html>

- [32] K. D. Dvijotham, J. Hayes, B. Balle, Z. Kolter, C. Qin, A. Gyorgy, K. Xiao, S. Gowal, and P. Kohli, “A framework for robustness certification of smoothed classifiers using f-divergences,” in *International Conference on Learning Representations*, 2020. [Online]. Available: <https://openreview.net/forum?id=SJKrkSFPH>
- [33] R. Mangal, K. Leino, Z. Wang, K. Hu, W. Yu, C. Pasareanu, A. Datta, and M. Fredrikson, “Is certifying ℓ_p robustness still worthwhile?” 2023. [Online]. Available: <https://doi.org/10.48550/arXiv.2310.09361>
- [34] D. Zügner and S. Günnemann, “Certifiable robustness of graph convolutional networks under structure perturbations,” in *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, ser. KDD ’20. New York, NY, USA: Association for Computing Machinery, 2020. doi: 10.1145/3394486.3403217. ISBN 9781450379984 p. 1656–1665. [Online]. Available: <https://doi.org/10.1145/3394486.3403217>
- [35] G. Bonaert, D. I. Dimitrov, M. Baader, and M. Vechev, “Fast and precise certification of transformers,” in *Proceedings of the 42nd ACM SIGPLAN International Conference on Programming Language Design and Implementation*, ser. PLDI 2021. New York, NY, USA: Association for Computing Machinery, 2021. doi: 10.1145/3453483.3454056. ISBN 9781450383912 p. 466–481. [Online]. Available: <https://doi-org.libproxy.aalto.fi/10.1145/3453483.3454056>
- [36] Y. Wu and M. Zhang, “Tightening robustness verification of convolutional neural networks with fine-grained linear approximation,” *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 35, no. 13, pp. 11 674–11 681, May 2021. doi: 10.1609/aaai.v35i13.17388. [Online]. Available: <https://ojs.aaai.org/index.php/AAAI/article/view/17388>
- [37] K. Hu, A. Zou, Z. Wang, K. Leino, and M. Fredrikson, “Unlocking deterministic robustness certification on imagenet,” in *Thirty-seventh Conference on Neural Information Processing Systems*, 2023. [Online]. Available: <https://openreview.net/forum?id=SHyVaWGTO4>

Exploring the Practical Application of Distributed Artificial Intelligence: Structures and Tools

Han Huazhi

huazhi.han@aalto.fi

Tutor: Vesa Hirvisalo, Anton Debner

Abstract

This literature review critically examines the utilization of Distributed Artificial Intelligence (DAI) in practical settings, specifically concentrating on the various platforms and tools integral to real applications. Through the analysis of recent literature, this paper reveals the transformative impact of DAI and related tools in fields such as multi-robot systems, healthcare, sustainable development and big data analytics. Comparative examinations of general-purpose platforms and application-specific tools illuminate their unique capabilities and roles in facilitating DAI solutions. Moreover, this review exposes the inherent challenges of distributed training, like communication overhead and data synchronization, and explores how contemporary tools are trying to address these challenges. This review serves not only as a summation of current practices but also as a lens through which the future trajectory of DAI can be conjectured.

KEYWORDS: *distributed artificial intelligence, applications, literature review*

1 Introduction

The rapid growth in machine learning and artificial intelligence (AI) has spawned numerous subfields that address specific challenges, such as computer vision, natural language processing, generative AI and distributed artificial intelligence (DAI). DAI is primarily a concept designed to address the problem of distributing and coordinating intelligent tasks across multiple computing nodes. Today DAI has demonstrated its adaptability and innovation with decentralized decision-making and excellent scalability.

As DAI continues to evolve, related tools and platforms have emerged to support DAI and its different phases, such as distributed training. Since understanding the practical applications of DAI necessitates a thorough examination of the platforms and tools serving as its backbone, this paper focuses on those tools and divides them into general-purpose platforms like Spark [1] and Ray [2] and application-specific tools such as multiuser mobile edge computing network [3] and aerial-ground network [4] to demonstrate the advantages of each for solving problems in related areas and applications.

This article first introduces the basics of DAI and then moves forward to the platforms and tools in DAI applications. Section 2 describes what is DAI and introduces the evolution of DAI and distributed training. Section 3 shows a detailed analysis of frameworks and structures for distributed training. Section 4 discusses the work of this article and how to promote the utilization of platforms to face future challenges of DAI. Section 5 is the conclusion.

2 Background

In this section, the concepts of AI and distribution are first introduced. Following this, the focus shifts to DAI, exploring its practical applications in real-world scenarios. Finally, the focus shifts to the training phase of DAI, where its origins, taxonomy, advantages, and challenges are described.

2.1 What Is AI and Distribution

Artificial Intelligence, or AI, is a part of computer science that works on making computers that can think and act like humans and are capable of intelligent behavior. It focuses on giving machines skills like learning, solving problems, recognizing things, making decisions, and understanding language. Modern AI encompasses multiple kinds of techniques, including machine learning, deep learning, and neural networks, which evolved due to advances in computational power and data availability[5]. A notable example is the progression of AI in face recognition technologies, evolving from basic pattern recognition to complex neural network-based applications, demonstrating major advancements in image processing and analysis [6].

The concept of distribution in computing refers to the spread of computational processes across multiple systems or locations, rather than relying on a single centralized unit. This approach is key to enhancing computational efficiency, reliability, and scalability. In distributed computing, tasks are divided and executed in parallel, allowing for faster processing and redundancy because one node's failure won't cause the failure of the entire system. The rise of cloud computing and big data further highlighted the importance of distributed systems, which are essential in handling the vast amounts of data generated in today's digital world [7, 8].

The convergence of AI and distribution systems utilizes both the data-driven capabilities of AI and the efficiency of distributed systems. This synergy is essential in scenarios involving large, geographically dispersed data sets, such as in IoT applications.

2.2 What Is Distributed Artificial Intelligence

Artificial Intelligence (AI) has undergone transformative change with the advent of DAI, a subfield of AI focusing on the decentralization of knowledge processing and problem-solving. DAI leverages the distribution of computational processes across multiple systems or agents, which can be geographically dispersed, to execute tasks that would be overly complex or time-consuming for a single entity. This approach not only enhances computational efficiency and task scalability but also introduces robustness and redundancy in system design, critical for tasks demanding high reliability [9].

The practical implications of DAI have been profound across various

domains. In smart cities, people implement DAI systems to monitor traffic volume to manage traffic lights better [10]. In healthcare, for example, DAI systems are used for patient monitoring and data analysis, integrating information from diverse sources to provide comprehensive patient care [11]. In the realm of data analytics, tools like MapReduce have revolutionized big data processing by enabling the distribution of data analysis tasks across multiple computing nodes, drastically reducing the time required for processing vast datasets [12].

Moreover, the rise of sophisticated DAI frameworks such as GraphLab and TensorFlow has significantly democratized and accelerated the deployment of machine learning models [13, 14]. These platforms provide the necessary infrastructure for developing and scaling machine learning applications across distributed systems, facilitating tasks like data parallelism and model serving.

As DAI continues to evolve, it is paving the way for more advanced, scalable, and efficient AI applications to suit the pressing needs of various domains. The ongoing research and development in DAI structures and tools underscore the immense potential and versatility this technology holds for future endeavors [15, 2].

2.3 Distributed Training in DAI

One of the key components in DAI is distributed training, a paradigm that has become essential due to the sheer volume of data and the computational intensity of modern algorithms.

The genesis of distributed training is deeply intertwined with the growth of big data and the need for real-time processing and analysis. Traditional single-node training methods encountered bottlenecks, becoming increasingly inefficient with the exponential growth in data volume, variety, and velocity [12]. The limitations of computational capacity, memory, and speed on single machines necessitated the evolution toward distributed training environments. This shift was not merely a matter of convenience but of absolute necessity, driving both research and practical, scalable applications [16].

At its core, distributed training involves the division of datasets and computational tasks across multiple machines or nodes. These distributed systems collaborate, processing data in parallel, and often employ sophisticated algorithms to optimize the operation and communication between nodes [1]. This method addresses several critical issues: it significantly

reduces the time required for training machine learning models, facilitates the handling of larger datasets, and enables the use of more complex, computationally intensive models [17].

There are primarily two approaches of distributed training: data parallelism and model parallelism. Data parallelism divides the dataset across different nodes, each of which computes model updates based on its subset of data. In contrast, model-parallelism involves splitting the model itself across different nodes, necessary for models too large to be implemented into the limited memory of a single device [18].

However, distributed training is not without its challenges. The communication overhead between nodes, especially in data parallelism, can be substantial, sometimes negating the advantages gained by parallel computation. Ensuring consistency and synchronization across nodes, particularly in non-convex optimization landscapes common in deep learning, is another area of active research [19].

The current landscape of distributed training is a vibrant array of open-source frameworks and proprietary systems, each offering different trade-offs between scalability, ease of use, and flexibility. Spark [1], Ray [2], and Apache MXNet [17] are prominent examples, each with its distinct approach to distributed training. And except for those general-purpose platforms, there are also many tools and structures specified for certain applications, such as aerial-ground deployment [4] and swarm learning in cancer histopathology [20]. Later the article will dive deeper into some application-specific models and structures.

Distributed training stands as a cornerstone technique in the era of big data and complex modeling. Its continued evolution holds significant promise for breakthroughs in machine learning capabilities, potentially transforming the landscape of artificial intelligence applications across diverse sectors.

3 Tools and Structures of Distributed Training

In this section, tools and structures for distributed training are divided into general-purpose and specific ones, as shown in Figure 1. General-purpose tools are platforms that can adapt to different applications in different fields, while specific tools usually aim to solve the problem found in certain areas.

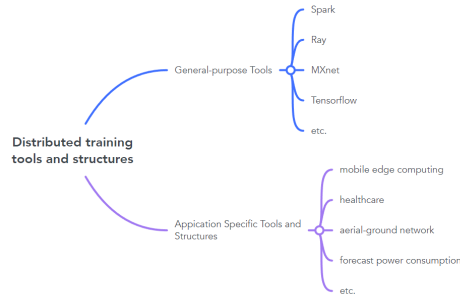


Figure 1. General taxonomy of tools and structures reviewed

3.1 General-purpose Tools

In this part, multiple popular general-purpose platforms for distributed training are reviewed. General-purpose platforms are designed to provide a versatile environment that can adapt to various computational tasks. These platforms excel in handling heterogeneous workloads, optimizing resource allocation, and ensuring scalability and fault tolerance across large-scale distributed systems. They form the backbone of many high-demand processing tasks, underpinning critical research and real-world applications in various fields.

Apache Spark is an open-source framework for large-scale data processing, capable of handling batch and real-time analytics [1]. Spark utilizes Resilient Distributed Datasets (RDDs) and Directed Acyclic Graphs (DAGs) to enable distributed data processing across a cluster of machines. With its in-memory computing feature, fault tolerance, and scalability, Spark provides an efficient and robust platform for big data analytics and machine learning in distributed environments. RDDs are a core concept in Apache Spark. They are collections of data distributed across a cluster, enabling parallel processing. They cannot be changed once created and maintain lineage information to reconstruct lost data due to node failures, ensuring fault tolerance.

Ray is another framework particularly designed to serve the needs of emerging AI applications that interact continuously with the environment and learn from these interactions, which is characteristic of Reinforcement Learning (RL) applications [2]. The key features of Ray are its bottom-up distributed scheduler and the global control store (GCS). The bottom-up distributed scheduler allows local nodes to submit to their local schedulers first and if the local nodes are overloaded, the task will be forwarded to the global scheduler, as shown in Figure 2. GCS logi-

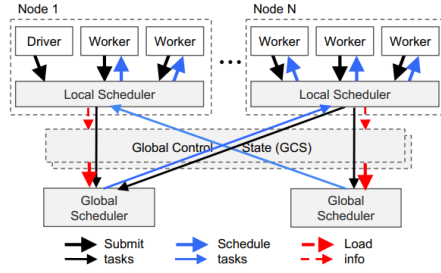


Figure 2. The structure of bottom-up scheduler of Ray [2]

ally centralizes the control state while physically distributing it across multiple nodes for scalability and fault tolerance. This design allows for informed decision-making in task scheduling and resource allocation and recovery from failures with the help of GCS coordination.

There are also other tools such as MXnet [17], `tf.distribute.Strategy` API of Tensorflow [13] and `torch.distributed` packet from Pytorch [21]. MXNet focuses on efficient parallel processing of multi-GPU data, while TensorFlow’s `tf.distribute.Strategy` API provides flexible distributed training strategies. Similarly, PyTorch’s `torch.distributed` package enhances support for massively parallel processing and communication. The diversity and complementarity of these tools provide a strong technical foundation for DAI research and practice, driving rapid growth and innovation in the field.

3.2 Specific Tools

There are also many unique structures developed for distributed training of targeted applications, such as mobile computing, healthcare, etc.

Mobile Edge Computing

Guo et al. propose a DAI approach in a multiuser mobile edge computing (MEC) network set within an unsafe environment [3]. The proposed distributed federated learning method significantly curtails system costs, notably in latency and energy efficiency. Moreover, it ensures a more substantial allocation of communication resources and computational capability to users bearing higher task priorities. The application scenario and workflow of the model are shown in Figure 3 and Figure 4. Users first train their DNN models on their own devices. After training, they upload their parameters to the central server for combining. The central node then shares the global parameters with all users, reducing the data

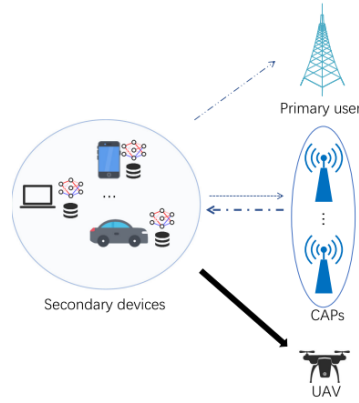


Figure 3. System model of a MEC network in cognitive eavesdropping environments, where CAP stands for computational access point [3]

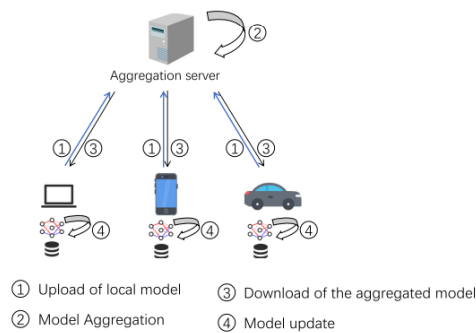


Figure 4. Workflow of the distributed training model [3]

transferred.

Healthcare

Saldanha et al. showcase the potential of Swarm Learning (SL) in dealing with challenges associated with data collection for AI in analyzing histopathology slides [20]. SL is proposed as a solution to ethical and legal challenges in collecting large datasets such as data privacy and monopolistic data governance by enabling decentralized training of AI models across different partners without the need for data transfer, as shown in Figure 5, thus averting monopolistic data governance. Utilizing SL, the paper demonstrates that AI models can predict molecular alterations directly from some kind of pathology slides of cancer. It also shows that adopting SL and separating data and models into different places creates strong incentives for participants to collaborate.

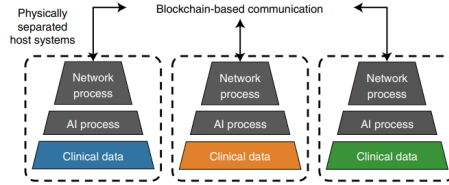


Figure 5. Model of swarm learning with data isolation [20]

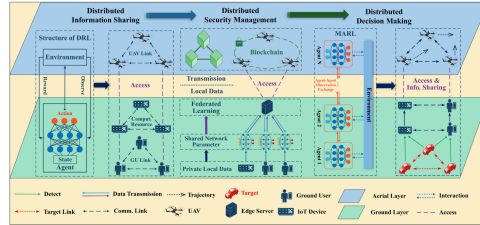


Figure 6. Overall architecture of DAIAGN [4]

Aerial-Ground Network

XIA et al. proposed a novel distributed AI-enabled aerial-ground network (DAIAGN) to fully take advantage of the distributed networks, conventional aerial-ground networks, and multi-agent reinforcement learning techniques [4]. DAIAGN, as shown in Figure 6 is a suitable solution to adapt DAI to manage multiple agents within a dynamic and time-varying network topology, which is very challenging. In centralized information-sharing systems, the same data or communication may occur several times, which wastes computing resources or communication bandwidth. In DAIAGN, a component called distributed information sharing is applied to promote data efficiency and optimize the process of decision-making. With deep reinforcement learning enabled distributed information sharing, agents only share their information with their neighbors instead of sharing it globally.

Forecast Power Consumption

Cerquitelli et al. propose an architecture known as the Scalable Predictor of Power Consumption (SPEC), which lays the foundation for the distributed framework and machine learning integration aimed at accurately forecasting power consumption [22]. SPEC is based on Apache Spark, it can apply a sliding window over the historical data stream when dealing with the data received by sensors and all of the random forests, artificial neural networks and ridge regression methods can be used with this distributed framework.

4 Discussion

The review of various tools and platforms for distributed training reveals significant advances in machine learning and artificial intelligence. Distributed training is characterized by computing tasks being dispersed across multiple nodes or even geographically distinct data centers, which shows great advantages in managing large-scale data and complex models. The general-purpose platforms and tools reviewed significantly speed up the training process by leveraging collective computing power [2], thereby reducing the time and resources required, and those specific structures make DAI more suitable for applications in certain fields. DAI is an evolving and very dynamic research area, and tools for DAI also need to evolve and be modified for different applications. However, the challenges such as lack of standardized protocols and interoperability among different distributed training platforms may impede the broad adoption and seamless integration of DAI, which needs to be addressed carefully [23].

5 Conclusion

This paper presents a review of DAI platforms and tools. General-purpose platforms like Spark and Ray leverage RDD and DAG techniques to effectively handle distributed data processing, enhancing system reliability. Application-specific tools, on the other hand, offer greater flexibility in structures, tailored to optimize performance in specific scenarios. This review underscores the diversity and the potential of these platforms and tools, illustrating how they collectively advance the field of DAI. Future research could build upon this foundation, exploring new ways to enhance interoperability between different DAI systems and further refining the efficiency of distributed data processing techniques.

References

- [1] M. Zaharia, R. S. Xin, P. Wendell, T. Das, M. Armbrust, A. Dave, X. Meng, J. Rosen, S. Venkataraman, M. J. Franklin, *et al.*, "Apache spark: a unified engine for big data processing," *Communications of the ACM*, vol. 59, no. 11, pp. 56–65, 2016. doi: 10.1145/2934664.
- [2] P. Moritz, R. Nishihara, S. Wang, A. Tumanov, R. Liaw, E. Liang, M. Elibol, Z. Yang, W. Paul, M. I. Jordan, *et al.*, "Ray: A distributed framework for emerging ai applications," in *13th USENIX symposium on operating*

- systems design and implementation (OSDI 18)*, pp. 561–577, 2018. doi: 10.48550/arXiv.1712.05889.
- [3] Y. Guo, R. Zhao, S. Lai, L. Fan, X. Lei, and G. K. Karagiannidis, “Distributed machine learning for multiuser mobile edge computing systems,” *IEEE Journal of Selected Topics in Signal Processing*, vol. 16, no. 3, pp. 460–473, 2022. doi: 10.1109/jstsp.2022.3140660.
- [4] Z. Xia, J. Du, Y. Ren, and Z. Han, “Distributed artificial intelligence enabled aerial-ground networks: Architecture, technologies and challenges,” *IEEE Access*, vol. 10, pp. 105447–105457, 2022. doi: 10.1109/access.2022.3210337.
- [5] S. J. Russell and P. Norvig, *Artificial intelligence a modern approach*. London, 2010. ISBN: 978-0-13-604259-4.
- [6] K. He, X. Zhang, S. Ren, and J. Sun, “Deep residual learning for image recognition,” in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 770–778, 2016. doi: 10.1109/cvpr.2016.90.
- [7] G. F. Coulouris, J. Dollimore, and T. Kindberg, *Distributed systems: concepts and design*. pearson education, 2005. ISBN: 978-0-13-214301-1.
- [8] A. S. Tanenbaum, *Distributed systems principles and paradigms*. 2007. ISBN: 978-0-13-088893-8.
- [9] P. Stone and M. Veloso, “Multiagent systems: A survey from a machine learning perspective,” *Autonomous Robots*, vol. 8, pp. 345–383, 2000. doi: 10.21236/ada333248.
- [10] A. Kirimtat, O. Krejcar, A. Kertesz, and M. F. Tasgetiren, “Future trends and current state of smart city concepts: A survey,” *IEEE Access*, vol. 8, pp. 86448–86467, 2020. doi: 10.1109/ISCC.2003.1214251.
- [11] M. Hassanalieragh, A. Page, T. Soyata, G. Sharma, M. Aktas, G. Mateos, B. Kantarci, and S. Andreescu, “Health monitoring and management using internet-of-things (iot) sensing with cloud-based processing: Opportunities and challenges,” in *2015 IEEE international conference on services computing*, pp. 285–292, IEEE, 2015. doi: 10.1109/scc.2015.47.
- [12] J. Dean and S. Ghemawat, “Mapreduce: simplified data processing on large clusters,” *Communications of the ACM*, vol. 51, no. 1, pp. 107–113, 2008. doi: 10.1145/1327452.1327492.
- [13] M. Abadi, P. Barham, J. Chen, Z. Chen, A. Davis, J. Dean, M. Devin, S. Ghemawat, G. Irving, M. Isard, *et al.*, “Tensorflow: a system for large-scale machine learning,” in *12th USENIX symposium on operating systems design and implementation (OSDI 16)*, pp. 265–283, 2016. doi: 10.48550/arXiv.1605.08695.
- [14] Y. Low, J. Gonzalez, A. Kyrola, D. Bickson, C. Guestrin, and J. M. Hellerstein, “Distributed graphlab: A framework for machine learning in the cloud,” *arXiv preprint arXiv:1204.6078*, 2012. doi: 10.14778/2212351.2212354.

- [15] R. Mayer and H.-A. Jacobsen, “Scalable deep learning on distributed infrastructures: Challenges, techniques, and tools,” *ACM Computing Surveys (CSUR)*, vol. 53, no. 1, pp. 1–37, 2020. doi: 10.1145/3363554.
- [16] M. Li, D. G. Andersen, J. W. Park, A. J. Smola, A. Ahmed, V. Josifovski, J. Long, E. J. Shekita, and B.-Y. Su, “Scaling distributed machine learning with the parameter server,” in *11th USENIX Symposium on operating systems design and implementation (OSDI 14)*, pp. 583–598, 2014. doi: 10.1145/2640087.2644155.
- [17] T. Chen, M. Li, Y. Li, M. Lin, N. Wang, M. Wang, T. Xiao, B. Xu, C. Zhang, and Z. Zhang, “Mxnet: A flexible and efficient machine learning library for heterogeneous distributed systems,” *arXiv preprint arXiv:1512.01274*, 2015. doi: 10.48550/arXiv.1512.01274.
- [18] N. Janbi, I. Katib, and R. Mehmood, “Distributed artificial intelligence: Taxonomy, review, framework, and reference architecture,” *Intelligent Systems with Applications*, p. 200231, 2023. doi: 10.2139/ssrn.4353525.
- [19] M. Li, D. G. Andersen, A. J. Smola, and K. Yu, “Communication efficient distributed machine learning with the parameter server,” *Advances in Neural Information Processing Systems*, vol. 27, 2014.
- [20] O. L. Saldanha, P. Quirke, N. P. West, J. A. James, M. B. Loughrey, H. I. Grabsch, M. Salto-Tellez, E. Alwers, D. Cifci, N. Ghaffari Laleh, *et al.*, “Swarm learning for decentralized artificial intelligence in cancer histopathology,” *Nature Medicine*, vol. 28, no. 6, pp. 1232–1239, 2022. doi: 10.1038/s41591-022-01768-5.
- [21] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga, *et al.*, “Pytorch: An imperative style, high-performance deep learning library,” *Advances in neural information processing systems*, vol. 32, 2019. doi: 10.48550/arXiv.1912.01703.
- [22] T. Cerquitelli, G. Malnati, and D. Apiletti, “Exploiting scalable machine-learning distributed frameworks to forecast power consumption of buildings,” *Energies*, vol. 12, no. 15, p. 2933, 2019. doi: 10.3390/en12152933.
- [23] M. Noura, M. Atiquzzaman, and M. Gaedke, “Interoperability in internet of things: Taxonomies and open challenges,” *Mobile networks and applications*, vol. 24, pp. 796–809, 2019. doi: 10.1007/s11036-018-1089-9.

Where is the Beef? Extending the Training Data

Ilmari Tarpila

ilmari.tarpila@aalto.fi

Tutor: Alexander Jung

Abstract

*This work addresses the challenge of having a robust model, such as a deep neural network with billions of tunable parameters, but limited training data. In this context, the primary question is: With an abundance of free online data sets, how to determine which ones could be useful to extend your training data with? This research delves into principled methods and explores the fundamental limits of the training data selection problem. **KEYWORDS:** Training Data, Data Selection, Data Extension, Data Quality, Clustering, Labels, Graph Learning*

1 Introduction

The modern world is thoroughly connected and filled with data, with huge machine learning models and blasting fast download speeds for them available to professionals and masses alike. But when availability ceases to be an issue, choice becomes one - what to do with all the power?

Large machine learning models can nowadays include billions of parameters, and are capable of even more specialized tasks. This drives the use of machine learning into ever smaller niches, which translates to ever smaller subsets of existing data sets being suitable for training. This

then makes it often difficult to tune these models for the specialized tasks expected from them.

A specialized tasks would require specialized data, which is a problem that can rarely be solved with computational power alone. Fortunately, the connectedness of the world gives rise to alternative ways of accruing training data for model training, in various shapes and sizes. Online, we can find all kinds of data sets for all kinds of training purposes, as well as data not perhaps directly meant as training data, but suitable for it nonetheless. Here, choice is truly the issue - how to select your necessary additional training material from this sea of data?

In this paper, we review key methods in selecting meaningful training data from large amounts of available data, as well as the fundamental limits of such methods.

The following sections will go through a few different identified methods useful when evaluating compatibility a of a candidate data set to extend training data with, the limitations and issues of these approaches in general, and finishing with a small review of implementations of these methods as machine learning models, that are able to extend their data sets.

2 Extending Training Data

This section discusses various considerations and strategies associated with expanding the data set to ensure the model's robustness and accuracy.

2.1 Problems of using data without discretion

A rudimentary approach to extending a machine learning model's data set is to include all available data, such as incorporating all the documents accessible on the internet. However, the fundamental challenge lies in the fact that a machine learning model is only as good as its data. It is built on the premise of *modeling* the behavior of the data, perhaps by understanding clustering tendencies, assigning labels to data, or by representing the data as a graph. Through training, models are conditioned to perform meaningfully well in the future when similar enough data is presented [1].

This succinctly captures the essence that feeding a machine learning

model with subpar or inaccurate data can lead to flawed or unreliable predictions [2]. Therefore, the process of extending a model's data set should prioritize not only quantity but, more importantly, the quality and relevance of the incorporated data to ensure the resulting model's robustness and accuracy. Selecting training data with good judgement helps the model specialize more effectively in addressing specific problems.

2.2 Using similar clusters of data

Clustering is a very analytical way of evaluating candidate data for use in training. When clustering data, we're looking for groups of clusters within the data, making a conclusion that the data points within a cluster are similar to each other in some. With proper selection of representation of the data, the features used in the clustering, this approach can produce clusters of data points that are similar to each other [3].

Clustering depends on the data actually having some grouping inherent to it. The essence of data clustering lies in partitioning a data set into several groups, ensuring that the similarity within each group is significantly larger than the similarities across different groups. If the data is uniformly distributed without any distinct groupings, attempts to cluster the data will likely result in failure, or at worst, lead to artificially introduced partitions that do not accurately reflect the underlying data structure [4].

There are many methods available to be used for clustering, one such method being K-means clustering. This technique partitions data into distinct groups by minimizing the dissimilarity (usually Euclidean distance) between data points and cluster centers. Each technique has its unique strengths and is best suited to different types of data and clustering requirements, and it's important to consider the specific characteristics of the data set and the desired outcome of the clustering process when selecting the method. [4].

When the inherent groupings align well with the intended use of the data, clustering can be a powerful tool for organizing and understanding complex data sets. However, when these inherent patterns are not present, or when groups overlap significantly, the efficiency and effectiveness of clustering techniques can be compromised. Overlapping groups, in particular, can present a challenge, as they may lead to ambiguities in classification and reduce the distinctiveness of the clusters formed. Understanding the nature of the data is therefore crucial when applying clus-

tering techniques. By acknowledging and addressing the inherent structures within the data, one can more effectively utilize clustering methods to reveal meaningful insights and patterns.

The principles discussed in [3] emphasize well the significance of choosing the right representation for data in clustering processes. The effectiveness of a clustering algorithm is inherently tied to how data is represented and interpreted. Expressive representations, which can capture a vast array of input configurations, are crucial. They allow for the grouping of data points not only based on superficial similarities but also in terms of deeper, more fundamental characteristics.

Clustering can be a straightforward way of finding data points or data sets with similar qualities to another, existing data set, but in their simplicity they still require a lot from the data itself and the selection of clustering method and the chosen features for it. If the chosen method clustering method does not represent actual data behavior in some way, an accepted candidate data set is not guaranteed to not be meaningfully connected or similar to the training data set.

2.3 Matching candidate training data using labels

Another intuitive way would be to use the labels of the data for matching candidate data sets with the training data set. Labels are often constructed features that represent the data in some meaningful way. Thus, it would make sense to use labels to evaluate candidate data, and add the data to the training set only if its labels correspond to the labels of the data set, or perhaps by comparing the distributions of the labels of the sets to maintain the original distribution, to represent the original problem statement of the model.

Any qualitative feature of the data could be selected as a label, and used basically as a filter field for the data set. However, often a label that is meaningful to the problem statement at hand is not available, and has thus to be constructed. Even if labels are available, the labelling logic across different data sets could be different, and thus the labels wouldn't actually end up meaning the same abstraction across different data points and sets.

Labelling in its simplest form can be simply manually going through data points, and assigning a given label to it. This could be for example one by one going through an image data set of dogs, and for each image assigning a label that corresponds to the dog breed of the dog in the

picture. This, however, is naturally time and resource intensive, even if the label quality can often be thought to be the best when doing manual labeling. Because a huge factor of engineering effort can in the end be put into labeling, automated labeling is very liked in the field [5].

Automatic labelling at its core is about building a classifier, an algorithm or a function that takes the data point as an input, and outputs a label for that data point. The construction of classifiers is a very discussed topic in machine learning, and many of the applications of machine learning themselves aim to create good classifiers for given types of data. [6]

Similar to different clustering methods, different types of classifiers are abundant, and the selection of classifier is important to base on the data and the problem at hand. [7] describes the process of selecting a classifier typically involving either relying on expert knowledge or empirically testing every possible classifier to identify the most effective one, which can be time-consuming and resource-intensive. Many studies like this one have thus been conducted to test different kinds of classifiers on different kinds of data sets to discover which classifiers are good in what type of problems. Still, understanding the underlying mechanics of the data is important to select a good classifier.

By choosing a good classifier that produces good labels for candidate data points, it's possible to identify data sets with similar features as your existing data set, without essentially having to go through every single data point manually. When choosing the labels and classifiers to use in a specific problem, it is again important to understand the underlying mechanics of the model, to produce labels and classifiers that are properly able to bring out an existing and real nature of the data for the purposes of the data selection.

2.4 Using graphs to find data connected to the original data

Graphs are constructs with nodes that are connected by edges. As a data set, a graph could be for example a web pages containing links to each other in their text - the web pages are the nodes, while the hyperlinks leading from a document to another would be the edges. Edges between nodes in a graph are often thought to imply relationships between the nodes [8], and that relationship can be used to infer similarity between the nodes. Thus, it would make sense to use these relationships as a similarity between the nodes - nodes that could represent data points or whole data sets that are candidates for extending your data set with.

Graphs can be especially intuitive to use when the data itself is already formatted or easily understandable as a graph, like in the case of the web page example. However, while the intuitive use case of graphs does apply, graphs can also be constructed when one is not immediately available using existing features - a distance function for example can generate values for each data point pair so that they could be used as edges, creating either a fully connected graph, or the values can be filtered by a given bound and only use the edges that are within the given threshold. [9]

The value in this constructed graph approach over clustering and labeling approaches can be seen in the idea that by using a graph, each data point has not just a relation to the nodes it is directly connected to with an edge, but to multiple data points through the complex relationships of multiple edges across many nodes. This complexity is often seen as a very favorable feature of modeling behavior of data in applications of graph-based methods, because it's able to more directly represent the complex relationships of real-world things. [8]

A graph of data can also be approached in a manual manner, using some root nodes or nodes that are of specific interest as a guide. Perhaps a root node could be the training data set itself, and all the other nodes connected to that data set through some edges could be evaluated by hand for their applicability. However, due to the complexity of the graph based approach, graphs are much more powerful when used through complex systems that model the relationships of the graph in an algorithmic way.

Algorithmic approaches to graph analysis, such as graph neural networks (GNNs) or other advanced machine learning techniques, can effectively leverage these complex relationships. These algorithms can traverse through the graph, assessing the connections and the overall structure, to identify nodes (data points) that are most relevant or similar to the original data set. Such techniques can discern subtle patterns and relationships that might be missed by more straightforward clustering or labeling methods.[9]

Graph-based methods offer a powerful tool for extending training data sets in machine learning. By utilizing the rich relational information embedded in graph structures and applying sophisticated algorithmic techniques, data points that are contextually and relationally relevant to the original data set can be uncovered. This approach is especially beneficial in situations where the relationships between data points are complex

and multi-dimensional, allowing for a more nuanced and comprehensive extension of the training data set.

3 Limits of Data Selection

In this section, we will discuss the inherent limitations of the approaches described so far in this paper, some of them already implied in the text. Understanding the limits of training data selection can help one to bound their data extension to meaningful amount of effort, and to also stay away from too hopeful scenarios.

3.1 Data quality

As mentioned before, it's widely thought that from bad data, only bad results will follow [2]. However, sometimes quality data is simply not available, and the only available data that is nearly topical for the problem being solved does not match the quality of the training data set or the required quality of the model.

Specifically, data quality can be broken down into several dimensions. One such dimension is the consistency of representation within the data set. For example, ensuring that a real-world entity is represented uniformly across the data set is crucial for maintaining consistency. A common issue in data sets is the presence of semantically equivalent but differently represented values, such as "Helsinki," "Helsingfors," or "HEL" in a city feature. [10]

Therefore, when selecting training data, it is not just the quantity of data that matters but also its quality across various dimensions. Understanding and addressing data quality issues are vital for ensuring that the training data contributes positively to the development of effective and reliable AI models.

3.2 Modeling

No matter the approach used in training data selection, understanding of the underlying mechanics of real-world phenomena is crucial. Each of the methods described here has a step that requires selection of features of the data to be used in modelling. If these decisions are not based on knowledge on the subject matter, no clustering method, classifier or graph-based approach can uncover true relationships between the data.

[7] [3]

Moreover, the choice of features significantly impacts a model's ability to generalize from the training data to real-world applications. Inadequate or irrelevant features can lead to models that are either overfitted to the training data or unable to capture the complexity of the phenomena they are intended to model. Therefore, a collaborative approach that combines domain expertise with advanced data analysis techniques is essential for building robust, effective models.

3.3 Scalability and Complexity

As models grow more complex, the time invested in them grows. While in this paper we don't go into ways of evaluating the results of data extension through the lens of the model performance, it must be remembered that in a real-world scenario most likely multiple different candidate data sets would be tested as an extension to the existing data set by running the training on the model. Because of this, turning the data selection process in itself into a complex machine learning problem can backfire in the way of slowing down the whole process considerably. It's thus important to consider ways to mitigate scaling problems as the data sets grow and models grow more complex. [11]

4 Machine Learning Methods for Data Extension

This section will go through some machine learning implementations of the previously detailed approaches. While many of them are not directly about extending a data set, they all do deal with evaluating data similarity and many have incorporated mechanisms of using only the parts of the supplied data that is applicable to their use case.

4.1 Self-training

In [12] contributions of semi-supervised learning are proposed, specifically in the context of self-training. The paper introduces two primary methods, both addressing the challenge of limited labeled data: active labeling and co-labeling. Active labeling focuses on manually or automatically labeling the most representative and informative data within certain clusters, thereby enriching the training data set. Co-labeling further extends this concept by employing an efficient classifier to automate

the labeling process within these clusters. These approaches not only enhance the quality of the training data set, but also significantly improve the accuracy of semi-supervised classification models.

4.2 Federated Learning

Many federated learning methods employ clustering as a means of matching data with statistically similar properties together for the use of personalized machine learning [13] and multi-task learning [14]. It's easy to see the connections between such tasks and the problem this paper discusses - while federated learning in these cases deals with partitioning existing masses of data to useful clusters, we are looking for additional data that wouldn't create more clusters, but add into the existing cluster. Another way to see this relation is to apply clustering methods to all of available public data to discover clusters useful to us, and while this is not applicable in the largest context, it will likely prove useful approach when the amount of data has been pruned to a meaningful amount.

4.3 Neural Clustering Processes

[15] proposes a method, which merges the capabilities of neural networks with clustering. This Neural Clustering Process (NCP) framework benefits from the flexibility and learning capacity of neural networks while dynamically determining the number of clusters based on the data, a feature of the Bayesian nonparametric approach. This methodology is a good example of innovative integration of neural network architectures with advanced clustering techniques.

4.4 Neural Graph Machines and Graph Agreement Models

Introduced in [9], Neural Graph Machines (NGM) represent a fusion of neural network architectures with graph-based semi-supervised learning, effectively utilizing both labeled and unlabeled data. This framework adds a graph-regularized objective to the training process of neural networks, compelling them to learn similar representations for adjacent nodes in the graph. NGM's versatility is highlighted by its compatibility with various neural network architectures, and its applicability to different forms of graphs. Experiments conducted by the authors demonstrate NGM's superiority over traditional methods in tasks like multi-label classification on social graphs, text categorization, and semantic intent clas-

sification.

As an extension of this research, [16] introduce Graph Agreement Models (GAM). GAM extends the concept of neural graph learning by incorporating an auxiliary agreement model that predicts the probability of label sharing between two nodes based on their features. This model significantly enhances the performance of node classification models by encouraging label agreement in cases where it is most probable. GAM's ability to apply to any semi-supervised classification problem, even in the absence of a pre-existing graph, makes it a versatile tool for training data extension. The implementation of GAM in various contexts, as shown in the paper's experiments, leads to marked improvements in accuracy, underscoring its potential in optimizing the selection and use of training data sets in machine learning.

5 Conclusion

This paper has explored various methods for optimally selecting and extending training data for machine learning models. We delved into methods such as using clustering, label matching, and graphs based methods for data extension. The criticality of data quality, the significance of modeling, and the scalability concerns in complex models in training data selection were also discussed. We also reviewed some existing implementations of these principles. The findings highlight the necessity of a balanced approach that considers both the quantity and quality of data while ensuring its relevance to the specific model requirements.

References

- [1] X. Wang, "Machine Learning Basics," 2015.
- [2] L. T. Rose and K. W. Fischer, "Garbage In, Garbage Out: Having Useful Data Is Everything," *Measurement: Interdisciplinary Research & Perspective*, vol. 9, no. 4, pp. 222–226, 10 2011. [Online]. Available: <https://www.tandfonline.com/doi/abs/10.1080/15366367.2011.632338>
- [3] Y. Bengio, A. Courville, and P. Vincent, "Representation Learning: A Review and New Perspectives," 6 2012. [Online]. Available: <http://arxiv.org/abs/1206.5538>
- [4] K. Hammouda and F. Karray, "A COMPARATIVE STUDY OF DATA CLUSTERING TECHNIQUES," University of Waterloo, Ontario, Tech. Rep., 2000.

- [5] T. Fredriksson, D. I. Mattos, J. Bosch, and H. H. Olsson, "Data Labeling: An Empirical Investigation into Industrial Challenges and Mitigation Strategies," 2020, pp. 202–216.
- [6] T. Calders, F. Kamiran, and M. Pechenizkiy, "Building classifiers with independency constraints," in *ICDM Workshops 2009 - IEEE International Conference on Data Mining*, 2009, pp. 13–18.
- [7] S. Moran, "Choosing the Best Bayesian Classifier: An Empirical Study," Tech. Rep., 2009. [Online]. Available: <https://www.researchgate.net/publication/40422668>
- [8] S. Wang, L. Hu, Y. Wang, X. He, Q. Z. Sheng, M. A. Orgun, L. Cao, F. Ricci, and P. S. Yu, "Graph Learning based Recommender Systems: A Review," *IJCAI International Joint Conference on Artificial Intelligence*, pp. 4644–4652, 5 2021. [Online]. Available: <https://arxiv.org/abs/2105.06339v1>
- [9] T. D. Bui, S. Ravi, and V. Ramavajjala, "Neural graph learning: Training Neural networks using graphs," in *WSDM 2018 - Proceedings of the 11th ACM International Conference on Web Search and Data Mining*, vol. 2018-February. Association for Computing Machinery, Inc, 2 2018, pp. 64–71.
- [10] L. Budach, M. Feuerpfeil, N. Ihde, A. Nathansen, N. Noack, H. Patzlaff, F. Naumann, and H. Harmouch, "The Effects of Data Quality on Machine Learning Performance," 7 2022. [Online]. Available: <http://arxiv.org/abs/2207.14529>
- [11] Y. SarcheshmehPour, Y. Tian, L. Zhang, and A. Jung, "Clustered Federated Learning via Generalized Total Variation Minimization," 5 2021. [Online]. Available: <http://arxiv.org/abs/2105.12769>
- [12] N. Piroonsup and S. Sinthupinyo, "Analysis of training data using clustering to improve semi-supervised self-training," *Knowledge-Based Systems*, vol. 143, pp. 65–80, 3 2018.
- [13] M. Werner, L. He, S. P. Karimireddy, M. Jordan, and M. Jaggi, "Provably Personalized and Robust Federated Learning," 6 2023. [Online]. Available: <http://arxiv.org/abs/2306.08393>
- [14] L. Jacob, F. Bach, and J.-P. Vert, "Clustered Multi-Task Learning: a Convex Formulation," Tech. Rep., 2008. [Online]. Available: <https://doi.org/10.48550/arXiv.0809.2085>
- [15] A. Pakman, Y. Wang, C. Mitelut, J. Lee, and L. Paninski, "Neural Clustering Processes," 12 2018. [Online]. Available: <http://arxiv.org/abs/1901.00409>
- [16] O. Stretcu, K. Viswanathan, D. Movshovitz-Attias, E. A. Platanios, A. Tomkins, and S. Ravi, "Graph agreement models for semi-supervised learning," in *Advances in Neural Information Processing Systems*, vol. 32, 2019.

Heimbjartur: Testing network policies with eBPF

Ingi Þór Sigurðsson

ingi.sigurosson@aalto.fi

Tutor: Jacopo Bufalino

Abstract

Securing network infrastructures in accordance with best practices remains an unexpectedly challenging task, particularly due to the complexities inherent in network policy management. Network policies, such as firewall rules, are pivotal in maintaining robust network security. However, they are often susceptible to flaws that can significantly compromise network integrity and unauthorized data access. To address this, this paper introduces Heimbjartur, a proof-of-concept software for verifying Layer-4 network policies by utilizing eBPF. Heimbjartur integrates into the operating system kernel, enabling efficient assessment of network packets to determine whether they are impacted by network policies or processed by the target application. This approach allows for a performant evaluation of millions of tests, offering a thorough and efficient method for network policy testing.

KEYWORDS: *eBPF, Linux, Computer networks, Network security, Firewall*

1 Introduction

Redundancy and fault tolerance in the operation of networking infrastructure has led to an increase in availability of online services around the world. However, network infrastructures across global network providers and cloud service providers are still susceptible to misconfigurations or faulty network policies. This can lead to unauthorized access of internal systems and an increase in security risk for organizations. Such misconfigurations may arise from human mistakes or unexpected consequences of software configuration and deployment tools. An illustration of this is the deliberate deletion of the NewsBlur database by an unauthorized entity [1]. The MongoDB instance powering NewsBlur was exposed to the internet, due to Docker silently configuring the firewall and thereby opening the database to the world. NewsBlur serves as a case in point for firewall misconfigurations. Quantitative studies of firewall configurations has been performed in [2, 3] concluding that they are inadequate. With the introduction of virtualization technologies such as virtual machines (VM), software containers and software-defined networks (SDNs), properly securing the network has become even more important and complex.

There is a clear need for tools and technologies that can verify network policies to make sure they are correctly configured and compliant with security requirements. By utilizing such tools, entities can prevent unauthorized data access or modification caused by erroneous policies.

This paper showcases Heimbjartur, a proof-of-concept Layer-4 network policy tester that utilizes eBPF (formerly known as *extended Barkley Packet Filter*), a technological framework that enables the secure and efficient execution of programs within the operating system (OS) kernel. By using eBPF, Heimbjartur can safely and efficiently determine how network policies affect the networking security aspect of a Linux host. Heimbjartur is open-source and available at <https://github.com/ingiths/heimbjartur>.

This paper is organized as follows. Section 2 presents necessary background of network policies and an introduction to eBPF and its wide use cases. Section 3 introduces Heimbjartur and its technical aspects. Section 4 evaluates Heimbjartur and its potential applications. Section 5 considers related work and comparisons to Heimbjartur. Finally, Section 6 considers further improvements to Heimbjartur and offers closing comments.

2 Background

This section offers an introduction to network policies and introduces eBPF, an essential component for enabling Heimbjartur's functionality.

2.1 Network policies

A network policy can be seen as an umbrella term for the various configurations in networking devices. Network policies affect the processing of network traffic in the various layers in the Open Systems Interconnection (OSI) model, and some of the more common network policies include but are not limited to: internet protocol (IP) packet filtering, network address translation (NAT), traffic shaping and quality of service (QoS). The configuration of network policies affects the security and performance of any network connected device. Therefore, it should be approached with utmost care and caution to ensure correct policy definitions.

Firewalls are a common tool to define network policies which affect protocols such as IP, Transmission Control Protocol (TCP), User Datagram Protocol (UDP) and Internet Control Message Protocol (ICMP). A firewall is a necessary component to maintain rigorous network security, but it can also lead to data breaches if configured incorrectly.

The most commonly used tools for defining network policies on a Linux host are provided by the Netfilter framework, such as iptables or the newer nftables user space programs, which allow for the definition of rules that determine the processing of network packets.

2.2 eBPF

The need for observation and manipulation of the various components that constitute the OS is necessary in maintaining rigorous network security policies. By leveraging eBPF, a modern method to extend the functionality of the Linux kernel, developers have gained the capacity to observe, trace, and manipulate events in the OS kernel. eBPF is restrictive by nature to ensure that user loaded programs do not crash the kernel and always terminate, and it provides this assurance by using an in-kernel verifier [4]. It has its own 64-bit instruction set (bytecode) that runs inside a VM to ensure the safety of the host kernel [4]. By design, programs are disallowed from exceeding 1 million instructions and unbounded loops are forbidden. For performance and optimization, Just-In-Time (JIT) com-

pilation is used to transform the eBPF bytecode into native machine instruction [4]. Since the initial release of eBPF in 2014, a wide variety of applications have emerged in the field of software engineering, such as eXpress Data Path (XDP) [5, 6], Cilium [7], Calico [8], and Katran [9].

By utilizing XDP, packet inspection and filtering can be done close the network interface card (NIC) before the networking stack of Linux processes the packet [5, 6]. XDP can yield up to 4x the performance compared to offloading the task to the kernel [10] when used for common packet filtering tasks, such as preventing denial-of-service (DoS).

The use of eBPF in large network infrastructures demonstrates its versatility across a range of use-cases. Numerous prominent networking corporations have started using eBPF for various tasks, such as Cloudflare for the Magic Firewall product [11], allowing them to craft intricate matching rules that were previously unattainable through the use of iptables or nftables alone. Facebook has created Katran [9], a Layer-4 load-balancer (L4LB), which enables them to handle network traffic at scale. Their previous generation of L4LB was based on the IP virtual server (IPVS) in the Linux kernel, and the use eBPF for packet related operations resulted in an increase in performance and reliability.

The most established and known application of eBPF are packet filters. However, in recent years, the tooling around eBPF has grown rapidly, enabling it for other computing tasks, such as networking in complex cloud-native environments and performance measuring of applications. The two most notable cloud-native solutions for networking that use eBPF are Cilium [7] Calico [8]. Both solutions can be used to harden and secure a Kubernetes (K8s) cluster [8]. In particular, Calico does this by utilizing eBPF networking features, and thus it is capable of hardening K8s clusters with thousands of network policies for ingress and egress traffic without significantly impacting performance.

Another powerful feature of eBPF is the rapid communication between the kernel space and the user space through key-value maps that enables the expansion of OS telemetry data for observability. As can be seen in ViperProbe [12], the use of eBPF enabled dynamic and adaptive metrics gathering for better observability without much increase in central processing unit (CPU) overhead for a microservice based system.

3 Heimbjartur

This section describes Heimbjartur and the technical aspects of the software. It begins with a brief discussion of the tooling, followed by an in-depth exploration of the use of eBPF for packet analysis. Finally, the section details network policy testing as conducted using Heimbjartur.

3.1 Tooling

The creation of eBPF programs has traditionally relied on the BPF Compiler Collection (BCC) [13] and the C programming language. However, new tools and programming languages have emerged, which promise memory safety without compromising performance, such as Rust [14]. Using Rust, projects such as Aya-rs [15] have appeared, which enables the creation of eBPF programs without relying on external libraries such as BCC or libpf. Hence, it is a sensible choice to write a eBPF program utilizing modern tools that offer performance similar to C without compromising safety.

3.2 Packet analyzing

For inspection of network packets, eBPF provides various program types that are related to networking but differ in invocation time in the Linux kernel. The most notable eBPF program types for this task are the XDP, traffic control (TC) and probe programs.

By using the XDP program type, an eBPF program can interpret and process packets before they reach the kernel networking stack. However, this is only available for incoming packets, therefore it is impracticable to inspect outgoing packets using XDP. It is necessary for Heimbjartur to be able to test network policies for incoming and outgoing network packets.

In contrast, a TC program attaches to the TC subsystem inside the Linux kernel and allows the exploration of incoming and outgoing packets after they have been processed by the Linux kernel. While a TC program appears to be a suitable program type for Heimbjartur, a disadvantage of using it is the inability to know what happens to processed packets inside the networking subsystem, as packet drops happen in the Netfilter framework, before they reach the TC subsystem.

While not directly used for network packet processing, function probes allow for the analysis of function arguments. Probes can be either kernel

probes (kprobes) or user probes (uprobes). It is possible to utilize function probes to attach to the main kernel functions related to networking inside the Linux kernel, such as `netif_receive_skb`, `dev_queue_xmit` or `ip_rcv` and gather information regarding the decision-making behind packet processing.

While XDP and TC are valid options for processing and inspecting network packets, a core goal of Heimbjartur is to know whether a packet was dropped and the reason for it. Only incoming packets are seen in XDP, and packet dropped by the Netfilter framework are not shown in TC. Therefore, to fulfil the requirement of knowing what happens to packets inside the Linux kernel, a kprobe program is the only method outlined above that can achieve this. A kprobe is attached to the kernel function `kfree_skb_reason`, which is a kernel function used to free kernel memory regions used by the `sk_buff` kernel data structure, the primary data structure within the kernel that represents a network packet. This function is called with two arguments: a pointer to the `sk_buff` structure and an integer that denotes the reason for why the `sk_buff` structure is being freed from memory. By utilizing these two data structures and with a kprobe eBPF program, it becomes feasible to monitor packet processing and the underlying reasons for the processing, thus facilitating a deeper understanding of the decisions made by the Linux networking stack.

3.3 Testing network policies

Heimbjartur was specifically created to monitor and gather information about what happens to TCP and UDP network packets, the predominant protocols within the Layer-4 of the OSI model. Therefore, protocols that reside on other layers such as HTTP, DNS and other protocols in Layer-4 are not considered.

To evaluate a network policy defined on a Linux host, the user must give Heimbjartur the following arguments: source host (IP:PORT), destination host (IP:PORT), the protocol to test, either TCP or UDP and finally whether the packet created from the aforementioned information is expected to be dropped or not. To facilitate this, Heimbjartur is equipped with a command line interface (CLI) capable of evaluating either an individual test or multiple tests sequentially.

Heimbjartur constructs a raw packet using information provided by the user. An important factor is to make sure that these testing packets do not leave the host and interfere with the external network. Therefore,

all packets are sent on the loopback interface (lo or lo0), which routes the packet from a host to itself. The eBPF program watches all packets that are processed, specifically packets created by Heimbjartur that are marked with a differentiated services code point (DSCP) value of 3 (binary: 000011). By employing a DSCP value of 3, packets are delivered with a best-effort approach and are discarded in favor of prioritizing more critical traffic. This strategy prevents the host from becoming overwhelmed by an excessive number of tests. The eBPF program parses packets and inserts their values into an eBPF map that is shared between the kernel space and the user space if and only if the DSCP value is equal to 3. The map consists of a simple data structure that contains the information about the source IP, source port, destination IP, destination port and drop reason. Using these fields, the user space part of Heimbjartur can interpret the results of a particular test packet and determine if the packet was allowed through the firewall or if it was dropped by it.

As mentioned before, Heimbjartur can be used in two modes: evaluation of a single test or evaluation of multiple tests. A simple test file can be created which Heimbjartur reads and for each test evaluates which hold and which do not. This feature allows users to define test files for different hosts and enables them to periodically run multiple tests to maintain rigorous network security.

4 Evaluation of Heimbjartur

This section evaluates Heimbjartur and how it can be used to test network policies. In addition, this section contains a performance analysis of Heimbjartur when running a test suite containing millions of tests.

4.1 Securing policies

With Heimbjartur it becomes possible to test network policies on a single Linux host and not require any external machines or networking equipment. As an example, in a given organization, system administrators should be the only ones able to access a management console for a particular system. Assume that system administrators reside in the 192.168.1.0/24 classless inter-domain routing (CIDR) range, other employees reside in the 192.168.2.0/24 CIDR range, the management console resides on the host 192.168.100.100 and the service is listening on port 443 using the

hypertext transfer protocol secure (HTTPS). To ensure that the firewall remains compliant with the security requirements, a test file could be created containing the following entries:

```
192.168.1.* 192.168.100.100:443 tcp pass
192.168.2.* 192.168.100.100:443 tcp drop
```

Which translates to: *“Expect the firewall to accept every network packet from an IP in the 192.168.1.0/24 CIDR range and drop all network packets from and IP in the 192.168.2.0/24 CIDR range”*. Using Heimbjartur, system administrators can simply test the firewall using the defined test file to make sure that it is compliant with security protocols.

4.2 Performance

Performance benchmarks were performed using test files containing millions of tests. These tests were run on an AArch64 Ubuntu 23.10 VM, using version 6.5.0 of the Linux kernel, with 4 processors and 2 GB of memory. The VM ran on a MacBook Pro with the Apple M1 Pro chip and 16 GB of memory. Figure 1 shows that the time it takes to evaluate tests grows linearly, with Heimbjartur being able to evaluate 10 million tests in about 2.5 seconds. At present, Heimbjartur processes tests sequentially, which presents an opportunity for parallelization to enhance its performance further.

While eBPF kprobes are a lightweight way to trace and observe kernel events, they do incur some performance overhead. As shown in figure 2, Heimbjartur inflicts some performance penalty on sending packets on the loopback interface. However, a test file that contains 10 million tests, which generates 10 million packets, only takes around 14 seconds to run.

5 Related work

The validation and testing of network policies for security has resulted in numerous academic research, emphasizing novel methodologies and sophisticated algorithms. This can range from generating test traffic between the sender and receiver [16] or through the application of static analysis on firewalls to simplify intricate rules [17].

Other approaches generate test cases through the analysis of existing firewall rules. El-Atway [18] utilizes a policy-based segmentation based technique to reduce the amount of test data needed to test a particular

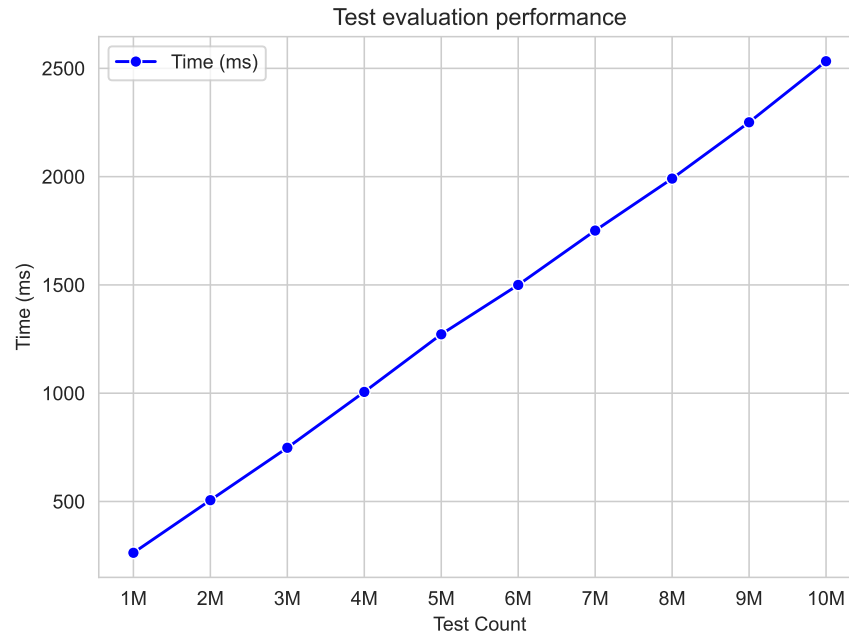


Figure 1. Linear scalability of Heimbjartur in test evaluation: demonstrating efficient processing of up to 10 Million tests in 2.5 seconds. This plot shows the linear relationship between the number of tests evaluated and the time taken, showcasing Heimbjartur’s consistent performance scalability.

network policy. In contrast, [19] generates abstract test cases through the use of Mealy automaton using unique input output (UIO) sequences. Concrete test cases are created from the abstract test cases, which results in test cases that are then used to test a particular network policy. However, [18] automates the approach by going through firewall logs and [19] does not offer any system or a concrete method to test a particular firewall policy. Compare this to Heimbjartur, which is able to hook into the kernel to gather detailed information about what happens to network packets inside a particular machine. While Heimbjartur does not possess the capability of test generation, rather relying on user input, it could be integrated into Heimbjartur in the future.

Another approach is the Blowtorch [20] C++ framework for firewall test generation. However, it is quite unclear on how it handles test packets for evaluation on whether network traffic was allowed through the firewall or not. Heimbjartur however is able to know exactly what happens to packets by attaching a kprobe into the Linux networking stack, knowing whether they are dropped due to the network policies in place or consumed by the intended recipient.

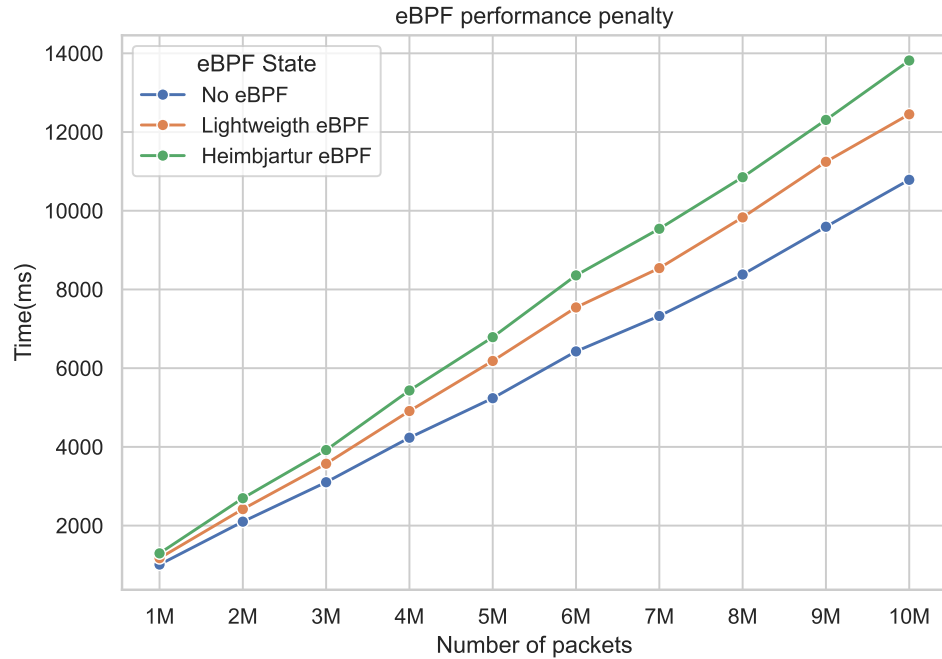


Figure 2. Performance impact of various eBPF implementations on packet transmission: comparing no eBPF, a basic eBPF kprobe with no logic, and Heimbjartur’s eBPF kprobe. The graph illustrates the time taken to send 10 million packets via the loopback interface, highlighting a 3-second performance impact introduced by Heimbjartur’s eBPF kprobe compared to no eBPF.

6 Conclusion

This paper introduced Heimbjartur and demonstrated its potential applications for evaluating and testing network policies. With the use of eBPF, Heimbjartur can hook into the Linux kernel to extract information about what happens to network packets and therefore providing a definitive answer. Heimbjartur is performative, being able to evaluate millions of network policy tests per second, thus supporting a large test suite for the testing of a particular network policy. A promising future direction for Heimbjartur is the use of automatic test case generation by utilizing techniques such as policy segmentation or abstract test cases to generate real test cases.

References

- [1] “How a Docker footgun led to a vandal deleting NewsBlurs MongoDB database.” <https://blog.newsblur.com/2021/06/28/story-of-a-hacking/>. (accessed: Oct. 11, 2023).
- [2] A. Wool, “A quantitative study of firewall configuration errors,” *Computer*, vol. 37, pp. 62–67, June 2004. 10.1109/MC.2004.2.

- [3] A. Wool, “Trends in Firewall Configuration Errors: Measuring the Holes in Swiss Cheese,” *IEEE Internet Computing*, vol. 14, pp. 58–65, July 2010. 10.1109/MIC.2010.29.
- [4] C. Authors, “BPF and XDP Reference Guide Cilium 1.15.0-dev documentation.” <https://docs.cilium.io/en/latest/bpf/>. (accessed: Oct. 22, 2023).
- [5] M. A. M. Vieira, M. S. Castanho, R. D. G. Pacífico, E. R. S. Santos, E. P. M. C. Júnior, and L. F. M. Vieira, “Fast Packet Processing with eBPF and XDP: Concepts, Code, Challenges, and Applications,” *ACM Computing Surveys*, vol. 53, pp. 16:1–16:36, Feb. 2020. 10.1145/3371038.
- [6] T. Høiland-Jørgensen, J. D. Brouer, D. Borkmann, J. Fastabend, T. Herbert, D. Ahern, and D. Miller, “The eXpress data path: fast programmable packet processing in the operating system kernel,” in *Proceedings of the 14th International Conference on emerging Networking EXperiments and Technologies*, CoNEXT ’18, (New York, NY, USA), pp. 54–66, Association for Computing Machinery, Dec. 2018. 10.1145/3281411.3281443.
- [7] Cilium, “Cilium - Cloud Native, eBPF-based Networking, Observability, and Security.” <https://cilium.io>. (accessed: Oct. 2, 2023).
- [8] Tigera, “eBPF use cases | Calico Documentation.” <https://docs.tigera.io/calico/latest/operations/ebpf/use-cases-ebpf/>. (accessed: Oct. 2, 2023).
- [9] N. Shirokov and R. Dasineni, “Open-sourcing Katran, a scalable network load balancer.” <https://engineering.fb.com/2018/05/22/open-source/open-sourcing-katran-a-scalable-network-load-balancer/>, May 2018. (accessed: Sep. 25, 2023).
- [10] D. Scholz, D. Raumer, P. Emmerich, A. Kurtz, K. Lesiak, and G. Carle, “Performance Implications of Packet Filtering with Linux eBPF,” in *2018 30th International Teletraffic Congress (ITC 30)*, vol. 01, pp. 209–217, Sept. 2018. 10.1109/ITC30.2018.00039.
- [11] C. J. Arges, “How We Used eBPF to Build Programmable Packet Filtering in Magic Firewall.” <https://blog.cloudflare.com/programmable-packet-filtering-with-magic-firewall/>, Dec. 2021. (accessed: Sep. 25, 2023).
- [12] J. Levin and T. A. Benson, “ViperProbe: Rethinking Microservice Observability with eBPF,” in *2020 IEEE 9th International Conference on Cloud Networking (CloudNet)*, pp. 1–8, Nov. 2020. 10.1109/CloudNet51028.2020.9335808.
- [13] IOVisor, “BPF Compiler Collection (BCC).” <https://github.com/iovisor/bcc>, Oct. 2023. (accessed: Oct. 17, 2023).
- [14] G. Hoare, “Rust Programming Language.” <https://www.rust-lang.org/>. (accessed: Oct. 17, 2023).
- [15] Aya Contributors, “Aya.” <https://github.com/aya-rs/aya>. (accessed: Oct. 17, 2023).
- [16] D. Hoffman, D. Prabhakar, and P. Strooper, “Testing iptables,” in *Proceedings of the 2003 conference of the Centre for Advanced Studies on Collaborative research*, CASCON ’03, (Toronto, Ontario, Canada), pp. 80–91, IBM Press, Oct. 2003.

- [17] C. Diekmann, J. Michaelis, M. Haslbeck, and G. Carle, “Verified iptables firewall analysis,” in *2016 IFIP Networking Conference (IFIP Networking) and Workshops*, pp. 252–260, May 2016. 10.1109/IFIPNetworking.2016.7497196.
- [18] A. El-Atawy, K. Ibrahim, H. Hamed, and E. Al-Shaer, “Policy segmentation for intelligent firewall testing,” in *1st IEEE ICNP Workshop on Secure Network Protocols, 2005. (NPSec.)*, pp. 67–72, Nov. 2005. 10.1109/NPSEC.2005.1532056.
- [19] D. Senn, D. Basin, and G. Caronni, “Firewall Conformance Testing,” in *Testing of Communicating Systems* (F. Khendek and R. Dssouli, eds.), Lecture Notes in Computer Science, (Berlin, Heidelberg), pp. 226–241, Springer, 2005. 10.1007/11430230_16.
- [20] D. Hoffman and K. Yoo, “Blowtorch: a framework for firewall test automation,” in *Proceedings of the 20th IEEE/ACM International Conference on Automated Software Engineering, ASE '05*, (New York, NY, USA), pp. 96–103, Association for Computing Machinery, Nov. 2005. 10.1145/1101908.1101925.

Prospects of WebAssembly to address performance and energy challenges of mobile clients

Jaakko Perttula

jaakko.perttula@aalto.fi

Tutor: Hannu Flinck

Abstract

WebAssembly has been studied as a high-performance and energy efficient alternative to other code representations. This paper reviews the challenges in mobile clients which WebAssembly may be able to solve.

KEYWORDS: WebAssembly, performance, energy efficiency, mobile client

1 Introduction

Mobile end devices, such as smart phones, have become powerful, multi-purpose, networked computing platforms. As a result, new computation-intensive and latency-critical services targeting mobile clients keep emerging, increasing the demand for efficient and portable client-side software. However, client-side computation is severely limited by inefficient web technologies and energy constraints.

WebAssembly (Wasm) is a new alternative to portable application code such as JavaScript. As its name suggests, it is a low-level instruction format primarily designed to overcome performance issues in web applications. Recent literature covers extensive performance analysis and comparison of WebAssembly and JavaScript. However, research on its

energy-efficiency is limited. Furthermore, some studies provide inconsistent and inconclusive results.

This paper reviews recent research on the performance and energy efficiency of WebAssembly. The focus is on its potential to enable high-performance computation on energy-constrained mobile platforms.

This paper is organized as follows. Section 2 briefly covers the design and the current state of WebAssembly. section 4 reviews key findings in literature on WebAssembly's performance and energy efficiency, with focus on mobile clients and comparison to JavaScript. Finally, Section 6 concludes this paper.

2 WebAssembly overview

WebAssembly is an open standard for a low-level code format. Its development was initiated by four major web browser vendors, Google, Mozilla, Microsoft and Apple, to solve efficiency issues attributed to JavaScript. The first version of WebAssembly was released in 2017, and it has since been subject to active development and research.

This section provides an overview of the design and core specification of WebAssembly, and of its implementation and application in practice.

2.1 Design and specification

Since WebAssembly is still in its infancy, its specification is under active development. This section describes the current draft version (2.0) of the WebAssembly core specification [10]. This version adds new features, such as a 128-bit vector datatype and related SIMD instructions, to the standard. These features are highlighted for clarity in the remainder of this section.

Design goals

The design of WebAssembly aims for an efficient code format for fast and safe execution, independent of platform, hardware, and programming language. It defines a virtual instruction set architecture with well-defined execution semantics, encoding, and validation rules. The binary encoding aims for a compact representation that can be decoded, compiled, and executed efficiently in parallel streams.

Computational model and representation

WebAssembly instructions operate on a virtual operand stack, with each consuming and producing a defined number of stack elements. The instruction set includes numeric, variable, memory, and control instructions. The instructions have defined operand and result datatypes of specific storage sizes. The numerical datatypes are 32-bit and 64-bit integers and floating-point numbers. In addition, draft version 2.0 of the specification defines a 128-bit vector type. Where applicable, there also exists separate instructions for unsigned and signed operations.

Apart from the virtual operand stack, WebAssembly programs may access a linear, byte-addressable memory arrays. The memory is accessed with load and store instructions of specific storage sizes. A memory array has an initial size but may be grown during execution.

Each instruction is encoded by a single-byte opcode, except for vector instructions that have a one-byte prefix and a variable length opcode. If an instruction has immediate arguments, they directly follow the opcode. Structured control instructions for loops and conditional branches have explicit terminating opcodes.

WebAssembly programs are split into modules that can be independently loaded, validated, compiled, and executed. A WebAssembly module contains zero or more sections for definitions of its imports, exports, memories, global variables, and functions. Imports and exports define which components, such as functions and memories, it expects from or exposes to an external environment for interaction. Memory definitions specify an initial size, maximum size, and initialization data segments. Functions contain all of the program logic. A module may also specify a start function, similar to a main function in a C program.

Execution

The WebAssembly core specification defines rules for module instantiation, which prepares the execution environment defined by the enclosed definitions.

Before its code can be safely executed, a module must be validated. Due to qualities such as a thorough type system, strict control flow, and a stack-based computational model, WebAssembly functions can be validated as independent streams in a single pass. In addition to functions, all imports and exports are verified to exist in their defined forms.

After validation, tables and memories are allocated and initialized with

specified contents if present, and finally, if a start function exists, it is invoked.

2.2 Implementation and application

The scope of the core specification of WebAssembly is limited to defining a virtual instruction set architecture with a specific binary representation. Therefore, WebAssembly can be embedded in a variety of execution environments. A typical use-case would be in web applications to replace or complement JavaScript. However, standalone WebAssembly runtimes enable a wide range of applications outside of web.

Major web browsers implement WebAssembly in the same runtimes that implement JavaScript. For example, Google Chrome runs WebAssembly in V8 [7], and Mozilla Firefox runs it in SpiderMonkey [6]. Similar to JavaScript, WebAssembly can be used to implement web application logic for both clients and servers. Node.js [5] and Deno [2], are examples of V8-based JavaScript/WebAssembly runtimes used in web service back-ends. In the web environment, WebAssembly modules are instantiated and run through JavaScript interfaces.

Beside web use-cases and Javascript runtimes like Node.js, there are standalone WebAssembly runtimes that extend the capabilities of pure WebAssembly programs with additional infrastructure. One such runtime environment is Wasmtime [8]. It implements the WebAssembly System Interface (WASI), which is a portable interface for host resources such as file systems and networking [12], and the Component Model which specifies WebAssembly wrappers that enable seamless interoperability with high-level datatypes [9]. Both WASI and the Component Model are in early phases of standardization. Such runtime environments open new dimensions for a future WebAssembly ecosystem.

As a virtual instruction set architecture with a binary representation, WebAssembly is a natural compilation target for high-level compiled languages. Clang, a C/C++ compiler using LLVM as a backend, supports WebAssembly as a compilation target as of version 8 [1][4]. Emscripten is a compiler toolchain targeting and optimizing specifically for WebAssembly [3]. These compilers enable the development of WebAssembly software through high-level programming languages.

Further, WebAssembly modules can be embedded in native software through libraries. For example, Wasmtime can be used as a library to embed WebAssembly in C/C++ and Rust programs.

3 Performance and energy challenges in mobile clients

Mobile clients, from laptops and smartphones to highly embedded acquisition nodes and controllers, have inherently higher constraints in both computational performance and energy, compared to stationary, wired computers. In order to be mobile, a device must include all required components, such as processors, radio modems and power sources, in a compact enclosure. Advances in hardware architecture and manufacturing have reduced component sizes and energy consumption per capability, which serves modern requirements for general-purpose mobile platforms such as smartphones. However, battery technology develops more gradually, which emphasizes challenges in energy efficiency.

This section briefly overviews the performance and energy efficiency challenges in smartphones, with the focus on software-related issues.

Various hardware components contribute to smartphone's total power consumption. The bulk of a smartphone's power is consumed by the processing units, most commonly CPUs and GPUs, volatile memory, a graphical display, and networking devices [25]. Power demand is directly linked to the utilization of each of these components. In CPUs and data storage, each operation requires transistor state manipulation, which always draws current. In radio devices, signal generation and amplification consume a significant amount of power, especially in poor channel conditions where relatively high amplification is required. In addition to highly dynamic changes in power consumption, maintaining responsiveness and availability requires keeping hardware components active.

A survey on performance optimizations for mobile applications by Hort et al. [18] highlights performance issues attributed to issues in mobile application software. The authors identify responsiveness, launch time, memory consumption and energy consumption as the predominant factors contributing to a consumer's perception of mobile application performance. They categorize researched optimization approaches for each of these characteristics. Responsiveness optimization was found to focus on computation offloading and prefetching. Research on launch time optimizations include preloading to eliminate cold-start delays, and strategies for prioritizing applications to keep in volatile memory. Research on applications' memory usage investigate optimizations involving garbage collection, deduplication, and memory management. Energy optimizations received the most attention in academia. In addition to optimiza-

tion approaches applying to other characteristics, research has focused on hardware features, application programming interfaces (APIs), and communication protocols. Further, the authors found extensive research on the impact of code refactoring, programming practices, and the choice of programming languages, on mobile application performance. The authors identify outstanding performance optimization challenges, including trade-offs between runtime performance and energy efficiency, and cost of comprehensive testing.

4 Performance and energy efficiency

WebAssembly is undergoing active development both in web contexts and as a general purpose program format. Its suggested use cases include computation-intensive applications such as video editing, gaming, and virtual reality in web browsers, as well as secure server-side computation of untrusted code, and portable components hybrid mobile applications [13].

Often characterized as having performance close to that of native code, WebAssembly has been subject to performance studies ranging from comparisons to JavaScript in a web browser [27], to benchmarks against native code in cloud and edge environments [21]. Some research also evaluate WebAssembly’s characteristics regarding energy efficiency, in environments ranging from desktop computers [14] to mobile devices [26].

Many of the published studies provide optimistic results regarding WebAssembly’s run-time performance and energy consumption [26] [15]. However, some studies highlight significant efficiency issues in the current implementations of WebAssembly [19]. Overall, efficiency results vary between benchmarks, runtime environments, and compilation methods.

4.1 Run-time performance

Comparison to JavaScript

WebAssembly is typically employed in web software. Consequently, its performance is often compared to that of JavaScript.

Many studies suggest that WebAssembly outperforms JavaScript in a variety of benchmarks. In particular, WebAssembly is almost always faster than JavaScript with relatively small inputs and short execution times [27][14]. The evaluations emphasize WebAssembly’s advantages in cases where its relatively low instantiation and compilation overhead

manifest the most.

Although WebAssembly has been suggested to be employed in numerically heavy computation in particular, its performance is sometimes manifested in more complex workloads. An application benchmark indicated an average of approximately 17% speed-up compared to JavaScript [15].

According to a study on HTML5 web worker migration from mobile clients to edge servers [20], offloading web workers using WebAssembly can result in 8.4 times faster execution compared to pure JavaScript implementations. In this use case, the speed-up is partly attributed to WebAssembly's efficient binary representation, which may have a significant impact on the resulting offloading overhead.

JavaScript may outperform WebAssembly in benchmarks where the same code path is executed repeatedly over an extended time period, as is the case in, for example, processing of large inputs with small mathematical kernels. This phenomenon has been attributed to the effectiveness of the just-in-time (JIT) compiler optimizations employed in JavaScript runtimes [29][27]. Another study showed that WebAssembly may be approximately 10% slower than JavaScript in micro-benchmarks [15]. These results highlight limitations of dynamic optimizations in WebAssembly runtimes.

Comparison to native

Many studies suggest that WebAssembly has performance characteristics comparable to native code execution. Native executables can be highly optimized with mature development tools and may omit restrictions that apply to WebAssembly due to, for example, its security requirements.

A study comparing WebAssembly's performance to that of container images [21] provides an example of a use case where WebAssembly can outperform native alternatives. The results show that, due to WebAssembly's fast cold-start, WebAssembly runtimes may outperform distroless container runtimes running native code with non-complex workloads.

Other studies suggest WebAssembly may be even around 50% slower than native code [14][19]. Particular issues degrading WebAssembly's relative performance have been identified, including inefficient usage of a processor's registers and addressing modes, which leads to a relatively high number of load and store instructions [19]. These studies were conducted on desktop computers employing the x86 processor architecture.

4.2 Energy efficiency

Most studies indicate that WebAssembly may be significantly more energy efficient than JavaScript. Some studies suggest improvements of around 30% to 40% [15][?]. Aggressive just-in-time compilation was pointed out as a factor that potentially increases JavaScript's energy consumption in comparison to WebAssembly. Another study comparing WebAssembly and JavaScript's energy consumption in Firefox and Chrome on a smartphone suggests WebAssembly is consistently more energy efficient than JavaScript, which consumed over twice as much energy [26].

When compared to native code, WebAssembly was shown to consume approximately twice as much energy [14].

5 Discussion

5.1 Limitations of studies in mobile clients

A notably large portion of studies on WebAssembly focus on runtime performance, while the number of studies concerned with its energy characteristics is limited. However, execution time and energy efficiency are often closely linked [23].

Many performance and energy benchmarks comparing WebAssembly to JavaScript and native code have been conducted in desktop environments. This may affect the applicability of perceived results to mobile platforms, since there may be significant variance in performance between desktop and mobile environments [29]. Beside the relatively constrained overall resources, mobile devices often employ different processor architectures, such as ARM, compared to the x86 architecture which is more common in desktop computers. Arguably, the differences in processor attributes, such as the number and function of available instructions and registers, has a non-negligible impact on compiler optimizations and achievable performance and energy figures. Therefore, more studies should be dedicated to analyzing WebAssembly on constrained embedded systems to clarify its viability in addressing performance and energy challenges in mobile clients.

Most of the currently published studies compare WebAssembly to either JavaScript or native binaries. This limits the evaluation of WebAssembly's potential to complement or replace other executable formats. In par-

ticular, there is little direct comparison between WebAssembly and Java bytecode, which is in extensive use in Android devices.

5.2 Prospects of WebAssembly

Being developed by the World Wide Web Consortium (W3C) with input from multiple high-profile technology companies, WebAssembly has an arguably solid foundation for future development.

Although initially developed for, and still mostly used in the web, WebAssembly may find increasing adoption in a wide range of non-web environments. Its inherent sandboxing and low cold-start overhead make it an attractive alternative to conventional containers, especially in serverless applications [17][21]. In the mobile domain, WebAssembly may have potential as a portable and safe alternative to native applications [28]. Furthermore, WebAssembly implementations for microcontrollers exist [16], which adds to its potential as a universal code representation across platforms. WebAssembly may receive significantly higher adoption as a general-purpose executable format once system interfaces and canonical ABIs are standardized.

Various new features, such as support for multithreading and garbage collection, have been proposed to alleviate WebAssembly's current limitations [11]. As WebAssembly is a relatively new binary format, there may be considerable opportunities for optimizations that are yet to be realized in compilers. A recent study discovered a large number of missed optimizations in an official WebAssembly optimizer and estimated that fixing them may result in a performance gain of over 17% [22]. Suboptimal compilation has been recognized as a non-negligible factor limiting WebAssembly's performance [29][19]. In addition, significant variation in performance across different execution environments has been observed [29]

6 Conclusion

A majority of studies evaluating WebAssembly's performance and energy consumption suggest that WebAssembly is a viable solution in addressing related challenges, especially in web applications, where the alternative solution would be JavaScript. Some outstanding issues in the execution and optimization of WebAssembly have been identified. However, some of

them may be solved in the near future.

References

- [1] Clang, 2023. <https://clang.llvm.org/>.
- [2] Deno, 2023. <https://deno.land/>.
- [3] Emscripten, 2023. <https://emscripten.org/>.
- [4] LLVM, 2023. <https://llvm.org/>.
- [5] Node.js, 2023. <https://nodejs.org/>.
- [6] SpiderMonkey JavaScript/WebAssembly engine, 2023. <https://spidermonkey.dev/>.
- [7] V8 JavaScript/WebAssembly engine, 2023. <https://v8.dev/>.
- [8] Wasmtime, 2023. <https://wasmtime.dev/>.
- [9] WebAssembly Component Model, 2023. <https://github.com/WebAssembly/component-model>.
- [10] WebAssembly core specification, 2023. <https://webassembly.github.io/spec/core/> (2023).
- [11] WebAssembly feature proposals, 2023. <https://github.com/WebAssembly/proposals>.
- [12] WebAssembly System Interface overview, 2023. <https://github.com/bytecodealliance/wasmtime/blob/main/docs/WASI-overview.md>.
- [13] WebAssembly's suggested use cases, 2023. <https://webassembly.org/docs/use-cases/>.
- [14] João De Macedo, Rui Abreu, Rui Pereira, and João Saraiva. On the Runtime and Energy Performance of WebAssembly: Is WebAssembly superior to JavaScript yet? In *2021 36th IEEE/ACM International Conference on Automated Software Engineering Workshops (ASEW)*, pages 255–262, 2021. <https://doi.org/10.1109/ASEW52652.2021.00056>.
- [15] João De Macedo, Rui Abreu, Rui Pereira, and João Saraiva. WebAssembly versus JavaScript: Energy and Runtime Performance. In *2022 International Conference on ICT for Sustainability (ICT4S)*, pages 24–34, 2022. <https://doi.org/10.1109/ICT4S55073.2022.00014>.
- [16] Robbert Gurdeep Singh and Christophe Scholliers. WARDuino: A Dynamic WebAssembly Virtual Machine for Programming Microcontrollers. In *Proceedings of the 16th ACM SIGPLAN International Conference on Managed Programming Languages and Runtimes, MPLR 2019*, pages 27–36. Association for Computing Machinery, 2019. <https://doi.org/10.1145/3357390.3361029>.
- [17] Adam Hall and Umakishore Ramachandran. An Execution Model for Serverless Functions at the Edge. In *Proceedings of the International Conference on Internet of Things Design and Implementation, IoTDI '19*, pages 225–236. Association for Computing Machinery, 2019. <https://doi.org/10.1145/3302505.3310084>.

- [18] Max Hort, Maria Kechagia, Federica Sarro, and Mark Harman. A Survey of Performance Optimization for Mobile Applications. *IEEE Transactions on Software Engineering*, 48(8):2879–2904, 2022. <https://doi.org/10.1109/TSE.2021.3071193>.
- [19] Abhinav Jangda, Bobby Powers, Emery D. Berger, and Arjun Guha. Not so Fast: Analyzing the Performance of Webassembly vs. Native Code. In *Proceedings of the 2019 USENIX Conference on Usenix Annual Technical Conference*, USENIX ATC '19, pages 107–120. USENIX Association, 2019. <https://doi.org/10.5555/3358807.3358817>.
- [20] Hyuk-Jin Jeong, Chang Hyun Shin, Kwang Yong Shin, Hyeon-Jae Lee, and Soo-Mook Moon. Seamless Offloading of Web App Computations From Mobile Device to Edge Clouds via HTML5 Web Worker Migration. In *Proceedings of the ACM Symposium on Cloud Computing*, SoCC '19, pages 38–49. Association for Computing Machinery, 2019. <https://doi.org/10.1145/3357223.3362735>.
- [21] Vojdan Kjorveziroski and Sonja Filiposka. WebAssembly as an Enabler for Next Generation Serverless Computing. *Journal of Grid Computing*, 21(3):1–20, 2023. <https://doi.org/10.1007/s10723-023-09669-8>.
- [22] Zhibo Liu, Dongwei Xiao, Zongjie Li, Shuai Wang, and Wei Meng. Exploring Missed Optimizations in WebAssembly Optimizers. In *Proceedings of the 32nd ACM SIGSOFT International Symposium on Software Testing and Analysis*, ISSSTA 2023, pages 436–448. Association for Computing Machinery, 2023. <https://doi.org/10.1145/3597926.3598068>.
- [23] Fernando Oliveira and Júlio Mattos. Analysis of WebAssembly as a Strategy to Improve JavaScript Performance on IoT Environments. In *Anais Estendidos do X Simpósio Brasileiro de Engenharia de Sistemas Computacionais*, pages 133–138. SBC, 2020. https://doi.org/10.5753/sbesc_estendido.2020.13102.
- [24] Rui Pereira, Marco Couto, Francisco Ribeiro, Rui Rua, Jácome Cunha, João Paulo Fernandes, and João Saraiva. Energy Efficiency across Programming Languages: How Do Energy, Time, and Memory Relate? In *Proceedings of the 10th ACM SIGPLAN International Conference on Software Language Engineering*, SLE 2017, pages 256–267. Association for Computing Machinery, 2017. <https://doi.org/10.1145/3136014.3136031>.
- [25] Pijush Kanti Dutta Pramanik, Nilanjan Sinhababu, Bulbul Mukherjee, Sanjeevikumar Padmanaban, Aranyak Maity, Bijoy Kumar Upadhyaya, Jens Bo Holm-Nielsen, and Prasenjit Choudhury. Power Consumption Analysis, Measurement, Management, and Issues: A State-of-the-Art Review of Smartphone Battery and Energy Usage. *IEEE Access*, 7:182113–182172, 2019. <https://doi.org/10.1109/ACCESS.2019.2958684>.
- [26] Max van Hasselt, Kevin Huijzendveld, Nienke Noort, Sasja de Ruijter, Tanjina Islam, and Ivano Malavolta. Comparing the Energy Efficiency of WebAssembly and JavaScript in Web Applications on Android Mobile Devices. In *Proceedings of the 26th International Conference on Evaluation and Assessment in Software Engineering*, EASE '22, pages 140–149. Association for Computing Machinery, 2022. <https://doi.org/10.1145/3530019.3530034>.

- [27] Weihang Wang. Empowering Web Applications with WebAssembly: Are We There Yet? In *2021 36th IEEE/ACM International Conference on Automated Software Engineering (ASE)*, pages 1301–1305, 2021. <https://doi.org/10.1109/ASE51524.2021.9678831>.
- [28] Elliott Wen, Gerald Weber, and Suranga Nanayakkara. WasmAndroid: A Cross-Platform Runtime for Native Programming Languages on Android. *ACM Trans. Embed. Comput. Syst.*, 22(1), oct 2022. <https://doi.org/10.1145/3530286>.
- [29] Yutian Yan, Tengfei Tu, Lijian Zhao, Yuchen Zhou, and Weihang Wang. Understanding the Performance of Webassembly Applications. In *Proceedings of the 21st ACM Internet Measurement Conference, IMC '21*, pages 533–549. Association for Computing Machinery, 2021. <https://doi.org/10.1145/3487552.3487827>.

Adversarial perturbation attacks on CNNs

Jere Hirviniemi

jere.hirviniemi@aalto.fi

Tutor: Blerta Lindqvist

Abstract

Convolutional Neural Network classifiers (CNNs) are widely used to classify data, such as images, to different classes. These classification networks have a wide range of applications. This paper examines different adversarial perturbation attacks on CNNs. Perturbation attacks aim to find an imperceptible change, a perturbation, that, when applied to the input, changes the networks classification of the input. It is crucial to understand the attack landscape on CNNs, as they are increasingly deployed to perform critical classification tasks. By understanding the attacks, strategies can be developed to defend against them.

KEYWORDS: Convolutional Neural Network, adversarial perturbation attack, classification task

1 Introduction

Convolutional Neural Network classifiers (CNNs) [1] are a popular approach to solving image classification tasks. Having a computer that is able to perceive visual data and accurately perform actions and make decisions based on visual information is highly useful for various industries. In image classification, a CNN is trained to classify a given input image

to one or multiple correct classes. It is crucial that CNNs perform the classification reliably and accurately. A classification mistake may lead to various issues depending on the field in which the CNN was used. For instance, an autonomous vehicle misclassifying a stop sign as a 80 km/h speed limit may lead to a crash, endangering human lives [2, 3]. In social media moderation, a CNN trained to recognize and delete harmful images might misclassify a disturbing image as a harmless dog, exposing users, especially children, to traumatizing content on their front page.

This paper broadly examines the different existing attacks that aim to compromise the integrity and correctness of a CNN. The goal is to survey the current landscape of attacks on CNNs and the threats they propose. The paper focuses on evasion attacks, where the attacker aims to create an input that evades correct classification by the network. The following evasion attacks are examined specifically: Fast Gradient Sign Method (FGSM) [4], Projected Gradient Descent (PGD) [?], CarliniWagner (CW) [5] and Zeroth Order Optimization (ZOO) [6].

This paper is organized as follows. Section 2 introduces general characteristics of the attacks. For the purposes of this paper, the specific attacks are divided into groups based on what level of access the attacker needs to the network. Each attack is then reviewed more closely in separate sections. Section 3 investigates white-box attacks, where the attacker has some level of access to the neural network and its parameters. In contrast, Section 4 examines black-box attacks, where the attacker has no access or information about the neural network. Section 5 summarizes the findings and evaluates the relative threat of the attacks.

2 Characteristics of perturbation attacks

Perturbation attacks on neural networks were first studied in [7]. The authors discovered an intriguing property in neural networks. A small indistinguishable perturbation in the input may change the classification made by the network. The perturbation that achieves this is not just any random noise, but a carefully calculated minimal change in the input, where the input should still be perceived to belong to the initial class by a resilient perceiver. The perturbation is generally calculated by finding the direction of the change, the gradient, in the input that leads to the most damage in the classification. The new input with the added perturbation is called an adversarial example. Finding the optimal perturbation

is a typical optimization problem, thus various existing optimization algorithms can be applied to solve for it. A common algorithm in machine learning optimization is Gradient Descent, and many other algorithms have been developed by expanding on it [8].

One of the challenges of perturbation attacks is calculating the perturbation efficiently. As neural networks tend to be deep and nonlinear, finding the gradient for the most effective perturbation is nontrivial. [7] introduced a function for approximating the "minimum distortion", but other faster methods, such as FGSM [4], have been developed since.

Szegedy et al. [7] found that the adversarial examples tend to transfer to other CNNs. This means that the adversarial examples generated for one network are also misclassified by other networks. Even networks with different layers, parameters and those trained on different training data are susceptible to misclassifying the transferred adversarial examples relatively frequently. Since adversarial examples generalize, combining multiple networks into an ensemble to perform the classification does not provide significant resistance against them [4].

The different attacks can be divided into two groups: white-box and black-box attacks. White-box attacks require the attacker to have all of the information about the target network. While it is relatively easy to defend against white-box attacks by keeping the network confidential, the generalizing nature of adversarial examples means even those networks can be vulnerable against adversarial examples created on other networks. Black-box attacks are credible threats to all networks, as the attacker does not need any information about the target network in order to perform the attack. In the following sections, the specific attacks are examined in detail.

3 White-box attacks

3.1 Fast Gradient Sign Method (FGSM)

Fast Gradient Sign Method (FGSM) was introduced in [4] as a method to quickly calculate the small perturbations used to attack the network. The paper argues that the networks vulnerability to perturbation attacks would stem from their linear nature. Expanding on that hypothesis, the paper introduces a method of calculating the perturbations by linearly es-

timating the cost functions around the model parameters, and efficiently calculating the gradient for the most effective change in the input using backpropagation. This method works well for finding small changes where the linear estimation is accurate. However, the authors noted that since the input is highly dimensional due to the large amount of pixels, applying the infinitesimal changes to each input dimension added up to a significant change in the resulting classification.

FGSM is a single-step attack that directly uses the sign of the gradient to update the input. It's a relatively simple and computationally efficient method. However, the method will not provide the most optimal perturbations. While the original paper did not apply FGSM for targeted misclassification, it is possible to apply the algorithm for targeted misclassification as well [9, 10]. Targeted misclassification means calculating the perturbation that changes the classification of a given input to a pre-determined target class.

The authors are able to use the FGSM to efficiently create adversarial examples for nonlinear models that are misclassified with high likelihood. Since these examples are relatively quick to compute, multiple adversarial examples can be quickly computed to be mixed in with the training data. The authors noticed that by adding correctly labeled adversarial examples in the training data, the trained model became more robust against perturbation attacks. Adversarial examples could be used as a form of regression to train networks defend against perturbation attacks.

3.2 Projected Gradient Descent (PGD)

Projected Gradient Descent (PGD) [11] is a simple algorithm for creating adversarial examples. It expands on the idea of finding the adversarial example with Gradient Descent. Unlike Gradient Descent, PGD minimizes a function subject to constraints. The constraints ensure that the perturbation does not exceed a certain magnitude or remains within a specific range to maintain imperceptibility. The constraints are applied by projecting the solution of each gradient descent step onto a predetermined feasible range of values before starting the next iteration.

3.3 CarliniWagner (CW)

Carlini and Wagner [5] introduce three attacks on CNNs, collectively known as CarliniWagner attacks, for different distance metrics: L_0 , L_2 and L_∞ .

These are slower methods of calculating the perturbation, but are much more effective in deceiving the network. In particular, L_0 was introduced as the first perturbation attack capable of targeted misclassification on ImageNet [12].

The CarliniWagner attacks were shown to be able to bypass defensive distillation [5, 13]. Defensive distillation is a defensive mechanism that aims to reduce the effectiveness of adversarial examples on networks. While defensive distillation was able to reduce the effectiveness of adversarial samples created with [14] from 95% to 0.5% [13], adversarial examples created with CarliniWagner were 100% effective [5].

4 Black-box attacks

4.1 Transferring examples from a substitute model

Since adversarial examples tend to generalize between models, it is possible to train a substitute model to resemble the target network. Thus, it is possible to use one of the white-box methods to create an adversarial example of a given input that should have a relatively high likelihood of being misclassified by the target network. While it is not necessary to know anything about the target network to perform this attack, any information can be used to create a more accurate substitute model of the target, which could increase the effectiveness of the generated adversarial example.

Substitute model attacks have a significant drawback. Creating a substitute model for a large neural network can be time and resource consuming. Many of the CNNs used in the industry tend to be too large to be effectively targeted by a substitute model attack [6].

4.2 Zeroth Order Optimization (ZOO)

Zeroth Order Optimization (ZOO) [6] is a black-box attack on CNNs that does not require training a substitute model. Traditional optimization methods used in perturbation attacks, like FGSM, rely on computing the gradients of the model's loss with respect to its parameters. However, in zeroth-order optimization attacks, the attacker does not have direct access to these parameters and must estimate or approximate them using only the model's predictions. ZOO is a variant of CarliniWagner attack that

achieves this by using ADAM coordinate descent algorithm [15] to numerically estimate gradients. ZOO was shown to provide results comparable to CarliniWagner attack and significantly improved results compared to a substitute model black-box attack. Performance wise, ZOO requires more iterations than CarliniWagner in order to approximate the gradient.

5 Summary

In conclusion, this paper underlines the threat posed by perturbation attacks on neural networks. Various effective methods have been developed to perform perturbation attacks on neural networks. While some strategies exist to defend against these attacks, finding more effective defenses against all of them is an open problem. As neural networks are deployed to wider use in various industries, the developers must remain mindful of possible perturbation attacks on them. Additionally, while this paper focused only on a small set of evasion attacks, many other types of attacks exist and are developed against CNNs.

Many of the attacks share the same characteristics, and expand on each other. Generally, finding an effective adversarial example is a non-trivial optimization problem. Thus, different attacks can be developed by applying different optimization algorithms for finding the optimal perturbation. Just as the network is trained with these optimization algorithms to perform correct classification, the optimization process can be also done in reverse to find the optimal input for a given classification.

Hiding the network from the attackers is an effective method for reducing the attack surface, by making it effectively impossible for the attackers to use strictly white-box methods. This in conjunction with training the network against adversarial examples provides an effective, but not complete, defense against attacks using transferred adversarial examples.

It is recommended to evaluate the robustness of networks against the possible attacks. CarliniWagner attacks are a good baseline for evaluation, as they are currently one of the most effective perturbation attacks developed, and work as the basis of other attacks, such as ZOO.

References

- [1] K. O’Shea and R. Nash, “An introduction to convolutional neural networks,” 2015.
- [2] S. Pavlitska, N. Lambing, and J. M. Zöllner, “Adversarial attacks on traffic sign recognition: A survey,” 2023.
- [3] N. Morgulis, A. Kreines, S. Mendelowitz, and Y. Weisglass, “Fooling a real car with adversarial traffic signs,” 2019.
- [4] I. J. Goodfellow, J. Shlens, and C. Szegedy, “Explaining and harnessing adversarial examples,” 2015.
- [5] N. Carlini and D. Wagner, “Towards evaluating the robustness of neural networks,” 2017.
- [6] P.-Y. Chen, H. Zhang, Y. Sharma, J. Yi, and C.-J. Hsieh, “ZOO,” in *Proceedings of the 10th ACM Workshop on Artificial Intelligence and Security*, ACM, nov 2017.
- [7] C. Szegedy, W. Zaremba, I. Sutskever, J. Bruna, D. Erhan, I. Goodfellow, and R. Fergus, “Intriguing properties of neural networks,” 2014.
- [8] S. Ruder, “An overview of gradient descent optimization algorithms,” 2017.
- [9] Q. Yao, Z. He, H. Han, and S. K. Zhou, “Miss the point: Targeted adversarial attack on multiple landmark detection,” 2020.
- [10] J. Xu, Z. Cai, and W. Shen, “Using fgsm targeted attack to improve the transferability of adversarial example,” in *2019 IEEE 2nd International Conference on Electronics and Communication Engineering (ICECE)*, pp. 20–25, 2019.
- [11] A. Madry, A. Makelov, L. Schmidt, D. Tsipras, and A. Vladu, “Towards deep learning models resistant to adversarial attacks,” 2019.
- [12] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, “Imagenet: A large-scale hierarchical image database,” in *2009 IEEE Conference on Computer Vision and Pattern Recognition*, pp. 248–255, 2009.
- [13] N. Papernot, P. McDaniel, X. Wu, S. Jha, and A. Swami, “Distillation as a defense to adversarial perturbations against deep neural networks,” 2016.
- [14] N. Papernot, P. McDaniel, S. Jha, M. Fredrikson, Z. B. Celik, and A. Swami, “The limitations of deep learning in adversarial settings,” 2015.
- [15] D. P. Kingma and J. Ba, “Adam: A method for stochastic optimization,” 2017.

Software supply chain security: the SLSA specification

Jonas Bäck
jonas.back@aalto.fi
Tutor: Jacopo Bufalino

November 21, 2023

Abstract

The number of attacks on software supply chains have seen a sharp increase in recent years, and therefore securing these processes has become ever more important. The SLSA (Supply Chain Levels for Software Artifacts) framework attempts to solve some of the issues related to supply chain security by providing a software bill of materials and a transparent and hardened build process for artifacts. We investigate the practical steps required to create and release a SLSA level 3 compliant npm package, and investigate the areas which the SLSA framework makes more secure, as well as discussing the areas which it does not affect. The SLSA framework is shown to allow users to verify how, where and by whom an artifact has been built. The framework does not have strict requirements for dependencies, leaving a clear attack vector that users of the framework should consider to further improve their supply chain security.

1 Introduction

Modern software supply chains typically consist of a large number of steps to deal with writing, testing, building and publishing artifacts. These steps in the supply chain tend to use multiple dependencies and tools, with the tools themselves having a varying number of further dependencies. The amount of different steps and tools used in the process increases the attack surface of supply chains, allowing attackers to probe for weaknesses in any of the links along the chain.

While many of the exploits have been possible for decades, the number of attacks on software supply chains has seen a sharp increase in recent years [18]. One notable example is the 2020 SolarWinds hack, where over 18000 customers were infected after installing a malicious version of the SolarWorks Orion network

management tool [19].

We investigate how supply chain security is impacted by creating a build system that is compliant with SLSA (Supply Chain Levels for Software Artifacts) Level 3. The focus lies on how the build system improves the trustworthiness of the artifact in practice, while investigating which aspects of the supply chain are still vulnerable.

2 Background

This section introduces the technologies that are used in the experiment. Supply chain security as a whole is introduced, with a closer look at the SLSA framework, digital signatures, in-toto attestations and Dead Simple Signing Envelope (DSSE).

2.1 Supply chain security

Enck And Williams (2022) identified a few key supply chain security challenges [18]. First, dependency vulnerabilities should be identified and updated, preferably using some automated method. Dependencies and their creators should also be trusted. This is by no means a given, as it is not uncommon for open source projects to have hundreds or thousands of dependencies, with a large number of organizations and individuals maintaining them. Artifacts should also provide a software bill of materials, which describes the source code, dependencies and version information. Finally, the build process should be secured by using transparent, consistent and hardened build processes.

A number of standards and frameworks exist that attempt to deal with these supply chain security issues. Many of these standards are internal to companies or government agencies, with one public framework being the Secure Software Development Framework (SSDF), which was created by the US National Institute of Standards and Technology (NIST). As many other similar frameworks, the software development practices included in SSDF aim to reduce the number of vulnerabilities and limit the impact of potential vulnerabilities in released software as well as providing a common vocabulary for software security [22].

2.2 SLSA

SLSA (Supply Chain Levels for Software Artifacts), attempts to tackle the challenges in securing the build process. Version 1.0 of the SLSA specification was released in April 2023 by The Linux Foundation. SLSA provides a common vocabulary for software supply chain security, a way to evaluate used artifacts and a checklist that can be used to improve software security [12]. SLSA aims to protect supply chains from tampering, giving users confidence that the program has not been modified after its initial analysis and build.

The SLSA specification defines three different levels of trustworthiness for provenance and resistance to tampering of the build process or artifact. Below is a description for these levels, including their requirements and what benefits each level provides to end users. By default, a specific level includes all the requirements for lower levels [8] [14].

Level	Requirements	Benefits
1	The build process has to be consistent and automated and has to generate provenance, providing metadata about how the artifact was built, including information about the build platform, build process, the source code as well as dependencies.	Offers a basic level of source code identification, allows consumers to make risk-based decisions and can aid in managing vulnerabilities. Does not provide tamper resistance.
2	Requires the use of a version control system and a hosted build system that generates authenticated provenance. End users must have a way to validate this provenance.	The authenticated provenance prevents tampering with the build service and gives end users greater confidence in the origin of the artifact.
3	Auditors must certify that the build platform meets specific standards that guarantee the integrity of the provenance and the auditability of the source. The access to secret material in user build steps has to be limited and one run can not influence another.	SLSA 3 provides stronger tamper resistance and prevents threats such as cross-build contamination.

2.3 Digital signatures

To verify network communication, digital certificates are commonly used. The basis for digital signatures is public-key cryptography. Public-key cryptography is based on public and private key-pairs, where the private key is only kept on the source machine, while the public key is shared freely with other machines. Public-key cryptography allows for the encryption of data using a public key, such that the data can only be decrypted using its private key pair. As such, a sender can be sure that only the intended recipient can decrypt a message. Additionally, public-key encryption allows for signing messages using a private key. This signature can then be verified with the corresponding public key, allowing the recipient to verify that the message was sent by the holder of a specific private key.

One of the issues that quickly becomes evident with digital signatures is that the verification using a public key only has meaning if a recipient can be sure that the key itself has not been tampered with, which could happen in the case of a supply chain attack. To partially help with this issue of trust, it is a common practice to use certificate authorities (CAs) as a source of trust. CAs store and

issue digital certificates, which are used to prove the ownership of a specific private key, allowing for the recipient of a message or artifact to reliably verify the identity of the sender [15].

2.4 DSSE and in-toto

The in-toto attestation framework, developed by the Cloud Native Computing Foundation (CNCF), is a format for authenticated metadata about software artifacts that is used by the SLSA framework. An attestation consists of a predicate, with information about a subject artifact, with a modifiable schema. The attestation also consists of a statement binding it to a specific subject, as well as an envelope that handles authentication and serialization[1].

Attestations are often contained in envelopes. In the case of SLSA, Dead Simple Signing Envelopes (DSSE) are used. DSSE messages contain payloads of arbitrary structure with their corresponding signatures [2].

3 Experiment

We demonstrate the process of creating and releasing a SLSA level 3 compliant artifact with the following steps:

1. Create a git repository for the project.
2. Host the repository on GitHub.
3. Create base node project.
4. Enable automated artifact building by adding the GitHub workflow *builder_nodejs_slsa3.yml@v1.9.0*, maintained by the SLSA project [11].
5. Enable automatic publishing to the npm repository by adding the GitHub workflow *nodejs/publish@v1.9.0*, maintained by the SLSA project [11].

The project now fulfills the requirements of SLSA level 1, as GitHub actions provide an automated build system and stores relevant metadata.

Level 2 requires the use of a version control system as well as using a hosted build system. The use of git fulfills the requirement for a version control system, while GitHub actions is a hosted build system. SLSA level 2 also requires generating authenticated provenance and providing a way to verify this provenance for the end user. To simplify the process, the SLSA project maintains builders for a number of languages and build systems that are designed to be used with GitHub actions [11]. In this case, the *builder_nodejs_slsa3.yml@v1.9.0* workflow is used for generating authenticated provenance, while the *nodejs/publish@v1.9.0* action of the same project is used to publish the artifact and its provenance to the npm registry. The npm registry also allows for the storage of provenance statements, making it easy for end users to independently download and verify packages. The SLSA project also provide tools that can verify this provenance on a number

of platforms, which gives end users some confidence that the downloaded artifact has not been tampered with [10].

To reach SLSA level 3, the builds have to be isolated and ephemeral. GitHub actions fulfill these requirements [21].

The implementation can be found in a public GitHub repository [17] and the generated package has been published to the npm repository as one-is-a-number [4].

4 Evaluation

4.1 Generated provenance

The generation and verification of provenance can be considered the critical part of implementing a build system with a SLSA level of 2 or higher. This section investigates the build and provenance generation process closer.

Investigating the JSON-formatted attestation file generated by our chosen GitHub builder we notice two separate attestation objects, with one being generated by npm and one being generated by our SLSA builder. Both attestation objects further utilise In-Toto bundles, with the provenance payload stored in DSSE envelopes [1] [2]. We will be focusing on the SLSA provenance object.

The DSSE payload contains an In-toto statement which in turn contains a subject and a predicate. The subject includes general information about the artifact, such as a name and a sha512 digest of the package. This same information is generated for all node packages, and can be viewed in the package-lock.json file if npm is used to handle dependencies [5]. The predicate object contains more detailed build information and follows SLSAs specification for provenance [6]. More specifically, the predicate lists what builder was used to generate the object, how the builder was invoked and with which parameters. Additionally, it lists all environment variables that were available during build time, as well as a reference to the source code that was used for the build. In our case, this is a reference to the git repository of the project as well as a digest for the commit that was built.

4.2 Provenance verification

Listing provenance information is useful, but its usefulness in preventing supply chain attacks is limited if its authenticity cannot be verified. In terms of SLSA levels, an artifact needs to provide authenticated provenance to reach a level of 2 or higher. The provenance in our case is signed by Sigstore Cosign, with log entries being stored in Rekor transparency log [7]. Sigstore also functions as the root of trust for all authenticated provenance [20]. The signing secrets are also ephemeral, as they are generated using a keyless signing procedure [9].

The main provenance content is contained in a DSSE envelope, which includes a signature verifying the payload.

When using `slsa-verifier` [10], the root of trust is fetched from Sigstore. Building on the trusted root, the provenance bundle and entry can be verified. As the provenance statement is now trusted, its content can be used to verify the artifact, including its attestation signature, package hash, attestation headers and subject digest. The end user can also choose to output the complete provenance statement or validate additional fields, such as the source branch or tag.

5 Discussion

SLSA helps projects to provide a software bill of materials for their artifact, and makes it easy for users of the artifact to verify its authenticity. The provenance and its verification also mitigates the risk of downloading artifacts that have been tampered with. Additionally, SLSA encourages packages to use transparent, consistent and hardened build processes, lowering the risk that internal processes are tampered with.

A notable aspect that the SLSA specification does not cover is the trustworthiness of dependencies, as SLSA levels have no requirements for the build level of used dependencies [3]. In practice, this means that a SLSA level 3 project can use level 0 dependencies. Additionally, the description of resolved dependencies is only described as best effort in the spec [8]. This leads to the completeness of dependencies to vary greatly between different build systems and languages. In our example using npm, the dependency description is quite complete, with a list of packages, including their versions and package hashes. Other languages and build systems might not generate complete dependency descriptions. One such language is C++, where the dependencies of standard library functions are not well-defined, which can include platform-dependent implementations [16].

How dependencies are packed also impacts the completeness of the provenance. In the case of node and npm, the `package-lock.json` provides some security against tampering with dependencies, but as they are still resolved from an external source (the npm registry), the possible attack surface is still significant. To improve the tamper resistance of dependencies, they can be shipped together with the artifact. Continuing with the example of node, an artifact would include the `node_modules` directory, keeping all relevant code contained in one project, allowing for more complete auditing.

While compiled languages, such as Go or C, might not always provide a complete list of dependencies, artifacts created using them are typically self-contained executables which include all dependencies. As such, it could be argued that these languages are more secure, as no dependency can be altered after the executable has been created. This does not however tell the complete story, as analyzing and auditing these dependencies in the first place can prove challenging.

Version 0.1 of the SLSA spec included a higher level 4 which required hermetic builds. Hermetic builds treat all system tools and dependencies as source code, guaranteeing that the provenance's dependency list is complete [13]. This level was however dropped when version 1.0 of the SLSA spec was released, lowering the security ceiling that the framework can provide. However, even in the now-removed level 4, a level 4 artifact could be built from level 0 dependencies, likely as requiring all dependencies to reach a specific SLSA level would be impractical for any real project.

At its core, SLSA only tries to provide tools to verify that an artifact has been built a specific way, and that it has not been tampered with since it was built. SLSA does not try to deal with vulnerabilities in the projects it is used for, which also extends to external dependencies. SLSA is indeed only a small part in the larger software security ecosystem, and is most useful when used together with rigorous manual and automated vulnerability analysis of the released artifact and all its dependencies.

6 Conclusions

We have shown a number of simple, practical steps that can be taken to make an npm-package SLSA level 3 compliant by utilizing the SLSA projects provided GitHub workflows. Very similar steps can be taken for artifacts created with other languages as well, as SLSA provides similar generators for a handful of build systems, such as Go, Gradle and Bazel. The SLSA project also allows users to use arbitrary build systems with their Build your own builder-framework.

SLSA is not a silver bullet to solve all software vulnerabilities and cyber attacks, nor is it intended as one. The framework focuses on providing a verifiable software bill of materials and on promoting transparent, hardened build systems. Thus it should be used together with other tools and processes that assist with writing and auditing source code, analyzing vulnerabilities and keeping systems up to date as new vulnerabilities are found. Describing and verifying dependencies is quite loosely defined in the SLSA specification, requiring users to independently implement systems to secure dependencies to their required level. Users will also have to consider if arbitrary packages and developers should be trusted or if some additional verification is needed. It is not very useful to know that a package has not been tampered with if you do not know if the package is secure to begin with.

References

- [1] Sigstore in-Toto Attestations, 2023. URL <https://docs.sigstore.dev/verifying/attestation/>.
- [2] DSSE: Dead Simple Signing Envelope, October 2023. URL <https://github.com/secure-systems-lab/dsse>. original-date: 2020-09-01T10:20:42Z.
- [3] Slsa frequently asked questions, 2023. URL <https://slsa.dev/spec/v1.0/faq>.
- [4] one-is-a-number, October 2023. URL <https://www.npmjs.com/package/one-is-a-number>.
- [5] package-lock.json | npm Docs, 2023. URL <https://docs.npmjs.com/cli/v10/configuring-npm/package-lock-json>.
- [6] Slsa provenance, 2023. URL <https://slsa.dev/spec/v1.0/provenance>.
- [7] Sigstore rekor log, 2023. URL <https://docs.sigstore.dev/logging/overview/>.
- [8] Slsa requirements, 2023. URL <https://slsa.dev/spec/v1.0/requirements>.
- [9] Sigstore signing, 2023. URL <https://docs.sigstore.dev/signing/>.
- [10] slsa-framework/slsa-verifier: Verify provenance from SLSA compliant builders, 2023. URL <https://github.com/slsa-framework/slsa-verifier>.
- [11] SLSA GitHub Generator, October 2023. URL <https://github.com/slsa-framework/slsa-github-generator>. original-date: 2022-03-28T15:57:17Z.
- [12] Supply-chain Levels for Software Artifacts, 2023. URL <https://slsa.dev/>.
- [13] Slsa security levels v0.1, 2023. URL <https://slsa.dev/spec/v0.1/levels>.
- [14] Slsa security levels, 2023. URL <https://slsa.dev/spec/v1.0/levels>.
- [15] Josh Aas, Richard Barnes, Benton Case, Zakir Durumeric, Peter Eckersley, Alan Flores-López, J. Alex Halderman, Jacob Hoffman-Andrews, James Kasten, Eric Rescorla, Seth Schoen, and Brad Warren. Let's encrypt: An automated certificate authority to encrypt the entire web. In *Proceedings of the 2019 ACM SIGSAC Conference on Computer and Communications Security, CCS '19*, page 2473–2487, New York, NY, USA, 2019. Association for Computing Machinery. ISBN 9781450367479. doi: 10.1145/3319535.3363192.

- [16] Bence Babati and Norbert Pataki. Analysis of include dependencies in c++ source code. In *FedCSIS (Communication Papers)*, pages 149–156, 2017. doi: 10.15439/2017F358.
- [17] Jonas Bäck. one-is-a-number, October 2023. URL <https://github.com/backjonas/slsa-node-example>. original-date: 2023-10-09T10:36:48Z.
- [18] William Enck and Laurie Williams. Top five challenges in software supply chain security: Observations from 30 industry and government organizations. *IEEE Security & Privacy*, 20(2):96–100, 2022. doi: 10.1109/MSEC.2022.3142338.
- [19] Jeferson Martínez and Javier M Durán. Software supply chain attacks, a threat to global cybersecurity: Solarwinds’ case study. *International Journal of Safety and Security Engineering*, 11(5):537–545, 2021.
- [20] Zachary Newman. Reducing trust in automated certificate authorities via proofs-of-authentication. 2023. doi: 10.48550/arXiv.2307.08201.
- [21] Chinenye Okafor, Taylor R. Schorlemmer, Santiago Torres-Arias, and James C. Davis. Sok: Analysis of software supply chain security by establishing secure design properties. In *Proceedings of the 2022 ACM Workshop on Software Supply Chain Offensive Research and Ecosystem Defenses, SCORED’22*, page 15–24, New York, NY, USA, 2022. Association for Computing Machinery. ISBN 9781450398855. doi: 10.1145/3560835.3564556.
- [22] Murugiah Souppaya, Karen Scarfone, and Donna Dodson. Secure Software Development Framework (SSDF) Version 1.1: Recommendations for Mitigating the Risk of Software Vulnerabilities. Technical Report NIST Special Publication (SP) 800-218, National Institute of Standards and Technology, February 2022. URL <https://csrc.nist.gov/pubs/sp/800/218/final>.

Third party cookies and how browser developers plan to deprecate them

Joonas Savola

joonas.savola@aalto.fi

Tutor: Sanna Suoranta

Abstract

Third party cookies are block of data, set by third party servers, without user directly visiting their websites. Third party cookies are used in digital advertisement to track users and by using gathered information to provide more relevant and profitable ads. User tracking may cause privacy concerns and web browser developers have reacted this by starting phase out third party cookies. Purpose of this study is to present how third party cookies can be used for user tracking and how web browser developers plan to replace third party cookies.

KEYWORDS: *cookie, third party, tracking, browser, advertising*

1 Introduction

HTTP cookies, that are also called just cookies, are block of data that consist of key value pairs, which HTTP server sends to a web browser [1] and browser saves it to the device. Cookies are useful and sometimes essential for web pages to work but they also serve many other purposes. Cookies are used, for example, in authentication, storing stateful data like website settings or online shopping cart content.

Cookies are regulated in EU by GDPR and ePrivacy Directive [2], but

cookies are used for purposes which may concern web users. Concerns are related to third party cookies which are used tracking users [1] and profiling users based on their online behaviour [3]. User tracking and profiling are done in commercial purposes but malicious entities may misuse them. Web browser developers have noticed the bad reputation of the third party cookies and are developing technologies for privacy preserving advertisement without third party cookies [4, 5, 6]. Mozilla and Apple are already blocking third party cookies by default in their browsers for privacy concerns [7]. Google plans to deprecated use of third party cookies by the end of 2024 [4].

In order to evaluate proposals to replace third party cookies, this paper presents how third party cookies can be used to gather information about web users and what kind of browser related technologies are planned for replacing them to ensure ad based funding of websites.

This paper has following structure. Section 2 presents the background of third party cookies and describes how third party cookies are used for user tracking and how third parties can share information between each other. Section 3 describes browser related proposals for replacing third party cookies in advertisement. Section 4 discusses the third party cookie alternatives presented in Section 3 and Section 5 is last chapter with some concluding remarks.

2 Third party cookies

Increasing amount of web pages are composed of content which are delivered from third party servers [8]. For example, news websites can have embedded content from social media [9], content delivery network (CDN) or advertising network which are received from third party servers. Third party servers can use cookies when their contents are used without user directly visiting their web pages [1]. These cookies received from third party servers are called third party cookies.

2.1 Third party tracking

Tracking user's web browsing history with third party cookies is one way of perform third party tracking. Purposes of tracking user's web behaviour are often related to personalized user experience, web analytics or targeted advertisement [10]. According to Somé et al. [10] third party needs

ability to recognize the web user and identify the website which user is visiting. User recognition is done with third party cookies.

Simple example of third party tracking with cookies is the following. User first visits a website A, which has third party content from *thirdparty.com*. Now *thirdparty.com* has possibility to set a cookie, which later on can be used to recognize the user. Also, *thirdparty.com* learned that user visited the website A. Next time when user visits different website B which has content from *thirdparty.com*, this third party recognize the user with the cookie set earlier and it also learned that user has now visited in web pages A and B. The more user visits web pages which contains content from the same third party, the more information the third party can gather from user's web history and behaviour [10].

2.2 Cookie synchronization

Papadoulos et al. [3] note that, due to the same-origin policy and using cookies for identifying users, different third party trackers have different identifier for the same user. Same-origin policy restricts communication between scripts or documents that have different origins [11]. One method to work around this limitation and bypassing the same-origin policy is cookie synchronization. They [3] describe it as a method to share user's web browsing behavioral between different third parties so that all third parties does not necessarily have direct information which web pages users have visited.

Cookie synchronization works as follows. User visits a web page A where are third party content from *thirdparty.com* and a web page B where are third party content from *anotherthirdparty.com*. Both of these third parties can set their own cookies for recognizing the user in the future but these third parties can recognize only their own user identifiers. If these two third parties have decided to collaborate, they can share their user identifiers with each other when user visits web page C where is third party content from *thirdparty.com*. When user request third party content from *thirdparty.com* via web page C, *thirdparty.com* will redirect user to *anotherthirdparty.com* with crafted URL which contains user identifier of *thirdparty.com* and original domain of web page C. Now *anotherthirdparty.com* knows also user identifier used by *thirdparty.com* and it can match that to user identifier used by itself. *thirdparty.com* and *anotherthirdparty.com* can share their knowledge in the background by synchronizing their databases to learn more about user's web browsing

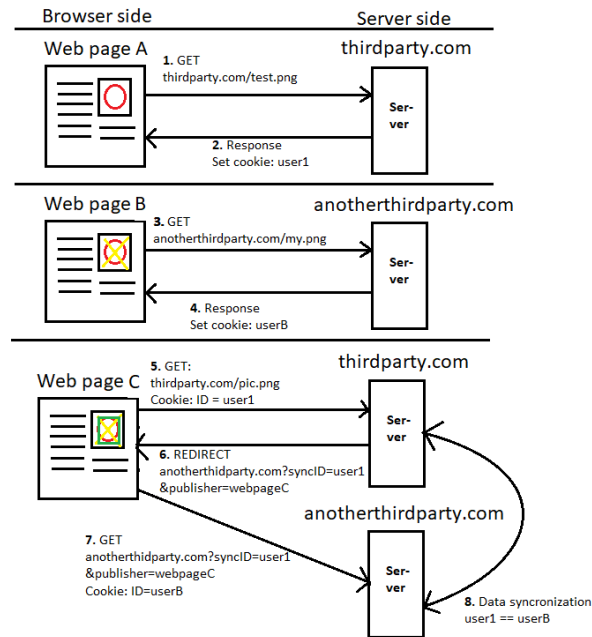


Figure 1. Cookie synchronization process based on Papadoulos et al. [3] example.

history from each other [3]. Figure 1. illustrates the example cookie synchronization process.

Papadopoulos et al. [3] found that 97% of the regular web browsing users are exposed to cookie synchronization and most of them are exposed to it within the first web browsing week. According to Papadoulos et al. [3], ad related domains participated in more than 75% of all cookie synchronizations.

2.3 General Data Protection Regulation (GDPR)

Tracking of web users has caused concerns related to privacy. Internet Engineering Task Force (IETF) state already in 2011 [1] that third party cookies are worrisome since those can be used to track the user without user ever visiting the third party server directly [1]. According to Urban et al. [9], website owners do not always know which third parties' content they are providing for the users and this may cause unwanted privacy and security problems. Legislators in European Union have notices worries related to web users privacy and have enacted GDPR to protect individuals right to their own personal data such as web browsing history. Even though a single cookie is not considered to be sufficient to identify a user [12], GDPR is applicable for cookies if they are used with other information stored on servers to create identifiable natural person profiles [13]. Basically this means that web pages which are using cookies must inform

users about personal data collection and request users consent for cookies [2].

3 Web browser without third party cookies

Future of third party cookies does not look bright. Privacy concerns related to third party cookies has been noticed by Apple and Mozilla. Safari and Firefox web browsers already block third party cookies by default [7]. According to StatCounter [14], Firefox had 3% browser market share and Safari had almost 20% browser market share in October, 2023. Google, whose Chrome web browser has almost two-third of the browser market [14], has planned to gradually phase out third party cookies starting from midway through 2024 [15].

3.1 Privacy Sandbox

Google is leading Privacy Sandbox initiative that target to protect people's privacy and provide tools to companies and developers for build prosperous digital businesses in the future. The Privacy Sandbox decreases the amount of third party tracking by deprecate use of third party cookies and replacing them with web standards that provides safer alternatives [4]. Google notes that deprecating and removing third party cookies is a challenge [16]. Their initial plan was to abandon third party cookies in 2022 but it has been postponed twice already [7]. According to current schedule, Google will disable third party cookies for 1% of Chrome users globally from the beginning of 2024 and starting gradually disabling third party cookies in Q3/2024 [4].

The Privacy Sandbox contains several proposals which are in different phase of development process. Proposals which are related to relevant content and ads are FLoC API, Topics API, Protected Audience API and Attribution Reporting API. FLoC means Federated Learning of Cohorts and its development is already stopped. Its purpose was to create large groups of people with similar browsing pattern to provide safety so that individual person does not stand out from the group [4]. According to Hana et al. [7], FLoC was ruled out since it did not comply with GDPR.

According to Google [4], Topics API is designed to maintain user privacy and enable showing relevant content and ads. Each website visited by user will be labeled by Topics API based on high-level topics of web-

sites. Then browser shares five most frequent topics collected from last three weeks [17] with the sites user visits and topics can be used to provide more relevant ads without revealing visited websites. Topics are selected from a publicly visible list which is human-curated and does not include sensitive categories like sexual orientation. Topics API was announced in Q1/2022 [17] and it is successor of discontinued FLoC with different approach to same use case. Chrome already supports Topics API starting from version Chrome Stable 115. According to Hana et al. [7], experts think that Topics API seems to be transparent and privacy protective but it is an issue for advertisers who want to target all website users with only one common topic.

Protected Audience API (formerly know as FLEDGE) is proposal of Google for having ad auctions without third party cookies by enabling on-device ad auctions by the browser [18]. Target of the Protected Audience API is to make remarketing possible without third party tracking. Process starts when user visits a website that wants to advertise its product. Then advertiser website can ask browser to add membership for the interest group. If the request is successful browser stores the name, owner and configuration information of the interest group. Configuration information contains access to bidding code, ad code and possibility to use real-time data depending on if the group owner is in the later phase invited to ad auction. When user visits a website which has ad space for sale, then the ad space seller specifies available ad space, who can bid and methods to score bids and asks browser to run auction. Only interest groups that browser is a member and whose owners are invited to auction can place their bids. Website will show winning ad in a fenced frame that securely shows ad without sharing cross-site data. Result of the auction can be reported by ad space seller and auction winner can report their wins. According to Dutton and Lee [18] using Protected Audience API advertiser does not learn websites user is visiting and ad publisher does not learn about user's interests. Geradin et al. [19] notes that there has been critique related to on-device ad auctions. Moving ad auctions to end users shoulder strain user devices with increasing use of bandwidth and computational power [20].

Proposal of Google for replacing third party cookies in conversion measurement is called Attribution Reporting API [21]. It provides two different kind of reports which are Event-level reports and Summary Reports. Event-level report provides data by combining a particular ad or view

with data on the conversion side. According to Nalpas and White [21], to preserve users privacy, data from conversion-side is very limited and noise is added to the data. With noise they mean random data instead of real data. Also another measure from privacy aspect is that there are delay between actual measurements and sending of the report. Summary reports offer more detailed data on aggregated level. Summary reports are encrypted by the browser and cannot be viewed without using aggregation service. Reason for this is that the aggregation service add noisy to the reports to add privacy. In both cases browser has a key role of combining clicks or views with conversion data and providing outputs to predefined endpoints. Cross-site identifiers are not used in either of the reports. Geradin at al. [19] notes that delay in Event-level reports makes campaign optimization impossible for advertisers.

3.2 Private Click Measurement

Apple has introduced Private Click Measurement (PCM) [6] for more privacy friendly web-to-web click measurement without cross-site tracking of users. PCM uses an 8-bit identifier for the ads in the source website side and a 4-bit identifiers for events on the conversion website side. This means that source website can use 256 identifiers for the ads and conversion side website can use 16 different identifier conversion events. When user clicks an ad that has a source side identifier, browser will store this event with source location, destination location and source side identifier for 7 days but this data is not accessible to the website. When user's activity on the destination website leads to a trigger event and if the triggering event matches with a stored click, then a single attribution report is sent out randomly between 24 to 48 hours later to the website where source click happened. Like Attribution Reporting API, PCM also handles data on-device.

3.3 Interoperable Private Attribution

Mozilla Foundation, developer of Firefox web browser, has made proposal called Interoperable Private Attribution (IPA), together with Meta, to provide privacy preserving advertising technology to the attribution problem [5]. IPA serves similar purpose as PCM and Attribution Reporting API but differs from them by using Secure Multiparty Computation (MPC) to join ad clicks, views and purchase events with conversion data instead of

using user device.

According to Mozilla [5], IPA uses Match Keys as user identifiers and these are stored locally by browser. Match Keys are set by websites and IPA enables websites to set same Match Key for multiple devices, if user logs in to the same website with different device. Match Keys are write-only identifier and are visible only to the browser. Match Keys are used together with source and trigger events. Source events are events, like when user sees or clicks an ad in a website. Trigger events are events which might be related to source events, like when clicking an ad has lead to purchase or application installation. Trigger event can contain a numerical trigger value which is meaningful related to conversion. Websites can request browser to create source or trigger event and log relevant metadata like time of action. Browser adds Match Key to the event and encrypts the data before sending it to the desired endpoint of the requester.

IPA [5] purposefully makes encrypted event data inaccessible for ad advertisers, publishers or parties representing them. Main idea of the proposal is that ad advertisers or publishers first collect large batch of encrypted event data and then make queries with encrypted data to MPC which decrypts the data but same time masks the Match Keys. MPC consists of at least two servers which perform actions. To make process easier to describe, let's assume that data is encrypted twice by using homomorphic asymmetric encryption, two servers are used for aggregation and each of them has their own private-public key pairs which public parts were used during encryption. Server one first decrypts the data and masks the Match Keys with deterministic algorithm which is commutative with the used encryption scheme. Server one shuffles the data before forwarding it to server two. Now server two performs decryption, masking and shuffling of data. Next server two joins source events with trigger events by using Match Keys, add some random noise to data and send the report back to requester. Since Match Keys are encrypted twice, server one cannot learn them after decrypting data. Server two cannot learn Match Keys after decrypting them since server one used deterministic algorithm to mask them. Original Match Keys are not exposed but if there were events which had same original Match Keys, those events also have same masked Match Keys due to the chosen algorithms and encryption schemes.

To achieve better privacy for web users, proposal [5] includes privacy

considerations. Differential privacy is consideration which adds small amount of random noise to the aggregate statistics. For example, if advertisers know from their own data that someone has used 200 euros to buy their product, aggregated statistics does not reveal this user since there is noise added to the value. It is possible to make multiple queries with small variation to the queried data and this way try to learn if specific purchase action was involved in the data. Privacy budgeting is consideration which are used to decide how much noise will be added to to queries. Mozilla suggest that website has a privacy budget for each week and they can decide how much of it will be used during one query. If website decided to use all of their privacy budget for one query during that week, they will receive only one report with small amount of noise. But they cannot make any more queries for the same set of users. On the other hand, websites can make multiple queries by using some portion of privacy budget for same set of users but now there are more additional noise with each query results.

3.4 PARAKEET

PARAKEET is proposal from Microsoft to improve end user privacy but still keep advertising as a profitable solution for funding websites [22]. Based on limited material available about PARAKEET [22], it is hard to conclude how it works and what is it current status. Microsoft Edge has 5% browser market share in the October, 2023, which makes it third largest browser in number of users.

4 Discussion

It seems inevitable, that using third party cookies for tracking and profiling users for advertisement purpose, will end in the future. Mozilla and Apple has released their proposals which mainly covers statistics about how many times ad placement has led to conversion. IPA from Mozilla differs from PCM and Attribution Reporting API by using MPC to join source events with trigger events. This means that MPC requires multiple trusted servers to provide user privacy. If using only one server, and it turns out to be malicious, server will have access to original Match Keys. If website provider owns this malicious server, it can directly combine original Match Keys with its own data and track users. On the other

hand, IPA does not strain user devices with additional tasks as much as PCM or Attribution Reporting API.

Google has almost two-third of the browser market share and this has reflected in their contribution to provide multiple proposals for different use cases in Privacy Sandbox. Developing alternative privacy preserving advertising technologies for third party cookies seems to be challenging. Google has already postponed deprecation of third party cookies twice and Chrome is still using third party cookies as default. Google's plan seems to be having full set of advertising tools available for advertisers and publishers before deprecating third party cookies. With this strategy they keep Chrome competitive in the eyes of ad tech during the change. This may lead to scenario, where Chrome has dominant tools for ad tech purposes and this leads to situation where Google can basically make all decisions related to the user privacy within the limits permitted by law and regulations. On the other hand, Apple and Mozilla or other web browser developers may join forces to develop and push forward proposal which they see the best.

5 Conclusion

This paper has reviewed how third party cookies are used for user tracking and what actions browser developers have taken to provide more privacy preserving tools to help ad based funding of website in the future without use of third party cookies. Third party cookies have has important role for making ad based funding of website more profitable. Due to privacy concerns related to third party cookies, Apple and Mozilla have already preventing the use of third party cookies by default in their browsers. Google is planning to deprecate use of third party cookies in Chrome in the end of 2024. Apple, Google and Mozilla have made proposals how to privacy preserving advertising could work in the future. Microsoft has also made their proposal but there were limited information available from it.

This papers relies heavily on browser developers own material about their proposal. There are opportunities for future work to test browser developers proposals in practice from user privacy perspective and their potential for advertisers. Additionally, there are other methods, like first party data, what can be used for making advertising more profitable.

References

- [1] A. Barth, “HTTP State Management Mechanism.” RFC 6265, The Internet Engineering Task Force, Apr. 2011. doi : 10.17487/RFC6265.
- [2] D. Bollinger, K. Kubicek, C. Cotrini, and D. Basin, “Automating cookie consent and GDPR violation detection,” in *31st USENIX Security Symposium (USENIX Security 22)*, (Boston, MA), pp. 2893–2910, USENIX Association, Aug. 2022.
- [3] P. Papadopoulos, N. Kourtellis, and E. Markatos, “Cookie synchronization: Everything you always wanted to know but were afraid to ask,” in *The World Wide Web Conference, WWW ’19*, (New York, NY, USA), p. 1432–1442, Association for Computing Machinery, 2019. doi: 10.1145/3308558.3313542.
- [4] Google, “Privacy Sandbox for the Web,” 2023. url: <https://privacysandbox.com/open-web/>. Accessed: 18.11.2023.
- [5] E. Taubeneck, B. Savage, and M. Thomson, “Interoperable Private Attribution (IPA),” 2022. url: <https://docs.google.com/document/d/1KpdSKD8-Rn0bWPTu4UtK54ks0yv2j22pA5SrAD9av4s/editheading=h.f4x9f0nqv28x>. Accessed: 18.11.2023.
- [6] J. Wilander, “Introducing Private Click Measurement, PCM,” 2021. url: <https://webkit.org/blog/11529/introducing-private-click-measurement-pcm/>. Accessed: 18.11.2023.
- [7] N. El Hana, M. Mercanti-Guérin, and O. Sabri, “Cookiepocalypse: What are the most effective strategies for advertisers to reshape the future of display advertising?,” *Technological Forecasting and Social Change*, vol. 188, p. 122297, 2023. doi: <https://doi.org/10.1016/j.techfore.2022.122297>.
- [8] J. R. Mayer and J. C. Mitchell, “Third-party web tracking: Policy and technology,” in *2012 IEEE Symposium on Security and Privacy*, pp. 413–427, 2012. doi: 10.1109/SP.2012.47.
- [9] T. Urban, M. Degeling, T. Holz, and N. Pohlmann, “Beyond the front page: measuring third party dynamics in the field,” in *Proceedings of The Web Conference 2020, WWW ’20*, (New York, NY, USA), p. 1275–1286, Association for Computing Machinery, 2020. doi: 10.1145/3366423.3380203.
- [10] D. F. Somé, N. Bielova, and T. Rezk, “Control what you include!,” in *Engineering Secure Software and Systems* (E. Bodden, M. Payer, and E. Athanasopoulos, eds.), (Cham), pp. 115–132, Springer International Publishing, 2017. isbn: 978-3-319-62105-0.
- [11] World Wide Web Consortium (W3C), “Same-origin Policy,” 2010. url: https://www.w3.org/Security/wiki/Same-Origin_Policy. Accessed: 18.11.2023.
- [12] C. Krisam, H. Dietmann, M. Volkamer, and O. Kulyk, “Dark patterns in the wild: Review of cookie disclaimer designs on top 500 german websites,” in *Proceedings of the 2021 European Symposium on Usable Security, EuroUSEC ’21*, (New York, NY, USA), p. 1–8, Association for Computing Machinery, 2021. doi: 10.1145/3481357.3481516.

- [13] European Parliament and Council of the European Union, “Regulation (EU) 2016/679 of the European Parliament and of the Council (GDPR).” url : <https://data.europa.eu/eli/reg/2016/679/oj>. Accessed: 18.11.2023.
- [14] GlobalStats, “Browser Market Share Worldwide,” 2023. url: <https://gs.statcounter.com/>. Accessed: 18.11.2023.
- [15] M. Mihajlija, “Prepare for phasing out third-party cookies,” 2023. url: <https://developer.chrome.com/docs/privacy-sandbox/third-party-cookie-phase-out/>. Accessed: 18.11.2023.
- [16] R. Merewood, “Preparing for the end of third-party cookies,” 2023. url: <https://developer.chrome.com/blog/cookie-countdown-2023oct/>. Accessed: 18.11.2023.
- [17] Google, “The Topics API,” 2023. url: <https://github.com/patcg-individual-drafts/topics>. Accessed: 18.11.2023.
- [18] S. Dutton and K. Lee, Kevin, “Protected Audience API,” 2023. url: <https://github.com/patcg-individual-drafts/topics> . Accessed: 18.11.2023.
- [19] Damien Geradin, Dimitrios Katsifis and Theano Karanikioti, “Google as a de facto privacy regulator: analysing the privacy sandbox from an antitrust perspective,” *European Competition Journal*, vol. 17, no. 3, pp. 617–681, 2021. doi : 10.1080/17441056.2021.1930450.
- [20] J. Hercher, “Criteo’s SPARROW Proposal Marks Ad Tech’s Venture Into Privacy Sandboxes And W3C,” 2020. url: <https://www.adexchanger.com/online-advertising/criteos-sparrow-proposal-marks-ad-techs-venture-into-privacy-sandboxes-and-w3c/>. Accessed: 18.11.2023.
- [21] M. Nalpas and A. White, “Attribution Reporting,” 2023. url: <https://developer.chrome.com/docs/privacy-sandbox/attribution-reporting/summary-reports/>. Accessed: 18.11.2023.
- [22] Microsoft, “PARAKEET,” 2022. url: <https://github.com/microsoft/PARAKEET/>. Accessed: 18.11.2023.

CDN attacks

Juho Pellinen

juho.pellinen@aalto.fi

Tutor: Jose Luis Martin Navarro

Abstract

Content Delivery Networks (CDN) are often used for serving content in the Internet. They are used for serving the content due to good peering, high bandwidth and redundancy. Also, they are often thought as a solution to improve security of the page. However, due to their nature, they can be used for causing denial of service attacks. This paper explains on different types of attacks CDN-services can be abused, and also demonstrates two theories.

KEYWORDS: *CDN, content delivery network, denial of service, DDoS*

1 Introduction

CDN-services are widely used in the Internet these days to serve content. It is estimated that majority of Internet traffic is transferred using CDN-services [1]. The main idea of a CDN-service is to reduce the resources used for serving data to client and increase the availability, and along with availability the security of the origin server increases because instead of going directly to the origin server, traffic goes through CDN-servers.

However, CDN-services cannot provide bulletproof protection against cyberattacks. There are different kinds of attack types which can exploit

CDN-services, and forward attacks towards the origin server. This paper focuses on telling about some attack types which are done exploiting CDN-services. Most of the presented attack types focus on either bypassing the CDN as a caching service, or caching a malicious response to CDN-servers which is showed to visitors.

The paper begins by talking about CDN-services in general followed by explanation of couple CDN-attack types. After that the paper focuses on creating demonstrations of two attack types using a commercial CDN-service to see if they are still valid, which is followed by discussing about the results and concluding the paper.

2 CDN basics

This section provides basic information about the idea of CDN, such as what is its purpose and what are its common use cases.

2.1 What is a CDN?

A CDN (*Content Delivery Network*) is a service often used for easing the resource usage of origin servers.

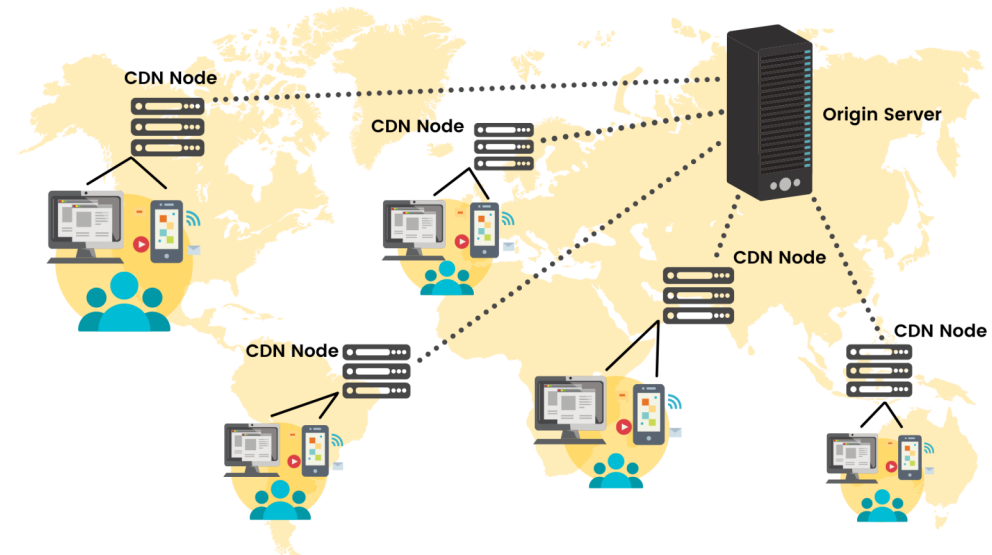


Figure 1. An example of architecture of a CDN-service [2]

The idea of a CDN-service is to provide a delivery network caching static data, such as images, videos and binaries, and static websites which are not changed that often [3]. Because usually these objects are usually larger ones than usually, downloading these from the origin server takes resources, and in worst case, jams the server traffic. This kind of jam-

ming can cause server inaccessibility. Because people cannot necessarily connect to the server, it can lead to a temporary loss of revenue. Caching makes it possible to store those files in CDN-servers (edge servers or CDN nodes) [4]. Users connect to CDN-service's servers instead of the original server and download cached files from CDN-servers as seen in Figure 1. This reduces the resources of origin server used for serving the files.

CDN-providers have edge servers in many continents. This is to provide a service with good connectivity worldwide. Multiple locations is necessary for improving availability. For instance, if user is fetching file from edge server which has issues, the data will be fetched from another server. In the best scenario, the client does not see anything changing.

Besides caching, an edge server is usually behind a high-speed connection. This is to ensure that downloading files would not slow when multiple users are downloading from the same edge server.

2.2 Security

CDNs can be used for defending security problems as well. Because the client fetches the data from an edge server, the real IP-address of the origin server will not be revealed to the client. This makes it harder for a potential attacker to get the real IP-address of the origin server. Also, certain services can be used for filtering given IP-addresses accessing the service or blocking a certain kind of traffic if the CDN-service provides a web application firewall.

Because of hiding the IP-address of the origin server, CDN can be used for protecting the origin server from Denial of Service (DoS)-attacks. DoS-attacks are done to make target's resources unavailable [5]. While a single DoS-attack is done by a single source, the attack can consist on many source, calling it DDoS-attack (*Distributed Denial of Service*). In DDoS-attacks many devices are used to exploit a single target device to prevent serving its data to users.

DDoS-attacks can happen by flooding target server with HTTP-requests by doing HTTP-requests from multiple terminals at the same time. Due to the tend of CDN-services, they can be used for blokcing certain attack types because the traffic does not reach the origin server. The main idea of a DDoS-attack is to cause harm for the target by preventing access to its server. In case of downtime, some CDN-services can display a cached version of the web page [3].

One type of DDoS-attack is a group consisting of thousands of devices

trying to connect to a single server. These devices almost always belong to a botnet so they can be controlled and set to attack to a target when wanted [6]. These days the peak rate of a DDoS-attack can be several terabits per second, for example Google has mitigated a DDoS-attack whose volume was over 2,5 Tb/s [7]. These days most of the attacks are amplified [8], and become DRDoS-attacks *Distributed Reflective Denial of Service*. The idea of DRDoS-attacks is to "amplify" the volume of the attack, often by exploiting a protocol, such as SSDP (*Simple Service Discovery*) and NTP (*Network time protocol*). The volume used does not originate from the attacker, but the attacker *reflects* the volume to the target server for example by spoofing the source IP- address sent to the service used for reflecting [8].

CDN-services can use Web Application Firewalls (WAF) to prevent attacks reaching the origin server because they can be used as a proxy. For example, it can be used for preventing XSS-attacks (Cross-Site Scripting), SQL-injections and other attacks at the application running [9]. Because the traffic is sent to edge server and if required, forwarded to the origin server by the edge server, they can be used for inspecting the packets using DPI (*Deep packet inspection*) [10].

2.3 Routing technologies

The routing is often thought to be from the closest location of the client to CDN-service's own network. This option is often more expensive because the bandwidth used in the service's own network costs as well. Other option is routing the traffic as little as possible in the CDN-service's own network, as close to the origin server. For example, Microsoft's Azure CDN-service can be configured to use either type of routing [11].

CDN-services can use Anycast-based IP-routing approach in their edge servers [12]. The idea is that a certain IP-address or a block is advertised in multiple locations, and while connecting, the shortest path to the edge server is chosen.

For web-traffic, different CDN-services might use different technologies for forwarding the traffic to origin server. For example, HTTP-redirection can be used [13] to forward requests to the origin server replicating the headers it received with the original request.

3 Attacking to CDN

Although CDN-services can be used for increasing security of the origin server, they cannot be considered as perfect options of protection from DDoS-attacks. There are methods of bypassing the given DDoS-attack protection in some cases due to the nature of CDN-services. However, CDN-services can be used for other forms of attacking the origin server.

3.1 Random String Denial Of Service

One method of bypassing CDN edge server is called *Random String Denial of Service* [3], which works by causing cache to miss the CDN edge server. Because the file does not exist in the edge server, the edge server tries to fetch it from the origin server and directs the load to the origin server [14]. Usually, static files are cached to edge servers of a CDN-service (for example `https://cdn.example.com/image1.png`). Random String Denial of Service works by adding a randomized string to the end of the URL (for instance `https://cdn.example.com/image1.png?id=1`).

Because the attack is abusing the connection between the CDN edge server and the origin server, the connection speed between the attacker and the edge server does not have to be fast. One of the solutions could be to have a real-time database of IP-addresses accessed the site [3]. If a certain IP-address tries to connect too frequently to a site, access to the resource can be blocked. Also, the edge server could throttle the speed towards to origin server to reduce the traffic in a short time [3].

3.2 Cache Poisoning Denial of Service

Cache poisoning works by sending malicious headers to the to the HTTP-query [15]. If the page has not been cached, the CDN edge server forwards the query to the origin server, and because the request contains additional headers, the origin web server responds with an error code. The edge server receives the error code and displays it for the clients trying to access to page. Because certain HTTP-request types (*DELETE*, *PUT*) are often forwarded to the origin server because the page functionality requires them, attacker can abuse them to cause a DoS-attack towards the server.

There are a few methods to defend against cache poisoning. For instance, a WAF can be set to block unnecessary request types. If that is

not possible, WAF could be set to detect anomalies in HTTP-headers, and block those requests, although the origin server has to do the filtering. Also, not caching error pages to CDN edge server can defend from this type of attack [3].

3.3 Port scanning using CDN-servers

CDN-services are often thought to connect to ports 80 and 443 (HTTP and HTTPS ports), providers can allow TCP-connections to other ports as well, making the CDN-service vulnerable for TCP port scanning of the origin server [16] [3]. The determination of whether the port is open, filtered or closed depends on the error code [16] and the body attacker receives in the response. One of issues of this behavior is that the origin server sees the scanning originating from the CDN-server. It might cause issues if the origin server tries to filter the traffic from the IP-address port scanning comes from.

The abuse of the information is not large because often the IP-address of the origin server is not known, it can still reveal some information about the origin server, and help the attacker to limit the number of probable targets. To protect the origin server from such attack, one solution could be to disable possibility to connect to unnecessary ports of the origin server (e.g., allowing 80 and 443 only). Also, the origin server is highly recommended to put behind a firewall, which could hide or filter unused ports.

3.4 XSS-attacks

The idea behind XSS-attacks is to override the origin policy [9]. By doing XSS-attacks, the attacker can inject malicious scripts to the site, which can affect other visitors (stored XSS-injections) [9]. Web application firewalls can be used for a protection from these attacks. Also, the structure of web applications must be made XSS-proof.

Because edge servers are often used for caching information from the origin server, they can accidentally be used for storing XSS-injections for other users for a longer time if the cache is not refreshed. edge servers can be used for protecting XSS-attacks.

4 Practical tests

Random string attack and cache poisoning can cause difficulties securing the origin server from DDoS-attacks. They are demonstrated in this section. The purpose is not to cause a large-scale DDoS-attack, but to see if these attacks work.

The CDN-provider chosen for these tests was Cloudflare [17]. Tests were run on a virtual machine running Ubuntu 23.04, and the web server run in the server is Nginx 1.24.0 unless stated otherwise. Cloudflare's free plan is used as a CDN service without further configuring. The only configurations done was adding the domain to Cloudflare and adding two A-records pointing to server. Another record was set to pass the traffic through Cloudflare's service and another one was pointing the server.

4.1 Random string attack

As mentioned in the subsection 3.1., random string attack is method used by adding a key with a random value to the end of an URL. Because the URL with the randomized string is not cached to the edge server, it has to fetch it from the origin server, which causes load to the server. Two virtual servers are used configured: one for accessing the server directly, and another for accessing server through Cloudflare's edge server.

To do the test without Cloudflare, a simple picture (*test.png*) was uploaded to */pictures/*, so the full was *<serverroot>/pictures/test.png*. The path for cached picture was *<serverroot>/cf-pictures/test.png*.

```
9H.237.14.6 - - "GET /pictures/test.png?cf=nocf HTTP/1.1" 200 5937 "-" "Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:109.0) Gecko/20100101 Firefox/119.0"
9H.237.14.6 - - "GET /pictures/test.png?cf=nocf HTTP/1.1" 304 0 "-" "Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:109.0) Gecko/20100101 Firefox/119.0"
9H.237.14.6 - - "GET /pictures/test.png?cf=false HTTP/1.1" 200 5937 "-" "Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:109.0) Gecko/20100101 Firefox/119.0"
9H.237.14.6 - - "GET /pictures/test.png?cf=nocf HTTP/1.1" 200 5937 "-" "Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:109.0) Gecko/20100101 Firefox/119.0"
```

Figure 2. Accessing the picture without CDN. Taken from Nginx's access log file (timestamps edited away).

As seen in the figure 2, each request to the picture is logged to the Nginx access log.

- The first line shows that the picture is fetched successfully (HTTP 200)
- The second line shows that the picture is fetched successfully, but it's cached in the client (HTTP 304)
- The third line shows that if the value of the key is changed, the picture is re-fetched from the server

- The fourth line shows that if client flushes their cache, the picture is fetched that it does not exist.

```
141.101.76.45 - -"GET /cf-pictures/test.png HTTP/1.1" 200 5937 "-" "Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:109.0) Gecko/20100101 Firefox/119.0"
172.70.46.151 - -"GET /cf-pictures/test.png?cf=true HTTP/1.1" 200 5937 "-" "Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:109.0) Gecko/20100101 Firefox/119.0"
172.71.94.127 - -"GET /cf-pictures/test.png?3 HTTP/1.1" 200 5937 "-" "Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:109.0) Gecko/20100101 Firefox/119.0"
```

Figure 3. Accessing the picture with CDN. Taken from Nginx’s access log file (timestamps edited away).

Security

WAF

Zone-level Web Application Firewall (WAF) detects and mitigates malicious requests across all traffic under this zone.

[WAF documentation](#)

Custom rules Rate limiting rules Managed rules Tools

We have updated **firewall rules** to more powerful **custom rules**. Some features work differently - [learn about what changed](#).

The **firewall rules** API is deprecated, but it will work until the sunset date to support any automation built on it. Refer to the [documentation](#) for details on migrating to the Rulesets API.

[← Back](#)

Create rule [About custom rules](#)

Rule name (required)

Give your rule a descriptive name.

If incoming requests match...

Field	Operator	Value	
URI Query Stri...	does not equal		And Or

e.g. page=1234

Expression Preview [Edit expression](#)

```
(http.request.uri.query ne "")
```

Then take action...

Choose action

Blocks matching requests and stops evaluating other rules

Figure 4. A WAF-rule used to block all queries in Cloudflare used for blocking queries in the end of URL

As seen in the figure 3, using re-accessing to the file does not appear in the log file (i.e. there is no HTTP 304). The reason is using Cloudflare as a caching service. However, it’s worth noting that using a key and variable after the end of the path directs the queries to the origin server, which suggests that Cloudflare’s CDN is vulnerable to random string attacks. Also, it’s the source IP-address differs from the ones in the figure 2. The reason is that instead of fetching the data directly, an edge server fetches it and serves it to the user.

However, a WAF-rule in Cloudflare can be used to block random ac-

cess queries. The following rule in Figure 4 blocks all the HTTP-traffic containing character `?` in the URL.

4.2 Cache poisoning

In this section cache poisoning is going to be tried as described in the section 3.2. The main idea is to cause issues so that due to an incorrect HTTP-method the site responses with an error code and therefore it does not show the page correctly but users see a cached version of the error page.

By default Nginx returns HTTP error code 405, which is also known as *"Method not allowed"*. This means that the web page is not configured to support the given request type.

Testing queries were done using cURL [18] with the following command: `curl -X PUT http://<domain-using-cloudflare>/new-path/test3.html`. The file contains the following information (in HTML): `<html>This is test3.html</html>`.

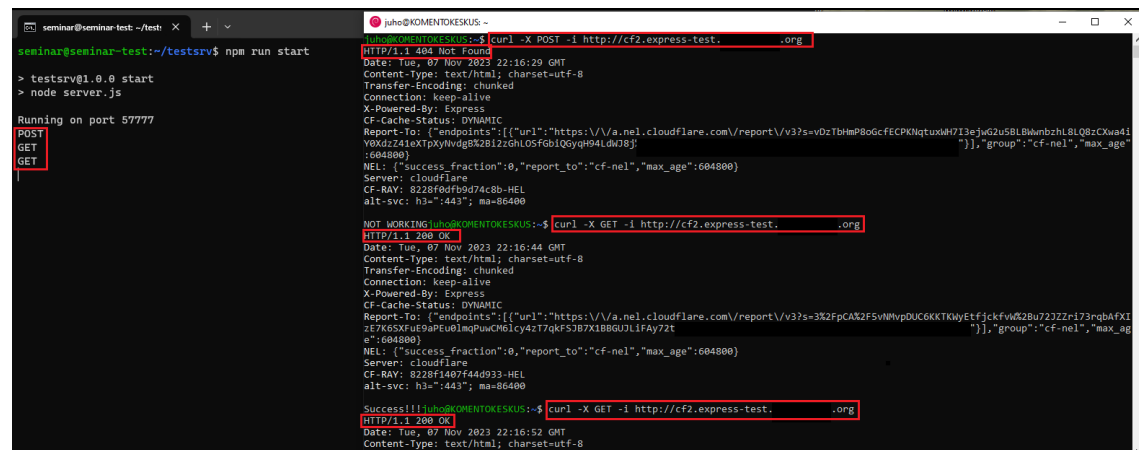
```
172.71.94.132 - - [30/Oct/2023:18:59:55 +0000] "PUT /new-path/test3.html HTTP/1.1" 405 166 "-" "curl/7.74.0"
172.71.94.47 - - [30/Oct/2023:19:00:10 +0000] "GET /new-path/test3.html HTTP/1.1" 200 55 "-" "Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:109.0) Gecko/20100101 Firefox/119.0"
172.71.94.47 - - [30/Oct/2023:19:00:17 +0000] "GET /new-path/test3.html HTTP/1.1" 200 55 "-" "Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:109.0) Gecko/20100101 Firefox/119.0"
172.71.94.47 - - [30/Oct/2023:19:00:23 +0000] "GET /new-path/test3.html HTTP/1.1" 200 55 "-" "Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:109.0) Gecko/20100101 Firefox/119.0"
172.71.94.47 - - [30/Oct/2023:19:00:28 +0000] "GET /new-path/test3.html HTTP/1.1" 200 55 "-" "Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:109.0) Gecko/20100101 Firefox/119.0"
```

Figure 5. Screenshot from Nginx's log file after send PUT-request to file and then fetching it four times using a GET-request

The PUT-request went through Cloudflare as expected. Although the PUT-request responded with HTTP error code 405, the page was not replaced with the error page and showed normally, as expected based on previous texts [15]. However, Cloudflare did not cache the page. As seen in the Figure 4, each new GET-request to `/new-path/test3.html` was fetched again from the origin server. This is quite interesting situation - because even though the content was not modified after the PUT-request, during each request Cloudflare fetches the page and does not seem to cache it. This kind of behavior may expose the origin server to DDoS-attacks. If it is known that creating a query which the server responses by HTTP error code 405 allows the caching to be bypassed, an attacker can send "usual" HTTP GET-requests to the web page and the edge server fetches the content each time.

Cloudflare has had issues related to cache-poisoning if the page returns HTTP error code 404 when sending HTTP-request [15]. The error code is used to indicate that the path does not exist on the server. To test this, a simple Express web-application was created which prints "GET" to

console and returns response with HTTP-code 200 if the server receives a GET-request. If a POST-request is received, server outputs "POST" to console, and returns HTTP-code 404 to client.



```
seminar@seminar-test: ~/test
seminar@seminar-test:~/test$ npm run start
> testserver@1.0.0 start
> node server.js

Running on port 57777
POST
GET
GET

jsho@KOMMENTOKESKUS:~$ curl -X POST -i http://cf2.express-test.
HTTP/1.1 404 Not Found
Date: Tue, 07 Nov 2023 22:16:29 GMT
Content-Type: text/html; charset=utf-8
Transfer-Encoding: chunked
Connection: keep-alive
X-Powered-By: Express
CF-Cache-Status: DYNAMIC
Report-To: [{"endpoints":[{"url":"https://va.nel.cloudflare.com/v/report/v3?s=y0zTbHhP8oGcFECPKNqtuxwH713eJw62u58Lbhmzbzl8L08:CKwa4I
Y8kdz241eXtpXyWdg8K28122GH05Fgl0GyqH94LdW38J"}]}], "group": "cf-nel", "max_age":
:604800}
NEL: {"success_fraction":0,"report_to":"cf-nel","max_age":604800}
Server: cloudflare
CF-RAY: 8228f0dfb9d74c8b-HEL
alt-svc: h3=":443"; ma=86400

NOT WORKING jsho@KOMMENTOKESKUS:~$ curl -X GET -i http://cf2.express-test.
HTTP/1.1 200 OK
Date: Tue, 07 Nov 2023 22:16:44 GMT
Content-Type: text/html; charset=utf-8
Transfer-Encoding: chunked
Connection: keep-alive
X-Powered-By: Express
CF-Cache-Status: DYNAMIC
Report-To: [{"endpoints":[{"url":"https://va.nel.cloudflare.com/v/report/v3?s=3X2FpCAk2F5wNwPDUc6KkTKWYEtFjckfw428u7232Zr173rqb4fX1
2E7K6SXfu9aEu81mqPuwCh0icy4217qkF53B7X188GUJ1Fay72t"}]}], "group": "cf-nel", "max_age":
:604800}
NEL: {"success_fraction":0,"report_to":"cf-nel","max_age":604800}
Server: cloudflare
CF-RAY: 8228f1487f44d933-HEL
alt-svc: h3=":443"; ma=86400

Success!!! jsho@KOMMENTOKESKUS:~$ curl -X GET -i http://cf2.express-test.
HTTP/1.1 200 OK
Date: Tue, 07 Nov 2023 22:16:52 GMT
Content-Type: text/html; charset=utf-8
```

Figure 6. Screenshot from the server console and sent requests to the server

As seen in the figure 6, the first sent POST-request receives HTTP-code 404 and GET-requests after that receive HTTP-code 200 fine, so the cache has not been *poisoned*. However, Cloudflare seems to have issues caching the GET-page, like with Nginx for some reason. The page is fetched two times with two GET-requests in total, which implicates that the edge server does not cache it. Also, it is worth noting that the domain hadn't been visited before requests done in Figure 6 so Cloudflare could not have cached it before the requests.

4.3 Discussion about the results

Considering the nature of the random string attack, it was not surprising that Cloudflare did not block the attack by default. However, Cloudflare's free plan has an option to set up a WAF-rule to block or limit queries used for random string attacks, or whitelist only the required ones, which was a surprise. If used properly, it will stop the attacking queries to the edge servers. However, the WAF-feature has to be enabled separately, so it is not automatically enabled. Also, the user is not necessarily informed about it.

However, the result related to cache-poisoning was a surprise. Even though the exploit did not work like it was supposed to work (responding with a cached 404-page instead of the real content), Cloudflare's edge servers still fetched the content from the origin server. This caused traffic between the edge server and the origin server, which can be used for causing a DDoS-attack towards the origin server. Cloudflare allows creating

a WAF-rule to allow/disallow different request types, but limiting access to the web page based on the methods can be difficult: what if different requests, like PUT- or POST-requests are required for the web page to functional? Limiting all other HTTP-requests except GET-requests would not be a solution then. The situation itself is admittedly quite difficult – these days web pages usually contain other type of requests than just GET-requests.

An improvement to the cache-poisoning configuration could adding a layer between the edge server and the origin server, such as a firewall to block traffic if it detects malicious behavior. Also, the used CDN-service could be configured that once a certain amount of traffic is configured, it would act as a limiter by blocking traffic unless a CAPTCHA [19] is resolved to prevent machine-created traffic accessing the origin server.

5 Conclusion

This paper gave introduction to CDN-services in general and explained how CDN-services can be used for attacking web services. Practical demonstrations were given for random string attacks and attempt for cache-poisoning, and after the demonstration, there was discussion about the results. In practical testing Cloudflare was used as a CDN-service. Even though random string attacks are known to be problematic for CDN-services, Cloudflare turned out to be vulnerable without further configuration. Regarding to cache-poisoning, Cloudflare did not perform as before, but strange behavior was encountered because pages were not cached.

CDN-services can often be used for improving the performance of web applications. They can be used for improving security as well. However, they should not be used as all-in-one protections because they have their own issues. If their issues are understood and their vulnerabilities, such as random string attacks, are limited somehow, CDN-services can be used to improve the performance and the security.

References

- [1] T. Barnett Jr, S. Jain, U. Andra, and T. Khurana, “Cisco visual networking index (vni) complete forecast update 2017–2022. december 2018,” *Accessed in April*, 2018.
- [2] “Content delivery network.” Image: <https://nexnetsolutions.com/wp->

content/uploads/2021/08/Content-Delivery-1536x864.png, article:
<https://nexnetsolutions.com/solutions/telecom-service-provider/content-delivery-network-cdn/>.

- [3] M. Ghaznavi, E. Jalalpour, M. A. Salahuddin, R. Boutaba, D. Migault, and S. Preda, "Content delivery network security: A survey," *IEEE Communications Surveys Tutorials*, vol. 23, no. 4, pp. 2166–2190, 2021.
- [4] A. Binder and I. Kotuliak, "Content delivery network interconnect: Practical experience," in *2013 IEEE 11th International Conference on Emerging eLearning Technologies and Applications (ICETA)*, pp. 29–33, 2013.
- [5] F. Lau, S. Rubin, M. Smith, and L. Trajkovic, "Distributed denial of service attacks," in *Smc 2000 conference proceedings. 2000 IEEE international conference on systems, man and cybernetics. 'cybernetics evolving to systems, humans, organizations, and their complex interactions' (cat. no.0, vol. 3, pp. 2275–2280 vol.3, 2000.*
- [6] S. T. Zargar, J. Joshi, and D. Tipper, "A survey of defense mechanisms against distributed denial of service (ddos) flooding attacks," *IEEE Communications Surveys Tutorials*, vol. 15, no. 4, pp. 2046–2069, 2013.
- [7] D. Menscher, "Exponential growth in ddos attack volumes," 2020.
- [8] C. Rossow, "Amplification hell: Revisiting network protocols for ddos abuse.," in *NDSS*, pp. 1–15, 2014.
- [9] J. P. N, N. Ravishankar, M. Raju, and N. C. S. Vyuha, "Secure software immune receptors from sql injection and cross site scripting attacks in content delivery network web applications," in *2021 9th International Conference on Reliability, Infocom Technologies and Optimization (Trends and Future Directions) (ICRITO)*, pp. 1–5, 2021.
- [10] A. Razzaq, A. Hur, S. Shahbaz, M. Masood, and H. F. Ahmad, "Critical analysis on web application firewall solutions," in *2013 IEEE Eleventh International Symposium on Autonomous Decentralized Systems (ISADS)*, pp. 1–6, 2013.
- [11] "Routing preference in azure - azure virtual network." <https://learn.microsoft.com/en-us/azure/virtual-network/ip-services/routing-preference-overview>.
- [12] S. Hao, Y. Zhang, H. Wang, and A. Stavrou, "End-Users get maneuvered: Empirical analysis of redirection hijacking in content delivery networks," in *27th USENIX Security Symposium (USENIX Security 18)*, (Baltimore, MD), pp. 1129–1145, USENIX Association, Aug. 2018.
- [13] A.-M. K. Pathan, R. Buyya, *et al.*, "A taxonomy and survey of content delivery networks," *Grid Computing and Distributed Systems Laboratory, University of Melbourne, Technical Report*, p. 44, 2007.
- [14] S. Triukose, Z. Al-Qudah, and M. Rabinovich, "Content delivery networks: Protection or threat?," in *Computer Security – ESORICS 2009* (M. Backes and P. Ning, eds.), (Berlin, Heidelberg), pp. 371–389, Springer Berlin Heidelberg, 2009.

- [15] H. V. Nguyen, L. L. Iacono, and H. Federrath, “Your cache has fallen: Cache-poisoned denial-of-service attack,” in *Proceedings of the 2019 ACM SIGSAC Conference on Computer and Communications Security, CCS ’19*, (New York, NY, USA), p. 1915–1936, Association for Computing Machinery, 2019.
- [16] R. Guo, J. Chen, B. Liu, J. Zhang, C. Zhang, H. Duan, T. Wan, J. Jiang, S. Hao, and Y. Jia, “Abusing cdns for fun and profit: Security issues in cdns’ origin validation,” in *2018 IEEE 37th Symposium on Reliable Distributed Systems (SRDS)*, pp. 1–10, 2018.
- [17] “Cloudflare.” <https://www.cloudflare.com/>.
- [18] “curl.” <https://curl.se/>.
- [19] M. Moradi and M. Keyvanpour, “Captcha and its alternatives: A review,” *Security and Communication Networks*, vol. 8, no. 12, pp. 2135–2156, 2015.

Exploiting HTTP/2: Investigating Attack Strategies against Content Delivery Networks

Jussi Impiö

jussi.impio@aalto.fi

Tutor: Jose Luis Martin Navarro

Abstract

This paper investigates known attacks against Content Delivery Networks (CDNs) arising from adopting HTTP/2. It focuses on how HTTP/2 can be exploited for denial-of-service (DoS) attacks, detailing various attack methodologies, such as amplification and slow-rate attacks. Additionally, the paper discusses the ethical implications and responsible disclosure practices in cybersecurity research related to CDNs.

KEYWORDS: content delivery network, HTTP/2, denial-of-service, amplification attack, slow-rate attack

1 Introduction

Content Delivery Networks (CDNs) play a crucial role in Internet infrastructure. CDNs efficiently deliver internet traffic by distributing content, optimizing load times, ensuring high availability, and providing security measures, making them vital for enhancing user experience and demand for online services. These attributes make CDNs very popular.

Different approaches exist to how CDNs can be utilized [1]. One type of CDN utilization involves CDN-based edge servers first requesting data from the origin server and storing the content of the website in a cache

before serving it to clients. Consequently, when clients seek the website, they are directed to the nearest CDN edge server, which provides the cached content. As illustrated in Figure 1, this results in a connection scenario: one between the client and the CDN edge server, as well as another between the CDN and the origin server. The interaction between the edge and the origin is the "CDN-origin" connection, while the communication between the CDN and the client is the "client-CDN" connection [2] [1].

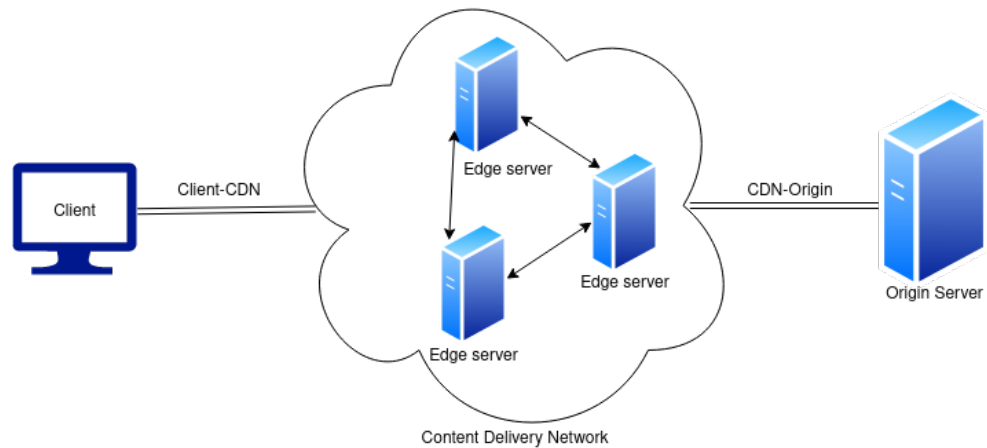


Figure 1. Illustration of Content Delivery Network interaction

Due to their critical role and the vast attack surface they expose due to their complexity, CDNs are vulnerable to security attacks and exploitation. These attacks may target the CDN infrastructure, the online service providers relying on the CDN, or the end-users who access these online services [1]. Given the critical significance of CDNs, CDN providers want to address vulnerabilities immediately. Consequently, the research may rapidly become outdated, as the attacks continually evolve.

Through a literature review, this paper aims to elucidate the current landscape of CDN security, focusing on HTTP/2 attacks. The research entailed an analysis of recent peer-reviewed journals, conference proceedings, and industry reports, selecting sources based on their relevance to CDN security and HTTP/2. Searches across databases, including IEEE Xplore, ACM Digital Library, and Google Scholar, utilized targeted keywords to gather relevant literature. Additionally, the review assesses ethical considerations in the literature.

This paper is organized as follows: Section 2 introduces relevant background information. Section 3 outlines methods by which HTTP/2 is lever-

aged in attacks against CDNs. Section 4 discusses ethical considerations and disclosures found in the referenced papers. Section 5 provides a discussion of the findings of this paper. Finally, Section 6 offers concluding remarks. ,

2 Background

2.1 Content Delivery Networks

Content Delivery Networks (CDNs) are a series of strategically dispersed servers designed to optimize the delivery of digital content to end-users. Their foundational purpose is to bridge the potential geographical gap between the source of web content (often referred to as the origin server, see Figure 1) and the end user, ensuring that content, such as web pages, videos, images, and other web resources, are delivered efficiently and swiftly [3] [4].

The growing complexity of web traffic has driven the evolution of CDNs. As the internet has become more ubiquitous, it became evident that serving users from a single, centralized server could lead to bottlenecks, especially when many users try accessing the same content simultaneously [5] [4]. CDNs emerged as a strategic solution to these challenges by redistributing the delivery load.

Functionally, when a user requests content, such as a video or an image, the request does not go directly to the origin server but is instead redirected to the nearest edge server in the CDN. This "nearest" server is selected based on algorithms considering proximity and server health [5] [4]. If the edge server already has the content cached, it delivers it to the user; otherwise, the CDN retrieves it from the origin server or another edge server that has cached the content. This approach reduces latency and ensures high availability and redundancy [3].

CDNs, however, are not just about speed. As security threats have become more sophisticated, CDNs have evolved to incorporate security measures. For instance, they can help mitigate threats by distributing traffic, making it harder for malicious actors to target a single point in the network. CDNs distributed nature provides a formidable defense against Distributed Denial of Service (DDoS) attacks. Additionally, many CDNs come equipped with Web Application Firewalls (WAFs) and other secu-

rity features to scrutinize and filter incoming traffic, thereby providing a layered shield against malicious activities [1].

2.2 CDN Security Challenges

Ghaznavi et al.[1] discuss the various security challenges faced by CDNs, categorizing them into three main components: edge servers, routing, and origin servers. All of these components have their own vulnerabilities and attack vectors.

Edge Servers

Edge servers play a pivotal role in CDNs, managing caching and content delivery to users. Ghaznavi et al. [1] identify several security challenges for edge servers, including application layer threats, vulnerabilities in caching mechanisms, susceptibility to DoS attacks, and risks of covert channel threats. These challenges necessitate distinct and robust security measures to ensure edge server integrity and resilience.

Origin Servers

Origin servers, which store the original content distributed by CDNs, also face security risks. The intermediary role of CDNs can lead to man-in-the-middle scenarios, potentially compromising SSL/TLS encryption. This split in secure sessions lets CDNs access unencrypted content, leading to concerns about possible data interception or alteration without adequate security measures. Additionally, malicious CDNs could exploit their position to collect IP addresses or circumvent geographical restrictions [2] [1].

Request Routing

Request routing within CDNs introduces security challenges, as outlined by Ghaznavi et al. [1]. Attackers can exploit vulnerabilities to redirect traffic or bypass security measures. These tactics include ingress and egress harvesting, targeting user data entering or leaving the CDN. Additionally, HTTP smuggling and multiple host ambiguities can lead to attacks, such as cross-site scripting or cache poisoning, by exploiting inconsistencies in HTTP request interpretation. DoS attacks, mainly through CDN network abuse for traffic amplification, pose a significant threat. The paper later discusses the utilization of HTTP/2 in CDNs for orchestrating DoS attacks.

2.3 HTTP/2

Hypertext Transfer Protocol (HTTP) is the foundation of data communication on the World Wide Web, facilitating the exchange of information between web browsers and servers. Its development has been pivotal in the evolution of the internet, from the early HTTP/0.9 to subsequent versions, HTTP/1.0 and HTTP/1.1, culminating in the more recent HTTP/2 and HTTP/3. Each version of HTTP has refined and expanded the capabilities of the protocol, catering to the growing demands of web communication [6].

The HTTP/1.1 protocol, widely used on websites, supports client-server communication through request-response exchanges. In this version, each resource request necessitates a new Transmission Control Protocol (TCP) connection, a process that becomes inefficient for web pages with many assets [6]. To address these inefficiencies, HTTP/2 introduces significant improvements for data transmission between clients and servers, such as multiplexing streams, server push events, and flow control mechanisms [6] [7].

Moreover, HTTP/2 uses frames, such as WINDOW_UPDATE and SETTINGS, to manage and adjust window sizes and optimize the data transmission process [7]. This advancement in HTTP/2 over its predecessor, HTTP/1.1, addresses the challenges of the older protocol and offers a more efficient, streamlined approach to web data transmission.

3 Leveraging HTTP/2 against CDNs

In their papers, both Song et al.[8], and Guo et al.[9] survey popular CDN vendors, analyzing their support for HTTP/2 in their networks. Their papers show that most CDN vendors support at least HTTP/2 between the client and edge server but not for the connection between the edge server and the origin. Adversaries can abuse this protocol change [8] [9]. This section will explore how adversaries can leverage HTTP/2 in CDNs to cause DoS attacks.

3.1 HTTP/2 Bandwidth Amplification Attack

Amplification attacks, a prevalent form of DDoS, exploit bandwidth discrepancies between attackers and origin servers. In the context of CDNs handling HTTP/2 traffic, these attacks are particularly concerning under the Ignore and Follow policies. The Ignore policy leads to the CDN overlooking HTTP/2 flow control, while the 'Follow' policy aligns the CDN flow control with the CDN-origin TCP connection. Attackers leverage these policies to initiate amplification attacks, reducing the receive window size and inundating the CDN with simultaneous HTTP/2 requests, significantly unbalancing traffic between the CDN-origin and client-CDN connections [8].

Executing a DoS amplification attack in CDNs via HTTP/2 involves several steps. Initially, the attacker sets the `SETTINGS_INITIAL_WINDOW_SIZE` parameter in the `SETTINGS` frame to 0 and sends a `HEADERS` frame to request resources through the CDN. This setting prevents the reception of any `DATA` frames. To bypass CDN caching, the attacker appends a random query string to the URL, prompting the CDN to fetch the resource using the HTTP/1.1 protocol [8].

Subsequent actions depend on the policy of CDN, either Ignore or Follow, affecting how much of the resource is retrieved. With the attacker's receive window at zero, the CDN is limited to returning only the `HEADERS` frame, leading to a substantial discrepancy in traffic volume between the client-CDN and CDN-origin connections, resulting in traffic amplification. Additionally, the exploitation of the multiplexing feature of HTTP/2 further amplifies traffic, as the bandwidth needed for a `HEADERS` frame on the client-CDN connection is significantly less than that required for a three-way handshake in a CDN-origin TCP connection. This disparity grows with increased concurrent streams [8].

Another aspect enhancing the potential for amplification attacks is using header compression of HTTP/2 (HPACK) encoding. The HPACK compression allows attackers to enlarge the size of response headers significantly. Attackers can craft requests with oversized header fields, exploiting the compression of the HPACK mechanism to create large headers that expand considerably upon decompression by the CDN. When the CDN decompresses these, they expand back to their original size. Such an approach is efficient with custom or commonly large headers, such as cookies or user-agent strings [9].

3.2 Slow rate attacks

As described by Song et al. [8], slow-rate attacks exploit the TCP kept-open time of the connection by manipulating the TCP receive window size. These attacks monopolize numerous TCP connections for extended periods, using up all available connections and disrupting regular service operations.

Adopting the Follow policy in a CDN allows adversaries to manipulate the WINDOW_UPDATE frame. As illustrated in Figure 2, this manipulation controls the duration for which the CDN-origin connection remains open. The attacker achieves this by continuously sending multiple WINDOW_UPDATE frames in a stream and setting the WINDOW_SIZE_INCREMENT parameter to a small value. Such a tactic reduces the speed at which DATA frames are transmitted on the client-CDN connection. Concurrently, the CDN might delay or regulate the speed at which it receives response data from the origin server. This method enables the attacker to indirectly extend the kept-open time of each CDN-origin connection. Consequently, an attacker flooding a CDN with numerous resource requests can deplete all available connections to the origin server, leading to significant service interruptions [8].

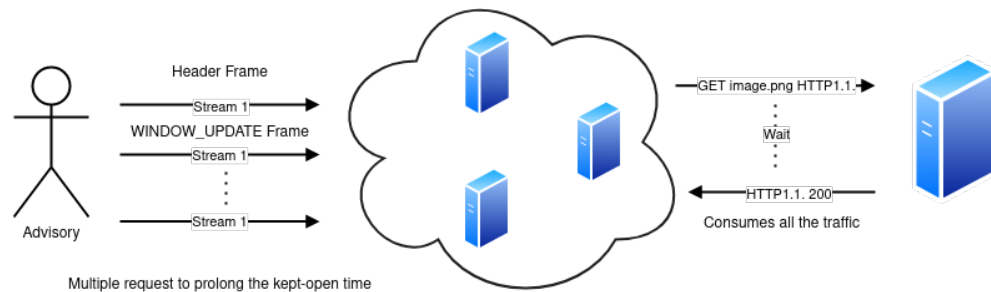


Figure 2. Illustration of a Slow-Rate Attack

Slow-POST Attack

Within the spectrum of slow-rate attacks, Slow-POST attacks, as elucidated by Guo et al.[9], are particularly impactful against CDNs. These attacks exploit processing of the HTTP POST request and forwarding mechanisms within CDN infrastructures, leading to significant service disruptions. The attacker initiates a POST request with a large declared content size in the Content-Length header and transmits the data slowly. This tactic purposefully extends the TCP connection duration, straining

the capacity of the server to its limits. The effectiveness of Slow-POST attacks stems from how certain CDNs handle the forwarding of POST requests. The vulnerability lies in CDNs that forward POST requests to the origin server immediately upon receiving just the POST header instead of waiting for the entire POST body. This premature forwarding mechanism is exploited by attackers to prolong connections between the CDN and the origin server, resulting in resource exhaustion and denial of service for legitimate requests [9]. The research of Guo et al.[9] also highlights the variability in CDN responses to Slow-POST attacks. Their experiments across various CDNs reveal differences in handling POST requests, with some CDNs, such as CloudFront and Fastly, starting the forwarding process upon receiving only the POST header. This variation signifies an exploitable weakness in CDNs that choose immediate forwarding over a more cautious, buffered approach.

4 Ethicality of papers

A notable aspect of contemporary research on Content Delivery Networks is the consistent attention to ethics and responsibility, particularly in handling disclosures and potential impacts of the findings. Studies in this domain, including those by Guo et al. [9] and Song et al. [8], demonstrate a commendable level of ethical awareness and responsibility.

In their research on CDNs, Guo et al. [9] and Song et al. [8] conducted their experiments using their origin servers, thereby avoiding any negative impact on public web services or user data. This method of operation reflects a careful and responsible approach to research in the field.

Both teams also displayed a commitment to responsible disclosure of their findings. They reported vulnerabilities to the involved CDN providers well before publishing their research. This approach to responsible disclosure is essential, as premature public disclosure could lead to unintended security risks. By communicating first with the CDN providers, Guo et al.[9] and Song et al.[8] ensured secure addressing of potential vulnerabilities.

5 Discussion

This paper reviews recent literature, particularly focusing on studies by Guo et al.[9] and Song et al.[8], which reveal significant insights into leveraging HTTP/2 against CDNs. These recent studies highlight the rapid evolution in this field¹.

5.1 Summary of Key Findings

In their research, both Guo et al.[9] and Song et al.[8] explore denial of service attacks, which align with the request routing security challenges identified by Ghaznavi et al. [1]. Their findings indicate that while HTTP/2-based denial-of-service attacks threaten origin servers, CDNs generally demonstrate resilience against these attacks. This resilience is attributed to proper configuration and effective mitigation strategies implemented on the origin servers. Despite this, the increasing deployment of HTTP/2 suggests a potential rise in DoS attacks, underscoring the importance of continuous vigilance and the development of adaptive security strategies. Interestingly, CDN providers showed little concern regarding the impact of leveraging HTTP/2 against CDNs and origin servers, reflected in the modest response to the researchers' disclosure of these vulnerabilities. Furthermore, the possibility of combining amplification strategies presented in this paper could pose a more significant denial-of-services. By manipulating the `SETTINGS_INITIAL_WINDOW_SIZE` parameter with the multiplexing feature of HTTP/2, attackers could control data transmission rates and inflate data sizes. This tactic can lead to a compounded amplification effect, significantly damaging the resources of the origin server. Such an approach could result in severe performance degradation or a total denial-of-service, with the attacker requiring only minimal data transmission to the CDN.

5.2 Future directions

As the adaptation of HTTP/3 commences, it presents opportunities for similar attacks against CDNs. However, the current body of research does not include papers directly leveraging HTTP/3 in the context of CDN attacks. This lack of specific research on HTTP/3 and CDNs indicates an area ripe for future exploration.

¹<https://nvd.nist.gov/vuln/detail/CVE-2023-44487> | Accessed: 2023-11-07

Additionally, while there is existing literature on the proxy conversion from HTTP/2 to HTTP/1 [10] [11] [12], there remains a noticeable gap in studies specifically addressing these protocols concerning CDN environments. This gap suggests a need for more targeted research to understand the implications of these newer protocols on CDN security.

Interestingly, while blog posts and informal articles provided by CDN providers addressing various CDN attacks are abundant, there is a noticeable scarcity of scientific papers on the subject. This disparity highlights a gap between industry knowledge and academic research. Blog posts from CDN providers offer valuable insights and practical solutions based on real-world experiences, yet they often lack rigorous analysis and peer-reviewed validation in scientific literature. The relative paucity of academic papers on CDN attacks points to a potential area for further research, where the depth and rigor of academic study can complement the practical understanding provided by industry professionals.

6 Conclusion

The reviewed literature provides valuable insights into the current state of CDN security, particularly in the context of HTTP/2. While current CDN configurations appear robust against specific attacks, the fast-evolving nature of internet protocols and cyber threats necessitates continuous research and adaptation of CDN security measures. This will be crucial in maintaining the security and robustness of CDN and origin server infrastructures in the face of new vulnerabilities and attack strategies.

References

- [1] M. Ghaznavi, E. Jalalpour, M. A. Salahuddin, R. Boutaba, D. Migault, and S. Preda, "Content Delivery Network Security: A Survey," *IEEE Communications Surveys & Tutorials*, vol. 23, no. 4, pp. 2166–2190, 2021.
- [2] B. Shobiri, M. Mannan, and A. Youssef, "CDNs' Dark Side: Security Problems in CDN-to-Origin Connections," *Digital Threats: Research and Practice*, vol. 4, pp. 3:1–3:22, Mar. 2023.
- [3] D. Robinson, *Content Delivery Networks: Fundamentals, Design, and Evolution*. Newark, UNITED STATES: John Wiley & Sons, Incorporated, 2017.
- [4] G. Carofiglio, G. Morabito, L. Muscariello, I. Solis, and M. Varvello, "From content delivery today to information centric networking," *Computer Networks*, vol. 57, pp. 3116–3127, Nov. 2013.
- [5] G. Peng, "CDN: Content Distribution Network," Nov. 2004.
- [6] S. Ludin and J. Garza, *Learning HTTP/2: A Practical Guide for Beginners*. Sebastopol, UNITED STATES: O'Reilly Media, Incorporated, 2017.
- [7] M. Belshe, R. Peon, and M. Thomson, "Hypertext Transfer Protocol Version 2 (HTTP/2)," Request for Comments RFC 7540, Internet Engineering Task Force, May 2015.
- [8] H. Song, J. Liu, J. Yang, X. Lei, and G. Xue, "Two Types of Novel DoS Attacks Against CDNs Based on HTTP/2 Flow Control Mechanism," in *Computer Security – ESORICS 2022* (V. Atluri, R. Di Pietro, C. D. Jensen, and W. Meng, eds.), Lecture Notes in Computer Science, (Cham), pp. 467–487, Springer International Publishing, 2022.
- [9] R. Guo, W. Li, B. Liu, S. Hao, J. Zhang, H. Duan, K. Sheng, J. Chen, and Y. Liu, "CDN Judo: Breaking the CDN DoS Protection with Itself," in *Proceedings 2020 Network and Distributed System Security Symposium*, (San Diego, CA), Internet Society, 2020.
- [10] B. Jabiyev, S. Sprecher, A. Gavazzi, T. Innocenti, K. Onarlioglu, and E. Kirda, "FRAMESHIFTER: Security Implications of HTTP/2-to-HTTP/1 Conversion Anomalies," pp. 1061–1075, 2022.
- [11] D. Beckett and S. Sezer, "HTTP/2 Tsunami: Investigating HTTP/2 Proxy Amplification DDoS Attacks," 2017.
- [12] S. Ismail, H. R. Hassen, M. Just, and H. Zantout, "A review of amplification-based distributed denial of service attacks and their mitigation," *Computers & Security*, vol. 109, p. 102380, Oct. 2021.

Microservices - when and how to use them

Jussi Vaitomaa

jussi.vaitomaa@aalto.fi

Tutor: Antti Ylä-Jääski

Abstract

Microservice architecture is a popular but complex architectural pattern. Implementing the architecture requires heavy understanding of its benefits and challenges, but the possibility of a scalable, cloud-native application drives many companies to adapt it. This paper provides an overview of microservice architecture, discussing factors contributing to successfully implementing it.

KEYWORDS: Microservices, MSA, cloud computing

1 Introduction

Microservices, an architectural pattern which has grown in popularity during the last decade, is continuously applied to many different types of software systems. Microservice architecture (MSA) promotes modularization and division of a larger solution into distributed, individual systems, which are easier to manage [1]. The current state of containerization technologies combined with the capabilities of cloud computing provide an even stronger incentive to adapt MSA [2].

However, due to the popularity of MSA, it is inevitable that it is oc-

asionally implemented for the wrong reasons [3]. Similarly to other architectural models, there are numerous challenges in designing and implementing a functional MSA [4]. Compared to monolithic approaches to software design, MSA requires a more complex deployment model, incurring costs and resources needed to implement it [3]. Therefore, it is important that the requirements for MSA are thoroughly evaluated before attempting to migrate an existing system or building an MSA from the beginning. The potential benefits of MSA can be greatly reduced, if it is implemented incorrectly or for the wrong reasons.

The aim of this paper is to provide an overview of the architectural pattern, covering common methods on how microservices are developed, deployed and managed. This paper also discusses factors which should be accounted for when deciding if MSA is suitable for a specific software solution.

This paper is organized as follows. Section 2 provides a background for microservices, discussing service-oriented and monolithic architectures. Section 3 covers the microservice architecture, its advantages and difficulties. Section 4 discusses how different properties of the microservice architecture account to the validity of the architecture as a choice.

2 Monoliths and service-oriented architecture

This section describes monolith architecture and service-oriented architecture.

2.1 Monoliths

Dragoni et al. [4] defined monoliths as software applications whose modules cannot be executed independently. Since each module is implemented as part of a larger application, the deployment of a monolith is typically a single, standalone program [5].

Because monolith architecture (MA) relies on a single program, a project utilizing it is likely to have a large and complex code base. Increasing the size and complexity of the monolith makes it harder for developers to work on the application: maintainability, dependency management, deployment, and scalability are all increasingly difficult for larger monoliths [4]. However, a well-designed monolith and good software development practices remedies most of these issues [3].

Recently, companies have been looking to migrate from monolithic applications to alternative architectural models [3]. One of the driving factors is that MA is not completely compatible with cloud computing [6]. MA is not suitable for incremental development of large-scale systems, in addition to being hard to scale efficiently [3].

2.2 Service-oriented architecture

For large-scale systems, service-oriented architecture (SOA) has proven suitable to remedy the issues of monolithic architecture [2]. The basic principle behind service-oriented architecture is the division of application modules into different cohesive services. The services communicate over a network and co-operate to complete business processes.

Services in SOA comprise of two parts: interface and implementation [7]. The interfaces can be shared with discoverable web service standards, for example the Web Service Description Language (WSDL), to enable communication with other services. This makes the implementation details of services ambiguous, and services can be implemented utilizing any technologies supporting the standardized interfaces. However, allowing services to communicate directly with each other increases coupling, which is why most SOA systems include an integration layer, separating the services from each other.

In practice, maintainable SOA requires a centralized component to provide smart routing and integration [2]. Services in SOA are not coupled, and should not directly communicate with other services. The centralized component manages the integration of services, which enables building business processes on the integration layer. The centralized component is often an enterprise service bus, a message broker that supports the web service standards used to describe service interfaces.

Despite the fact that each service can be developed individually, the integration component may cause dependencies in deploying the different services [2]. In the worst case, this results in the entire system acting similarly to a monolith application.

3 Microservices

This section describes the microservice architecture (MSA), its advantages and difficulties.

3.1 Common properties

Microservices are individual network-accessible systems, implementing a specific functionality [1]. Because individual microservices are responsible for performing closely related tasks, they have high cohesion and are often small in size.

In MSA, a software system is built utilizing microservices. There is no strict definition for microservices, and it is debatable whether MSA is a subtype of SOA or not [2]. However, similarly to SOA, the goal of MSA is to create independent and distributed services [1]. Cerny et al. differentiate MSA from SOA with microservices having a higher autonomy [2].

In a well-designed MSA, the microservices are loosely coupled, meaning that they are least dependant on each other [8]. Loose coupling is an important property, because it prevents changes in one microservice propagating to multiple other services.

Communication

There are two ways to organize communication in a multiple-service system: orchestration and choreography [2]. Orchestration refers to the use of a centralized component, which oversees and integrates services. The services do not directly communicate with each other, but messages from one service to another are managed by the orchestrator. On the other hand, choreography lacks a centralized component, and there is no single point for all control logic. This is typical for MSA, where the autonomy of services is important.

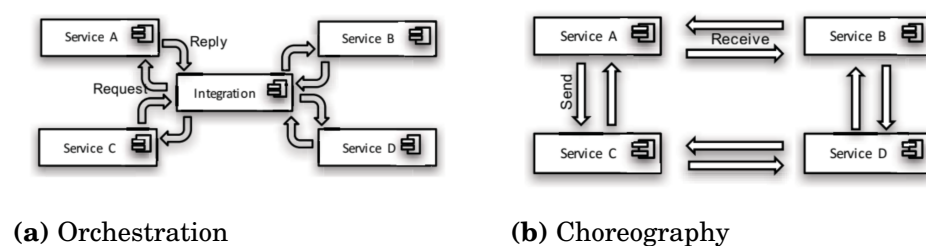


Figure 1. Interaction patterns. Figures from [2]

In [9], Aksakalli et al. conducted a systematic literature review on deployment and communication patterns in MSA. Numerous communication patterns for microservices were identified. Synchronous communication based on request/response is suitable for fetching data, but it increases coupling and service availability strategies must be enforced. On the other hand, asynchronous communication via message queues

or message-oriented middleware uncouples services, and can be used for event-driven systems. Direct point-to-point communication is risky, as it can result in overly complex dependencies among services. Communication patterns are often adopted individually on a requirement basis, hence a combination of approaches can be leveraged for optimized communication. To further optimize communication, binary protocols such as Apache Thrift or Google's protobuf can be implemented for well-defined interfaces.

Data persistence

Data persistence is an important aspect of software architecture. For distributed services, the services can either share the same persistence solution (by shared databases or tables), or independently manage their own.

In [8], Mishra et al. compare shared and independent approaches to databases for MSA. To facilitate loose coupling of microservices, it is better to provide each service with an independent view of data. Having individual databases for each microservice may result in duplication of data, but it prevents changes in a database schema requiring modifications to multiple microservices. Additionally, the choice of database can be optimized for a specific microservice, because other services are not dependant of it. However, independent databases do not support traditional transactions, requiring an alternative system to ensure data consistency.

Container orchestration and service mesh

Microservices are containerized to form a uniform approach to their management [10]. Furthermore, to automatically manage the containers, orchestration systems such as Kubernetes can be utilized. Container orchestration systems monitor and manage the state of containers, and can be configured to dynamically scale containers or handle failures.

Containerization and container orchestration provide tools for efficient development and deployment of microservices, but lack in operational functionality [11]. There are operational problems that are not possible to achieve without introducing further capabilities [11]. For example, as the number of microservices grows in an MSA system, a lack of visibility on the complex communication patterns makes it difficult to troubleshoot or monitor the state of the entire system.

Service mesh [11], an infrastructure layer built for container orchestration platforms, can deliver multiple operational enhancements. Commonly implemented as sidecar network proxies and a control plane for centralized governance, it increases the system's visibility and manage-

ability. Because the network proxies are deployed next to microservices as sidecars, there is no requirement for microservices to programmatically support them. Additionally, a service mesh can strengthen the security of an MSA system, by enforcing access control policies.

3.2 Advantages

This section covers the most prominent advantages of MSA.

Scalability

Applications can be scaled vertically or horizontally [12]. In vertical scaling, computing resources available to an application are adjusted to match demand. However, vertical scaling is restricted by the underlying hardware, and there is a ceiling for how much an application can be vertically scaled. Horizontal scaling combats this by creating multiple instances of an application, providing scalability by concurrency. Microservices are highly scalable vertically and horizontally, because of the precision at which they can be scaled. Monolithic applications benefit less from scaling due to the entire system having to be scaled by demand.

Due to the complexity of MSA, automatic scaling of microservices is important but difficult. However, containerization of microservices enables the use of service fabrics, such as Kubernetes, to manage autoscaling [10].

It is too early to determine if scaling microservices is more cost-effective than alternative architectural models [12]. Some studies have found cloud-native microservices optimized for scaling to be cheaper than monolithic approaches. However, there are counterexamples, and not enough empirical evidence for such a large ecosystem that the cost-effectiveness of scaling could be generalized.

Agility of MSA development

It is important for businesses to conform to rapidly changing requirements and to be able to quickly deliver new features or processes. MSA can be used to provide fast deliverance via efficient continuous delivery practices and DevOps [13]. Due to the small size of microservices, their deployments are simple and quick. Moreover, a small code base simplifies the development process and enables developers to quickly understand a microservice's functionality.

In addition to fast development time, the development of microservices

is horizontally scalable [1]. Multiple teams can develop different, individual microservices in parallel. Individual teams can utilize most suitable tools and technologies to implement the microservices. This is further supported by containerization, because the packaging and deployment of containers is agnostic to the technologies used.

Cloud-native

Microservices are designed to be cloud-native [10]. Service-orientation, lightweight virtualization and high ability to scale allows an MSA to operate on the cloud with optimized resource usage. These are important properties for large systems receiving heavy network traffic from millions of users [6].

3.3 Difficulties

Microservice modularization

Modularizing a software system into multiple cohesive parts is not a trivial task. In [14], Schröder et al. conducted a literature review on different approaches to identifying microservices. Some methods analyze existing applications via static code or runtime execution, while other methods focus on MSA design as a greenfield project. It is also possible to implement an application as a monolith first, and then convert it to MSA.

Consequences of improper MSA design are not minor. Multiple microservice smells have been identified, which hinder the maintainability of microservices [15]. Bad design can lead to cyclic dependency between microservices and high coupling.

In [16], Zhao et al. investigate the occurrence of code clones, duplicate lines of code across microservices. The existence of code clones may suggest the coupling of microservices. When a code clone is modified, there is a possibility that other microservices would have to be modified synchronously. However, code clones can be a result of multiple microservices sharing the same frameworks or utilities. On the other hand, cross-microservice code clones exist [17], and mitigating them increases the maintainability of microservices.

Monitoring microservices

Monitoring complex microservice systems is difficult [18]. To gain visibility, data must be collected and processed from multiple microservices. Fortunately, many monitoring solutions have been developed (ELK stack,

Prometheus and Grafana). However, these solutions will still have to be implemented, and each microservice must provide logging capabilities for monitoring internal logic.

The importance of traceability grows with the complexity of a microservice system [19]. To be able to successfully troubleshoot issues and discover root cause for failures, communication between microservices must be properly captured by the monitoring infrastructure.

Operational overhead

MSA is dependant on multiple different systems and methodologies to support its development and management efficiently. Deploying all of these systems introduces noticeable operational overhead, which much be dealt with by engineers. Misconfigured systems may present issues in terms of usability and security.

Sometimes, the benefits of MSA are not worth the operational cost. For example, Istio, which previously had implemented its control plane with microservices, decided to migrate the architecture back into a monolith [20]. Only one of the microservices managed most of the traffic, which meant that there was no use in decomposing the system as multiple services. Additionally, the use of MSA caused increased burden while deploying the system.

4 Discussion

This section discusses the appropriate use cases for MSA based on the overview of the architecture and its requirements.

4.1 Microservice scalability

Improved ability to cost-efficient scaling can be seen as one of key factors for MSA migration. However, there is a lot of overhead required to realize the full potential of microservices. Migrating from a functional MA to MSA does not guarantee reduced costs or higher availability. A poorly designed MSA with tightly coupled microservices may even increase the costs of running the system, as scaling out highly dependent microservices is equivalent to horizontally scaling a monolith.

To analyze whether an existing monolith could benefit from being migrated to MSA, many approaches to microservice identification should be utilized. The design should be evaluated to show that further scalability

is achievable and required.

4.2 Development overhead

Scalable, parallel development of MSA is another desirable property for companies looking to increase the efficiency of their development teams. The technological freedom provided by MSA encourages developers to optimize the choice of tools used for an implementation without affecting other projects. However, to properly manage MSA, investing in solutions regarding proper deployment, management and monitoring of microservices is mandatory. Infrastructural and operational requirements should be emphasized when evaluating the resources needed to implement MSA.

It is self-evident that organizations with less developmental resources are not able to heavily benefit from scaling their development across multiple projects. When a single developer must develop and maintain multiple microservices, it might be more efficient for the developer to work on a monolithic application instead. Additionally, the MSA must be operated by people with knowledge on how to do it, which incurs further overhead for a smaller team. However, other aspects of MSA may still provide enough benefits for MSA to be more cost-effective than alternative approaches.

In all likelihood, to successfully implement MSA, a clear view of wanted benefits and how they are realized is required. As mentioned in [3], the simplicity provided by monolithic applications may be much more valuable than the benefits of MSA.

5 Conclusion

The aim of this paper was to provide an overview of microservice architecture and identify situations where the architecture is applicable. Background information on monolithic and service-oriented architectures highlighted alternative approaches. In addition with an overview of how MSA is typically implemented, high-level considerations for MSA were discussed.

Because there is no clear definition for microservices, it is difficult to reason definite rules that indicate the success of MSA. However, general pitfalls of MSA are well understood. As MSA is implemented on more software projects, better understanding on the topic will be gained.

Microservice architecture is a continuously developing area. The promise of a cloud-native architecture with major potential for scalability and reliability is of high interest to many. This paper only scratched the surface, and the readers are recommended to further investigate the topic for more in-depth understanding.

References

- [1] P. Jamshidi, C. Pahl, N. C. Mendonça, J. Lewis, and S. Tilkov, "Microservices: The journey so far and challenges ahead," *IEEE Software*, vol. 35, no. 3, pp. 24–35, 2018. doi: 10.1109/MS.2018.2141039.
- [2] T. Cerny, M. J. Donahoo, and M. Trnka, "Contextual understanding of microservice architecture: Current and future directions," *SIGAPP Appl. Comput. Rev.*, vol. 17, p. 29–45, jan 2018. doi: 10.1145/3183628.3183631.
- [3] D. Gravanis, G. Kakarontzas, and V. Gerogiannis, "You don't need a microservices architecture (yet): Monoliths may do the trick," in *Proceedings of the 2021 European Symposium on Software Engineering, ESSE '21*, (New York, NY, USA), p. 39–44, Association for Computing Machinery, 2022. doi: 10.1145/3501774.3501780.
- [4] N. Dragoni, S. Giallorenzo, A. L. Lafuente, M. Mazzara, F. Montesi, R. Mustafin, and L. Safina, *Microservices: Yesterday, Today, and Tomorrow*, pp. 195–216. Cham: Springer International Publishing, 2017. doi: 10.1007/978-3-319-67425-4_12.
- [5] K. Gos and W. Zabierowski, "The comparison of microservice and monolithic architecture," in *2020 IEEE XVIth International Conference on the Perspective Technologies and Methods in MEMS Design (MEMSTECH)*, pp. 150–153, 2020. doi: 10.1109/MEMSTECH49584.2020.9109514.
- [6] M. Villamizar, O. Garcés, H. Castro, M. Verano, L. Salamanca, R. Casallas, and S. Gil, "Evaluating the monolithic and the microservice architecture pattern to deploy web applications in the cloud," in *2015 10th Computing Colombian Conference (10CCC)*, pp. 583–590, 2015. doi: 10.1109/ColumbianCC.2015.7333476.
- [7] M. P. Papazoglou and W.-J. van den Heuvel, "Service oriented architectures: approaches, technologies and research issues," *The VLDB Journal*, no. 16, pp. 389–415, 2007. doi: 10.1007/s00778-007-0044-3.
- [8] R. Mishra, N. Jaiswal, R. Prakash, and P. N. Barwal, "Transition from monolithic to microservices architecture: Need and proposed pipeline," in *2022 International Conference on Futuristic Technologies (INCOFT)*, pp. 1–6, 2022. doi: 10.1109/INCOFT55651.2022.10094556.
- [9] I. Karabey Aksakalli, T. Çelik, A. B. Can, and B. Teki nerdoğan, "Deployment and communication patterns in microservice architectures: A systematic literature review," *Journal of Systems and Software*, vol. 180, p. 111014, 2021. doi: 10.1016/j.jss.2021.111014.

- [10] D. Gannon, R. Barga, and N. Sundaresan, “Cloud-native applications,” *IEEE Cloud Computing*, vol. 4, no. 5, pp. 16–21, 2017. doi: 10.1109/MCC.2017.4250939.
- [11] W. Li, Y. Lemieux, J. Gao, Z. Zhao, and Y. Han, “Service mesh: Challenges, state of the art, and future research opportunities,” in *2019 IEEE International Conference on Service-Oriented System Engineering (SOSE)*, pp. 122–1225, 2019. doi: 10.1109/SOSE.2019.00026.
- [12] G. Blinowski, A. Ojdowska, and A. Przybyłek, “Monolithic vs. microservice architecture: A performance and scalability evaluation,” *IEEE Access*, vol. 10, pp. 20357–20374, 2022. doi: 10.1109/ACCESS.2022.3152803.
- [13] L. Chen, “Microservices: Architecting for continuous delivery and devops,” in *2018 IEEE International Conference on Software Architecture (ICSA)*, pp. 39–397, 2018. doi: 10.1109/ICSA.2018.00013.
- [14] C. Schröer, F. Kruse, and J. Marx Gómez, “A qualitative literature review on microservices identification approaches,” in *Service-Oriented Computing* (S. Dustdar, ed.), (Cham), pp. 151–168, Springer International Publishing, 2020. doi: 0.1007/978-3-030-64846-6_9.
- [15] D. Taibi and V. Lenarduzzi, “On the definition of microservice bad smells,” *IEEE Software*, vol. 35, no. 3, pp. 56–62, 2018. doi: 10.1109/MS.2018.2141031.
- [16] Y. Zhao, R. Mo, Y. Zhang, S. Zhang, and P. Xiong, “Exploring and understanding cross-service code clones in microservice projects,” in *Proceedings of the 30th IEEE/ACM International Conference on Program Comprehension, ICPC ’22*, (New York, NY, USA), p. 449–459, Association for Computing Machinery, 2022. doi: 10.1145/3524610.3527925.
- [17] R. Mo, Y. Zhao, Q. Feng, and Z. Li, “The existence and co-modifications of code clones within or across microservices,” in *Proceedings of the 15th ACM / IEEE International Symposium on Empirical Software Engineering and Measurement (ESEM), ESEM ’21*, (New York, NY, USA), Association for Computing Machinery, 2021. doi: 10.1145/3475716.3475784.
- [18] M. Waseem, P. Liang, M. Shahin, A. Di Salle, and G. Márquez, “Design, monitoring, and testing of microservices systems: The practitioners’ perspective,” *Journal of Systems and Software*, vol. 182, p. 111061, 2021. doi: 10.1016/j.jss.2021.111061.
- [19] B. Li, X. Peng, Q. Xiang, H. Wang, T. Xie, J. Sun, and X. Liu, “Enjoy your observability: an industrial survey of microservice tracing and analysis,” *Empirical Software Engineering*, vol. 27, pp. 1–28, 2022. doi: 10.1007/s10664-021-10063-9.
- [20] N. C. Mendonça, C. Box, C. Manolache, and L. Ryan, “The monolith strikes back: Why istio migrated from microservices to a monolithic architecture,” *IEEE Software*, vol. 38, no. 5, pp. 17–22, 2021. doi: 10.1109/MS.2021.3080335.

Large language models for protein structure prediction and design

Lauri Ahlberg

lauri.ahlberg@aalto.fi

Tutor: Katsiaryna Haitsiukevich

Abstract

¹ *Large language models (LLMs), neural networks pre-trained on vast amounts of text, have demonstrated impressive capabilities for generalisation, also performing tasks they were not trained for, often in zero-shot (i.e. on the first try) or few-shot fashion. This performance can be further improved by adapting pre-trained models to a downstream task via fine tuning. In this paper I explore the use of these models for protein-sequence tasks, such as structure prediction and design. Proteins are a class of functional molecules which possess huge potential for novel applications in many biological contexts such as medicine. They permit a natural textual representation while exhibiting complex structural relationships, reminiscent of natural languages. I present brief background on LLM architecture, protein structure and some recent deep-learning methods used in the domain before taking a look at LLM-based approaches, some requiring pre-training on protein sequence data, while others seek to leverage existing models. The field is developing at such a rapid pace that lack of established methods makes assessing novel approaches somewhat unpredictable.*

KEYWORDS: *large language models, transformers, biology, proteins, struc-*

¹The original proposed title was: *Large language models for modeling of physical systems*

1 Introduction

Large language models (LLMs), neural networks trained on vast amounts of data to understand language, currently occupy the centre stage in many discussions around the future of research. This is largely because of the capabilities they have demonstrated for zero-shot learning, i.e. being able to do an arbitrary task based on the prompt alone without any extra training [1]. The extent of this generalisation is currently being explored in the research of proteins, which has, until recently, been lacking accessible and efficient tools.

Proteins are among nature's most remarkable molecules as they are responsible for a multitude of different functions in living organisms that are needed to sustain life. Protein structures are exceedingly complex, consisting of multiple levels of organisation within sequences, and often also interactions between them. Because of this hierarchical structure, there is a certain resemblance to human languages. Until recently, the way that the one-dimensional amino acid sequences which make up proteins correspond to their final, precise three-dimensional structures, and vice versa, was one of the untamed frontiers in biology as the available methods were expensive and time consuming. However, this may be beginning to change.

The availability of large, publicly accessible databases of protein (amino acid) sequences and the development of open-source LLM-architectures has paved way for the adoption of LLMs for protein research. These new advances hint at a future where tailor-made proteins for, among other things, novel medications could be at our fingertips. They have enabled the creation of LMs that have been pre-trained on protein sequences and that are able to infer structural relationships between different parts of the sequence [2]. Previous methods relied on extensive background knowledge about protein structure that was incorporated into the deep learning models.

The aim of this paper is to explore some of the current state-of-the-art methods for using LLMs for questions related to protein structure prediction and design.

The structure of this paper is the following. Section 2 presents tech-

nology behind current LLMs as well as the biological concepts around protein structure and research methods used in the study thereof. Section 3 presents recent, LLM-based advances in this domain as well as hands-on experiences around these themes. Finally, Section 4 contains conclusions and speculations, as well as looking to the future.

2 Background

2.1 Large Language Models

Let us take a look at the structure and training process behind one of the most successful LLMs at the moment, the Generative Pre-Trained Transformer (GPT) -line of models. These are based on so-called attention mechanisms and the transformer-architecture [3].

The transformer is based on a two-stack structure, outlined in Figure 1 [3]. There is an encoder stack, which transforms the input into embeddings that aim to capture some of the meanings in the input. The decoder stack is connected to the encoder and it gives probabilities over the different possible outputs of the model. The embeddings are computed, at their core, as linear projections. At first the query is projected into the space of keys corresponding to output values. This projection is used as the weights for a weighted sum over the output values, yielding the final output of this attention mechanism. There are multiple such attention-heads and multiple multi-head attention layers as well. Large parts of this network can be executed in parallel, and thus benefit from the power of GPUs for faster computations. In both encoder and decoder layers, there is further a feed-forward network before the final output.

The GPT-models are developed using a two-step workflow consisting of pre-training the transformer followed by fine tuning. Firstly, the pre-training step is done on a large corpus of unlabelled data, such as all outbound links with 3 or more karma on reddit, in the case of GPT2 [1]. This is unsupervised training and the model begins to learn the structure contained in the training data without any explicit direction from without. After the pre-trained model is ready, what follows is a fine tuning step, which calibrates a reward model that indicates preferred outputs for a query. In the case of the more recent GPT models, this is performed as a reinforcement learning task on a human-labelled dataset demonstrat-

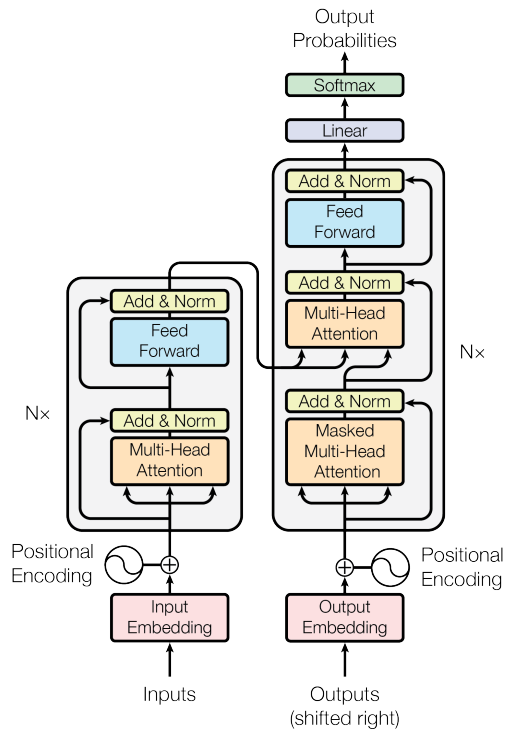


Figure 1. An outline of the transformer architecture as presented in [3].

ing desired model behaviour [4]. However, there is evidence that even the pre-trained model without fine tuning performs well on many natural language processing (NLP) tasks [1].

The GPT-approach to training is such that it only allows the model to see parts of the input before the point for which we want a prediction. It has been argued that this means the model is, in some cases, missing useful context for inferences [5]. A different approach would be to mask a random word in the input and ask the model to predict it based on the context in both directions. This method has been implemented in the Bidirectional Encoder Representations from Transformers (BERT) [5]. This model relies on the same transformer architecture and a similar split into pre-training and fine tuning phases, but the training is done on the masked queries.

2.2 Proteins and Structure Prediction

Most complex structures found in nature, especially outside the plant kingdom, consist of proteins. They are everywhere from muscle tissue and organs all the way to hormones, neurotransmitters and nerve toxins.

Proteins

The way that nature stores the information about different proteins is in the genetic material within each cell's nucleus – in the DNA. To construct a protein molecule, DNA is first converted to RNA through a process known as transcription. The RNA-molecule is then (often modified, and) translated to a sequence of amino acids. The exact ordering of these amino acids determines the chemical interactions between them which result in the exact three-dimensional structure of the final protein [6, p. 387]. Even just due to the size of the molecules, this structure can be very complex, consisting of three levels of different interactions. A fourth level is brought on by the creation of protein complexes, which consist of more than one amino acid chain.

The exact shape of the molecules is intricately tied to their function. A protein may, for example, bind to a receptor on some nerve cell that is shaped to recognise exactly that protein and trigger a nerve signal resulting in some more widespread response. Thus the study of protein structure is a field that offers a lot of potential for example for the discovery of novel medical compounds, as many of the processes in the human body are in some way controlled by proteins.

Classical Research Methods

Until recently, inferring protein structure from the DNA sequence alone, known as the protein folding problem, [7] was one of the important problems in biology for which there were very few methods. The most promising way for 'imaging' proteins was X-ray crystallography, which is expensive, labour intensive, and requires the protein to be available. Some electron-microscopy-based methods can also be used [6, p. 131]. It is easy to understand that these approaches are rather limited, offering no answers to predicting, for example, what kind of protein a currently unstudied gene codes for. Neither do they help if we would wish to build a protein suited for a specific task.

Current Best Practices

More recently, there has been an increase in deep-learning-based methods for inferring protein structure. AlphaFold [8] made headlines a few years ago by predicting protein structures to an unprecedented level of accuracy in the 2020 critical assessment of structure prediction (CASP) competition. Since then, AlphaFold-inspired approaches such as RoseTTAFold [9] have achieved similar results.

AlphaFold makes use of the same transformers-architecture as LLMs. It runs in two phases. A central component of the first phase is an Evoformer block, which works on Multiple Sequence Alignment (MSA) representations and pair representations of amino acid sequences. MSAs encode information about shared evolutionary history of sequences, while pairwise features indicate regions of similar structure between two sequences. In the Evoformer, these representations are iterated through several attention updates as well as graph-based triangle updates. That yields the output of the first phase. The second, structure prediction phase, computes three-dimensional coordinates of single atoms in the final folded (equilibrium) shape of the amino acid chain and rotations of the side chains.

3 Towards the Brass Tacks

As we have seen, working with protein sequences has seen major leaps in accessibility as computational advances have come alongside purely laboratory-based methods. In fact, some of these models are within grasp for anyone with internet access [10].

3.1 An LLM is all you need

The task of structure prediction, addressed by AlphaFold and RoseTTAFold, has recently also seen purely LLM-based solutions. The idea is that as LLMs have learned to infer the syntax and semantics of natural language from large amounts of text alone, LLMs pre-trained on datasets consisting of protein (amino acid) sequences could learn the rules governing the ways that stable proteins are formed. The researchers at Meta AI have demonstrated that as the size of the model grows large enough, there seems to be an emergence of understanding of the structural relationships that govern the way proteins fold [2]. The resulting model is considerably faster to run than previous state-of-the-art structure predicting models and the accuracy is on a comparable level as demonstrated in Figure 2. Their results are based on unsupervised training of an LM with up to 15 billion parameters on protein sequences with a randomised mask, a similar approach as in BERT [5]. They also used the model to create the ESM Metagenomic Atlas, a database with over 700 million predicted protein structures.

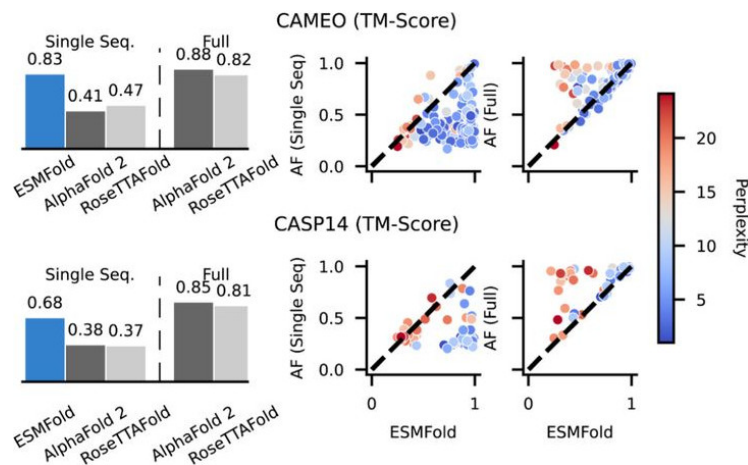


Figure 2. Comparisons between ESMFold, AlphaFold and RoseTTAFold on some structure prediction metrics. Image reproduced from [2].

As generative LLMs pre-trained on natural language can produce suitable answers to text inputs, novel protein sequences with desired properties can be generated by LLMs pre-trained on protein datasets. Ferruz et al [11] present ProtGPT2, an autoregressive transformer model (i.e. same pre-training method as for GPT-models) and demonstrate that their model is capable of generating stable structures unseen in nature, which nonetheless exhibit similar functional regions as natural proteins. The model can be prompted to create proteins from different naturally occurring protein classes. The researchers from Salesforce AI develop this approach further and show that such a model can be further refined via fine tuning [12]. They also partnered with experimental scientists to devise laboratory procedures for verifying the generated protein structures by comparing them with some naturally occurring counterparts.

The success of general-purpose LLMs in the NLP domain has led to explorations into ways that we could use the weights produced by the extensive pre-training as a foundation model for more specific use-cases. In fine tuning, the pre-trained weights of the foundation model are kept unchanged, but an additional weighting is performed on the outputs to adapt them better to the task at hand. In the case of chatGPT the model was tailored to dialogue via reinforcement learning [4]. A memory-efficient approach is to reduce the rank of the pre-trained weight matrix in order to select those weights most relevant for our task of interest [13]. For biomolecule-oriented tasks such as protein function prediction or protein design, there is some yet-unpublished work detailing an instruction dataset that can be used to adjust the fine tuning weights [14].

3.2 Experiences

Inspired by the success that general purpose LLMs have shown in zero-shot timeseries prediction [15], I designed some experiments for protein sequence prediction. The referenced article indicates that LLMs can infer patterns without any domain-specific training, which makes one wonder whether this could be used in the case of incomplete sequencing data, for example. I ran some toy experiments on Google Colab using the Llama-2 [16] 7b parameter version, available through the Hugging Face platform, as well as the corresponding chat-optimised model. Further, I input some of the same prompts into chatbot demo versions of the 7 and 13b parameter versions. In all the experiments, I provide a brief context about the task (predicting the next amino acids after given sequence stub) and then give a segment of length 20-50 amino acids randomly selected from the insulin sequence from the UniProt database [17]. Using Colab turned out to be quite challenging as the model which could run on the free tier (as per the Hugging Face blog post introducing Llama 2 models [18]) was running out of GPU memory after a few calls.

Qualitatively, I can say that the results from this brief foray were not very promising. The explanations given by the chat versions were mislabelling amino acids, skipping over some of them and reporting input sequences as clearly shorter than they actually were. On one occasion the 7b chat also started inventing its own amino acids. For the most part, the table of correspondences between letter representations and amino acid names were mostly correct, though rarely entirely so. The predicted continuations, when present, often contained amino acids that were in completely different category (e.g. hydrophobic when real amino acids were charged). The raw model (untrained for chat) was outputting code in different programming languages.

It must be noted, however, that due to time constraints the experiments conducted were very superficial. It would be interesting to select, for example, a segment of an amino acid chain corresponding to a known sub-structure (such as an α -helix or β -sheet) and see if the LLM is able to continue the pattern. Tweaking some of the configuration options (temperature, repetition penalty etc.) or iterating through a number of different versions of the system prompt may also lead to better results. Quantifying the results via some metric representing how close the LLM predictions were to the real amino acids in terms other than just the binary

correct/incorrect would also be worthwhile.

4 Discussion/Conclusions

We are at a very interesting time as the deep-learning technologies presented in this report can go in both directions. They can read and interpret protein sequences, but they can also design and write them. Both of these tasks have seen progress specifically from the LLM-direction. There are still ethical questions that need to be addressed before they can be used for some purposes, namely those dealing with sensitive medical data where confidentiality needs to be ensured. New advances have made this technology much more approachable, but the point remains that understanding the results and data itself requires domain expertise, which restricts these technologies to those who have enough background in molecular biology to be able to gauge what use cases are suitable for these tools. Time will tell whether their accuracy is good enough for practical purposes, although initial results seem promising.

Based on the experiments I wonder if LLMs could one day be used to help, for example, as part of a pre-processing step in the analysis of incomplete sequencing data, such as might be obtained from wet-lab experiments. This is a relevant question, as dealing with DNA- and amino acid sequences is what large swathes of the discipline of bioinformatics is focused on. One related question concerns the context length of LLMs. In the case of proteins this is hardly a problem as e.g. Llama2 boasts 4k tokens [16] whereas the average protein lengths across numerous different taxa are in the order of hundreds of amino acids [19]. The case may be different for DNA-sequences, however, as average gene length in humans is over 20 000 basepairs [6], but then again, perhaps not the entire gene is needed as context for predictions. Perhaps LLMs will not work directly out of the box, but I cannot help thinking that those tasks are very similar to the text-generation task GPT-models are trained for and to predicting the masked input used in training BERT models. LLMs may be of use particularly in cases where we are lacking a suitable reference genome or proteome, which many classical sequencing methods in bioinformatics rely on. One example may be with data obtained from sequencing environmental samples where we may often not even know what organism the DNA or protein segment came from.

Recent years have seen great leaps forward in the development of ma-

chinery for language modelling. The successes exhibited in NLP-related tasks have aroused curiosity about whether these technologies could be applied in other domains as well, in what concerns this report, particularly to working with protein sequences. Protein structure prediction has seen recent advances from deep learning approaches leveraging considerable domain specific knowledge, but since then it has also been shown that LLMs with enough parameters, pre-trained on amino acid sequences, can rival those results as well as generate novel proteins. LLMs trained on natural language could also potentially be adapted to tasks related to protein sequences via fine tuning. However, these approaches are still so young that they have yet to prove their worth in practice. The next years and decades will tell which combinations of technologies will surmount future obstacles and yield the most useful results.

Bibliography

- [1] A. Radford, J. Wu, R. Child, D. Luan, D. Amodei, and I. Sutskever, “Language Models are Unsupervised Multitask Learners,” *OpenAI blog*, vol. 1, no. 8, p. 9, 2019.
- [2] Z. Lin *et al.*, “Evolutionary-scale prediction of atomic-level protein structure with a language model,” *Science*, vol. 379, no. 6637, pp. 1123–1130, 2023, doi: 10.1126/science.ade2574.
- [3] A. Vaswani *et al.*, “Attention Is All You Need,” Aug. 2023, doi: 10.48550/arXiv.1706.03762.
- [4] D. M. Ziegler *et al.*, “Fine-tuning language models from human preferences,” 2020, doi: 10.48550/arXiv.1909.08593.
- [5] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, “Bert: Pre-training of deep bidirectional transformers for language understanding,” 2019, doi: 10.48550/arXiv.1810.04805.
- [6] N. A. Campbell, L. A. Urry, M. L. Cain, S. A. Wasserman, P. V. Minorsky, and R. B. Orr, *Biology : A Global Approach, Enhanced Global Edition*, 12th ed. Harlow, United Kingdom: Pearson, 2021, isbn: 9781292341705.
- [7] K. A. Dill, S. B. Ozkan, M. S. Shell, and T. R. Weikl, “The protein folding problem,” *Annual Review of Biophysics*, vol. 37, no. 1, pp. 289–316, Jun. 2008, doi: 10.1146/annurev.biophys.37.092707.153558.
- [8] J. Jumper *et al.*, “Highly accurate protein structure prediction with AlphaFold,” *Nature*, vol. 596, no. 7873, pp. 583–589, Jul. 2021, doi: 10.1038/s41586-021-03819-2.
- [9] M. Baek *et al.*, “Accurate prediction of protein structures and interactions using a three-track neural network,” *Science*, vol. 373, no. 6557, pp. 871–876, 2021, doi: 10.1126/science.abj8754.

- [10] M. Mirdita, K. Schütze, Y. Moriwaki, L. Heo, S. Ovchinnikov, and M. Steinegger, “ColabFold: making protein folding accessible to all,” *Nat. Methods*, vol. 19, no. 6, pp. 679–682, Jun. 2022, doi: 10.1038/s41592-022-01488-1.
- [11] N. Ferruz, S. Schmidt, and B. Höcker, “ProtGPT2 is a deep unsupervised language model for protein design,” *Nature Communications*, vol. 13, no. 1, Jul. 2022, doi: 10.1038/s41467-022-32007-7.
- [12] A. Madani *et al.*, “Large language models generate functional protein sequences across diverse families,” *Nature Biotechnology*, vol. 41, no. 8, pp. 1099–1106, Jan. 2023, doi: 10.1038/s41587-022-01618-2.
- [13] E. J. Hu *et al.*, “LoRA: Low-rank adaptation of large language models,” in *International Conference on Learning Representations*, 2022, doi: 10.48550/arXiv.2106.09685.
- [14] Y. Fang *et al.*, “Mol-Instructions: A Large-Scale biomolecular instruction dataset for large language models,” *arXiv*, 2023, doi: 10.48550/arXiv.2306.08018.
- [15] N. Gruver, M. Finzi, S. Qiu, and A. G. Wilson, “Large language models are zero-shot time series forecasters,” *arXiv*, 2023, doi: 10.48550/arXiv.2310.07820.
- [16] H. Touvron *et al.*, “Llama 2: Open foundation and fine-tuned chat models,” *arXiv*, 2023, doi: 10.48550/arXiv.2307.09288.
- [17] A. Bateman *et al.*, “Uniprot: the universal protein knowledgebase in 2023,” *Nucleic Acids Research*, vol. 51, no. D1, p. D523–D531, Nov. 2022, doi: 10.1093/nar/gkac1052.
- [18] P. Schmid, O. Sanseviero, P. Cuenca, and L. Tunstall, “Llama 2 is here - get it on hugging face,” Jul 2023. [Online]. Available: <https://huggingface.co/blog/llama2>
- [19] Y. Nevers, N. M. Glover, C. Dessimoz, and O. Lecompte, “Protein length distribution is remarkably uniform across the tree of life,” *Genome Biol.*, vol. 24, no. 1, p. 135, Jun. 2023, doi: 10.1186/s13059-023-02973-2.

Software supply chain security: the SLSA specification

Leevi Parkkinen

leevi.parkkinen@aalto.fi

Tutor: Jacopo Bufalino

Abstract

The vulnerabilities introduced by complex software supply chains and the popularity of third-party package repositories have made them an attractive target. Current approaches have not been effective at preventing supply chain attacks and SLSA has appeared to tackle the issue. This paper introduced SLSA with its security guarantees to the reader by producing an SLSA L3 npm package. Producing a new SLSA L3 package proved to be straightforward with the existence of premade GitHub actions SLSA builders. The generated provenance along with the digital signature gives strong guarantees that the package was built accordingly. However, the SLSA's lack of protection against tampering with source code, vulnerable and malicious dependencies, and usage of compromised packages, leaves room for improvement. Future versions of the framework combined with vulnerability detection audit tools can help achieve a more holistic supply chain security.

KEYWORDS: SLSA, Software supply chain security

1 Introduction

Software supply chain is the process of delivering software artifacts, including components and steps such as the source code, dependencies, the build process, and delivery. Supply chains can have vulnerabilities at every stage from source code to the released package including its dependencies. Attacks can include compromising build process [1], uploading modified packages [2] and building from modified source [3]. Combined with the popularity of third-party package repositories such as NPM and PyPI, supply chain attacks have become an attractive target [4]. As a result, all parties utilizing and developing packages including companies, developers, and customers require security guarantees for attacks against supply chains.

In one example, IT management product, Orion, was maliciously subverted to distribute malware to create backdoors on victims' networks [2]. This malware enabled spying on at least 100 companies and nine U.S. government agencies [4]. Clearly, current measures are not enough, and new ones are needed. Some specifications with security guarantees have been proposed, one of which is Supply Chain Levels for Software Artifacts (SLSA). [5]. SLSA is a security framework organized in levels of increasing security guarantees for supply chains, to prevent tampering, improve integrity, and secure packages and infrastructure [5]. This paper will introduce readers to SLSA by producing an SLSA-compliant project and analyzing its security guarantees.

The motivation of our work is to address the lack of practical research about the novel SLSA specification. This paper's goal is for the reader to understand what SLSA is, what it offers for supply chain security, and how to produce an SLSA-compliant npm package in practice.

This paper is organized as follows: Section 2 briefly introduces software supply chains and common threats. Section 3 introduces SLSA and its main points. Section 4 describes the process of producing an SLSA Level 3 NPM package with GitHub Actions, how it achieves SLSA L3 requirements, and the security guarantees it offers. Finally, section 5 summarizes the findings with a conclusion.

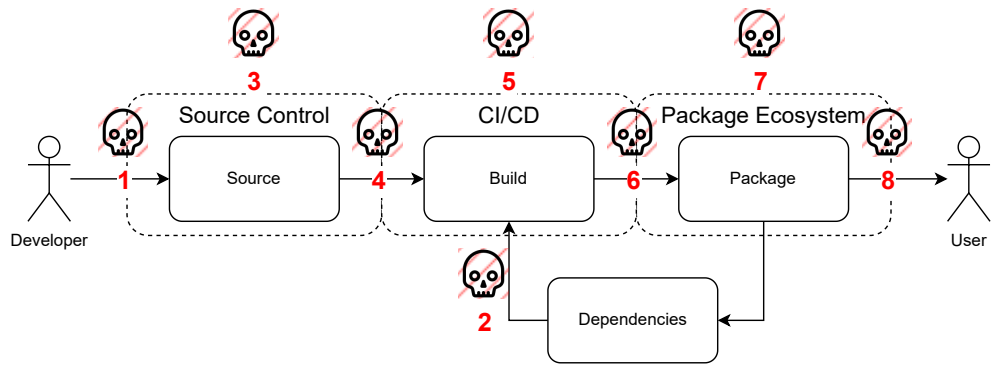


Figure 1. Software supply chain threats

2 Background

Software supply chain is the process of creating, building and delivering software. It includes the steps from the source code to a released software artifact. This production chain introduces multiple points of vulnerability, as risks do not only exist at source code but at multiple points in the complex process of producing and distributing software artifacts from that code. The next subsection introduces some of the supply chain threats.

2.1 Supply chain threats

The SLSA specification and Ruian Duan et. al [5, 6] introduce several possible threats for software supply chains highlighted in figure 1. Attacks can exploit vulnerabilities in source code and dependencies through actions like injecting malicious code (1) or leveraging malicious dependencies (2). They can also compromise the infrastructure by targeting source code platforms (3) or build platforms by building from a modified source (4) or modifying the build process (5). Additionally, attackers may upload vulnerable or malicious package versions via compromised accounts (6), directly target vulnerabilities within the package registry service (7) or target users with compromised packages (8).

Among the attack vectors targeting different parts of package management supply chains, the most popular ones are typosquatting, account hijacking, and social engineering [6]. Typosquatting is an attack where the attacker publishes a malicious package with a similar or misspelled name to a popular package. Typosquatting and account hijacking are low-effort attacks exploiting negligence and carelessness. Social engineering attacks can exploit the collaborative nature of open-source projects, trick owners of package repositories to transfer ownership, or publish "useful"

looking packages only to wait until it is used by their target to update it with malicious payloads [6].

A multitude of attacks have been made against package ecosystems. In one instance, an attacker compromised the npm account of an ESLint maintainer and published malicious versions of the `eslint-scope` and `eslint-config-eslint` packages that sent the contents of the user's `.npmrc` file to the attacker [7]. Another attack involved the publication of `rest-client` versions with hidden backdoors, utilizing a compromised `RubyGems.org` account belonging to a maintainer [8]. A separate incident involved a malevolent version of the widely used `bootstrap-sass` package, uploaded to the official `RubyGems` repository, which included a backdoor allowing for remote command execution on server-side Rails applications [9].

To study the security of package managers and registry abuse Ruian Duan et. al [6] introduced MALOSS, a custom pipeline based on well-known program analysis techniques such as metadata, static, and dynamic analysis. They found 339 new malicious packages, with three of them having more than 100,000 downloads. Dramatically, 20% of the malicious packages persisted in the NPM, PyPI, and `RubyGems` ecosystems for over 400 days [6]. MALOSS is just one approach to package management security and Enck William Chinenye et.al and Okafor et.al acknowledge the disjoint efforts and challenge of getting industry-wide participation in supply chain security [4, 10]. Introduced in the next section, is another approach to improve software supply chain security called SLSA. SLSA encompasses all parties involved in producing, consuming, and providing infrastructure for software, which can create more trust across the entire supply chain [5]

3 SLSA

SLSA is a set of incrementally adoptable guidelines for supply chain security, established by industry consensus [5]. SLSA is useful for both producers and consumers of software artifacts. For producers SLSA offers a checklist to improve software security, while consumers can evaluate the trustworthiness of the artifacts they consume and both parties, SLSA offers terminology to talk about software supply chain security.

SLSA offers integrity by adding security guarantees that an artifact is produced from the expected source, with the expected build process on a safe build platform. If "software bill of materials" (SBOM), tells what the

software is made of, SLSA can be thought of as all the food safety handling guidelines that make this ingredient list credible [5].

3.1 Security levels

SLSA is organized into levels with increasing security guarantees for supply chains. Levels range from L0 with no security guarantees to L3 with the most security guarantees. Levels are split into tracks, and each track measures a single aspect of supply chain security. Higher levels have the requirements and benefits of lower levels. SLSA version 1.0 defines only a Build track, while a Source track may be added in a future version [5].

The build track is designed to ensure that package artifacts are trustworthy and complete in terms of their provenance. Provenance includes details about the entity that built the artifact, the process they used, and the inputs involved. Build track offers protection to safeguard against tampering with the build, the provenance, or the artifact in increasing levels. [5] The build track's main objective is to verify that the artifact has been built according to the expected standards. Consumers can determine what the expected provenance should look like for a particular package and then compare the actual provenance of each package artifact to those expectations.

Each ecosystem or organization can implement the Build track their way, including means of defining expectations, accepted provenance formats, whether reproducible builds are accepted, distributing provenance, verifying it, and actions taken on failure [5]. The next three paragraphs summarise SLSA build levels L1-L3.

Build L1 has a provenance that describes the build process and identifies the resultant package through cryptographic digests. This provenance, while serving as a useful tool for error prevention, may be incomplete or unsigned, lacking absolute tamper-proof qualities.

Build L2 has the requirements and benefits of L1, while introducing an additional layer of security to prevent tampering with the provenance through digital signatures. Build steps, provenance generation and signing are run on a hosted build platform on shared or dedicated infrastructure. Forging the provenance or evading verification may be easy, but it can be used to deter less sophisticated adversaries and entities exposed to legal or financial risks [5].

Build L3 offers strong unforgeability of the provenance and isolation of the build process on top of the previous levels. In practice, builds run on a hardened build platform that offers strong tamper protection [5]. Runs are isolated and do not influence one another, even within the same tenant's project. Secret material used to sign the provenance are stored in a secure management system not accessible to the user-defined build steps, and any remote influence on the build is recorded in the provenance. [5] Build L3 effectively thwarts tampering threats during the build process, be they from insider threats, compromised credentials, or other tenants.

3.2 Attestation and SLSA Provenance

A software attestation is signed metadata about a software artifact or collection of software artifacts. The metadata is explicit, and the signature only denotes who created the attestation. Attestations can express an arbitrary amount of information. For example, an attestation might state exactly how an artifact was produced. Attestations can be fed into automated policy engines, such as in-toto-verify and Binary Authorization. [5]

For SLSA attestation, any format may be chosen, however, there needs to be a way for external users to consume and possibly verify the provenance [5]. For most cases, SLSA provenance is recommended [5]. SLSA Provenance offers interoperability and easy verification using Generic SLSA Verifier.

Provenance is information, or metadata, about how an artifact or set of artifacts was produced. This could include information such as what source code, build system, and build steps were used, as well as by whom and why the build was initiated. Provenance can be used to determine the authenticity and trustworthiness of software artifacts. To achieve integrity guarantees, provenance must exist from the beginning, as it is the verifiable information about software artifacts describing where, when, and how something was produced [5]. Higher SLSA levels bring more resilient integrity guarantees, with stricter provenance requirements [5].

4 Build L3 NPM package

This section introduces the practicalities needed to produce a SLSA Build L3 Node.js project with provenance and analyzes the offered security guar-

```

1  name: Node slsa
2
3  on: [push]
4
5  jobs:
6    build:
7      permissions:
8        id-token: write # For signing
9        contents: read # For repo checkout.
10       actions: read # For getting workflow run info.
11       if: startsWith(github.ref, 'refs/tags/')
12       uses: slsa-framework/slsa-github-generator/.github/workflows/builder_nodejs_slsa3.yml@v1.9.0
13       with:
14         run-scripts: "ci, build"
15
16     publish:
17       needs: [build]
18       runs-on: ubuntu-latest
19       steps:
20         - name: Set up Node registry authentication
21           uses: actions/setup-node@64ed1c7eab4cce3362f8c340dee64e5eaef8f7c # v3.6.0
22           with:
23             node-version: 18
24             registry-url: "https://registry.npmjs.org"
25
26         - name: publish
27           id: publish
28           uses: slsa-framework/slsa-github-generator/actions/nodejs/publish@v1.9.0
29           with:
30             access: public
31             node-auth-token: ${ secrets.NPM_TOKEN }
32             package-name: ${ needs.build.outputs.package-name }
33             package-download-name: ${ needs.build.outputs.package-download-name }
34             package-download-sha256: ${ needs.build.outputs.package-download-sha256 }
35             provenance-name: ${ needs.build.outputs.provenance-name }
36             provenance-download-name: ${ needs.build.outputs.provenance-download-name }
37             provenance-download-sha256: ${ needs.build.outputs.provenance-download-sha256 }

```

Figure 2. Project Workflow

antees.

4.1 Producing the package

The project is a sample Hello World Express TypeScript project built on GitHub with GitHub Actions and released as a npm package. A premade Node.js builder is used to meet L3 provenance generation and isolation requirements. It handles the complex orchestration of reuseable workflows, signing, intoto, Sigstore, etc. The builder is based on the Build Your Own Builder (BYOB) framework and tools offered by SLSA GitHub Generator [11]. These tools allow projects to generate SLSA provenance safely and accurately using GitHub Actions.

Figure 2 describes the created project workflow. It calls the SLSA-compliant Node.js builder on line 12. The required npm build scripts defined in the project’s package.json file are provided as input to the builder. This yaml file is the only required configuration file to create an SLSA 3-compliant Node.js project from scratch. After the build is done the package is published to the npm registry with the provided publish action.

```

_type: https://in-toto.io/Statement/v0.1
subject:
  - name: pkg:npm/slsa-node-project@0.1.4
    digest:
      sha512: >-
        421e4b95bdba74d0d881939e1138de079d8dbe7df6894d17cd180e60fa94a69c1afb56874d3d579f6970c45c50c88c6de23ce960805194a44af9dc8564cb7c
predicateType: https://slsa.dev/provenance/v0.2
predicate:
  builder:
    id: >-
      https://github.com/slsa-framework/slsa-github-generator/.github/workflows/builder_nodejs_slsa.yml@refs/tags/v1.9.0
    buildType: https://github.com/slsa-framework/slsa-github-generator/delegator-generic@v0
  invocation:
    configSource:
      uri: git+https://github.com/slebes/slsa-node-project@refs/tags/v0.1.5
      digest:
        sha1: 42734bad08ba5efea3b0bf58b7ce05b667cb9407
      entryPoint: .github/workflows/github-acrtions.yaml
    parameters:
      inputs:
        directory: .
        node-version: ''
        node-version-file: ''
        rekor-log-public: false
        run-scripts: ci, build
      environment:
        GITHUB_ACTOR_ID: '48884966'
        GITHUB_EVENT_NAME: push
        GITHUB_BASE_REF: ''
        GITHUB_REF: refs/tags/v0.1.5
        GITHUB_REF_TYPE: tag
        GITHUB_REPOSITORY: slebes/slsa-node-project
        GITHUB_REPOSITORY_ID: '694706733'
        GITHUB_REPOSITORY_OWNER_ID: '48884966'
        GITHUB_RUN_ATTEMPT: '1'
        GITHUB_RUN_ID: '6265477715'
        GITHUB_RUN_NUMBER: '19'
        GITHUB_SHA: 42734bad08ba5efea3b0bf58b7ce05b667cb9407
        GITHUB_TRIGGERING_ACTOR_ID: '48884966'
        GITHUB_WORKFLOW_REF: >-
          slebes/slsa-node-project/.github/workflows/github-acrtions.yaml@refs/tags/v0.1.5
        GITHUB_WORKFLOW_SHA: 42734bad08ba5efea3b0bf58b7ce05b667cb9407
    metadata:
      buildInvocationId: 6265477715-1
      completeness:
        parameters: true
  materials:
    - uri: git+https://github.com/slebes/slsa-node-project@refs/tags/v0.1.5
      digest:
        sha1: 42734bad08ba5efea3b0bf58b7ce05b667cb9407

```

Figure 3. Provenance

4.2 Generated provenance

The provenance is logged in a public transparency ledger along with a signing certificate. The generated provenance is shown in figure 3 and the main contents are described below.

The schema is identified by **_type** to be an in-toto attestation. It applies to the artifacts described in **subject**, i.e., the produced npm package, which contains the name and the cryptographic digest of the artifact.

The **predicate** is the arbitrary metadata about the subject, the provenance. It conforms to the SLSA provenance v0.2 format as stated in **predicateType**.

The **builder** is the entity that executed the invocation trusted to have correctly performed the operation and populated this provenance [5]. The **id** field indicates that the builder is the Node.js builder we used. The **buildType** contains a URI that should resolve to a human-readable specification with overall description of the build type. It determines the mean-

ing of **invocation** and **materials** [5].

The **invocation** object identifies the event that kicked off the build [5]. It contains **configSource**, which describes where the config file that started the build came from. This includes the URI of the source tag, the entrypoint file in figure 2, and its cryptographic digest. External inputs that influence the build are listed in **parameters**. These can include inputs set by a user or tenant of the build platform. Most importantly, it includes the build scripts and node version we have defined. To trust the build process and achieve Build L3, this should be complete, meaning that every external input is included and no external influence is left unnoted [5]. Parameters that are under the control of the builder are listed in **environment**. These may be necessary for reproducing the build and are mainly used for debugging, incident response, and vulnerability management [5].

Other properties of the build are included in **metadata**. For example, **completeness** indicates that **parameters** is complete. Lastly, the collection of artifacts that influenced the build is included in **materials**, in our case, the URI of the source tag.

4.3 Security guarantees

So, what are the security guarantees of the produced project? The security guarantees are primarily tied to the provenance and how it was generated. The generated provenance includes important details such as the source tag, inputs to the build, and the identity of the builder. From this one can infer from what, how, and who made the build. The provenance is automatically generated and signed on a hosted platform making it tamper-proof after the build. By forming expectations and checking whether those expectations are met in the provenance we can detect builds from a modified source with unofficial parameters (figure 1 4). Even if an adversary gains access to a compromised package repository account and uploads a modified package or provenance, verifying the provenance fields and validating the digital signature will detect these changes, rendering threats 6 in figure 1 ineffective.

The user build process is run in isolation and any external influence is recorded in the provenance, while any secret material used in signing of the provenance is not accessible to the build. Thus compromised build workers cannot steal secrets, forge the provenance, or influence concurrent or sequential builds, leaving threats 5 in figure 1 ineffective. This

guarantees a high degree of confidence that the package was constructed exclusively from the official source with the correct build process, free from external influences.

On the other hand, SLSA doesn't offer protection against submitting unauthorized changes or compromise of the source platform (figure 1 1 and 3) [5]. Another inherent limitation is that the build platform must be trusted. If the control plane responsible for the orchestration of builds and generation and signing of the provenance is compromised, the provenance cannot be trusted. Although typosquatting is the most popular attack vector for supply chains, SLSA offers no security guarantees against the usage of compromised packages (figure 1 8). In addition, SLSA doesn't have a standardized way to apply to dependencies recursively [5]. This means that dependencies must be verified separately and a Build L3 project with vulnerable or lower SLSA level dependencies is still considered L3 (figure 1 2).

To address these issues, SLSA could be used together with current audit tools like npm-audit or Snyk. Both tools can be used to detect vulnerabilities in dependencies (figure 1 2), while Snyk can also address suspicious source code detecting vulnerable and malicious packages, which could help organizations detect malicious code injections and users detect malicious or vulnerable packages (figure 1 1 and 8).

5 Conclusion

In this paper, we introduced SLSA and the practicalities needed to produce an SLSA L3 npm packet. We analyzed the security guarantees offered by SLSA L3. In its current state, SLSA Build L3 with its isolation and provenance generation requirements offers strong guarantees that the artifact was built with the correct build process. However, SLSA alone doesn't address tampering with the source, vulnerable dependencies, or actual usage of compromised packages. By combining SLSA with audit tools and the possibility of future versions of the framework addressing source vulnerabilities, a more holistic supply chain security could be achieved.

References

- [1] C. I. Team, “Sunspot: An implant in the build process.” [Online]. Available: <https://www.crowdstrike.com/blog/sunspot-malware-technical-analysis/>.
- [2] Codecov, “Post-mortem / root cause analysis (april 2021).” [Online]. Available: <https://about.codecov.io/apr-2021-post-mortem/>.
- [3] J. Cameron, I. Rostovtsev, and W. community, “Security.” [Online]. Available: <https://www.webmin.com/exploit.html>.
- [4] W. Enck and L. Williams, “Top five challenges in software supply chain security: Observations from 30 industry and government organizations,” *IEEE Security & Privacy*, vol. 20, no. 2, pp. 96–100, 2022. doi: 10.1109/MSEC.2022.3142338.
- [5] Open Source Security Foundation (OpenSSF), *Supply-chain Levels for Software Artifacts (SLSA)*, 2023. <https://slsa.dev/spec/v1.0/>.
- [6] R. Duan, O. Alrawi, R. P. Kasturi, R. Elder, B. Saltaformaggio, and W. Lee, “Towards measuring supply chain attacks on package managers for interpreted languages,” 2020. doi: 10.48550/arXiv.2002.01139.
- [7] J. Foundation and other contributors, “Postmortem for malicious packages published on july 12th, 2018.” [Online]. Available: <https://eslint.org/blog/2018/07/postmortem-for-malicious-package-publishes/>.
- [8] J. Koljonen, “[cve-2019-15224] version 1.6.13 published with malicious backdoor.” [Online]. Available: <https://github.com/rest-client/rest-client/issues/713>.
- [9] L. Tal, “Malicious remote code execution backdoor discovered in the popular bootstrap-sass ruby gem.” [Online]. Available: <https://snyk.io/blog/malicious-remote-code-execution-backdoor-discovered-in-the-popular-bootstrap-sass-ruby-gem/>.
- [10] C. Okafor, T. R. Schorlemmer, S. Torres-Arias, and J. C. Davis, “Sok: Analysis of software supply chain security by establishing secure design properties,” in *Proceedings of the 2022 ACM Workshop on Software Supply Chain Offensive Research and Ecosystem Defenses*, (New York, NY, USA), p. 15–24, Association for Computing Machinery, 2022.
- [11] O. S. tooling team, “Slsa github generator v1.9.0.” [Online]. Available: <https://github.com/slsa-framework/slsa-github-generator/tree/v1.9.0>.

Caching is vulnerable: Modern threats facing CDN Edge Servers

Leevi Rönkkö

leevi.ronkko@aalto.fi

Tutor: Jose Navarro

Abstract

In many ways, Content Distribution Networks act as a backbone for the modern interactive web. They hold manage and deliver most of the media that gets consumed over the internet. This paper studies how the caching systems of CDNs may be exploited to hurt network infrastructure and the customers of such systems. Some specific attack types are studied and possible mitigations are presented. A general discussion on the security implications of cache systems is included.

KEYWORDS: CDN, web, security, attack, DoS, Denial of Service, Cache Poisoning, content, Content Distribution Network

1 Introduction

During the earlier days of the growing world wide web, websites, including the media content therein were usually hosted on the same web server. As demand for media, and web content was rapidly increasing, this centralized approach for content distribution was proving itself more and more unfit for purpose. The ever increasing content delivery delay was one of the most significant problems to solve in the field of web technologies.

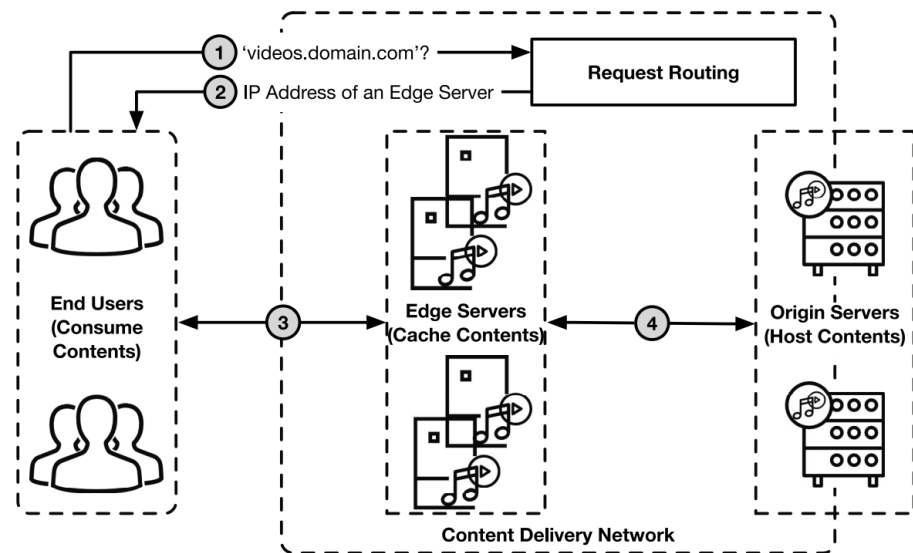


Figure 1. Common structure of a CDN its request-response process [3]

During the late 90s, Akamai brought to market the first ever global Content Distribution Network, or **CDN** [1]. Their goal was to move media content away from the central web servers, and distribute it across data centers all over the globe, caching it as close to the edge of the networks as possible. This approach was immensely successful and by 2010 Akamai was delivering 15-20% of all internet traffic worldwide [2].

Figure 1 shows the common structure of CDN services and demonstrates how a request is handled from end user to origin and back. An end user requests a resource and the CDN routes the request to and edge server. If the requested resource is cached, the CDN can return it to the user directly, otherwise it makes a round trip to the origin to retrieve the resource and possibly caches the result for future use [3].

While CDNs certainly can speed up load times for web pages and media content as well as take some load off the central servers, a common marketing strategy for CDNs has been to highlight their potential security benefits. Naively, it is quite easy to see how putting the strength of a global CDN service between would be attackers and the web servers they want to target can mitigate the chances of, say, a successful denial of service attack.

Unfortunately, by the 2020s security researchers and criminals alike had discovered multiple ways to exploit CDN infrastructure, both to bypass their defences and even weaponize their capabilities to make customers more, not less susceptible to some fairly powerful attacks [3].

A 2021 survey [3] published by IEEE, outlines some of the security

challenges that CDNs face. It splits CDN security challenges into three categories based on which part of the CDN infrastructure they pertain to (Edge Servers, Request Routing and Origin Servers) and further, by the nature of the threat itself (eg. Denial of Service).

This paper focuses on the security challenges that the edge servers of CDNs face, introduces some specific attacks that modern CDN systems are vulnerable and studies ways to mitigate risks now and in the future. The next section outlines some background on CDN edge servers and two important basic modes of attack.

2 Background

Edge servers are the part of CDN infrastructure that cache content as close to the end user as possible [1]. According to the IEEE survey [3] they must deal with a trade off between security, privacy and performance. In order to protect the origin servers, an edge server may want to perform some analysis on the application layer packets. End to end encryption limits the edge server's ability to do so, but disabling it raises concerns about privacy. Use of holomorphic encryption may aide in the edge server's ability to perform packet analysis, but implementing it often comes with a prohibitive performance cost.

2.1 Caching

One major way edge servers are targeted, is by exploiting their caching functionality to either degrade quality of service, or trick them into serving unintended content [3]. These types of attacks are often referred to as *Cache Pollution* and *Cache Poisoning*, respectively.

Edge servers utilize a locality principle to improve the rate at which a requested resource can be found in cache, that is, they will cache recently requested objects. An attacker who knows this, can intentionally request unpopular resources frequently to achieve a *polluted cache state* [3]. The unpopular resources will remain in cache while legitimate requests have to be fulfilled by the origin servers themselves.

In contrast, a Cache Poisoning attacks attempt to replace the cache value for some legitimate resource key with incorrect and often malicious contents [3]. Edge servers often use some combination of HTTP headers as cache keys. This can be exploited, when the origin server is configured

to include some unkeyed request headers in its response or to generate some content dynamically based on unkeyed request headers. The response including poisoned headers or content will then be stored in the cache and served to end users that send a request with a matching set of keyed headers. Cache poisoning can also be used as one step of larger attack to, for example steal users' private information or aide in a Denial of Service attack [3]. This type of DoS attack studied in detail in the next section of this paper.

2.2 Denial of Service

Denial of Service attacks are any attacks on a network that aim to degrade a network's or host's quality of service by consuming it's resources with illegitimate requests or traffic [4]. During the late 90s and early 00s they were becoming a major security issue for web systems world wide and in 2006, researchers from The University of California San Diego released a paper outlining their techniques for categorizing DoS attacks and quantifying their prevalence on the internet [4]. Their backscatter analysis identified approximately 70,000 attacks on 5,300 different organizations.

Denial of Service attacks continue to be one of the most financially, and materially damaging modes of attacks on internet based systems in the modern day. While plain, distributed DoS attacks are still common, modern attacks are often increasingly reliant on design flaws in internet infrastructure, including CDN services.

3 Cache Poisoned Denial of Service

In November of 2019, security researchers introduced a way to exploit the cache poisoning vulnerabilities of CDNs and caching services to perform a devastating denial of service attack [5]. They aptly called this new type of attack *Cache Poisoned Denial of Service*. Although attacks that weaponize cache services to cause denial of service had been found in past, CPDoS is special in the fact that it requires staggeringly little skill or resources to pull off [5].

Figure 2 demonstrates the basic concept of the attack. It exploits a difference (or *semantic gap*) in interpretation of HTTP request headers between the cache and the origin server, in order to poison the cache of a legitimate resource with an error response [5]. In principal, only a sin-

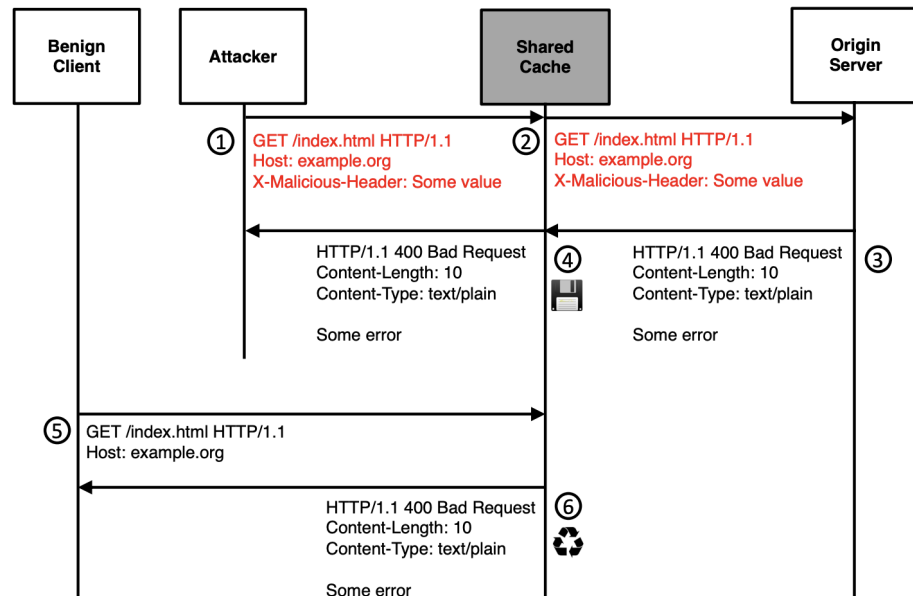


Figure 2. The basic process of a CPDoS attack [5]

gle HTTP request is necessary, to make one resource completely inaccessible for some geographical area. Depending on the importance of that resource, it can be possible to bring down and entire web service with a single request [5].

3.1 Headers and caches

A successful CPDoS attack relies on nothing more than a semantic gap in HTTP header interpretation between a caching server and an origin server. Commonly, a combination of the HOST and METHOD headers are used as cache keys and only the results of GET requests are cached [5]. Any unkeyed headers will generally be passed along to the origin server as they are. If some specific header value is ignored by the cache, but causes an error on the origin server, an error message will replace the cache value of a legitimate resource, and end users requesting resource will be served the error message.

One naive mitigation for this problem is not caching error responses at all, and indeed, some services never cache certain HTTP status codes [5]. This however, can present another problem. If responses for code 404 and 400 (Not found and Bad Request, respectively) are never cached, the origin is directly vulnerable to a CDN bypass when a nonexisting resource is requested. The malicious requests can in theory be completely indistinguishable from benign requests and since only a single request is needed to poison the cache, attacks are very difficult to detect.

The only real barriers to entry for performing a successful attack are in finding the relevant semantic gaps and figuring out when the cache for the targeted response expires [5]. A recent development in semantic gap discovery, *HDiff* [6], will be described with more detail in the next subsection. The expiration time of the cache can be obtained in multiple possible ways, the simplest of which are the HTTP *Expires*, *Age*, *max-age* and *s-maxage* headers [5].

3.2 HDiff

In 2022 security researchers presented a semi automated framework to find semantic gaps in HTTP implementations, HDiff [6]. The researchers were concerned about the rise of semantic gap attacks, including CPDoS. They found that a large portion of the vulnerabilities were caused by implementations which did not adhere to RFC standards, either due to programming errors or intentional relaxations. According to them, RFC specifications are difficult to formalize, as they are written in natural language.

HDiff uses natural language processing techniques to convert RFC documentation into two types of formalized structures *SRs* and *ANBFs* [6]. The first structure, the SR or *Specification Requirement*, semantically describes actions that an agent in the HTTP protocol ought to take when processing a specific signal, while ANBFs formalize the syntax and structure of signals in HTTP [6].

Once the documentation is processed, HDiff generates test cases and compares the outputs of two target HTTP implementations, discrepancies in the outputs are identified as semantic gaps between the two [6]. Figure 3 shows the architecture of the framework. First is the Documentation Analyzer which again, produces formalized structures based on human language protocol documentation. Then Differential Testing is used to find semantic gaps in HTTP implementations.

Out of the ten popular HTTP implementations the researchers tested, they found that six were vulnerable to CPDoS attacks, including Apache Nginx and Squid [6]. Apache and Nginx alone account for over 60% web servers accounted for in a 2023 W3Techs survey [7].

Figure 4 shows the workflow the researchers used to test HTTP implementations. They used a proxy, an echo server and an origin server. In step one, a proxy (edge server) forwards a request to the echo server which returns it back as-is. In step two, the proxy forwards the request to

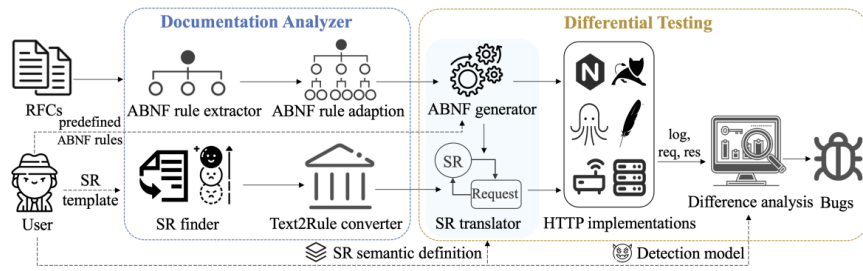


Figure 3. The architecture of HDiff [6]

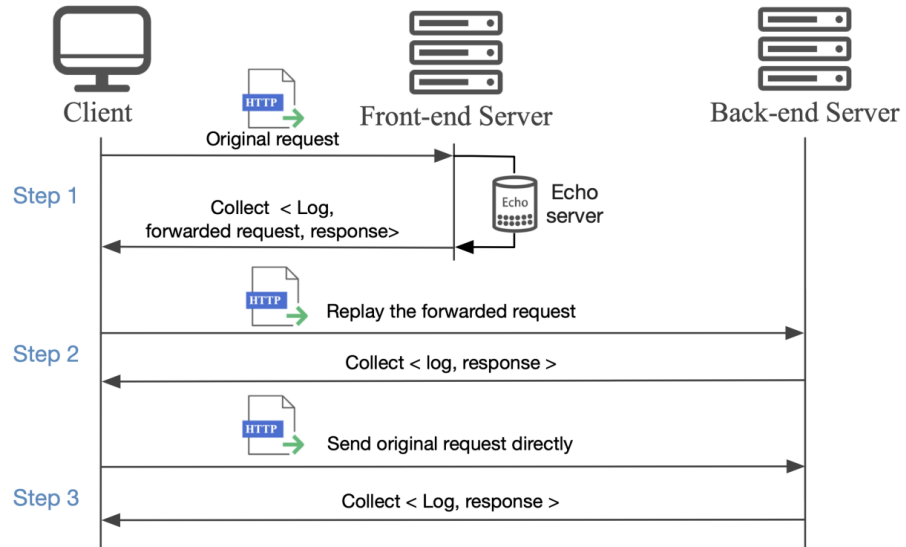


Figure 4. Example of the HDiff test workflow [6]

an origin server and the response is logged and in step 3 the same request is sent to origin direct. Discrepancies in responses between steps 1 and 3 can be used to test for RFC compliance and discrepancies between responses between steps 1 and 2 can be used to find vulnerable server pairs [6].

3.3 Mitigations

As earlier established, it is difficult to detect CPDoS attacks by analyzing requests as the amount requests required for a successful attack is small and distinguishing a legitimate error state from one caused by a malicious request is non-trivial. The only 100% effective way to block CPDoS attacks is to stop caching error responses [5]. While that approach is certainly worth considering in some instances, most services cannot reasonably bear the performance cost.

In lieu of completely abandoning caches for error messages, the usage of frameworks and tools such as HDiff is vital in determining which implementations and services might be vulnerable. While fixing implementation problems in server technologies is left to the developers of the HTTP implementations themselves, users of such implementations should be encouraged to use the most up-to-date and least vulnerable solutions. Origin server operators should also make sure that their implementations follow RFC specifications and limit the scope of HTTP headers they allow in requests and responses.

4 Discussion

As the world wide web has grown, it has become more commercialized and as a consequence, in some ways less distributed [8]. The idea of an equal and accessible network of computers distributing information in the form of interlinked documents has now become a collection of a few large services and applications that compete to fulfil more and more of a user's needs. If a CDN giant like Akamai goes down, a large amount of seemingly unrelated internet services will become inaccessible or lose functionality.

As responsibility of maintaining global web infrastructure is consolidated to fewer entities, they naturally become targets for abuse. This encourages security researchers and criminals alike to be the first ones to discover new vulnerabilities. Service providers want to sell the idea of security and reliability to their customers and their customer's users. However, collectively and individually it is important to avoid placing too much responsibility on a single entity. A single point of failure in a society's infrastructure has technical, financial and geopolitical consequences [8].

The findings of security researchers on the topic of CDN security and CPDoS in particular are no doubt significant, but arguably even more important is their reception. A tool such as HDiff, serves as a great example of this. Anyone who has some stake in the security of CDN services can potentially take advantage of the research. A potential customer can use it to find which services might be vulnerable to specific threats and make a more informed purchase decision. The companies responsible for the CDN services can use it to find potential flaws in their own design and improve it. Criminals can make use of the tool as well. They can find vulnerable services and turn them into potential targets for attack.

From the perspective of a potential CDN customer, threats like CPDoS highlight complexities in security analysis and the trade offs that secure systems come with. On the modern web, hosting a a complex web application with a critical mass of users can get expensive relatively quickly. Maintaining and serving everything from the origin with no caching can be unfeasible. Having a large CDN provider between a service and the open web can indeed protect the service from traditional DoS attacks. On the other hand, becoming functionally reliant on a global service can cause one's web application to be hurt from attacks and outages that it is seemingly not the target of. And as in the case of CPDoS, it can make one's application vulnerable to amplified attacks that weaponize the CDN against the application.

In the modern day, the selection of CDN, cloud and hosting services can have implications on the functionality of the application itself and selecting the right services for one's application is vitally important. Even after the selection of services is made, it is helpful to be aware of the implications of said selection on the rest of the stack. If one knows of, say, some CPDoS vulnerability in the CDN service they use, they maybe able to mitigate risk with their own implementation by being careful about what headers their service accepts and sends in responses and how their service handles errors for example.

From the perspective of the web as a whole and the threat landscape it represents, consolidation and the proliferation of CDN services means the threat of single attacks and exploits which can affect the entire web or the entire web of some part of the world. Let's say, for example, that a large web-based authentication service gets disrupted due to a poisoned caching of the login page in some data center. Now nobody is able to authenticate to any service that relies on it for sign-in. This list of affected services may include banking, government websites, tax services, email and much else.

5 Conclusions

In conclusion, as CDNs and large caching services have grown in prevalence and network infrastructure has consolidated to be run by a smaller number of companies, the security of large web services has become more vital. Content Delivery Networks have historically been sold as a possible security solution to protect smaller services from Denial of Service

but recently, they have been successfully weaponized against their own customers.

The study of semantic gaps in HTTP implementations has revealed many subpar design patterns utilized by large players in the CDN market. Semantic gaps may be exploited to achieve poisoned cache states denial of service in a large scale or in targeted attacks. This has implications for every stakeholder in CDN security. Namely, CDN companies, their customers, regular users on the web and criminals who wish to exploit security flaws in web infrastructure.

Security researchers have developed tools and a framework for detecting semantic gaps in HTTP implementations. This can be useful in identifying potential security and performance issues in web technology both to improve it and to exploit existing infrastructure.

Web developers and system administrators who wish to avoid semantic gap attacks ought to stay up to date on which web technologies are known to be vulnerable and choose services that are safe for their purposes. Developers should be mindful of error handling and the processing of HTTP on the servers that they control.

The companies that maintain CDN or other caching services should remain increasingly vigilant of the semantics gaps in the protocol implementations they use in their infrastructure. It would be prudent to provide customers with detailed and up-to-date documentation on how their caching services work and how they can be used in a safe manner. Documentation should include guidelines on which HTTP headers are included in cache keys and which error codes may be cached in responses to requests.

It would also be prudent for governing bodies to prevent network infrastructure from becoming too consolidated. It is vital that the functioning of our networking services are not reliant on only a few companies.

References

- [1] J. Dille, B. Maggs, J. Parikh, H. Prokop, R. Sitaraman, and B. Weihl, "Globally distributed content delivery," *IEEE Internet Computing*, vol. 6, no. 5, pp. 50–58, 2002.
- [2] E. Nygren, R. K. Sitaraman, and J. Sun, "The akamai network: A platform for high-performance internet applications," vol. 44, p. 2–19, aug 2010.
- [3] M. Ghaznavi, E. Jalalpour, M. A. Salahuddin, R. Boutaba, D. Migault, and S. Preda, "Content delivery network security: A survey," *IEEE Communica-*

tions Surveys & Tutorials, vol. 23, no. 4, pp. 2166–2190, 2021.

- [4] D. Moore, C. Shannon, D. J. Brown, G. M. Voelker, and S. Savage, “Inferring internet denial-of-service activity,” *ACM Trans. Comput. Syst.*, vol. 24, p. 115–139, may 2006.
- [5] H. V. Nguyen, L. L. Iacono, and H. Federrath, “Your cache has fallen: Cache-poisoned denial-of-service attack,” in *Proceedings of the 2019 ACM SIGSAC Conference on Computer and Communications Security, CCS ’19*, (New York, NY, USA), p. 1915–1936, Association for Computing Machinery, 2019.
- [6] K. Shen, J. Lu, Y. Yang, J. Chen, M. Zhang, H. Duan, J. Zhang, and X. Zheng, “Hdiff: A semi-automatic framework for discovering semantic gap attack in http implementations,” in *2022 52nd Annual IEEE/IFIP International Conference on Dependable Systems and Networks (DSN)*, pp. 1–13, 2022.
- [7] w3techs.com, “Comparison of the usage statistics of nginx vs. apache for websites,” tech. rep., W3Techs, 2023. Accessed 10/15/2023.
- [8] T. V. Doan, R. van Rijswijk-Deij, O. Hohlfeld, and V. Bajpai, “An empirical view on consolidation of the web,” *ACM Trans. Internet Technol.*, vol. 22, feb 2022.

Unidirectional gateways for security of OT networks

Markus Granström

markus.granstrom@aalto.fi

Tutor: Mohit Sethi

Abstract

It is beneficial from cost and operational point of view to integrate Operational Technology (OT) networks with corporate Information Technology (IT) networks. However, this integration will expose the OT network to general cybersecurity threats. The exposure introduces a need to strictly isolate the OT networks from the IT networks but still retain access to the information inside the OT network. Air-gapping the networks has been used to achieve the isolation but it also prevents all data flow from the OT network. Unidirectional gateway is a way to physically prevent signals from reaching OT network from external network while simultaneously allowing out bound traffic.

KEYWORDS: operational technology, information technology, security, unidirectional gateway

1 Introduction

The term OT is used to refer to systems and devices that interact with the physical environment [1]. Interaction can be either observation (e.g. temperature measurement) or manipulation (e.g. adjustment of air flow). An OT system can be composed of a wide variety of components, such as

electrical, mechanical, pneumatic, and hydraulic. OT also covers systems that are used to manage these devices and systems. Practical examples of OT systems include for example Industrial Control Systems (ICS), Building Automation Systems (BAS), and transportation systems.

Traditionally, OT systems have been isolated, specialized systems, running on purpose-built hardware and software [1]. The current trend of converging OT systems with IT systems provides new opportunities with integration to corporate business systems and cost benefits but also introduces threats from the outside world into the previously isolated environment.

Incorporating standard IT network solutions introduces cost and functional benefits including flexibility of hardware and configuration [2]. Instead of proprietary network technologies, standard Ethernet networks can be deployed using off-the-shelf Local-Area Network (LAN) and Wide-Area Network (WAN) solutions, with the trade off of exposing the OT systems to wide variety new threats.

OT specific solutions to address these issues include unidirectional gateways, OT-specific firewalls, Virtual Local Area Networks (VLAN), and Software-Defined Networking (SDN) [1]. This paper will take a closer look at the use of unidirectional gateways in securing OT networks.

This paper is organized as follows: Section 2 briefly overviews OT in general and the security challenges involved from networking point of view. Section 3 looks at unidirectional gateways and their use in isolating OT networks. Section 4 explores commercial offering to address the identified issues. Section 5 discusses unidirectional gateways as a solution for OT security. Section 6 concludes the paper.

2 Overview of OT

The evolution of OT from traditional physical systems, such as analog mechanical controls, towards digital and IT solutions introduces cost benefits and the connectivity allows for better control over large systems [1]. However, today's connected *smart* systems also introduce new concerns around security.

OT systems can be categorized in the following way [1]:

- **Supervisory Control And Data Acquisition (SCADA)** systems are used to control distributed systems of sensor and instruments that are

connected over the network to the control center where Human Machine Interface (HMI) software allows for centralized monitoring and control [3]. The systems can be widely geographically distributed. They are utilized for example in water and electricity distribution systems and natural gas and oil pipelines. The large scale and geographical distribution of SCADA systems emphasize the need for reliable and secure network communications.

- As opposed to the geographically distributed nature of SCADA systems, **Distributed Control Systems (DCS)** control a geographically co-located production system [1] [4]. Such systems would be used in for example water treatment plants, power stations, natural gas and oil refineries. The DCS is often connected to the corporate network for integration into business processes.
- **Programmable Logic Controllers (PLCs)** can be used to control a singular process in the production pipeline without centralized control but they are also used as low-level components in DCS and SCADA systems [1]. PLCs connect to the manufacturing process via specialized communication network, such as Modbus, but are also connected to the engineering workstation, database server and other control systems via LAN.
- **Building Automation Systems (BAS)** are architecturally similar to DCS as they monitor and control an environment in a single location [1]. Typical responsibilities of a BAS are for example control of temperature and ventilation and physical access control of the building. The sensors and control components communicate over wired or wireless network.
- **Physical Access Control Systems (PACS)** allow for discrete control over who is granted access to an area [1]. Typical examples of PACS include doors, locks, gates, and turnstiles. Often the user is required to have credentials for example in the form of a PIN, a key card, or a fob to be granted access. The reader located at the access point will request authorization based on the credentials from the access control server. Based on the response the control system will either allow or deny access.

- **Industrial Internet of Things (IIoT)** connects interconnected edge devices, such as sensors and instruments, to business processes and applications over an internet connection [1]. The three tiers (edge, platform, and enterprise) of IIoT architecture, as proposed by Lin et al. [5], can be connected over corporate network or a private network overlay over public Internet.

2.1 OT security challenges

The introduction of standard IT technologies, such as Ethernet, Internet Protocol (IP), and remote access and monitoring capabilities exposes OT systems to wide range of cybersecurity threats [1]. Originally OT systems enjoyed a certain amount of security stemming from the use of non-standard software, hardware, and isolation from common communication networks. Connectivity and standard technologies enable easier integration into corporate systems and increase efficiency and control but expose the systems to standard threats as well.

Typical IT security solutions, such as firewalls, might cover some aspects of OT security as well but OT systems often introduce special requirements that need to be considered [1]. Some of the special requirements stem from different risk factors, prioritization and regulatory requirements.

OT systems often have different priorities when it comes to security. As cited by Larkin et al. [6], traditional IT security prioritizes confidentiality over integrity and availability whereas in OT security availability is prioritized higher than integrity and confidentiality [7]. The availability of control and monitoring systems is crucial as any disruption in these systems might lead to catastrophic failure with environmental impact.

3 Unidirectional Gateways

To ensure the availability of OT systems, the secure OT network needs to be isolated from the external network. Whereas traditional firewalls are susceptible to exploitation and misconfiguration, unidirectional gateways can be used to physically prevent data flow from external network into the OT network [1]. This is an effective way to prevent attacks from the external network reaching the secure network. The main concern is to

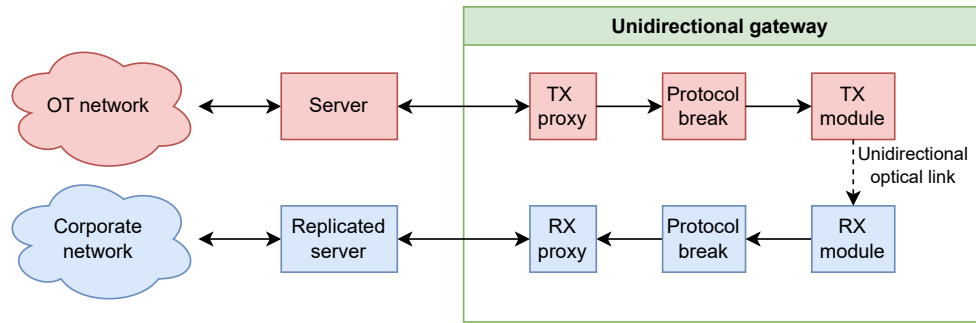


Figure 1. High level design of unidirectional gateway [9] [10]

protect the OT network from outside influence to avoid disruption in critical equipment, such as electrical generators and water pumps, the disruption of which could have serious consequence for property, environment, or the population. Physically isolating the critical system from external networks also reduces the regulatory requirements for the system as no Externally Routable Connectivity (ERC) exists [8].

Unidirectional gateways are useful at all levels of OT networks from large geographically distributed SCADA networks to local DCS and protecting PLCs and other low-level components inside a facility.

Sometimes the terms *unidirectional gateway* and *data diode* are used interchangeably. Even though their hardware can be similar, a unidirectional gateway is more software bound than a data diode [8]. Often data diodes only include support for limited set of protocols whereas a unidirectional gateway can support a wide range of industrial and application protocols.

This section presents the high-level design of a unidirectional gateway and its key features.

3.1 Gateway design

The unidirectional gateway design consists of hardware that only implements outbound traffic and software that replicates the services to the external network [8]. Any disruption incurred towards the exposed replicated services have no effect on the isolated services. Figure 1 shows the high level design of a unidirectional gateway.

The hardware itself is incapable of transmitting data into the higher security OT network. Most often the one way traffic is achieved by optical isolation [8]. The gateway hardware consists of separate transmit (TX) and receive (RX) modules connected only via a fiber optic cable [10]. The

transmitter is typically an Light Emitting Diode (LED) and the receiver is a photodetector. There is no electrical connection between the transmit and receive side and the optical link only allows for one way data transfer.

The gateway software consists of an application proxy and a protocol break [11]. The proxy is responsible for receiving data transmissions from inside the OT network, for example from a data historian server. The proxy takes care of two way communication, such as handshakes and acknowledgements, towards the OT network which might require the support for specific protocols, such as Modbus and Distributed Network Protocol 3 (DNP3). The protocol break takes care of encapsulating the transmission in a connection-less protocol, such as User Datagram Protocol (UDP), for transport over the unidirectional link. At the receiving end the process is reversed and the data fed it into the the replicated server in the external network.

As the data transfer is one way no acknowledgments or other connection-oriented measures can be used over the unidirectional link. To improve the reliability of the unidirectional data transfer message sequence numbers and forward error correction are employed [12].

3.2 Alternative designs

Although the optical isolation is the prevalent solution for achieving unidirectional connectivity, other solution have also been proposed.

Ha et al. [13] propose a Field-Programmable Gate Array (FPGA) based data diode to protect smaller scale components, such as PLCs, with unidirectional data transfer capability. The FPGA is divided into high- and low-security zones. The unidirectional data transfer is achieved using a Modified Dual-port Block Random Access Memory (MDBRAM) shared between the high and low-security zones. The low-security zone is limited to read-only access to the memory whereas the high-security zone would also have write access enabled. The low cost of this approach can enable widespread adoption of the technology in small, dispersed deployments.

Azarmipour et al. [14] explore a virtualization based approach to protecting OT networks with unidirectional data transfer. Virtualization enables the high-security OT application, that resides in the OT network, to run on the same hardware as the low-security replica of that application which is connected to the external network. In their proposal the strict separation between the secure OT network and external network is guaranteed by the hypervisor. The hypervisor is responsible for providing

the unidirectional connectivity between the low- and high-security partitions. The hypervisor chosen is PikeOS by SYSGO [15] which is certified according to the generic standard common criteria.

3.3 Remote management of isolated OT network

Unidirectional connectivity limits the access to the OT network to monitoring only. It is not possible to manage the devices from outside as inbound traffic is not supported. However, often it is necessary to configure or manage the devices or adjust the processes inside the OT network from a remote control center. This can be enabled by deploying an *operations* WAN with remote management capabilities and direct access from the remote control center [8]. Even though this allows for bi-directional traffic into the OT network, as long as any connection between the operations WAN and external network is unidirectional, the system as a whole is still protected.

3.4 Testing

The primary focus in testing is to ensure the integrity and completeness of the transferred data. The received data must match the sent data exactly and no part can be lost during transfer [16]. The general test methodology consists of generating data on the sender side and recording its characteristics such as file name, content hash, and time stamp. The data is then transferred over the unidirectional gateway using the supported protocols, such as TCP, UDP, or FTP. For OT specific protocols, such as Modbus and Open Platform Communications Unified Architecture (OPCUA), the data can be discrete measurements from instruments instead of data files, but the same methodology applies: record the data at the sender side and compare it to the data recorded at the receiving end. The connection is also stressed by using high data volumes.

4 Commercial offering

There is plenty of choice when it comes to commercial unidirectional gateway and data diode offering. OPSWAT, a US based cybersecurity company, has compiled a review of 30 industry leading products [17]. The devices differ greatly in their size and feature set.

The key characteristics of unidirectional gateways and data diodes can

be summarized as follows:

- **Certifications:** Which international and national certifications are applicable to the device and required by the target environment, e.g. *International Electrotechnical Commission (IEC) 62443, Common Criteria, and NATO Information Assurance Product Catalogue (NIAPC)*.
- **Gateway technology:** What technology base is used by the device to achieve unidirectional connectivity, e.g. *optical, electrical, electromagnetic, hardware, or software*.
- **Transfer protocols:** Which transfer and application protocols are supported by the device, e.g. **industrial protocols:** *Modbus, Open Platform Communications (OPC), Message Queuing Telemetry Transport for Sensor Networks (MQTT-SN)*, **IT monitoring applications:** *Security Information and Event Management (SIEM) Integration, Syslog*, **standard:** *UDP, Transmission Control Protocol (TCP)*, **file transfer:** *File Transfer Protocol (FTP), Secure File Transfer Protocol (SFTP), Network File System (NFS), Server Message Block (SMB)*, **email transfer:** *Simple Mail Transfer Protocol (SMTP), Post Office Protocol 3 (POP3)*.
- **Cryptographic standards:** Cryptographic algorithms supported by a device can include **symmetric** (e.g. *Advanced Encryption Standard (AES)*) and **asymmetric** (e.g. *Elliptic Curve Cryptography (ECC)*). The devices might further be characterized based on the detailed cryptographic primitives supported (e.g. elliptic curves *Curve25519* and *Ed25519*).
- **Transfer rate:** What kind of transfer rates is the device capable of? Ranging from *1 Mbps* to *10 Gbps*.
- **Physical Dimension and weight:** The physical size of the device. Ranging from small *0.5 kg units* to *2 X 1U rack mountable servers 12.2 kg / server*
- **Cost:** The cost of unidirectional gateways can vary from a few hundred US dollars to thousands of US dollars depending on the hardware capabilities and protocols supported.

4.1 Product comparison

Commercial unidirectional gateway products can have vastly different features and properties [17]. Five products are chosen to highlight the differences:

- **OPSWAT** *NetWall USG*
- **Advenica** *SecuriCDS Data Diode DD1000i*
- **Arbit Cyber Defence Systems** *Data Diode 1 Gbps*
- **Bilge SGT** *DataBrokerX 10G*
- **Owl Cyber Defense** *OPDS-1000*

Table 1 compares the gateway technologies and the transfer rates. Table 2 compares the supported transfer protocols. Table 3 compares the physical dimensions and weight.

Product	Gateway Technology	Transfer rate
NetWall USG	Electrical	10 Gbps
DD1000i	Hardware	1 Gbps
Data Diode 1 Gbps	Optical	1 Gbps
DataBrokerX 10G	Optical	3.6 Gbps
OPDS-1000	Optical DualDiode	1 Gbps

Table 1. Comparison of gateway technologies and transfer rates.

Product	Transfer protocols
NetWall USG	Modbus, OPC, MQTT-SN, SIEM Integration, Syslog, FTP, SMB
DD1000i	UDP, TCP, Syslog, NTP, FTP, SFTP, NFS, SMB, SMTP, POP3
Data Diode 1 Gbps	SMTP, FTP, SFTP, SMB, NFS, NTP, TCP, UDP, HTTP, HTTPS, NiFi, SysLog
DataBrokerX 10G	HTTP, HTTPS, LDAP, LDAPS
OPDS-1000	UDP, TCP/IP, SNMP, SMTP, NTP, SFTP, FTP

Table 2. Comparison of transfer protocols.

Product	Rack units	Weight
NetWall USG	2 X 1U	2 x 12.2kg
DD1000i	1U	9 kg
Data Diode 1 Gbps	2 x 1U	9.6 Kg
DataBrokerX 10G	1U	22.0 kg
OPDS-1000	1U	3.6 kg

Table 3. Comparison of physical dimensions and weight.

5 Discussion

When considering protecting e.g. critical infrastructure or manufacturing plants unidirectional gateways offer stronger security guarantee over conventional firewalls. The inability to physically relay traffic from external network into the high security OT network effectively blocks any attempts of online infiltration originating from outside. This has to be a priority consideration when disruptions in manufacturing or delivery processes are unacceptable. The wide range of commercial products available caters for the needs of large geographically distributed SCADA systems as well as small localized DCSs.

6 Conclusion

Special care needs to be taken when securing critical OT networks because of different priorities compared to normal IT networks. Unidirectional gateway is an efficient way to isolate critical networks and prevent attacks or signals from reaching the network from outside. Most often optical isolation is used to achieve unidirectional connectivity but other approaches exist as well. Many commercial options are available ranging from small light weight device to heavy duty rack servers.

References

- [1] K. Stouffer, M. Pease, C. Tang, T. Zimmerman, V. Pillitteri, and S. Lightman, "Guide to operational technology (ot) security," tech. rep., National Institute of Standards and Technology, April 2022. doi: 10.6028/NIST.SP.800-82r3.ipd.
- [2] S. Hachana, F. Cuppens, and N. Cuppens-Boulahia, "Towards a new gener-

- ation of industrial firewalls: Operational-process aware filtering,” in *2016 14th Annual Conference on Privacy, Security and Trust (PST)*, pp. 615–622, IEEE, December 2016. doi: 10.1109/PST.2016.7906996.
- [3] G. Yadav and K. Paul, “Architecture and security of scada systems: A review,” *International Journal of Critical Infrastructure Protection*, vol. 34, p. 100433, 2021. doi: 10.1016/j.ijcip.2021.100433.
- [4] S. Ali, “Cybersecurity management for distributed control system: systematic approach,” *Journal of Ambient Intelligence and Humanized Computing*, vol. 12, no. 11, pp. 10091–10103, 2021. doi: 10.1007/s12652-020-02775-5.
- [5] S.-W. Lin, B. Miller, J. Durand, G. Bleakley, A. Chigani, R. Martin, B. Murphy, and M. Crawford, “The industrial internet of things volume g1: Reference architecture,” tech. rep., Industrial Internet Consortium, June 2019.
- [6] R. D. Larkin, J. Lopez, J. W. Butts, and M. R. Grimaila, “Evaluation of security solutions in the scada environment,” *ACM SIGMIS Database: the DATABASE for Advances in Information Systems*, vol. 45, p. 38–53, February 2014. doi: 10.1145/2591056.2591060.
- [7] J. Weiss, *Protecting industrial control systems from electronic threats*. Momentum Press, 4th ed., 2010.
- [8] A. Ginter *et al.*, “Unidirectional security gateways: stronger than firewalls,” *Proceedings of the ICALEPCS, San Francisco, CA, USA*, pp. 6–11, 2013.
- [9] Y. Heo, B. Kim, D. Kang, and J. Na, “A design of unidirectional security gateway for enforcement reliability and security of transmission data in industrial control systems,” in *2016 18th International Conference on Advanced Communication Technology (ICACT)*, pp. 310–313, IEEE, 2016. doi: 10.1109/ICACT.2016.7423372.
- [10] P. T. N. T. Đ. An and T. T. Tin, “Design of unidirectional security gateway device for secure data transfer,” December 2021. doi: 10.15625/vap.2021.00107.
- [11] M. B. de Freitas, L. Rosa, T. Cruz, and P. Simões, “Sdn-enabled virtual data diode,” in *Computer Security: ESORICS 2018 International Workshops, CyberICPS 2018 and SECPRE 2018, Barcelona, Spain, September 6–7, 2018, Revised Selected Papers 2*, pp. 102–118, Springer, 2019.
- [12] L. Honggang, “Research on packet loss issues in unidirectional transmission,” *Journal of Computers*, vol. 8, pp. 2664–2671, October 2013. doi: 10.4304/jcp.8.10.2664-2671.
- [13] S. S. Ha, H. Beuster, T. R. Doebbert, and G. Scholl, “An fpga-based unidirectional gateway proposal for ot-it network separation to secure industrial automation systems,” in *2023 IEEE 21st International Conference on Industrial Informatics (INDIN)*, pp. 1–6, IEEE, 2023. doi: 10.1109/INDIN51400.2023.10218126.
- [14] M. Azarmipour, C. von Trotha, C. Gries, T. Kleinert, and U. Epple, “A secure gateway for the cooperation of information technologies and industrial automation systems,” in *IECON 2020 The 46th Annual Conference of the IEEE Industrial Electronics Society*, pp. 53–58, IEEE, 2020. doi: 10.1109/IECON43393.2020.9254634.

- [15] SYSGO, “Pikeos certifiable rtos & hypervisor,” 2023. <https://www.sysgo.com/>. Last accessed 12 October 2023.
- [16] OPSWAT, “Testing a unidirectional gateway/data diode,” 2023. <https://www.opswat.com/blog/how-we-test-unidirectional-gateways>. Last accessed 9 November 2023.
- [17] OPSWAT, “Unidirectional security gateway and data diode guide,” 2023. <https://www.opswat.com/products/unidirectional-security-gateway-guide>. Last accessed 10 October 2023.

A Survey of Methods for Robust Camera based Pose Estimation

Max Ulfves

max.ulfves@aalto.fi

Tutor: Aziza Zhanabatyrova

Abstract

Camera based pose estimation is relevant in many fields, from augmented reality to autonomous driving. This paper presents a survey of selected methods for camera-based pose estimation, with an emphasis on robustness in a variety of conditions. The study discusses both methods using marker based methods such as RANSAC, and more recent SLAM methods. Furthermore, we see how Kalman filters and Extended Kalman filters, has been used to improve the quality of measurements.

KEYWORDS: robust camera based pose estimation, camera relocalization, robust pose estimation, robust slam

1 Introduction

Camera based pose estimation is the act of estimating the position and orientation of a camera in 3D space, based on the image data produced by the camera. This is generally performed by knowing the intrinsic parameters of the camera and real-world coordinates of a feature set. A set of image features is identified in the camera image, after which we try to estimate the extrinsic camera matrix E , so that we minimize the reprojection error of the observed features. (1)

This process is generally used within augmented reality and autonomous driving, as it can function as a relatively inexpensive way of obtaining a more precise estimate than what could be provided by GPS. Unlike GPS it need not depend on satellite data, which makes potentially more reliable in situations where it is not available. Especially in the context of autonomous driving, it is imperative to know where exactly on the lane the car is, where the car is positioned in a curve. An incorrect estimate of less than a metre could have disastrous consequences for the passengers of the vehicle.

Robustness in computer science refers to “the degree to which a system or component can function correctly in the presence of invalid inputs or stressful environmental conditions” (2). In camera based pose estimation, robustness can mean how well the model functions when features are detected incorrectly, when the real-world 3D-coordinates for certain features are incorrect, or when other factors could cause faulty data. (1)

This paper will go over different studies that are used for camera based localization. We will consider methods relying solely on the camera’s image data, of various formats and quality. Studies primarily relying on LIDAR and other sensory information will not be considered. In the background section, we will discuss how many current implementations work, and what principles are central in many, if not most procedures. In the third section, we go over a selection of recent methods that have shown promising results, especially for achieving robustness in difficult conditions. The fourth section clarifies the findings of this study, and gives an overview over what has been discussed previously.

2 Background

In order to understand the recent advances in camera based pose estimation, and possible future research, it is essential to know what methods have been used so far.

2.1 Camera space, world space, image space

Let us start by clarifying some concepts related to camera based pose estimation. **World space** refers to the coordinate system of the real world. In the case of a self driving vehicle, it could be earths coordinate system for example or something to represent the surroundings in which the car

is moving. **Camera space** on the other hand is a coordinate system in which the camera's origin point is in the origin, and the camera is pointing along some axis. Lastly, **image space**, is a 2d coordinate system of the image captured by a camera.

$$p_{image} = p_{world} \cdot I \cdot M \quad (1)$$

Equation 1 describes the transformation of coordinates from world space to image space. I represents the intrinsic parameters of the camera, such as lens distortion or focal length. p_{world} are the coordinates in world space, and p_{image} are the coordinates in image space. (3)

M is the transformation matrix of the camera, which includes the translation and rotation of it. When trying to estimate a camera's pose in world space, we typically want to find a matrix M , so that the re-projection error of Equation 1 is minimized. (3)

2.2 Marker-based pose estimation

The perspective n-point (PnP) problem was formally described by Fischler and Bolles as "Given the relative spatial locations of n control points, and given the angle to every pair of control points from an additional point called the Center of Perspective (CP), find the lengths of the line segments ("legs") joining the CP to each of the control points". (3)

Fischler and Bolles also describes Random sample consensus (RANSAC), which is a method for producing an estimate for a solution to the PnP-problem when there is an abundance of points, using random sampling. (3)

One practical implementation of this is marker-based pose estimation, where the world coordinates, $p_{world} \in \mathbb{R}^3$, of observed features are known. Then we can use various image processing techniques for estimating the image coordinates, p_{image} , of these features. (3)

A suitable image feature can be just about anything that a camera can detect, and bind to some mathematical constraint. Typically, augmented reality tags, such as April tags or ArUco markers Figure 1, are used in applications where high precision is required. (4) These markers are useful, as they are computationally easy to detect, and decode, due to their rectilinear nature. Furthermore, the image position of the markers' corners can be estimated with subpixel precision, as they are in the intersection of two sides of the markers. (4)

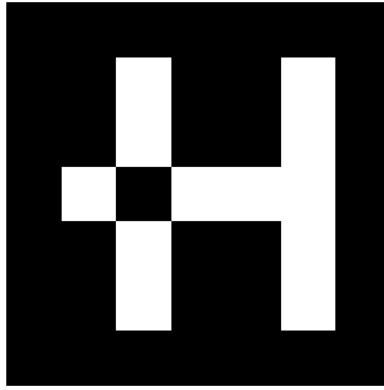


Figure 1. ArUco marker 0x29A, generated using the OpenCV library

One could also train a neural network to detect other types of markers in images, for example if it is not desirable to have visible markers in the scene. Such marker could be almost anything that is easily detectable, and that facilitates a low amount of false-positive identifications.

Challenges in marker-based pose estimation are related to producing accurate estimates when the markers are obscured, or not detectable. We also typically want to obtain robustness against misclassification of markers, which can be caused by poor image quality. Having a predefined set of markers also requires a high level of control over the environment. In the context of a national road network, the relocation or obstruction of markers renders this form of pose estimation ineffective.

Line based feature tracking

(author?) (1) presents a method for tracking lines, in combination of points. In this study, a box with known proportions is tracked using a camera, and the edges of the box are used instead of markers. The line constraint is very similar to a point constraint, in that it measures how far the projected line is from the detected line in the image. This presents an advantage as lines are often easier to detect consistently, compared to points. Ababsa and Mallem shows experimentally that their method is robust against partial occlusion of the tracked object, and that their model is able to track the object, even in a cluttered environment. (1)

2.3 Markerless pose estimation

Although we do not have access to placed markers with known position we can still find traceable features in most images. Convolutional neural net-

works (CNNs) are particularly good at identifying features in images, as demonstrated in for example an experiment by Melekhov et al.(5). Finding a certain set of features can, when combined, narrow down the state space enough to allow for a good estimate of a camera's position.

There are challenges with these types of methods. As neural networks generally works like black boxes, in that once they are trained, we do not precisely know exactly how they produce certain results. If an error is discovered, it is often difficult to correct it by other means than introducing more training data.

2.4 Kalman Filters

Many modern algorithms for camera relocalization, use extended Kalman filters (EKF) for stabilizing their pose estimates. Kalman filters, introduced by Kalman in 1961, are a collection of recursive methods for estimating measurements in a dynamic systems, that are often subject to noisy data(6).

Kalman filters are generally useful in camera based pose estimation, as the object the camera is attached to, is subject to several physical constraints. With Kalman filtering, one can for instance account for the measured acceleration between two position estimates, and after that update the state space according to the new measurement.

3 Methods for robust Pose Estimation

In the following section we will go over a variety of methods that have shown promising results, especially in environments with noisy input data.

3.1 SLAM

Simultaneous Localization and Mapping (SLAM), is a collection of techniques used for both estimating the position of an actor in an environment, and collecting information about the environment (7).

There are five parts of SLAM, initialization, tracking, mapping, relocalizaion and global map optimization (7). Data collection in SLAM can be difficult for several reasons. The cameras used can have different intrinsic parameters, they can be pointed in different directions in relation to the direction of the vehicle, and be subject to different weather condi-

tions. Environments also change over time, as trees grow, buildings are built, repainted and destroyed.

RobustLoc

RobustLoc is a model for estimating a camera's pose in a small world map, such as a university campus. RobustLoc uses a CNN for creating a feature map from the input images, vector embeddings and a branched decoder to produce an estimate for the position and rotation of the camera. An advantage of RobustLoc is that it can utilize images from neighboring vehicles as well, to produce a more accurate estimate. Several experiments using the model showed promising results, as it seems to be fairly robust against various distortions of the captured images, such as adding noise. (8) The model is tested and compared to other models on the 4Seasons dataset, and it outperforms in most reported categories (8). As the 4Seasons dataset includes data from each season, it is remarkable that RobustLoc achieves the relatively high accuracy of 1 to 22 meters. (8)

RobustLoc requires extensive training to perform well, and it could be questioned how well scales (8). The model would likely be useful in small scale environments, such as for relocalization of autonomous delivery robots, operating on a confined area. It is however not proven that RobustLoc would work very well in a larger environment such as a city or a country. Delivery trucks, which often stick to the trunk of the road network, could potentially benefit from RobustLoc, as they mostly need to know where they are on the route they are driving.

Semantic SLAM

Semantic SLAM, is based on the idea of assigning semantic labels to observed objects using sensory data. The idea is that a location can be estimated, based on the occurrence of observed objects, which can be updated. (9)

By labeling observed data, and categorizing it as objects, we can account for different states of an object. This model can potentially provide robustness against changes in the environment, as many changes would produce semantically similar objects.

As image data is converted into object metadata, before the location is estimated, it does not really matter what type of camera is used, given that the image classifier is trained to process data from that camera. The orientation of a camera in relation to the car is also irrelevant, as the location is estimated based on the 3D scene.

(An object based model could also potentially distinguish between permanent and non-permanent objects in the environment)

LSD-SLAM

Large-Scale Direct monocular SLAM (LSD-SLAM) distinguishes itself from other SLAM methods, in that it is a featureless algorithm (10). For feature-based algorithms to work efficiently, there needs to be an agreement on what features are significant(10).

In LSD-SLAM, the global map consists of several key frames, which are obtained from captured point clouds using a monocular camera. The key frames are structured in a graph-like manner, so that each key frame has some neighbours(10). Edges in the graph-model, represent the transformation between two key frames. In addition to the image of the key frame, an inverse depth map is also stored, as well as the variance of the inverse depth map. (10)

LSD-SLAM provides robustness against different camera setups, in that it builds a complete map of the environment. Even with differing camera equipment and other conditions, the produced representation of the global map will be fairly similar. There may however be differences in the produced quality of each contribution, but these could be mitigated by imposing quality requirements for updating the map.

It is unclear how robust this method is against changing environmental conditions, like weather. As the model uses 3D-depth-map data to produce position estimates, it should be fairly consistent against different lighting conditions. However, snow may disrupt the textures of the surrounding environment, which could, unless mitigated by other means, reduce the model quality.

Key frame quality can be improved by using better sensory equipment, such as stereo cameras and LIDAR. We will not delve deeper into this in this paper.

3.2 Image Enhancement Techniques

For many problems, relatively simple techniques can be utilized for achieving robustness. Especially problems related to varying light conditions can in many cases be mitigated using image filtering.

Image exposure

There are several techniques to correct over/under exposure. One way is to use a trained neural network, which has been trained on large dataset of images(11). For marker based pose estimation, one could train a large dataset to identify markers in various conditions. In SLAM, this is usually not a problem, as

Identifying regions of interest

In a scenario where we need the position in relation to the road, there are several ways of refining the training data, to remove obscured or otherwise irrelevant parts of the image. Hillel et al. lays out several methods used in the context of autonomous driving. (12) The difficult part is often to identify precisely what identifies an obscuring object. In autonomous driving it is usually anything that is not the road, or a sign, but depending on the environment it could be subjected to a wide range of requirements.

4 Discussion

Clearly, camera based pose estimation is a very broad field. From this follows that there is a multitude of different scenarios in which this form of pose estimation is needed. Each of these methods have vast differences in underlying conditions and requirements, and it can be difficult to compare performance of different methods. Additionally, the way in which the different methods attempt to add robustness is very different. For example, with methods such as RobustLoc, we have very different conditions, compared to a scenario using camera based pose estimation for some augmented reality project.

As briefly mentioned earlier, marker based methods require highly controllable, unchanging environments. While it may not work very efficiently in certain environments, such as large road networks, marker based methods offer flexibility, in that we can completely control what parts of the image the model uses for pose estimation. It is also comparatively easy to construct filters that detect AR markers, and reduce noise in images in which they occur, as they are constructed to be specifically easy to detect for cameras.

Some SLAM methods are computationally complex, and it is unlikely that many of them would work well on a large road network. Especially methods that use point clouds as a means to map the environment, may

struggle with computation of the environment. Recognizing what part of the images are going to be part of the environment long-term is an area that is going to be interesting in future studies.

5 Conclusion

In this article we have analyzed a variety of methods used in camera based positioning. We started by discussing methods based on markers, where we have a high control of the environment. These methods can be extended to use other forms of markers, such as lines on objects of known position. We also briefly discussed how any type of feature that can be detected can be used as a marker.

Furthermore, we analyzed some recently introduced methods such as RobustLoc and CNN-SLAM, which have shown promising results for determining camera poses. We also looked into semantic slam, and discussed the prospects of using it in larger environments.

References

- [1] F. Ababsa and M. Mallem, “Robust camera pose estimation combining 2d/3d points and lines tracking,” in *2008 IEEE International Symposium on Industrial Electronics*, pp. 774–779, 2008. doi: 10.1109/ISIE.2008.4676964.
- [2] “Ieee standard glossary of software engineering terminology,” *IEEE Std 610.12-1990*, pp. 1–84, 1990. doi: 10.1109/IEEESTD.1990.101064.
- [3] M. A. Fischler and R. C. Bolles, “Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography,” *Communications of the ACM*, vol. 24, no. 6, pp. 381–395, 1981. doi: 10.1145/358669.
- [4] J. Wang and E. Olson, “Apriltag 2: Efficient and robust fiducial detection,” in *2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 4193–4198, 2016. doi: 10.1109/IROS.2016.7759617.
- [5] I. Melekhov, J. Ylioinas, J. Kannala, and E. Rahtu, “Relative camera pose estimation using convolutional neural networks,” pp. 675–687, 2017. doi: 10.1007/11744023₃2.
- [6] R. E. Kalman, “A new approach to linear filtering and prediction problems,” *Transactions of the ASME—Journal of Basic Engineering*, vol. 82, no. Series D, pp. 35–45, 1960. doi: 10.1115/1.3662552.
- [7] T. Taketomi, H. Uchiyama, and S. Ikeda, “Visual slam algorithms: A survey from 2010 to 2016,” *IPSJ Transactions on Computer Vision and Applications*, vol. 9, no. 1, pp. 1–11, 2017. doi: 10.1186/s41074-017-0027-2.
- [8] S. Wang, Q. Kang, R. She, W. P. Tay, A. Hartmannsgruber, and D. N. Navarro, “Robustloc: Robust camera pose regression in challenging driving environments,” 2023. doi: 10.1609/aaai.v37i5.25765.
- [9] J. Civera, D. Gálvez-López, L. Riazuelo, J. D. Tardós, and J. M. M. Montiel, “Towards semantic slam using a monocular camera,” in *2011 IEEE/RSJ international conference on intelligent robots and systems*, pp. 1277–1284, IEEE, 2011. doi: 10.1109/IROS.2011.6094648.
- [10] J. Engel, T. Schöps, and D. Cremers, “Lsd-slam: Large-scale direct monocular slam,” in *European conference on computer vision*, pp. 834–849, Springer, 2014. doi: 10.1007/978-3-319-10605-2₅4.
- [11] M. Afifi, K. G. Derpanis, B. Ommer, and M. S. Brown, “Learning multi-scale photo exposure correction,” in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 9157–9167, 2021. doi: 10.1109/CVPR46437.2021.00904.
- [12] A. B. Hillel, R. Lerner, D. Levi, and G. Raz, “Recent progress in road and lane detection: a survey,” *Machine vision and applications*, vol. 25, no. 3, pp. 727–745, 2014. doi: 10.1007/s00138-011-0404-2.

Advancements in Neural Radiance Fields: A Comparative Study of Instant NeRF and BakedSDF

Niko Nästi

niko.nasti@aalto.fi

Tutor: Matti Siekkinen

Abstract

This paper explores the significant advancements in the domain of Neural Radiance Fields (NeRF), a pivotal technology in neural rendering and computer graphics. The primary focus is on two notable developments: Instant NeRF and BakedSDF. This paper begins by detailing the key concepts underlying NeRF and its implementation in creating hyper-realistic view synthesis, essential for applications like virtual reality. Then the paper proceeds into presenting Instant NeRF and BakedSDF, highlighting the nuances of each method in terms of training speed and image quality. A comparative analysis is presented, examining these techniques across three parameters: Peak Signal to Noise Ratio (PSTNR), Structural Similarity Index (SSIM), and Learned Perceptual Image Patch Similarity (LPIPS). The paper concludes with a discussion on the implications of these advancements for the future of neural rendering, emphasizing the potential for wider application and efficiency in real-time rendering scenarios. Through this study, the paper aims to provide a comprehensive understanding of the current state and future potential of NeRF technology in computer graphics and virtual environment creation.

KEYWORDS: *Neural Radiance Fields (NeRF), review, Instant NeRF, BakedSDF, Neural Rendering, View Synthesis*

1 Introduction

In computer graphics, there has been significant efforts to make view synthesis look hyper realistic. These techniques are needed for example for virtual reality applications to give users a realistic feeling of the virtual reality. Mildenhall et al.[1] made a breakthrough in this field by introducing Neural Radiance Fields (NeRF) that could produce hyper realistic view synthesis. Although that was a remarkable event the view synthesis was relatively slow since training of the neural network takes several days. A couple of years after the release of the original NeRF there have been multiple steps to make the technology faster. One worth of attention is Instant Nerf [2] that can produce same Neural Radiance Fields much faster while the image quality preserves.

View synthesis is done from a set of images that captures an object from different viewpoints. These images are given to the multilayered perceptrons (MLP) that produce a model from the input data that. The MLP is doing the rendering and the object creation for the output image. This image can be rendered in diverse ways, for example by using classical rasterization techniques or ray marching.

This paper goes through the key concepts of the NeRF and introduces the original NeRF, Instant NeRF, and BakedSDF. After introduction there is a comparison between Instant NeRF and BakedSDF with three different quality estimators. There are three parameters: Peak Signal to Noise Ratio (PSTNR), Structural Similarity Index (SSIM), and Learned Perceptual Image Patch Similarity (LPIPS) that are used to compare the NeRF methods. Although there is a minor difference between those two methods it is still significant, because if the context of application requires fidelity image quality even the fraction of a difference is significant.

Finally, there is a discussion on how hashing affects the training time and what rendering techniques should be considered when rendering neural volumes.

2 Key Concepts

2.1 Multilayer perceptron

Multilayer perceptron (MLP) is a perceptron-based neural network capable of learning nonlinear optimization problems. MLPs are simple fully

connected networks. Perceptron algorithms are based on feed forward network backpropagation, which is used to optimize the parameters of the neural network. MLP uses non-linear activation functions that produce solutions on a continuous plane. [3]

This paper concentrates on MLPs that are coordinate based. This means that the MLPs input is low-dimensional point. The coordinate based MLP is trained to provide representation measurement such as images, volume density, signed distance or novel view synthesis. [4]

To produce state-of-the-art view synthesis that was produced in the beginning of the neural radiance fields the input coordinates were mapped as Fourier features before feeding it to the MLP [1]. This is referred to as positional encoding. Positional encoding is used in log-linear spaced frequencies for each dimension [4]. Tancik et al [4] has demonstrated this method to be more effective than wrapping coordinates around the circle or Gaussian encoding.

MLPs are good to learn higher dimensional frequency functions and MLPs are biased when trying to use lower dimensional points as inputs. This is why positional encoding is used in lower dimensional input data. This kind of positional encoding maps the coordinates to a higher dimension so the MLP can learn higher frequency details.

3 Neural Radiance Fields

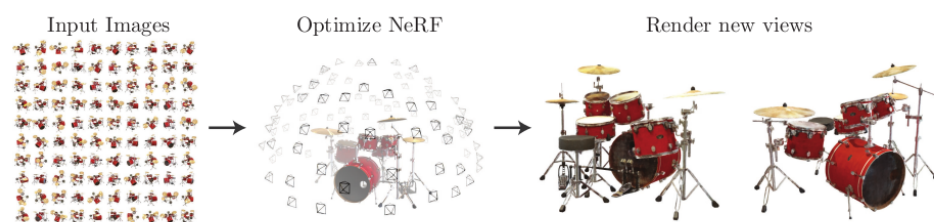


Figure 1. Overview of NeRF process. This picture is from the original NeRF paper [1].

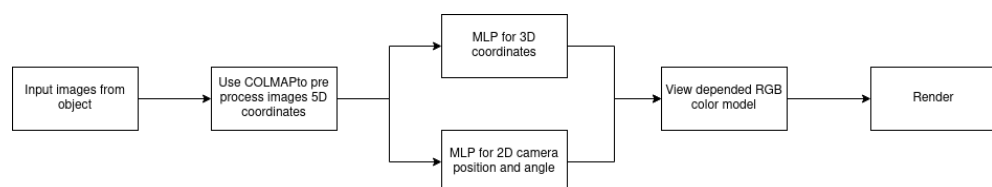


Figure 2. Work flow of NeRF

Neural radiance fields (NeRF) is a method in neural rendering research to produce state-of-the-art rendering results [1]. The name comes

from a neural network describing the scene's radiance in a continuous function. NeRF uses five-dimensional input (3 dimensions for a point in space and 2 dimensions for viewing direction) for two ReLU activated MLPs that produce volume density and directional emitted color [1].

The first MLP utilizes positional encoding to the 3D coordinate with 8 fully connected layers with 256 channels per layer and generates the volume density and a 256-dimensional feature vector. The feature vector is then linked together with the camera viewing direction that is also positionally encoded and transmitted to a supplementary fully connected layer that outputs the directional emitted color. [1]

One main technique to make NeRF work is to use positional encoding when teaching the network. This technique has a significant effect compared to the other methods for coordinate mapping [4]. This technique maps coordinates, in this case 3D point and 2D point, to a hypersphere that has a set of sine waves. The network that is trained in NeRF is modeling low dimensional functions, this means that the MLP is a kernel machine based around dot production of the inputs and the dot product of Fourier map inputs is a stationary kernel.

A Neural Radiance Field represents a scene describing the volumetric density and directional emitted radiance within a continuous spatial domain. Then rendered color of any scene piercing ray with classical volume rendering. The volume density can be understood as the incremental probability of a ray ending at an infinitesimal particle located at point in space. [1]

In the original paper Mildenhall et.al. [1] compared NeRf to other neural rendering techniques including Neural Volumes, Scene Representation Networks, and Local Light Field Fusion. In conclusion other techniques were slower to render images, produced lower quality images or is more space consuming for the neural models, for example Local Light Field Fusion made a model that took 15GB of space, when compared to NeRF that needed 5MB for the weights of the network [1].

4 Disadvantages of NeRF

The main disadvantage of NeRF is that it is slow to produce the neural network needed for rendering. Mildenhall et.al. [1] states that all scenes that were produced took about 12 hours of training even with a relatively good graphics card. This means that NeRF is not ready for real time

applications.

Also, the original NeRF produces stationary scenes, this means that the object that is modeled cannot be animated. This also reduces the usability of the technique in hand. This does not mean that the model is not interactive, in the scene you can move the camera around and the specular artifacts alter according to the view.

The original NeRF lacks any parameter tweaking features such as changing the color or size of the modeled object in the scene. Even if the model were modifiable, the editing process would be cumbersome since the final image that is rendered is not done in real time.

There has been an attempt to have NeRF-style rendering for consumer grade hardware called Mobile NeRF [5]. This technique makes editing of the object possible in real time. The main idea is to utilize the scene in a traditional rendering pipeline. This way the renderer MLP can be executed in a shader making the process massively parallelized. The final image is made in real time, but it still lacks such quality as the original NeRF can produce.

Mobile NeRF stores the produced model in OBJ-format. This enables variety applications since OBJ-format is widely used. Also, by using familiar object representation format the threshold for useful applications is lower.

One technique that this paper is going to review is Instant NeRF [2]. Instant NeRF is successfully reducing time used for training the networks and rendering.

5 Instant NeRF

Instant NeRF was introduced by Müller et. al.[2] The difference between NeRF and Instant NeRF is clear. The former produces neural network and renders models in magnitude of seconds. There still is a trade off in produced quality. In Instant NeRF the training can be completed in seconds or if necessary the training time can be increased to minutes [2]. This means that the training time is lowered by magnitude of hours compared to the test data between the Instant NeRF and NeRF. The rendering of the final image is also significantly faster in Instant NeRF. The rendering of final image is done in magnitude of milliseconds making this technique real time application.

The increment in training speed is done with hash tables in positional

encoding with fewer parameters. Müller et. al. uses multiple hash tables with distinct resolutions and trainable encoding parameters. This ensures that there is no hash collision since the encoding has different emphasis in each resolution level as long as the resolution is high enough. This technique anticipates on the neural network to learn hash collisions. This method reduces the complexity of the network while improving the performance. [2]

These hash tables also make the memory layout more predictable [2], so the utilization of GPU is much better, since there is no need to wait for fetching data from outer levels of caches.

Parametric encoding comes with a tradeoff between quality and training speed. With sparse encoding the quality might get better but then there is going to be an unoptimized number of parameters. Unoptimized in a way that if the input picture has empty background and the object in the middle there is going to be as many features in the background as in the object itself. On the other hand, without any encoding the quality of the final image is poor. Müller et. al.[2] overcame this problem with multiple separate hash tables and relaying that the learned has tables are optimal. The performance is even more increased because in Instant NeRF you can set the cache size for the hash tables.

6 BakedSDF

Yariv et. al. [6] took a different approach to accelerate NeRF-like volume rendering. Instead of optimizing the MLP's they made a high-quality triangle mesh out of the volume rendered surface reconstruction. This enables consumer grade hardware to utilize NeRF in real time since common hardware is made for triangle rendering rather than neural rendering.

BakedSDF first constructs neural radiance field representation from the input images. This is done in three stages. First the surface of the object to be rendered is reconstructed using volume rendering. The volume rendered surface is then baked to triangle mesh and used for appearance model trained with spherical Gaussians.[6]

The triangle mesh is made by the volume rendering representation to ordinal 3D grid which is then run through Marching Cubes [6]. While marching cubes the 3D points are sampled from training data along the ray. The samples give the rendering weight for each sample. Then there is decided a threshold for weights to be added to the final grid. This prevents

	Outdoor Scenes			Indoor Scenes		
	PSNR \uparrow	SSIM \uparrow	LPIPS \downarrow	PSNR \uparrow	SSIM \uparrow	LPIPS \downarrow
BakedSDF (offline)	23.40	0.619	0.379	30.21	0.888	0.243
BakedSDF (real time)	22.47	0.585	0.349	27.06	0.836	0.258
Instant NeRF (offline)	22.90	0.566	0.371	29.15	0.880	0.216

Table 1. Section of results that is produced by Yariv et. al. [6]

noise artifacts from the final mesh.

The triangle mesh representation is widely used graphics primitive that allows consumer grade hardware to accelerate rendering. The triangle mesh also enables modifications as well as simulation for the object to be rendered [6]. This makes NeRF more useful for computer graphics applications. There are still limitations for this technology. Because the final representation is made with triangle mesh the common problems of high detailed surfaces are still apparent. Also, the renderer is not suited for transparent and semi-transparent materials. [6]

7 Comparison between Instant NeRF and BakedSDF

Yariv et. al. [6] did not mention neural network training time so the comparison between Instant NeRF and BakedSDF is going to be limited in rendering time, Peak Signal To Noise Ratio (PSTNR), Structural Similarity Index (SSIM), and Learned Perceptual Image Patch Similarity (LPIPS). In the table 1 Outdoor Scenes and Indoor Scenes are scenes that are made of pictures of the object in either outdoor environment or indoor environment. The main difference between these scenes are that the outdoor scenes have more complicated background textures than in indoor scenes.

In offline BakedSDF outperforms Instant NeRF in all but one parameter: LPIPS. In other words, BakedSDF produces outputs that are more similar to the ground truth than Instant NeRF. Instant NeRF has lower LPIPS in indoor scenes indicating that in a less noisy environment the Instant NeRF captures textures and shapes well.

One interesting result is that the real time BakedSDF outperforms in some parameters the offline BakedSDF. This is likely because the offline BakedSDF is taken from intermediate volume rendering step that is before the "baking" step that is included in the real time renderer.

Yariv et. al. did another test to compare the frame rates between Instant NeRF and BakedSDF. The test was made in 1920×1080 resolution. BakedSDF had on average 72.21 fps and Instant NeRF had 3.78 fps. [6]

This proves that the BakedSDF is faster to render than Instant NeRF. This is likely because the BakedSDF utilized triangle mesh that can be accelerated easier than other neural rendering methods. Also, BakedSDF uses more memory than Instant NeRF. [6]

8 Discussion

There are multiple ways to increase performance of the Neural Radiance Fields. In every part of the process there is room for speed ups and quality enchantments. One enchantment possibility for the NeRF is the positional encoding. It is one of the most crucial parts since this step makes the complexity of the model. The complexity comes from the feature vectors stored in suboptimal data structure. To increase training speed Instant NeRF [2] uses hash table data structure to store feature vectors.

Another way to increase performance is to utilize triangle mesh in rendering phase. Since triangle meshes are widely used in the computer graphics community, the industry has made improvements in the hardware side that is meant to accelerate triangle mesh structures.

Marching cubes are preferred way to render final image since it uses volumetric data. There is hardware accelerated designs for ray tracing, but since the input data is volumetric ray tracing would not fit for in the purpose of NeRF. Relying only on hardware acceleration could be deemed a less suitable strategy to increase performance.

One limitation of the NeRF is that it requires fine quality photos to reconstruct the model. This restricts usability in sensor data picture reconstruction. For example, if there is only sensor data that is from a wider spectrum than visible light, these NeRF methods introduced in this paper are not suitable for this kind of data representation.

There is promising ongoing research to construct NeRF models for various kinds of data that was represented in this paper. Wysocki et. al. [7] has been researching NeRF for constructing images from ultrasound. This proves that the NeRF is versatile technique to produce images and videos from different data than visible light in RGB format.

Finally, NeRF is a way to bring scenes to a virtual environment. This enables a more interactive way to represent real life objects and scenes.

NeRF can be utilized in many fields from architecture to video games. It enables interactive ways to display and document historical sites.

9 Conclusions

This paper introduced the beginning of NeRF research [1]. and presented two NeRF based techniques: Instant NeRF [2] and BakedSDF [6]. One of the key techniques to get meaningful results from NeRF is parametric encoding. Parametric encoding maps low dimensional data to higher dimensions and using Fourier features in this step gives higher quality output from NeRF. Training time can be decreased by implementing hash tables for the feature vectors and rendering time can be decreased by baking the model to triangle mesh.

References

- [1] B. Mildenhall, P. P. Srinivasan, M. Tancik, J. T. Barron, R. Ramamoorthi, and R. Ng, "Nerf: Representing scenes as neural radiance fields for view synthesis," in *ECCV*, 2020.
- [2] T. Müller, A. Evans, C. Schied, and A. Keller, "Instant neural graphics primitives with a multiresolution hash encoding," *ACM Trans. Graph.*, vol. 41, pp. 102:1–102:15, July 2022.
- [3] J. Naskath, G. Sivakamasundari, and A. A. S. Begum, "A study on different deep learning algorithms used in deep neural nets: Mlp som and dbn," *Wireless Personal Communications*, vol. 128, no. 4, pp. 2913–2936, 2023.
- [4] M. Tancik, P. P. Srinivasan, B. Mildenhall, S. Fridovich-Keil, N. Raghavan, U. Singhal, R. Ramamoorthi, J. T. Barron, and R. Ng, "Fourier features let networks learn high frequency functions in low dimensional domains," 2020.
- [5] Z. Chen, T. Funkhouser, P. Hedman, and A. Tagliasacchi, "Mobilenerf: Exploiting the polygon rasterization pipeline for efficient neural field rendering on mobile architectures," 2023.
- [6] L. Yariv, P. Hedman, C. Reiser, D. Verbin, P. P. Srinivasan, R. Szeliski, J. T. Barron, and B. Mildenhall, "Baked sdf: Meshing neural sdfs for real-time view synthesis," *arXiv*, 2023.
- [7] M. Wysocki, M. F. Azampour, C. Eilers, B. Busam, M. Salehi, and N. Navab, "Ultra-nerf: Neural radiance fields for ultrasound imaging," 2023.

Password usability and security

Nuutti Henriksson

nuutti.henriksson@aalto.fi

Tutor: Sanna Suoranta

Abstract

As password-based authentication has solidified its status as the most commonly used authentication method, the number of attacks against passwords has increased. Service providers often require passwords to follow predefined entropy criteria to create more attack-resistant passwords. Adhering to these criteria can complicate password utilization and thus negatively affect their usability. This paper is a literature review on password usability and security, discussing both password memorization and password entry, two common issues in modern-day password usage. The results showed that users could adopt techniques, such as mnemonic memorization and password managers, to increase password usability, while service providers can implement additional password verification during password creation to increase password memorability and thus usability. Additionally, further research into mobile password managers and the usability and security of other platforms, such as smart televisions, were identified as important subjects for future research.

KEYWORDS: *password, usability, security, password manager, mnemonic memorization*

1 Introduction

Since the introduction of computers, users have used passwords to authenticate themselves. A user had between 2 and 200 password-protected accounts already in 2010 [1]. Since then, password-based authentication has solidified its status as the most commonly used authentication method, meaning that the number of accounts per user has also likely risen. As password-based authentication has increased in popularity and computers have become more powerful, the number of attacks against this form of authentication has also increased. One of the most common and widely used methods to protect against these attacks is to create passwords with high unpredictability, or entropy. This makes it harder to perform common attacks, such as brute-force and shoulder surfing [2].

Currently, when creating a new password for an account, it has to often fulfill predefined entropy criterion. This criterion can include password length and different character classes. Additionally, some sites might impose wordlist-based and pattern-based checks, prohibiting the use of easy-to-remember patterns or words. It is also well known that passwords should not be re-used on multiple accounts, as this can result in multiple accounts becoming compromised if a password is leaked.

While adhering to the password entropy criterion creates stronger passwords, it can complicate their utilization. Shay et al. [3] showed that additional requirements can make it difficult for users to remember their passwords. Additionally, special character requirements make it more complicated to enter passwords on devices other than computers [4].

This paper summarizes the main difficulties with modern-day password usage and reviews currently available solutions and techniques for utilizing them in a user-friendly and secure way.

Section 2 discusses password use and usability issues, while Section 3 presents both potential and already existing solutions to solve password usability problems. Section 4 discusses the advantages and disadvantages of different approaches, while Section 5 concludes the paper.

2 Issues with modern day passwords

Password-based authentication is currently the most commonly used authentication method. As mentioned above, one of the most common approaches to protect against attacks on password-based authentication is

to increase password entropy and use multiple different passwords. While these are effective methods to protect against attacks, password usability can become a problem [3].

Section 2.1 highlights problems with password creation and recollection, while Section 2.2 discusses password entry issues when using different devices.

2.1 Creating and memorizing passwords

While it is essential to ensure that passwords can withstand attacks, it is also important that even strong passwords remain usable. Shay et al. [3] studied the effects of requirements on password strength and usability. They found that only increasing the length resulted in users predictably filling the required characters. Although it is a poor and insecure habit, users create predictable passwords to help reduce the cognitive task of having to remember multiple passwords over a long period [5]. Imposing additional pattern and blacklist checks result in stronger passwords, but further compromise the usability as users are more likely to forget their password [3].

Although a user might be able to retain one or a few strong passwords in their memory, it has been shown that attempting to remember multiple passwords causes users to require more guesses to recall the correct one [6]. This can be an issue since many sites allow users to try their passwords only a few times before locking the account or issuing a cooldown. While this does protect accounts against brute-force attacks, it also negatively affects real users who use strong passwords and require a few attempts to recall them.

2.2 Password entry

Complex passwords can also introduce new usability issues on mobile platforms. Greene et al. [4] studied the effects special characters and length have on the usability of passwords on mobile devices. The research showed that both aspects negatively affected the usability, although it was much more apparent with special characters. This was linked to mobile devices not allowing users to enter special characters from the first keyboard screen. Instead, the keyboard view has to be changed once or twice, depending on the character used. Melicher et al. [7] showed that mobile users used fewer special characters, capitalized letters and numbers when

creating passwords. The study also found that passwords mainly intended for mobile platform use often had special characters bundled together, reducing the amount of times a user would have to switch between different keyboard views. While this does make the password more usable, it also introduces a pattern that can be easier to guess. Furthermore, passwords created on mobile platforms were 32% easier to guess when an attacker had more than 10^{13} tries [7]. This could be a security concern if attackers are able to execute an offline brute-force attack, which is likely to happen when password hashes leak.

In addition to password entry being harder on mobile devices, there seems to be differences between different mobile platforms. Schaub et al. [2] studied password entry times on mobile platforms. They found that typing on an iOS virtual keyboard was slightly faster than on Android keyboards, while the more complex Symbian keyboard was considerably slower and more error prone. Although the study was done 10 years ago, the keyboards have remained similar. Virtual keyboards could possibly be additionally improved, particularly on devices that utilize harder to use variants.

3 Overcoming password usability issues

This section highlights strategies for improving password usability while also analyzing the possible impact on password security. As one of the main issues was recollection, increasing password memorability is an appropriate approach to improve usability. In addition, there exists different software based solutions, such as password managers, which can be used to keep track of multiple accounts.

Section 3.1 explores ways to increase password memorability and Section 3.2 discusses the use of different software-based solutions, mainly password managers.

3.1 Remembering passwords

Linguistic and phonological difficulty, or pronounceability, have been suggested for creating more memorable and thus more usable passwords [8]. A common strategy to approach pronounceable passwords is the *Modify Until Not Guessed Easily* (munge) method, which replaces alphabetical characters with special characters [9]. While munge does create easy-to-

remember passwords, it does not necessarily offer any additional protection against attacks, as character conversions are well-known and can be incorporated into attacks [9]. Gasser [10] introduced another well-known method for creating pronounceable passwords in 1975. The method uses 34 different phonemes to create randomized strings. A phoneme is a small, one or two-character, element that produces a sound [10]. Combining these phonemes creates a string, such as "qeshquo", that is easy to pronounce, but means nothing. While this might make passwords more memorable and resistant against dictionary attacks, it also limits the character set and the combination of characters that can be utilized. Thus, the length would play a critical role in creating a pronounceable password that also has high entropy.

Another proposed method to help with password memorability is mnemonic memorization. Mnemonic memorization utilizes a piece of information, such as phrases [11] or images [1], to compose passwords.

Phrase-based mnemonic passwords are commonly constructed by selecting a memorable sentence and substituting each word for a letter, symbol or number. The idea is that the sentence can then be used to recall the password. Yan et al. [12] found that phrase-based mnemonic memorization made passwords both more memorable and stronger. The study compared mnemonic passwords to both randomly generated passwords and user chosen passwords. The researchers found that the mnemonic passwords were as secure as randomly chosen passwords, while being nearly as memorable as normal user created passwords.

While mnemonic passwords are effective against normal dictionary and user information attacks [12], some specialized attacks can be made to target mnemonic passwords. Kuo et al. [11] found that many users use common phrases, that can be found online, when choosing their mnemonic phrase. The researchers compiled a dictionary using song lyrics, slogans and quotations, which was then used to try and crack mnemonic-based passwords. The study found that 11% of normal passwords could be cracked with a regular dictionary attack, while 4% of mnemonic passwords were cracked using the dictionary they compiled. Although this seems promising at first, the researchers found out that 65% of the studied mnemonic passwords were based on phrases that could be found online. They note that their dictionary was relatively small in size and a highly effective dictionary could be built for phrases found online. Instructing users to only use personalized sentences that are not widely

known could prove to be an effective strategy to construct unique sentences and thus stronger passwords.

Image-based mnemonic passwords are created with the help of information gathered from an image. Nelson and Vu [1] compared the memorability of phrase-based, image-based and regularly created passwords. The methods all used the same password criterion. The image-based technique associated a phrase with an image and created a password based on it, similar to phrase-based mnemonics. The main differences between the three approaches were the number of attempts needed, the number of passwords forgotten and password complexity. The number of attempts required to recall the password was significantly higher for the phrase-based method compared to the two other methods. The phrase-based group also had more forgotten passwords than the image-based group. However, password complexity was significantly higher with both mnemonic groups compared to the control group. In addition, the researchers note that the control group created passwords in highly predictable ways. These results correspond to the ones by Yan et al. [12], indicating that utilizing mnemonic techniques helps to create strong and memorable passwords.

In addition to user actions, service providers can also help users with password usability. Woods and Siponen [13] studied the effects of password verification during password creation. They found that requiring verification multiple times during password creation increased both first-time recollection accuracy and long-term recollection accuracy. The study showed that requiring verification twice, as opposed to just once, increased password recollection by 17% while requiring verification three times increased recollection by 28%. The researchers also conducted a user convenience survey, which showed that users found verifying the password twice to be more convenient than once or three times. Thus, it seems that adding an extra verification into password creation prompts would increase password memorability significantly, while not lowering the user experience.

3.2 Software-based solutions

Password managers are a popular software-based solution to overcome password usability problems. As some users can have more than 200 accounts [1], remembering and creating a unique and strong password for each account becomes impossible. Password managers try to overcome

this issue by securely storing all the passwords with the help of one strong master password. A study by Ion et al. [14] showed that most security experts would recommend using a password manager. However, the same study also noted that only 24% of non-expert users use password managers, while 73% of experts use them. This raises the question: Is there something wrong with password managers?

Ion et al. [15] focused on determining how usable password managers are on computers. The researchers conducted a study where participants used LastPass, a popular password manager, for a short period and then answered questions regarding its usability. They found that a majority of the participants thought the manager to have high usability. However, many participants found the password manager untrustworthy as they were not familiar with how the passwords were stored. Additionally, the researchers found that even many password manager users avoided inputting all their passwords into the manager. These observations suggest that the poor adaptation of password managers has to do with not understanding the working principles of a password manager and a lack of transparency from the providers.

In addition to the already mentioned benefits, password managers are able to help users with password entry issues on mobile platforms. As previously discussed, users use fewer character groups when creating and using passwords on mobile devices because the keyboards require at least one view change before allowing number and special-character entry [7]. Auto-filling passwords could prove as a solution to this issue. Seiler-Hwang et al. [16] studied the usability of four popular mobile password managers in 2019. The researchers used a standardized 10-question test to grade each password manager on the System usability scale [17]. The questionnaire results determined that only two of the managers could be considered as having adequate usability, while the two others were ranked average or below average. The researchers highlighted application integration and user instructions among the most crucial factors in the future development of mobile password managers.

Little to no recent research has been done on mobile password managers, making it difficult to estimate how much updates have focused on these areas. However, one recent study by Oesch et al. [18] reported the usability experiences of password manager users. The study found that many avoid using mobile password managers as they work inconsistently and have issues with even some of the most important features, such as

auto-fill. It is still safe to assume that at least some advances have been made in mobile password manager usability, shifting the experience towards good, very good and excellent.

4 Discussion

As there are many approaches to increasing password usability, choices have to be made between the solutions. While the highlighted solutions work on their own, a combination of different solutions could provide additional security and increase usability.

For example, combining an effective and secure memorization technique, verification and a password manager could promote stronger passwords and better usability. The memorization technique can be used to create a strong and memorable master password, while the password manager can be used for storing and creating strong passwords. Password manager providers could incorporate an additional verification for the master password during creation, which would further improve the memorability of the master password [13]. This approach would eliminate the need for additional password verification in other services because the password manager would store the passwords instead of the user. While the usability of mobile password managers is not excellent [16], password managers do also offer auto-filling, which can help users with password entry issues on mobile platforms.

While the above-mentioned combination of approaches solves the mentioned usability issues, a majority of users are currently not using password managers [14]. Another approach could be to only combine password verification with memorization techniques to help passwords to become more memorable. However, this approach fails to address password entry issues.

Researching the usability of modern password managers is important as they tackle many password usability issues present in modern-day usage. Requiring more transparency from password manager providers and educating the public on safe password storage could help to encourage more people to utilize password managers. While password managers can aid computer and mobile device users with password entry and security, other devices might still face these issues. Thus, it is important to research password usability and security on other increasingly popular platforms, such as smart televisions.

5 Conclusion

This paper summarized the main difficulties with modern-day password usage and reviewed solutions and techniques for utilizing passwords in a user-friendly and secure way. The main summarized difficulties were password memorability and password entry on mobile platforms. Pronounceable passwords and mnemonic memorization were identified as potential techniques for increasing password memorability, while the more popular technique, munge, raised crucial security concerns. Password managers were found to be a convenient software-based alternative to the already mentioned solutions while also assisting mobile users with password entry. Password manager usability, especially on mobile devices, and usability and security on other popular platforms, such as smart televisions, were identified as important topics for future research.

References

- [1] D. Nelson and K.-P. L. Vu, “Effectiveness of image-based mnemonic techniques for enhancing the memorability and security of user-generated passwords,” *Computers in Human Behavior*, vol. 26, no. 4, pp. 705–715, 2010. doi: 10.1016/j.chb.2010.01.007.
- [2] F. Schaub, R. Deyhle, and M. Weber, “Password entry usability and shoulder surfing susceptibility on different smartphone platforms,” in *Proceedings of the 11th International Conference on Mobile and Ubiquitous Multimedia, MUM '12*, (New York, NY, USA), Association for Computing Machinery, 2012. doi: 10.1145/2406367.2406384.
- [3] R. Shay, S. Komanduri, A. L. Durity, P. S. Huh, M. L. Mazurek, S. M. Segreti, B. Ur, L. Bauer, N. Christin, and L. F. Cranor, “Designing password policies for strength and usability,” *ACM Trans. Inf. Syst. Secur.*, vol. 18, may 2016. doi: 10.1145/2891411.
- [4] K. K. Greene, M. A. Gallagher, B. C. Stanton, and P. Y. Lee, “I can’t type that! p@\$\$w0rd entry on mobile devices,” in *Human Aspects of Information Security, Privacy, and Trust* (T. Tryfonas and I. Askoxylakis, eds.), (Cham), pp. 160–171, Springer International Publishing, 2014.
- [5] A. Hanamsagar, S. S. Woo, C. Kanich, and J. Mirkovic, “Leveraging semantic transformation to investigate password habits and their causes,” in *Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems, CHI '18*, (New York, NY, USA), p. 1–12, Association for Computing Machinery, 2018. doi: 10.1145/3173574.3174144.
- [6] K.-P. L. Vu, R. W. Proctor, A. Bhargav-Spantzel, B.-L. B. Tai, J. Cook, and E. Eugene Schultz, “Improving password security and memorability to protect personal and organizational information,” *International Jour-*

nal of Human-Computer Studies, vol. 65, no. 8, pp. 744–757, 2007. doi: 10.1016/j.ijhcs.2007.03.007.

- [7] W. Melicher, D. Kurilova, S. M. Segreti, P. Kalvani, R. Shay, B. Ur, L. Bauer, N. Christin, L. F. Cranor, and M. L. Mazurek, “Usability and security of text passwords on mobile devices,” in *Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems*, CHI ’16, (New York, NY, USA), p. 527–539, Association for Computing Machinery, 2016. doi: 10.1145/2858036.2858384.
- [8] J. R. Bergstrom, S. A. Frisch, D. C. Hawkins, J. Hackenbracht, K. K. Greene, M. F. Theofanos, and B. Griepentrog, “Development of a scale to assess the linguistic and phonological difficulty of passwords,” in *Cross-Cultural Design* (P. L. P. Rau, ed.), (Cham), pp. 131–139, Springer International Publishing, 2014. doi: 10.1007/978-3-319-07308-8_13.
- [9] K. S. Walia, S. Shenoy, and Y. Cheng, “An empirical analysis on the usability and security of passwords,” in *2020 IEEE 21st International Conference on Information Reuse and Integration for Data Science (IRI)*, pp. 1–8, 2020. doi: 10.1109/IRI49571.2020.00009.
- [10] M. Gasser, “A random word generator for pronounceable passwords, mtr-3006,” *ESDYTRY75Y97, ADYA017676*. Bedford, Mass: MITRE Corp, 1975.
- [11] C. Kuo, S. Romanosky, and L. F. Cranor, “Human selection of mnemonic phrase-based passwords,” in *Proceedings of the Second Symposium on Usable Privacy and Security*, SOUPS ’06, (New York, NY, USA), p. 67–78, Association for Computing Machinery, 2006. doi: 10.1145/1143120.1143129.
- [12] J. Yan, A. Blackwell, R. Anderson, and A. Grant, “Password memorability and security: empirical results,” *IEEE Security & Privacy*, vol. 2, no. 5, pp. 25–31, 2004. doi: 10.1109/MSP.2004.81.
- [13] N. Woods and M. Siponen, “Improving password memorability, while not inconveniencing the user,” *International Journal of Human-Computer Studies*, vol. 128, pp. 61–71, 2019. doi: 10.1016/j.ijhcs.2019.02.003.
- [14] I. Ion, R. Reeder, and S. Consolvo, ““...No one can hack my Mind”: Comparing expert and Non-Expert security practices,” in *Eleventh Symposium On Usable Privacy and Security (SOUPS 2015)*, (Ottawa), pp. 327–346, USENIX Association, jul 2015.
- [15] F. Alodhyani, G. Theodorakopoulos, and P. Reinecke, “Password managers—it’s all about trust and transparency,” *Future Internet*, vol. 12, p. 189, Oct 2020. doi: 10.3390/fi12110189.
- [16] S. Seiler-Hwang, P. Arias-Cabarcos, A. Marín, F. Almenares, D. Díaz-Sánchez, and C. Becker, ““i don’t see why i would ever want to use it”: Analyzing the usability of popular smartphone password managers,” in *Proceedings of the 2019 ACM SIGSAC Conference on Computer and Communications Security*, CCS ’19, (New York, NY, USA), p. 1937–1953, Association for Computing Machinery, 2019. doi: 10.1145/3319535.3354192.
- [17] P. W. Jordan, B. Thomas, I. L. McClelland, and B. Weerdmeester, *Usability evaluation in industry*. CRC Press, 1996.

- [18] S. Oesch, S. Ruoti, J. Simmons, and A. Gautam, ““it basically started using me:” an observational study of password manager usage,” in *Proceedings of the 2022 CHI Conference on Human Factors in Computing Systems*, CHI '22, (New York, NY, USA), Association for Computing Machinery, 2022. doi: 10.1145/3491102.3517534.

Neural Radiance Fields And Its Extensions And Applications In Computer Graphics

Otso Laasonen

otso.laasonen@aalto.fi

Tutor: Matti Siekkinen

Abstract

Neural radiance fields is a state of the art view synthesis method that utilizes neural networks. This seminar paper reviews latest articles and literature on the subject and explores the extensions and applications that have been created using this technology. For example, the technology has been extended to feature variable lighting conditions, and it has been used to create 3D-textures. It is a heavily studied subject and new articles are published constantly. Some of the latest applications using the technology are limited to offline rendering due to the high cost of the neural network querying and many computations.

KEYWORDS: View synthesis, 3D reconstruction, Image-based modeling and rendering, Volumetric rendering, 3D deep learning

1 Introduction

In computer science, view synthesis aims to create new views of a scene when supplied with a set of images and their corresponding camera poses. Neural radiance fields (NeRF) is a state-of-the-art method for achieving view synthesis using multilayer perceptrons[1]. As NeRF is one of the most researched methods currently in view synthesis, finding comprehen-

sive information about the topic can be difficult.

This paper is a literature review that compiles some of the latest literature and research on the subject with a focus on the derivatives of NeRF and its applications in computer graphics. This means explaining what NeRF is, along with the research built on top of it. Technical details of the introduced techniques are also discussed briefly.

The structure of this document is as follows. In chapter two, previous research on view synthesis is discussed briefly. Chapter three describes the basic operation of Nerf. Chapter four elaborates on some of the extensions built on top of NeRF and chapter five expands on the other computer graphics applications that NeRF can provide. Lastly, chapter six ends the paper by providing conclusive statements and discussing the future of the technology.

2 View synthesis

As mentioned in the introduction, view synthesis, or novel view synthesis, is the act of creating new views from a scene based on existing ones. Numerous techniques have been used for view synthesis in the past. Most of the original work involved establishing a correspondence of shapes and points between two images. This was often done by a human and not by computational methods. Then, an algorithm is deployed to find a mapping for the rest of the images[2]. Optical flow information is, however, difficult to obtain and relies on certain conventions, such as diffuse reflectance in order to establish the mapping. A 1996 paper, the *Lumigraph*[3], suggested sampling the full plenoptic function. This technique was not restricted by the reflectance of the properties of the scene. In addition, the 5D plenoptic function could be presented as a 4D lightmap of the radiance leaving the scene, due to radiance being constant along a ray. [3] However, both of these techniques require dense sampling resolution in order to be effective.

Mesh-based representations are also a popular class of techniques for scene representation. They are optimizable using gradient descent either by differentiable rasterizers or path tracers. Difficulties arise when optimizing the mesh: Local minima can easily trap the optimizer, and there is not a simple and efficient loss function available[4].

Another class of techniques for creating views are volumetric methods. Early approaches represented the scene as a voxel grid or a point cloud [1].

These representations have also recently been paired with convolutional neural networks that try to predict the volumetric representation of the scene[5]. However, sampled representations are fundamentally limited in their ability to scale to a higher resolution, due to the discrete sampling of the scene space. NeRF avoids this problem by representing the scene as a continuous volumetric representation that is encoded as parameters of a neural network[6, 1].

3 NeRF

Neural radiance fields represent the scene as a continuous function in the form of a multilayer perceptron (MLP). This MLP takes in a 3D position, and a viewing direction for a combined 5D vector, similarly to a plenoptic function. The vectors represent coordinates along camera rays tracing the scene. These vectors are then fed as an input to the network, which outputs a color and a volume density. Figure below provides a visual representation of the technique (figure 1).

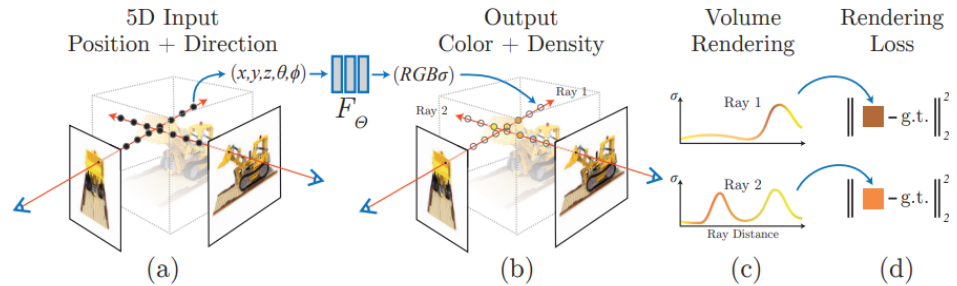


Figure 1. Overview of NeRF[1].

These are compiled into an image by numerically approximating multiple scattering of light in a medium[7]. The expected color is calculated using the following formula:

$$C(\mathbf{r}) = \int_{t_n}^{t_f} T(t)\sigma(\mathbf{r}(t))\mathbf{c}(\mathbf{r}(t), \mathbf{d})dt \quad (1)$$

Here the parametric ray $\mathbf{r}(t) = \mathbf{o} + t\mathbf{d}$ has its origin at \mathbf{o} and direction points to \mathbf{d} . Volume density is $\sigma(\mathbf{x})$ at point \mathbf{x} . Function $T(t)$ is the accumulated transmittance acquired among a ray from t_n to t [7].

The integral above needs to be numerically estimated using a quadrature. Stratified sampling is used to uniformly sample from evenly spaced bins along t [1].

$$t_i u \left[t_n + \frac{i-1}{N}(t_f - t_n), t_n + \frac{i}{N}(t_f - t_n) \right] \quad (2)$$

Just feeding the parameters to an MLP results in poor performance in areas where there are large deviations in color and geometry. This can be alleviated by mapping the inputs to a higher dimensional space[1].

The dataset for optimizing the network consists of captured images of the scene. At each iteration of the optimization algorithm, a random mini-batch of camera rays from the set of all pixels is used, and for each ray, a random amount of points along the ray is queried. The loss function becomes the squared error between the actual pixel color, and the estimated pixel color[1].

4 Extensions of NeRF

While standard NeRF improved upon the previous view synthesis work, it struggles when the scene is no longer static, or it features refractive mediums. After its release, NeRF has been a subject to numerous studies that aim to improve upon the original paper[8, 9].

4.1 NeRF and refractive medium

As mentioned previously, NeRF uses volume rendering techniques involving ray casting to output the density and radiance along a ray. However it is not possible to use viewing direction and parameterization t to uniformly sample the bins when the ray path curves inside the medium due to refraction. Therefore the sampling technique used needs to be extended to take this into account. One such extension is introduced by Pan et. al [8]. In this paper, the refraction paths are constructed using the following derivation of the Eikonal equation:

$$\mathbf{x}_{i+1} = \mathbf{x}_i + \frac{\Delta s}{n} v_i, \quad v_{i+1} = v_i + \Delta s \nabla n \quad (3)$$

Here, Δs is the step size, n is the refractive index and ∇n is the gradient index. Some preprocessing is required before a path can be traced using the piecewise linear approximation above 3. A proxy geometry of the refractive object is constructed, and noisy components are removed manually. Aforementioned equation is used to bend the path at each step when collecting the samples. A subset of these samples are fed to the coarse network for processing. The loss function consists of three components,

weighted by their corresponding hyperparameters. The first component is the squared error between the ground truth pixel values and the fine and the coarse pixel values. It is called the re-rendering error. The second part is a boundary regularizer that is a masked re-rendering error. The third is an L2 gradient penalty to boundary[8].

This extension improves on the visual quality of NeRF when faced with refractive geometry. However even with this extension, foggy artifacts may appear on refractive surfaces, and using the extension requires a manual step of removing noisy components[8].

4.2 NeRF-W

NeRF in the wild (NeRF-W) is a collection of extensions built on top of NeRF to enable the handling of real-world uncontrolled images[10]. One of the most central limitations of regular NeRF is its poor performance when presented with moving objects, or with variable illumination. This is due to the fact that it assumes that the given images are from a static scene, and therefore the density and the radiance are constant for a given view direction. For example, in outdoor photography the time of day directly affects the illumination of the scene, violating NeRF's requirement for constant radiance along a set ray[10].

Adapting NeRF to account for a variable like this involves making a latent embedding out of the variance in the emitted radiance. Now the latent embedding are optimized and the model uses these embeddings to guess the image-dependent radiance[9, 10].

Transient objects are dealt with introducing a separate estimator for transient color and density. In addition, this estimator also outputs a field of uncertainty, which can be used to mask off pixels when calculating reconstruction loss. Each pixel color is modeled as a normal distributed variable, and the variance of that distribution is rendered using the same methods as NeRF[9].

The model is optimized similarly to NeRF, with two volumetric density representations: a coarse and a fine one. The coarse model uses only the latent appearance model, excluding the transient object model. Alongside the models, both the transient and the appearance embeddings are optimized. One advantage of modeling and optimizing the appearance separately is the fact that the lighting can be modified without altering the base geometry[9].

NeRF-W is able to address many of regular NeRF's weaknesses. How-

ever it remains susceptible to poor camera calibration and it suffers quality degradation when details appear sparsely in a training set or when they are viewed from heavily slanted angles[9].

5 Applications of NeRF in computer graphics

Even though NeRF is a new method, it has already been used in some applications with great success. In this section three such applications are reviewed.

5.1 NeRF relighting

Neural Reflectance and Visibility Fields for Relighting and View Synthesis, also known as NeRV, is both an application and an extension. As mentioned in the previous chapter, NeRF assumes constant radiance along a ray, meaning that it does not separate the scene lighting condition from the scene itself. NeRF relighting changes the network querying logic: instead of querying the network for color and volume density, the network is queried for bidirectional reflectance distribution function (BRDF) parameters[11]. This means that instead of assuming the scene as a field of particles absorbing and emitting light, the field absorbs and reflects light that is emitted by an external light source. This changes the volume rendering integral in the following way:

$$L(\mathbf{c}, \boldsymbol{\omega}_o) = \int_0^\infty V(\mathbf{x}(t), \mathbf{c}) \sigma(\mathbf{x}(t)) L_r(\mathbf{x}(t), \boldsymbol{\omega}_o) dt \quad (4)$$

$$L_r(\mathbf{c}, \boldsymbol{\omega}_o) = \int_S L_i(\mathbf{x}, \boldsymbol{\omega}_i) R(\mathbf{x}, \boldsymbol{\omega}_i, \boldsymbol{\omega}_o) d\boldsymbol{\omega}_i \quad (5)$$

However, this change poses a challenge: since the lighting is not assumed to be a part of the scene anymore, each quadrature sample of the integral also needs samples from each light source. This renders brute force solutions unusable, due to the sheer amount of computations involved, as computations of direct lighting and indirect lighting scale quadratically and cubically per ray, respectively. This is illustrated in (Figure 2)

For this reason, another MLP is introduced that produces an approximation of the lighting visibility for any input direction at a given point, along with the expected ray termination depth[11]. Using the visibility queried from the MLP reduces the complexity of computing direct lighting from quadratic to linear per ray. Two approximations are required

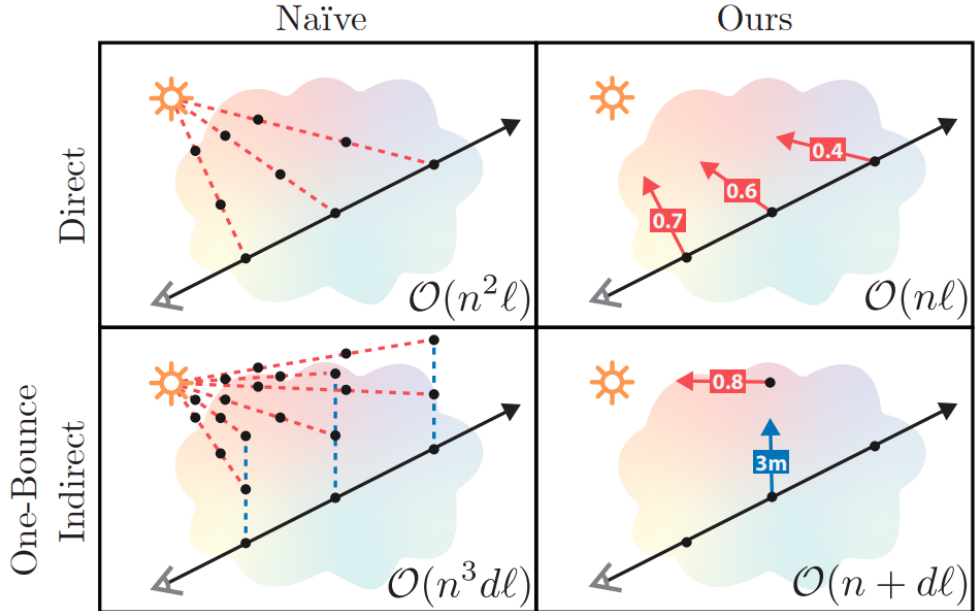


Figure 2. Computational cost of querying external light sources scales poorly if not approximated[11].

to accelerate indirect lighting calculations. Firstly, accumulated radiance is reduced to a single point evaluation by treating the volume as a hard surface at the termination depth. This termination depth is not the one queried from the visibility MLP. Secondly, indirect recursion is limited to a single bounce with the same hard surface approximation. This single bounce is approximated with the depth received from the visibility MLP. This also reduces the complexity to linear[11].

Similarly to the regular NeRF networks are trained by tracing rays into the scene and querying the parameters from the network. These parameters are used to calculate the shading at each point. The loss is a sum of three mean squared losses. The first one consists of the calculated color compared to the ground truth image color, similarly to NeRF. The visibility MLP is optimized by comparing the visibility and depth predictions to estimates given by the actual shape MLP, so it is not compared to any ground truth value. This is to match both MLP’s estimates of the scene geometry[11].

NeRV is able to create very convincing scenes that can be relit with different lighting configurations[11]. This technology could be used to solve global illumination problems in video games. However there is a noticeable error when compared to the ground truth scene renderings.

5.2 Virtual reality and NeRF

While NeRF is able to produce high quality views of the scene, querying the network for high resolution stereoscopic images features a substantial rendering cost. This makes real time rendering using NeRF unfeasible[12].

However, a recent paper by Rolff et al.[12], suggests grouping pixels in 16 by 16 pixel groups to be queried once and sharing this value between the pixels. The technique is called variable rate shading NeRF (VRS-NeRF)[12]. Rolff et al. [12] improved upon the original VRS-NeRF by introducing further optimizations, such as choosing the shading rate based on different heuristics. Three different methods are used to determine this shading rate. The first is to render in high resolution at the center of the view and in lower resolution when image area is in peripheral vision. The second is to render high edge and color frequency areas in higher resolution. Lastly, a mapping can be created from a pre-rendered image to determine prominent areas of the actual image[12]. An additional optimization method is to increase the step size depending on the shading rate.

These optimizations allow for a 20-30 frames per second real time virtual reality experience. This was achieved on a high end mobile graphics card.

5.3 NeRF texture synthesis

NeRF is not limited to synthesizing image data. Traditionally, methods for synthesizing textures were limited to two dimensions. Multitude types of different geometry require more structure than what is provided by a regular 2D texture. For example, grass, fabric or leaves consist of fine detail in three dimensions, and are therefore poorly represented by a two-dimensional image.

Huang et al. [13] synthesized meso-structure textures from real-world images. Additionally, the model learns the base shape of the object in question. The first step of this technique is to extract the base shape of the object. This step is done in 3 separate actions. Base shape extraction uses a result of Müller et al. [14] to obtain the coarse mesh of the object. Then the mesh is transformed into union of convex hulls containing the mesh, and finally the mesh is subdivided and re-meshed to obtain the base shape.

After base shape determination, query points are projected onto the surface of the coarse mesh using ray casting along the negative normal. This allows for getting the signed distances from the points. Unfortunately, this ray cast makes the projected coordinates non-differentiable at the input point. This requires constructing a differentiable projection layer[13].

The latent features of the texture presentation is stored into a hashmap for constant access complexity. This helps with overcoming the overhead from querying the latent codes. The texture is applied in patches to the mesh using a patch matching algorithm[13].

Regular NeRF does not learn to disentangle shading properties like material and lighting from the base representation of the object. This texture synthesizing technique encodes illumination and materials into spherical basis functions. This encoding into basic function is known as spherical harmonics[13]. The resulting synthesized textures look convincing and they qualitatively outperform 2D textures.

6 Discussion

This paper compiled results from the latest research in NeRF. It is clear that neural radiance fields are the focus area in view synthesis with numerous articles written in recent years. As explored in chapter five, the trained networks can also be useful outside pure view synthesis for applications such as texture synthesis, or even in real time rendering. The amount of research done on the topic is large, with new papers being constantly published. This makes it difficult to predict what can and will be done with the technology in the future. However, one of the key abilities of the deep neural networks are their adaptability for different functions. Perhaps NeRF could be leveraged to solve traditionally hard problems in computer graphics space, such as global illumination. NeRV was already able to create convincing scenes with variable lighting conditions. This could be used, for example, in video games for more realistic lighting when complex geometry is present in the scene. This however requires faster querying for parameters from the network than what can currently be done. NeRF can also create realistic textures, as seen in chapter five. This could also be used to create variable quality 3D-textures based on users hardware capabilities for more realistic surface presentation in games.

References

- [1] B. Mildenhall, P. P. Srinivasan, M. Tancik, J. T. Barron, R. Ramamoorthi, and R. Ng, “Nerf: Representing scenes as neural radiance fields for view synthesis,” 2020.
- [2] S. E. Chen and L. Williams, *View Interpolation for Image Synthesis*, p. 381–390. New York, NY, USA: Association for Computing Machinery, 1998.
- [3] S. J. Gortler, R. Grzeszczuk, R. Szeliski, and M. F. Cohen, “The lumigraph,” in *Proceedings of the 23rd Annual Conference on Computer Graphics and Interactive Techniques*, SIGGRAPH ’96, (New York, NY, USA), p. 43–54, Association for Computing Machinery, 1996.
- [4] T.-M. Li, M. Aittala, F. Durand, and J. Lehtinen, “Differentiable monte carlo ray tracing through edge sampling,” *ACM Trans. Graph.*, vol. 37, dec 2018.
- [5] T. Zhou, R. Tucker, J. Flynn, G. Fyffe, and N. Snavely, “Stereo magnification: Learning view synthesis using multiplane images,” *ACM Trans. Graph.*, vol. 37, jul 2018.
- [6] B. Mildenhall, P. P. Srinivasan, R. Ortiz-Cayon, N. K. Kalantari, R. Ramamoorthi, R. Ng, and A. Kar, “Local light field fusion: Practical view synthesis with prescriptive sampling guidelines,” *ACM Trans. Graph.*, vol. 38, jul 2019.
- [7] J. T. Kajiya and B. P. Von Herzen, “Ray tracing volume densities,” *SIGGRAPH Comput. Graph.*, vol. 18, p. 165–174, jan 1984.
- [8] J.-I. Pan, J.-W. Su, K.-W. Hsiao, T.-Y. Yen, and H.-K. Chu, “Sampling neural radiance fields for refractive objects,” in *SIGGRAPH Asia 2022 Technical Communications*, SA ’22, (New York, NY, USA), Association for Computing Machinery, 2022.
- [9] J. Sun, X. Chen, Q. Wang, Z. Li, H. Averbuch-Elor, X. Zhou, and N. Snavely, “Neural 3d reconstruction in the wild,” in *ACM SIGGRAPH 2022 Conference Proceedings*, SIGGRAPH ’22, (New York, NY, USA), Association for Computing Machinery, 2022.
- [10] R. Martin-Brualla, N. Radwan, M. S. M. Sajjadi, J. T. Barron, A. Dosovitskiy, and D. Duckworth, “Nerf in the wild: Neural radiance fields for unconstrained photo collections,” 2021.
- [11] P. P. Srinivasan, B. Deng, X. Zhang, M. Tancik, B. Mildenhall, and J. T. Barron, “Nerv: Neural reflectance and visibility fields for relighting and view synthesis,” 2020.
- [12] T. Rolf, K. Li, J. Hertel, S. Schmidt, S. Frintrop, and F. Steinicke, “Interactive vrs-nerf: Lightning fast neural radiance field rendering for virtual reality,” in *Proceedings of the 2023 ACM Symposium on Spatial User Interaction*, SUI ’23, (New York, NY, USA), Association for Computing Machinery, 2023.

- [13] Y.-H. Huang, Y.-P. Cao, Y.-K. Lai, Y. Shan, and L. Gao, “Nerf-texture: Texture synthesis with neural radiance fields,” in *ACM SIGGRAPH 2023 Conference Proceedings*, SIGGRAPH ’23, (New York, NY, USA), Association for Computing Machinery, 2023.
- [14] T. Müller, A. Evans, C. Schied, and A. Keller, “Instant neural graphics primitives with a multiresolution hash encoding,” *ACM Trans. Graph.*, vol. 41, jul 2022.

Privacy and Dark Patterns

Reda Bellafqih, Exchange Student, 101619963

reda.bellafqih@aalto.fi

Tutor: Suoranta Sanna

Abstract

In today's digital landscape, safeguarding user privacy is crucial, particularly in user interface design and under the General Data Protection Regulation (GDPR). This article delves into the interplay between GDPR and dark patterns - deceptive design tactics that compromise user privacy by manipulating behavior and decision-making. Through analyzing dark patterns in e-commerce, gaming, and UX design, the paper reveals their impact on privacy and exploitation of cognitive biases.

The research evaluates GDPR's effectiveness in countering these practices, noting the limitations of current regulations and stressing the need for a comprehensive approach. This includes developing protective tools, enhancing regulatory measures, offering economic incentives, and educating users about such practices.

In conclusion, the study highlights the need for evolving regulatory frameworks and stakeholder collaboration to ensure a digital future that values user privacy and autonomy. It advocates for a transparent, trust-based, and empowering digital environment, ensuring that technological advancements do not undermine user freedom and privacy.

1 Introduction

Data has become increasingly important in today's society. Protecting user privacy is now a priority, in the European Union. Over time, it has become clear that strict regulations are necessary to safeguard data. The General Data Protection Regulation (GDPR), which was introduced by the European Union in 2018, is a testament to these efforts. Since its implementation, the GDPR has given users control over their data, allowed them to set limits for those who handle their information. However, despite having the GDPR in place, there are still design tactics called dark patterns that persist. These tactics use psychology to obtain information. Such manipulative approaches can intrude on user privacy by influencing behavior and choices often resulting in consent and disclosure of information. This article explores the connection between GDPR and dark patterns, evaluates how effective current legal frameworks are at addressing these practices and protecting user privacy. Through this examination, we aim to shed light on the mechanics of patterns, identify the limitations of existing regulations and suggest ways to strengthen data protection measures.

2 Environment: Privacy Protection and User Interfaces

2.1 Studies on GDPR

Several researchers have dissected the GDPR from both theoretical and practical perspectives. Voigt and Von dem Bussche [1] provided an in-depth analysis of the GDPR, examining its foundational principles, scope, and its impact on both organizational and individual data processing dynamics. In contrast, Kuner [2] explored data flows in the context of data protection and privacy laws. This exploration is pivotal as it offers a deeper understanding of the influence of GDPR on global data exchange paradigms, emphasizing its implications beyond Europe. In 'Regulation of Transborder Data Flows under Data Protection and Privacy Law,' the authors call for a nuanced approach to data privacy regulation. They advocate for a balance between geographical and organizational factors, enhanced regulatory efficiency, and transparency. Additionally, they emphasize the need for continuous evaluation of underlying policies and their impacts, recognizing the complexities of transborder data flow in the mod-

ern digital landscape.

2.2 Exploration of Dark Patterns:

Dark patterns have been subjected to rigorous scrutiny. Brignull [3], who is credited with coining the term, highlighted numerous instances where design strategies were employed to mislead users, prompting them to act against their best interests. His groundbreaking work has paved the way for subsequent research into these deceptive design strategies and their implications on user autonomy and informed decision-making. Mathur et al. [4] further illuminated the topic by analyzing real-world data, revealing the prevalence of dark patterns across thousands of e-commerce platforms. Their findings underscore the pressing need for more stringent regulations to counteract these manipulative designs. Additionally, several studies have delved into the psychological underpinnings of dark patterns, illustrating how these tactics exploit inherent cognitive biases, sometimes leading to inadvertent data sharing or privacy breaches [5].

2.3 GDPR and Dark Patterns: Navigating the Interplay

It is essential to grasp how dark patterns can bypass the safeguards set by the GDPR. This article endeavors to fill the existing knowledge void by examining the interplay between these deceptive design tactics and data protection regulations. Through this exploration, the objective is to identify potential enhancements to legal frameworks that can effectively combat such misleading online strategies.

3 Examples of Dark Patterns

Dark patterns have become a concern, in the world particularly in relation to user experience and interface design. These deceptive design techniques are intentionally implemented to deceive users leading them to make decisions that may not be in their interest. In this section we will explore instances of patterns drawing from recent research and real life examples.

3.1 Deceptive User Experience (UX) Design

One common form of dark patterns is deceptive user experience (UX) design. These designs often manipulate users into making choices, such as subscribing to newsletters making purchases or sharing personal information [5]. An illustration of the prevalence of these patterns across various websites can be seen in Figure 1. This figure shows the distribution of dark patterns discovered over the Alexa rank of the websites, indicating that a significant percentage of popular shopping websites employ at least one dark pattern.

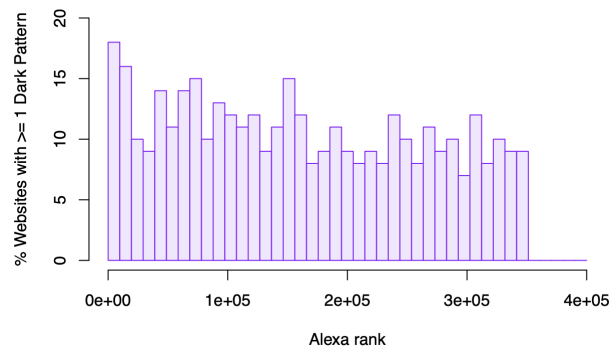


Figure 1. Distribution of the dark patterns discovered over the Alexa rank of the websites. Each bin indicates the percentage of shopping websites in that bin that contained at least one dark pattern [5].

3.2 E commerce Manipulation

E-commerce platforms are notorious for using dark patterns. A comprehensive study analyzing than 11,000 shopping websites revealed that many of these platforms employ deceptive tactics to influence user behavior including hidden charges misleading countdown timers and bait and switch strategies [4]. However, it is not just the E-commerce that employs these tactics. Another prevalent category of dark patterns is "Misdirection." This category encompasses design strategies that intentionally divert users' attention away from certain actions or information, leading them to make unintended choices. Figure 2 showcases four distinct types of the Misdirection category, it illustrates how these patterns can be subtly integrated into user interfaces to mislead or confuse users

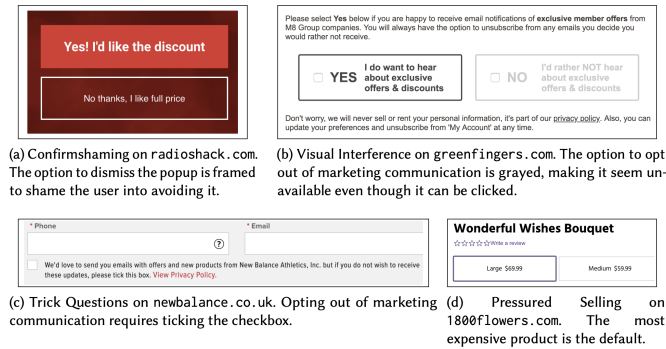


Figure 2. Four types of the Misdirection category of dark patterns [4].

The study 'Dark Patterns at Scale: Findings from a Crawl of 11K Shopping Websites' by Mathur et al. [4] sheds light on the extensive use of dark patterns in e-commerce platforms. The authors reveal that a significant percentage of shopping websites employ deceptive tactics such as hidden charges, misleading countdown timers, and bait-and-switch strategies. Their findings emphasize the pressing need for more stringent regulations to counter these manipulative designs and protect user privacy.

3.3 Dark Patterns in Gaming

Even the gaming industry does not shy away from employing dark patterns. Certain games utilize design strategies to encourage in game purchases or retain players, for durations—sometimes at the expense of the users well being. In 'Dark Patterns in the Design of Games' by Zagal et al. [6], the authors highlight the ethical concerns surrounding the use of dark patterns in the gaming industry. They emphasize the need for greater awareness and understanding of these patterns and their impact on the player experience. The study concludes with a call for a more conscientious approach to game design, focusing on ethical considerations and the well-being of players

3.4 Strategies to Protect Privacy

The issue of privacy patterns is quite alarming because they have an effect on user data and personal information. These patterns are designed to deceive users into sharing more information than they actually want to or make it difficult for them to comprehend the consequences of their choices when it comes to sharing their data [5].

3.5 Real-world Instances

Several real-world examples highlight the pervasive nature of dark patterns. These instances range from misleading subscription models to convoluted opt-out processes, all designed to exploit the user's cognitive biases and influence their decisions [7].

4 Impact on Privacy

The evolution of the online world has brought about benefits, such as improved connectivity and streamlined services. However, along with these advancements come challenges, specifically, when it comes to safeguarding user privacy. Dark patterns, which are design tactics, have emerged as a threat to maintaining privacy. Their strategic implementation in interfaces carries implications for data protection and individual rights.

4.1 Coercive Data Collection

Dark patterns often employ techniques to coerce users into sharing more personal information than they originally intended. These tactics take advantage of biases causing users to inadvertently disclose their data. Such practices not breach user trust, also expose them to potential misuse and breaches of their data [8].

4.2 Misleading Consent Mechanisms

Consent banners and pop ups are used tools to obtain user consent for data collection purposes. However, dark patterns can manipulate these mechanisms by making them confusing or misleading. Users may believe they are opting out of sharing their data when, in reality, they are granting permission unknowingly. This deceptive approach undermines the concept of informed consent, which is a cornerstone of privacy regulations [9].

4.3 Privacy Paradox

The privacy paradox refers to the discrepancy between users expressing concerns, about privacy and their actual online behaviors. Dark patterns worsen this paradox by taking advantage of the way our minds work causing users to make choices that go against their privacy preferences. This

situation highlights the importance of user friendly design approaches that prioritize user control and informed decision making [10].

5 Possible Solutions

With the increasing prevalence of patterns and their impact on user privacy, it is important to explore remedies to counteract these deceptive tactics. Dealing with the challenge of patterns requires an approach that combines technological advancements, regulatory measures and educational initiatives. In this section we will delve into solutions that could mitigate the effects of dark patterns on user privacy.

5.1 Tools

One way to address patterns is by integrating tools that promote consumer privacy as an initial defense. For example browser extensions or applications that alert users about patterns on websites can empower them to make informed decisions [4]. A practical example of such a tool is depicted in Figure 3. This mockup showcases a possible browser extension developed using researcher dataset. The extension is designed to flag instances of dark patterns with a distinct red warning icon. By simply hovering over the icon, users can gain insights into the specific pattern, further enhancing their online browsing experience and safeguarding their privacy.

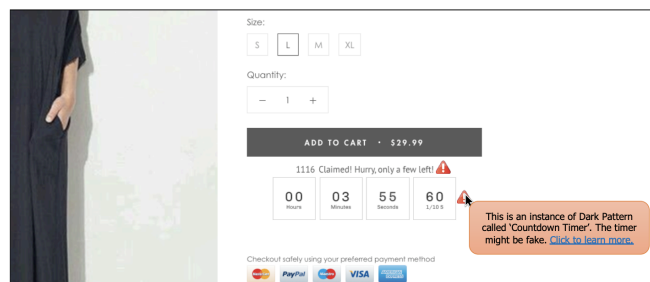


Figure 3. Mockup of a possible browser extension that can be developed using our data set. The extension flags instances of dark patterns with a red warning icon. By hovering over the icon, the user can learn more about the specific pattern [4].

5.2 Regulatory Measures

Regulatory solutions can play a key role in reducing the prevalence of dark patterns. Governments and regulatory bodies can establish guide-

lines and penalties for websites and platforms that employ design tactics. By this organizations will be encouraged to prioritize user design instead of manipulative strategies [9]. In the context of regulatory measures, the insights from 'Tales from the Dark Side: Privacy Dark Strategies and Privacy Dark Patterns' by Christoph et al. [9] are particularly relevant. The study delves into the manipulation of privacy through dark patterns, highlighting the need for regulatory frameworks that specifically address these privacy-invasive tactics. The authors advocate for guidelines and penalties for platforms that exploit user data and privacy, underlining the role of regulatory bodies in ensuring digital environments prioritize user autonomy.

5.3 Economic Incentives

Employing economic incentives can encourage organizations to adopt privacy friendly solutions. For instance providing tax breaks or other financial benefits to companies that prioritize user design may motivate them to avoid using dark patterns [11].

5.4 User Education and Awareness

It is crucial to educate users about the existence and implications of patterns in order to combat them effectively. Users can gain the knowledge to recognize and navigate patterns through awareness campaigns, workshops and online resources. This empowers them to retain control over their personal data. In 'What makes a dark pattern... dark? Design attributes, normative considerations, and measurement methods' by Arunesh et al. [12], the authors discuss the importance of user education in recognizing and navigating dark patterns. The study provides a framework for understanding the design attributes and normative considerations of dark patterns, offering insights that can empower users in their digital interactions. This emphasizes the critical role of user education and awareness in combating deceptive design strategies.

6 Discussion

Exploring the relationship between patterns and user privacy is crucial due to its complexity. As the digital world evolves, manipulative techniques used to influence users have become more sophisticated. In this

section we delve into the implications of dark patterns the difficulties in tackling them, and potential ways forward.

6.1 The Widespread Use of Dark Patterns

The extensive utilization of patterns across digital platforms emphasizes the urgency of addressing this matter. While these deceptive strategies may provide short term benefits for businesses, they undermine user trust. This can have lasting consequences on brand reputation and user loyalty.

6.2 The GDPR Paradox

The implementation of GDPR marked an advancement in protecting user privacy. However, the persistence of patterns after GDPRs enforcement reveals gaps in regulations and raises questions about their effectiveness in genuinely safeguarding users from deceitful online practices.

6.3 Balancing User Experience and Privacy

One challenge in combating patterns lies in finding a delicate equilibrium between creating captivating user experiences while ensuring adequate privacy protection. Although it is important for digital platforms to prioritize user design, it should not be achieved by compromising user autonomy and the ability to make decisions.

6.4 The Importance of Industry and Regulatory Involvement

Both the technology industry and regulatory bodies have responsibilities in minimizing the influence of dark design techniques. By working they can establish guidelines and best practices that focus on transparency and empowering users.

6.5 Future Research Directions

Further research, into the workings of patterns and their practical consequences is urgently needed. This research can provide insights to create solutions implement regulatory measures and educate users on how to combat these deceptive tactics. In summary it is important to understand that the issue of dark patterns extends beyond design or technology challenges; it encompasses wider societal concerns. To address this issue effectively, a comprehensive approach involving stakeholders from all

sectors is necessary. This approach ensures that our digital future is built upon principles of transparency, trust and empowering users.

7 Conclusion

In this age we have witnessed advancements and conveniences. However, along with these benefits come challenges that put user autonomy and privacy to the test. Dark patterns, which are design tactics that exemplify these challenges as they manipulate users and often lead them to make decisions that may not be in their best interest. Despite the implementation of regulations like the GDPR, these dark patterns persist, highlighting the complexity of the issue. Our exploration into dark patterns, their impact on privacy, and potential solutions has shown that while there are measures in place to counteract these tactics, there is still work to be done. Addressing this multifaceted problem requires an approach that combines innovation, strict regulations, educating users and fostering collaboration within industries. Moreover, the GDPR paradox emphasizes the necessity for evolution in frameworks. As digital strategies evolve over time, regulations must also adapt accordingly. It is not about creating laws; it is about ensuring their effectiveness in the changing digital landscape. The significance of user education cannot be emphasized enough. Empowering users with knowledge to identify and navigate around patterns is a defense against these deceitful tactics. However, it is important to note that the responsibility should not rest on the user. The technology industry, regulatory entities and other parties involved must collectively assume accountability, for establishing an ecosystem based on principles of openness, equity and consideration, for user freedom. In closing, while the challenges posed by dark patterns are significant, they are not insurmountable. With concerted efforts from all stakeholders, a digital future that prioritizes user privacy and autonomy is achievable. The journey towards this future requires continuous vigilance, collaboration, and a commitment to placing user interests at the forefront of digital innovation.

References

- [1] a. V. A. Voigt, Paul, *The EU General Data Protection Regulation (GDPR)*. The EU General Data Protection Regulation (GDPR): A Practical Guide, 2017.
- [2] C. Kuner, *Regulation of Transborder Data Flows under Data Protection and Privacy Law*. TILT Law Technology Working Paper, 2017. https://papers.ssrn.com/sol3/papers.cfm?abstract_id=1689483.
- [3] H. Brignull, *Dark Patterns: Deception vs. Honesty in UI Design*. Master thesis Faculty of Law. University of Oslo, 2011. <https://alistapart.com/article/dark-patterns-deception-vs-honesty-in-ui-design/>.
- [4] A. Mathur, A. Gunes, M. J. Friedman, E. Lucherini, J. Mayer, M. Chetty, and A. Narayanan, *Dark Patterns at Scale: Findings from a Crawl of 11K Shopping Websites*. Proceedings of the ACM on Human-Computer Interaction, 2019. <https://dl.acm.org/doi/abs/10.1145/3359183>.
- [5] C. M. Gray, Y. Kou, B. Battles, J. Hoggatt, and A. L. Toombs, *The Dark (Patterns) Side of UX Design*. Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems April 2018 Paper No.: 534 Pages 1–14, 2018. <https://dl.acm.org/doi/abs/10.1145/3173574.3174108>.
- [6] J. P. Zagal, S. Björk, and C. Lewis, *Dark patterns in the design of games*. Proceedings of the International Conference on Foundations of Digital Games, 2011. <https://www.diva-portal.org/smash/record.jsf?pid=diva2%3A1043332&dsid=-8399>.
- [7] L. Jamie and S. Lior Jacob, *Shining a light on dark patterns*. Journal of Legal Analysis, Volume 13, Issue 1, 2021, Pages 43–109,, 2021. <https://academic.oup.com/jla/article/13/1/43/6180579>.
- [8] C. M. Gray, C. Shruthi Sai, B.-B. Kerstin, a. J. T. G. Arunesh, Mathur, and S. Brennan, *Emerging Transdisciplinary Perspectives to Confront Dark Patterns*. Extended Abstracts of the 2023 CHI Conference on Human Factors in Computing Systems April 2023 Article No.: 522 Pages 1–4, 2023. <https://dl.acm.org/doi/abs/10.1145/3544549.3583745>.
- [9] B. Christoph, E. Benjamin, K. Frank, K. Henning, and S. Pfattheicher, *Tales from the dark side: privacy dark strategies and privacy dark patterns*. Proceedings on Privacy Enhancing Technologies 2016(4):237–254 DOI:10.1515/popets-2016-0038, 2016. https://www.researchgate.net/publication/303814886_Tales_from_the_Dark_Side_Privacy_Dark_Strategies_and_Privacy_Dark_Patterns.
- [10] A. E. Waldman, *Cognitive biases, dark patterns, and the 'privacy paradox'*. Current Opinion in Psychology Volume 31, February 2020, Pages 105-109, 2019. <https://www.sciencedirect.com/science/article/abs/pii/S2352250X19301484>.
- [11] G. Saul, B. Sebastian, V. Jo, and D. Jakub, *Dark patterns in proxemic interactions: a critical perspective*. Proceedings of the 2014 conference on Designing interactive systems June 2014 Pages 523–532, url = <https://dl.acm.org/doi/abs/10.1145/2598510.2598541>, note = <https://dl.acm.org/doi/abs/10.1145/2598510.2598541> , 2014.

- [12] M. Arunesh, K. Mihir, and M. Jonathan, *What makes a dark pattern... dark? design attributes, normative considerations, and measurement methods*. Proceedings of the 2021 CHI Conference on Human Factors in Computing Systems May 2021 Article No.: 360, 2021. <https://dl.acm.org/doi/abs/10.1145/3411764.3445610>.

New challengers in the field of stance detection

René Steeman

rene.steeman@aalto.fi

Tutor: Tianyi Zhou

Abstract

Stance detection is a method that allows for the automated analysis of opinions on a large scale, by determining if a text is in favor of, neutral, or against a certain topic or statement. Many approaches exist, with this paper dividing them in three groups: classical approaches, deep learning, and large language models. We find that especially GPT-4 performs well on semEval 2016, but takes significant time to run. It is however easier to setup than classical or deep learning approaches. Deep learning methods in turn generally outperform classical approaches.

KEYWORDS: stance detection, large language models, natural language processing, neural networks

1 Introduction

With the increasing amount of information being shared online, being able to determine the opinions towards subjects of interest can be immensely valuable. Stance detection is one of the most powerful methods to accomplish this with. It takes in the text that is to be analysed, which will be referred to as the item. This could be a tweet, a review, or a news article amongst others. The object that this item may be expressing an

opinion towards is referred to as the target. This could for example be a politician, a feature of a product, or a political statement. The output is the stance label, which is often restricted to be either against, neutral, or in favour, with some variations not using the neutral label or adding an option for when the item and target are unrelated. Such systems could help governments to better understand public opinions, or help companies find elements of their service that could be improved.

This paper will map out and analyse three main categories of stance detection systems. The first being the traditional approaches, under which we find systems that use features such as TF-IDF, keywords, and N-grams that are often combined with support vector machines (SVM). The second category consists of deep learning approaches, such as long short-term memory (LSTM) and convolutional neural networks (CNN). Lastly, we have the most recent approach, using large language models such as GPT-4.

1.1 Uses of stance detection

Stance detection has many use cases, ranging from political applications to commercial uses. Bergam et al. [3] for example used it for the analysis of opinions expressed by the US Supreme Court. Karande et al. [13] made use of it to determine the credibility of information shared on social media, similarly Hardalov et al. [12] describes applications for misinformation identification. Another popular application is the analysis of online discussions [14]. As an example of a commercial use-case, Wang et al. [23] used it for predicting stock prices based on the opinions that were expressed by experts on social media.

Some potential use cases that have not yet been extensively reported on in research would be analysing reviews to find which part of a product could be improved, using stance as a signal for recommending more diverse news content, and seeing how people’s opinions change over time.

1.2 Difference between stance and sentiment

Stance detection and sentiment analysis are closely related, but it is important to understand their differences. Sentiment analysis aims to capture the tone used by the item, with it ranging from being negative to positive, but not being in relation to a target. An example to illustrate this difference would be “Policy x would be an awful idea”, which has a

negative sentiment, but depending on the target could have a positive stance. If the target would be a government that wants to end policy x, then the item would be in favour. In case the target would be the policy itself, then the stance would be against. It has been shown that sentiment on its own is not a good predictor for stance, although it may be an additional feature that could be used.

The correlation between sentiment and stance on the SemEval 2016 task 6b dataset is a good example. In the cases where the stance was in favor, the sentiment was negative 56% of the time, neutral for 9%, and positive for just 35% of cases [1]. When the stance was against, this was 67%, 3%, and 30% respectively.

2 Methods of stance detection

The first group of stance detection approaches that we analyse consists of classical machine learning approaches. These are approaches that do not make use of deep learning both for their pre-processing as well as classification algorithms. They commonly use inputs such as: n-grams, sentiment, TF-IDF, part of speech (POS) tags, and hashtags. Such inputs are generally not able to capture the full meaning of the text and the relations between words within a sentence as well as between sentences, but are however easy and quick to generate. As their main classification algorithms, SVMs are particularly commonly used. Others include random forests, k-Nearest Neighbours (kNN), decisions trees, Hidden Markov Models (HMM), and Naïve Bayes (NB).

The second group consists of deep learning approaches. Here we see that common inputs are embeddings, both on the word and sentence or even full text level, along with the inputs earlier mentioned for the first group. Many flavours of classification algorithms are used, popular ones being convolutional neural networks (CNNs) and long short term memory (LSTM), with many variations such as gated recurrent units (GRU) and bidirectional LSTMs.

The final group consists of large language models. Because of the wide range of applications without requiring many context-specific modifications, they can be promising methods that can quickly be deployed without the need for additional adaptations. Popular examples include: GPT-3.5, GPT-4, and Llama2. They are capable of taking in a natural language representation of the classification task and directly return the predicted

classification.

3 Performance comparison

To give context to the kind of data that these algorithms could be used on, as well as possible sources of training data, and benchmarking options, we present several datasets. We shall then perform a comparison to understand the characteristics of the groupings presented earlier.

3.1 Datasets

The field of stance detection has a variety of datasets, however, these do not share a common format making it difficult to find the right benchmark data. Therefore, we shall present a list of some of the most famous datasets and their characteristics in tables 1 and 2.

Name	# stance samples	Classes	Input format	Notes	Availability
Argmin [20]	24430	opposing 24.3%, no argument 56.2%, and supporting 19.5%	topic, sentence	Focuses on fake news detection	Can be requested from the Technische Universität Darmstadt
Arc [10] ¹	4448	agree 40.0%, disagree 45.8%, no stance 24.3%	comment, topic	Focuses on argumentation and only has stance as a sub-task	https://github.com/UKPLab/argument-reasoning-comprehension-task
Fnc1 [8]	99261	unrelated 55.3%, discuss 5.6%, agree 13.5%, disagree 1.5%	article body, title	It labels the stance of an article's body towards its own title and focuses on fake news detection	https://github.com/FakeNewsChallenge/fnc-1

Table 1. A selection of available datasets for stance detection, part 1

¹Based on exported-4448-comments-incl-no-stance.tsv

Name	# samples	Classes	Input format	Notes	Availability
Ibmcs [2]	2394	pro (55.3%), con (44.7%)	debate topic, claim, target		Can be requested from IBM [No response obtained]
Perspectrum [5]	5095	support 51.6%, opposing 48.4%	claim, perspective, evidence	It is based on debates and uses a nested json format instead of being row based	https://github.com/CogComp/perspectrum
Semeval 2016 task 6A [16]	4163	favor 25.3%, against 50.7%, neither 23.9%	target, Tweet		https://alt.qcri.org/semeval2016/task6/index.php?id=data-and-tools
RumourEval 2019 task 7A [9]	8574	supporting 13.8%, denying 7.1%, querying 7.1%, commenting 72.0%	source message, conversation thread	It focuses on rumour evaluation, but contains stance elements. It has both Twitter and Reddit data.	https://competitions.codalab.org/competitions/19938#learn_the_details
Snopes+ [11] ²	16508 ³	support 40.8%, refute 13.7%, no stance 45.5%	claim, evidence snippets	The paper focuses on fact checking, but contains stance data	Can be requested from the Technische Universität Darmstadt
WTWT [6]	51284	support 13.0%, refute 8.2%, comment 40.7%, unrelated 38.1%	Tweet id, merger	Focuses on financial data (mergers and acquisitions) for rumour verification, but contains stance	https://github.com/cambridge-wtw/ac12020-wtw-tweets

Table 2. A selection of available datasets for stance detection, part 2

There are some significant limitations of these datasets. Firstly, they do not share a common structure. Their set of possible labels is not uniform, as well as the structure in which the data is recorded being very different. This makes preparing the files for benchmarking difficult and limits their use. It also makes it difficult to compare methods tested on other datasets. Secondly, the range of content covered by them is narrow. Most datasets only cover a few, often related, topics and especially political statements are overrepresented. Additionally, stance datasets in multiple languages are scarce, making it difficult to see how stance detection systems would perform on languages other than English. Then there is also the issue of most of the texts being very short, partially because of the prevalence of Twitter data. Performance on large texts therefore remains unclear. The number of entries per dataset is also limited, making it hard to see how methods would compare on larger real-world applications where more data is available. Finally, the agreement between labellers is also limited. Therefore it can become unclear whether the more powerful methods are limited by the quality of the data, or other factors. It is therefore generally recommended to have 5 or more labellers per item [10], with Walker et al. [21] noting that humans achieve an accuracy of 77% on average while Habernal et al. [10] reports 80%.

3.2 Performance comparison

In order to compare these models, we first need a shared metric and task definition. The most commonly used metric to measure the performance of stance detection algorithm is the F1-score.

$$F1 = \frac{2 \times (\text{Precision} \times \text{Recall})}{\text{Precision} + \text{Recall}} \quad (1)$$

Where:

$$\text{Precision} = \frac{\text{True positive}}{\text{True positive} + \text{False positive}}$$

$$\text{Recall} = \frac{\text{True positive}}{\text{True positive} + \text{False negative}}$$

The F1 score is the harmonic mean between precision and recall, being in the range of $[0, 1]$. This makes it particularly useful when working with

²The dataset was not named in the paper, but is based on Snopes and therefore referred to as Snopes+

³Based on evidence text snippets 'ETS'

imbalanced datasets. In case of multiple topics, the macro F1 score can be used:

$$\text{Macro F1} = \frac{\sum_{i=1}^N F1_i}{N} \quad (2)$$

3.3 Model type comparison

As the scope of this paper is limited, we shall focus on the Semeval 2016 task 6A dataset as it is the most popular in research, widely available, uses a clear format, and is made specifically for stance detection. As the results for this task are generally reported with an F1-score for the FAVOR and AGAINST classes, even though the algorithms were also classifying for the NONE class, we shall report them in the same way. For each category we shall demonstrate a small selection of models to understand their general performance and the types of inputs that they use.

Classical methods

Firstly, the baseline model created for the competition is an SVM using ngrams [16]. It managed to score 0.69, using a separate classifier for each target.

Zhang and Lan [27] uses a two-step approach, first classifying if a tweet is relevant before then deciding if it is in favor or against. This is a common setup that can be beneficial in cases where the main classifier has difficulties dealing with the NONE class as a third option. It uses a combination of n-grams, POS tags, clustering, whether a word is in the top 20 highest TFIDF scores for a topic, punctuation, latent Dirichlet allocation (LDA), non-deep learning based similarity measures, sentiment, and several other features based on classical methods. Having two classifiers per topic, as described by the two-step approach, they obtained a score of 0.66 with logistic regression as the classification algorithm.

Wojatzki and Zesch [24] made us of n-grams, syntactic features, stance-lexicon, concept features, and target transfer features. Like the previous methods, it has a separate classifier trained for each of the topics. It managed to achieve a score of 0.62 using a SVM.

Deep learning

Schiller et al. [19] reached an F1-score of 0.70 using their MT-DNN_{MDL} model. This model was trained on a collection of 10 datasets. When training their model on semeval only, we see that it reached 0.69 instead (MT-

DNN_{SDL}). The system is based on the BERT transformer architecture [7], extended with a classification layer in the form of a stochastic answer network. The initial weights used for BERT were obtained from a model that was fine-tuned on the GLUE Benchmark [22]. Without using these previously fine-tuned weights, a score of 0.68 was achieved ($BERT_{SDL}$).

LSTMs (Long Short-term Memory) are also a popular method, with Zarrella and Marsh [25] using an input based on word2vec and word2phrase. They managed to achieve a score of 0.68 when training a separate classifier on each topic.

The use of multi-task learning and attention can result in even stronger performance as shown by Li and Caragea [15] which got a score of 0.72 using Bidirectional Long Short-Term Memory.

GPT

Performance for GPT is heavily dependent on the prompt used and can be strongly improved by providing an example. We therefore test two prompts. The first one using the system prompt

“Given a text and a target, return the stance of the text towards the target. 0 means against, 1 is neutral, and 2 means in favor. Only return the number. An example would be text: SpaceX Starship launch from South Texas will happen soon - The Washington Post target: SpaceX return 2” with the user prompt being constructed as "text: [‘body’] target: [‘target’]" where [‘body’] refers to the tweet and [‘target’] to the topic.”

Zhang et al. [26] obtained promising results using chain-of-thought prompting, we therefore also test such a prompt. This prompt is as follows

“Q: What is the stance of the tweet: "I bet Hillary Clinton is the best choice for President in 2016, she’s the next best one for the people." to the target "Hillary Clinton" select from FAVOR, NONE, AGAINST.

A: Let’s think step by step. The tweet uses positive language to describe Clinton, stating that she is the "best choice". Therefore the stance belongs to the FAVOR class.”

Performance can also be strongly improved by fine-tuning the model on a small data set. For the GPT 3.5 Fine Tuned model, as suggested by OpenAI [17], we used 50 samples. These samples were randomly selected from the training set of all topics and the model was trained for 3 epochs. As we do not have chain-of-thought example answers, only prompt 1 could be tested for fine-tuning. The results in terms of F1 score, inference time, and costs can be found in table 3.

Model	F1 Score	Time (seconds)	Cost
GPT 3.5 Turbo prompt 1	0.43	2752	\$0.35
GPT 3.5 Turbo prompt 2	0.68	6065	\$0.66
GPT 3.5 Fine Tuned	0.68	544	\$2.79
GPT 4 prompt 1	0.74	1457	\$6.72
GPT 4 prompt 2	0.68	7700	\$14.03

Table 3. Comparison of GPT Models

Overview

To get a clear overview of how the F1 scores for these models compare, table 4 can be used.

Model	F1 Score
Baseline SVM [16]	0.69
Two-step logistic regression [27]	0.66
SVM [24]	0.62
MT-DNN _{MDL} [19]	0.70
BERT _{SDL} [19]	0.68
LSTM [25]	0.68
Bi-LSTM with attention [15]	0.72
GPT 3.5 Turbo (prompt 2) [4]	0.68
GPT 3.5 Turbo Fine Tuned (prompt 1) [4]	0.68
GPT 4 (prompt 1) [18]	0.74

Table 4. This table gives a short overview of how the models from this section compare on their F1 score

4 Advantages and disadvantages per model type

For the classical methods we can see that they often require extensive manual tuning to find suitable input features for the specific task. They do however still show quite strong performance compared to the other options and can be more computationally efficient. However, their ability to generalise across datasets and tasks is quite limited, with them generally requiring training for each topic individually to achieve the best results.

Deep learning methods on the other hand can lead to better performance when including data from sources other than the exact topic from the benchmark. They also allow for more general inputs, such as embeddings

instead of a large range of (task) specific features. The performance of the deep learning methods is strong, being generally better than the classical methods.

Lastly, we have the GPT models. The default GPT-3.5 Turbo can give strongly varying results depending on the prompt used. Fine-tuning it can deliver vastly better results for some cases, on par with some of the deep learning methods and the best of the classical methods, although it is still rather slow. GPT-4 gave the best score from the methods tested and from what we could find elsewhere in the literature, although it is a very slow and a rather expensive model. The main advantage of the GPT models is their ease of use, GPT-4 can give extremely strong results without even requiring any training data and can be setup with minimal effort. It is also interesting to note that the prompt that worked best for GPT3.5, gave worse results for GPT4, and vice-versa.

5 Conclusion

Our findings indicate that traditional algorithms can still offer robust performance, but at the cost of extensive manual parameter tuning. By contrast, deep learning methods generally yield superior outcomes with less manual intervention. Among the evaluated models, GPT-4 emerged as the most effective, slightly outperforming the deep learning methods, while also being the simplest to configure. However, this advantage comes with the trade-offs of slow inference speed and high costs. Future work could focus on the variability of performance between datasets and the use of other languages.

References

- [1] Abeer Aldayel and Walid Magdy. Assessing sentiment of the expressed stance on social media. In *SocInfo 2019*, pages 277–286. Springer, 2019.
- [2] Roy Bar-Haim, Indrajit Bhattacharya, Francesco Dinuzzo, Amrita Saha, and Noam Slonim. Stance classification of context-dependent claims. In *EACL 2017: Vol. 1*, pages 251–261, 2017.
- [3] Noah Bergam, Emily Allaway, and Kathleen Mckeown. Legal and political stance detection of SCOTUS language. In *Natural Legal Language Processing Workshop 2022*, pages 265–275. Association for Computational Linguistics, December 2022.
- [4] Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Ka-

- plan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901, 2020.
- [5] Sihao Chen, Daniel Khashabi, Wenpeng Yin, Chris Callison-Burch, and Dan Roth. Seeing things from a different angle: discovering diverse perspectives about claims. In *NAACL-HLT 2019: Vol. 1*. Association for Computational Linguistics, June 2019.
- [6] Costanza Conforti, Jakob Berndt, Mohammad Taher Pilehvar, Chryssi Giannitsarou, Flavio Toxvaerd, and Nigel Collier. Will-they-won't-they: A very large dataset for stance detection on Twitter. In *ACL '58*. Association for Computational Linguistics, July 2020.
- [7] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: Pre-training of deep bidirectional transformers for language understanding. In *NAACL-HLT 2019: Vol. 1*, pages 4171–4186. Association for Computational Linguistics, June 2019.
- [8] FakeNewsChallenge. Fake news challenge, 2017. URL <http://www.fakenewschallenge.org/>. Accessed: 16/09/2023.
- [9] G. Gorrell, E. Kochkina, M. Liakata, Ahmet Aker, Arkaitz Z., Kalina B., and L. Derczynski. Semeval-2019 task 7: Rumoureeval 2019: Determining rumour veracity and support for rumours. In *Semantic Evaluation: NAACL HLT 2019*, pages 845–854. Association for Computational Linguistics, 2019.
- [10] Ivan Habernal, Henning Wachsmuth, Iryna Gurevych, and Benno Stein. The argument reasoning comprehension task: Identification and reconstruction of implicit warrants. In *NAACL-HLT 2018: Vol. 1*, pages 1930–1940. Association for Computational Linguistics, June 2018.
- [11] Andreas Hanselowski, Christian Stab, Claudia Schulz, Zile Li, and Iryna Gurevych. A richly annotated corpus for different tasks in automated fact-checking. In *CoNLL '23*. Association for Computational Linguistics, November 2019.
- [12] Momchil Hardalov, Arnav Arora, Preslav Nakov, and Isabelle Augenstein. A survey on stance detection for mis- and disinformation identification. In *NAACL 2022*, pages 1259–1277. Association for Computational Linguistics, July 2022.
- [13] Hema Karande, Rahee Walambe, Victor Benjamin, Ketan Kotecha, and TS Raghun. Stance detection with bert embeddings for credibility analysis of information on social media. *PeerJ Computer Science*, 7:e467, 2021.
- [14] Peter Krejzl, Barbora Hourová, and Josef Steinberger. Stance detection in online discussions. *arXiv preprint arXiv:1701.00504*, 2017.
- [15] Yingjie Li and Cornelia Caragea. Multi-task stance detection with sentiment and stance lexicons. In *EMNLP-IJCNLP*, pages 6299–6305, 2019.
- [16] Saif Mohammad, Svetlana Kiritchenko, Parinaz Sobhani, Xiaodan Zhu, and Colin Cherry. Semeval-2016 task 6: Detecting stance in tweets. In *SemEval-2016*, pages 31–41, 2016.

- [17] OpenAI. Gpt fine-tuning. URL <https://platform.openai.com/docs/guides/fine-tuning/preparing-your-dataset>. Accessed: 2023-09-22.
- [18] OpenAI. Gpt-4 technical report, 2023. URL <https://arxiv.org/abs/2303.08774>. Accessed: 2023-09-22.
- [19] Benjamin Schiller, Johannes Daxenberger, and Iryna Gurevych. Stance detection benchmark: How robust is your stance detection? *KI-Künstliche Intelligenz*, pages 1–13, 2021.
- [20] Christian Stab, Tristan Miller, Benjamin Schiller, Pranav Rai, and Iryna Gurevych. Cross-topic argument mining from heterogeneous sources. In *2018 Conference on Empirical Methods in Natural Language Processing*, pages 3664–3674. Association for Computational Linguistics, October–November 2018.
- [21] Marilyn A. Walker, Pranav Anand, Rob Abbott, Jean E. Fox Tree, Craig Martell, and Joseph King. That is your evidence?: Classifying stance in online political debate. *Decision Support Systems*, 53(4):719–729, 2012. ISSN 0167-9236.
- [22] Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel Bowman. GLUE: A multi-task benchmark and analysis platform for natural language understanding. In *2018 EMNLP: BlackboxNLP*, pages 353–355. Association for Computational Linguistics, November 2018.
- [23] Heyuan Wang, Tengjiao Wang, and Yi Li. Incorporating expert-based investment opinion signals in stock prediction: A deep learning framework. In *AAAI Conference on Artificial Intelligence*, volume 34, pages 971–978, 2020.
- [24] Michael Wojatzki and Torsten Zesch. ltl. uni-due at semeval-2016 task 6: Stance detection in social media using stacked classifiers. In *SemEval-2016*, pages 428–433, 2016.
- [25] Guido Zarrella and Amy Marsh. MITRE at SemEval-2016 task 6: Transfer learning for stance detection. In *SemEval-2016*, pages 458–463. Association for Computational Linguistics, June 2016.
- [26] Bowen Zhang, Xianghua Fu, Daijun Ding, Hu Huang, Yangyang Li, and Liwen Jing. Investigating chain-of-thought with chatgpt for stance detection on social media. *arXiv preprint arXiv:2304.03087*, 2023.
- [27] Zhihua Zhang and Man Lan. Ecnu at semeval 2016 task 6: relevant or not? supportive or not? a two-step learning system for automatic detecting stance in tweets. In *SemEval-2016*, pages 451–457, 2016.

Testing network policies with eBPF

Riste Ristov

riste.ristov@aalto.fi

Tutor: Jacopo Bufalino

Abstract

With the continuous emergence of new technologies that depend on network connectivity, the development of the Internet of Things(IoT), and the increased amount of available software, the amount of devices being targeted by malicious parties is continuously increasing. That is why it is more important than ever to have effective firewall rules to protect the infrastructure from possible intruders. Despite these firewall rules being human-defined and error-prone, few tools are publicly available for efficiently testing the effectiveness of the firewall rules. This paper introduces a hybrid software solution that allows to effectively test the aforementioned firewall rules. This software is compounded from a low-level Extended Berkeley Packet Filter(eBPF) program which runs in the kernel space and a higher-level Python program which processes the output from the C program. The solution is intended to fill in the gap of firewall testing software solutions that can run on low-end hardware. The abstract rule here is that if the hardware can handle the interfaces, it can run the software to test the firewall for those interfaces.

KEYWORDS: *eBPF, firewall, IPTables, kernel, kernel hooks, network, network policy*

1 Introduction

There are many advantages to living in a connected society, such as being able to communicate with anyone, anywhere and at any time, automate factories, buildings, production lines, utilize cloud computers both for professional and personal use. However, there is a significant amount of complex networks involved to ensure that this is feasible. This complexity opens doors for attackers that want to steal data and/or gain control over the desired device. This can be prevented with network policies and firewalls.

The network policies are rules that govern the network traffic flow or it can be said that the network policies define what is allowed, who is allowed and dictate how something is allowed. Most of the network policies are designed and implemented by humans, which means they can be error prone or not effective enough. A simple approach is needed to test this. Monitoring, benchmarking and observations are significant part of the testing process, for which some tool that allows monitoring the effectiveness of the network policies is used. Such tool is the Extended Berkeley Packet Filter (eBPF).

The eBPF is a technology developed for running low-level applications in a privileged manner interacting directly with the core of the operating system without the need to rebuild the core itself. Therefore, eBPF applications can extend capabilities of the system at runtime [1]. This is combined with Just-In-Time (JIT) compiler and verification engine that ensures that the kernel will not be compromised by the application at any time. Typical eBPF applications include network policies enforcement, load-balancing on various scales, tracing applications in the user-space, and network traffic monitoring. The main motivation for this paper is the error-prone network policies defined by humans and the lack of tools that enable to efficiently test them on low-end hardware.

This paper is organized as follows. Section 2 presents the technologies that are generally used for defining network policies and the technologies used in this research paper. Section 3 discusses the setup, structure, and design of the eBPF software solution. Section 4 discusses the results and effectiveness of the solution and finally, Section 5 briefly summarizes the whole paper and presents the conclusions. The complete software solution can be found in [this repository](#)¹.

¹The repository is based off [2]

2 Background

2.1 Network Policies

Formally, as specified in [3], a network policy represents a collection of rules. Each rule is a set of conditions and respective sets of actions. The relation is that conditions define when the set of actions is in effect. When a rule is activated, one or multiple actions from that policy can be executed. The execution of an action can be triggered by meeting or rather not meeting a set of conditions. These actions are in most cases defined by network administrators, which essentially means that they are defined by humans and are error-prone.

2.2 Firewalls & IPTables

The aforementioned network policies with one word can be called a firewall. Two general types of firewalls exist, i.e., network firewalls and host-based firewalls, as demonstrated in [4]. Network firewalls provide security between networks. However, host-based firewalls are used to filter traffic within a network between the hosts [5]. The main difference is that the network policies specified in the network-based firewalls are utilized to conclude whether a packet should be accepted in the network or not, and the network policies in the host-based firewalls are used to decide whether a packet that is already in the network should be accepted by the destination or it should be dropped. As demonstrated on Figure 1, the host-based firewall rules are triggered after the packet has been accepted by the network-based firewall. In this paper, only the host-based firewall is relevant. Hence, from this point on, the term firewall references the host-based firewall.

IPTables is a software tool in UNIX-like operating systems which helps to manage the network policies or more precisely, rules [6]. As the name suggests, the rules are organized in multiple tables. Our paper utilizes the so-called "Filter Table", which contains rules for packet filtering. However, the rules are further organized in so-called "chains", where each policy rule or network policy, which arguably at this point are interchangeable, has its own order and the filtering is done based on this order. How the order of the rules affects packet filtering is out of the scope of this

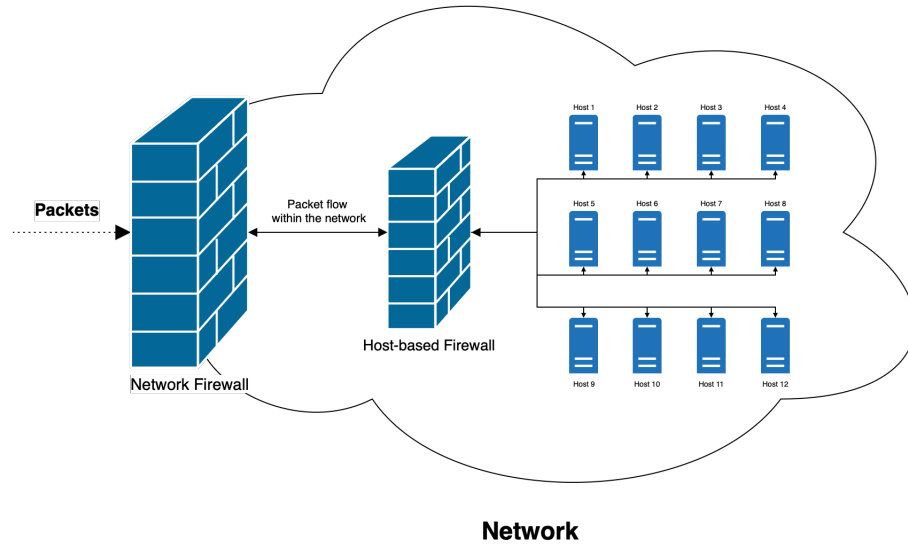


Figure 1. Firewalls and network structure

paper. Generally, there are three chains, **INPUT**, **OUTPUT**, and **FORWARD**, each containing rules that apply to incoming packets, outgoing packets, and forwarding packets, respectively. These tables offer different kernel hooks, which can be used by eBPF to analyze events/packets. Such hooks are used in this paper to test the network policies whether they are having the effect that we expect them to have.

2.3 Extended Berkeley Packet Filter eBPF

The predecessor of **eBPF** is Berkeley Packet Filter (BPF) whose roots can be traced back to McCanne and Jacobson's analysis and description in [7]. Per their analysis, BPF is a kernel architecture for packet capture whose primary objective is to manage and handle packet copying from kernel space to userspace. However, eBPF has increased the amount of functionalities to the point where, as stated in [1], the packet filter relation is deprecated, and it does not accurately represent what eBPF can do. Currently, Linux kernels run only eBPF. The original BPF is translated to eBPF.

The eBPF is a small virtual machine (VM) in the kernel that allows the use of kernel hooks and interaction with almost any sub-process running in the kernel without the need to rebuild the kernel or even restart the system. An important feature of eBPF is that it can be injected in various places in the system, such as between the hardware and kernel space, the kernel itself, on the passage between user space and kernel space, and even in a user space application. In our research, the eBPF application is

running in the kernel space, and it is used as a testing tool to test firewall policies by utilizing IP tables kernel hooks. This means that the program intercepts network packets when they arrive and monitors the rules that get triggered from the IP Tables.

3 Testing network policies with eBPF

Despite network policies being around for a few decades now, few publicly available tools exist for testing them efficiently. This is not an issue when the hosts in question are powerful machines, such as personal computers and servers, because they have the resources, but it becomes an issue when the machines in question are low-powered IoT devices. Recent years have seen increased use of low-powered hardware as a network gateway, which means that this low-powered hardware simply has insufficient resources to run such tools directly. We decided to create such tool using eBPF and document it in a paper that contributes to the IT community by filling in a software gap.

We try to create software that utilizes eBPF to test network policies by monitoring the IPTables. The output of this software is a feedback to the user whether the network policies were set up correctly or not purely based on whether a network packet is dropped or accepted. In this section, first we elaborate on the used infrastructure, continue with the firewall rules that were utilized for testing and finalize by elaborating on the technologies used for development of the software, and the software structure itself.

3.1 Structure

In our development environment, we utilized Vagrant to set up a virtual machine that has predefined firewall rules in the **INPUT** chain². The virtual machine itself uses the *ubuntu/focal64* image offered by Vagrant, and has attached three IPv4 addresses to the `enp3s0` interface illustrated on Figure 2. The interface is specified to have the following IP address range 10.10.0.0/16. There are four http Node.js servers running on 10.10.0.10, 10.10.0.11, 10.10.0.12 and 10.10.0.20. The explanation of Vagrant, Node.js and the other tools used to create this development environment are outside the scope of this research paper. Hence, they will

²Modified version of [8]

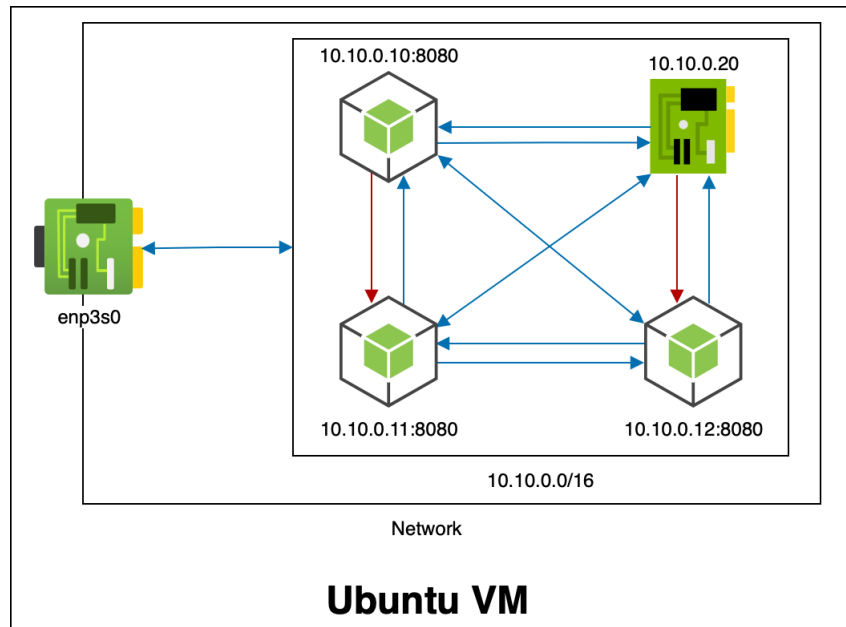


Figure 2. Vagrant Ubuntu VM and network configuration

not be explained.

3.2 Firewall Rules

Between the addresses on the virtual machine, which in our case represent virtual hosts, are set certain firewall rules. The firewall rules are set with the following commands:

```
$ sudo iptables -A INPUT -s 10.10.0.10 -d 10.10.0.11 \
-p tcp --dport 8080 -j DROP
$ sudo iptables -A INPUT -s 10.10.0.10 -d 10.10.0.11 \
-p udp --dport 8080 -j DROP
$ sudo iptables -A INPUT -s 10.10.0.20 -d 10.10.0.12 \
-p tcp --dport 8080 -j DROP
$ sudo iptables -A INPUT -s 10.10.0.20 -d 10.10.0.12 \
-p udp --dport 8080 -j DROP
```

These firewall rules block incoming TCP and UDP traffic from 10.10.0.10 to 10.10.0.11:8080 and from 10.10.0.20 to 10.10.0.12:8080. If we take the first firewall rule as an example, this means that if the initiator of the communication is 10.10.0.10 to 10.10.0.11:8080, the packets will be dropped and the communication will fail, but if the communication is from 10.10.0.11 to 10.10.0.10 then the packets will be accepted. The next

Section elaborates on what technologies we used to create the software, testing procedures, kernel probes, and technologies.

3.3 eBPF program

The eBPF program is composed of two parts, the C program which attaches to IPTables kernel probes and monitors crucial metrics, such as network packets, actions taken by the IPTables, and the rules that are taking place. Terminologically speaking, this is the whole eBPF program. The second part is Python application which is used to attach the C program to the kernel and parse the eBPF output. The Python program requires two arguments, source and destination. Based on these arguments and the parsed output from the C(eBPF) program, which includes taken actions and made decisions for a network packet, a simple boolean value that represents the final verdict is outputted. This boolean value indicates whether the packet from the specified source can reach the specified destination.

Python & C

As mentioned, the technologies used to develop this software are Python and C. To be more specific the Python version used is 3.x (x indicating minor version which is irrelevant in our case). The Python program uses only one external module, **bcc**. This module enables the Python program to attach the eBPF program to the kernel and capture the output. The program itself is very simple, however, only the function used to determine the final verdict will be shown. No explanation will be provided since that is out of scope of this paper. The next code block is the part of the code that parses the output from the eBPF program and determines whether the destination is reachable from a specified source or not:

```
def is_reachable(cpu, data, size):
    event = ct.cast(data, ct.POINTER(TestEvt)).contents
    if event.flags & ROUTE_EVT_IF != ROUTE_EVT_IF:
        return

    if event.ip_version == 4:
        saddr = inet_ntop(AF_INET, pack(="I", event.saddr[0]))
        daddr = inet_ntop(AF_INET, pack(="I", event.daddr[0]))
    else:
        return
```

```

if event.flags & ROUTE_EVT_IPTABLE == ROUTE_EVT_IPTABLE:
    verdict = _get(NF_VERDICT_NAME, event.verdict, "~UNK~")
    hook = _get(HOOKNAMES, event.hook, "~UNK~")

if(f'{saddr}' == BPF_SOURCE and f'{daddr}' == BPF_TARGET and hook):
    if(hook == 'INPUT'):
        global final_verdict
        if(verdict == 'DROP'):
            final_verdict = False
        else:
            final_verdict = True

```

The next code block is part of the eBPF program written in C. The functions shown are used to attach to the IPTables kernel hooks and read specifics from IPTables, such as network packets, IPTables chain(s), and route of these packets.

```

int kprobe__ipt_do_table(struct pt_regs *ctx, struct sk_buff *skb,
const struct nf_hook_state *state, struct xt_table *table)
{ return __ipt_do_table_in(ctx, skb, state, table); };
int kretprobe__ipt_do_table(struct pt_regs *ctx)
{ return __ipt_do_table_out(ctx); }

```

4 Results

We performed two tests. In the first test the previously described Vagrant VM was allocated only 512MB of Random Access Memory(RAM), and in the second test, the same VM was allocated 2048MB of RAM. In these two tests, two metrics were measured. The first metric, shown on Figure 3., measures what is the execution time of the whole program in seconds(*s*), and the second metric, shown on Figure 4. measures the amount of time it takes to make the final verdict about the policy in microseconds(μs). In addition to these measurements, the output from each iteration was captured and compared to an expected value.

Both of tests were performed over 1000 iterations, where the source and destination addresses were randomized with each single iteration to

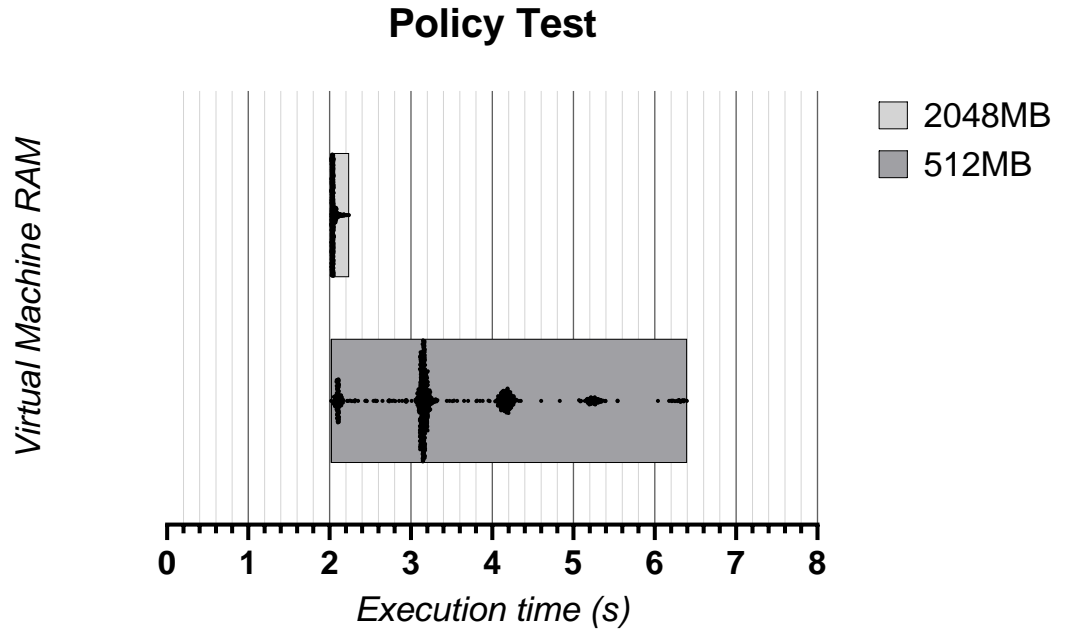


Figure 3. Policy test program running

ensure variations. The tests achieved 100% success rate which means that each of the final verdicts matched with the expected value.

On Figure 3. is shown the time it takes for the Python program to run, i.e. start the C program in the kernel space, generate http packet using "client for URL(cURL)", process the output buffer from the eBPF program, conclude the final verdict, return the output and stop the C program. When the VM was configured with 512MB RAM, the mean execution time of the Python program was $3.407s$, with minimum of $2.020s$ and maximum of $6.394s$. The standard deviation in this case was $0.8680s$. However, when the VM was configured with 2048MB RAM, the mean execution time dropped by $1.364s$, which results in mean time of $2.043s$, with minimum of $2.018s$ and maximum of $2.237s$. Furthermore, the standard deviation dropped to $0.0247s$ which is substantial difference. Moreover, it is interesting to see that the difference between the minimum times is $2ms$. However, the mean time and maximum time differ by substantial amount.

On Figure 4. can be seen the second metric, i.e. the execution of *is_reachable* function which makes the final verdict, whether the destination is reachable from the source. In the first test, the mean execution time of this function was $10.89\mu s$, with minimum of $0.477\mu s$, maximum of $458.2\mu s$, and standard deviation of $20.56\mu s$. The average performance in the second

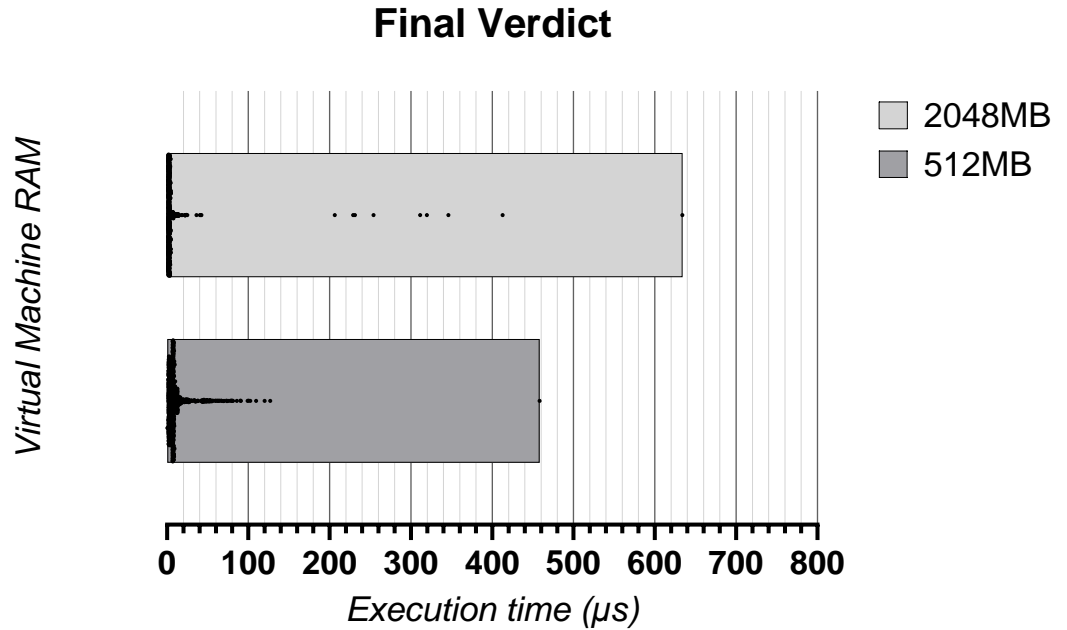


Figure 4. Final verdict execution time

test was better, with mean execution time of $6.091\mu s$, minimum of $1.669\mu s$, maximum of $634\mu s$, and standard deviation of $33.02\mu s$.

The relevance of these measurements depends on the scenario that this program is used in. If the user wants to test firewall policy without going through the complexity of constantly running a custom eBPF program in the kernel space, then the first measurements of whole program execution would be more relevant. However, if the scenario is that the eBPF program is running constantly and only the buffer from it is read on demand and based on that a final verdict is made, the second measurement is more relevant.

5 Conclusion

The presented results are excellent indicator of how effective is this solution. Due to the nature of the software, almost any machine that uses Linux OS distribution is able to run this solution, thus reducing costs and efficiency. Furthermore, this software solution allows the users to effectively test any firewall policy, or multiple policies, within seconds on devices that are not able to run any of the existing solutions.

References

- [1] “What is ebpf? an introduction and deep dive into the ebpf technology.” eBPF, <https://ebpf.io/what-is-ebpf/>, (Accessed: September 20th 2023).
- [2] A. M. (ariel miculas), “Tracepkt.” <https://github.com/ariel-miculas/tracepkt>, 2017.
- [3] G. Stone, B. Lundy, and G. Xie, “Network policy languages: a survey and a new approach,” *IEEE Network*, vol. 15, no. 1, pp. 10–21, 2001. 10.1109/65.898818.
- [4] P. P. Mukkamala and S. Rajendran, “A survey on the different firewall technologies,” *International Journal of Engineering Applied Sciences and Technology*, vol. 5, no. 1, pp. 363–365, 2020. <https://ijeast.com/papers/363-365,Tesma501,IJEAST.pdf>.
- [5] V. T. Patil, P. R. Patil, V. O. Patil, and S. V. Patil, “Performance and information security evolution with firewalls,” *International Science and Technology Journal*, vol. 7, no. 2, pp. 33–38, 2018. https://www.researchgate.net/profile/Vinay-Patil-12/publication/372832097_Performance_and_information_security_evolution_with_firewalls/links/64ca3b95d394182ab3991f64/Performance-and-information-security-evolution-with-firewalls.pdf.
- [6] S. Miano, M. Bertrone, F. Risso, M. V. Bernal, Y. Lu, and J. Pi, “Securing linux with a faster and scalable iptables,” *ACM SIGCOMM Computer Communication Review*, vol. 49, no. 3, pp. 2–17, 2019. <https://doi.org/10.1145/3371927.3371929>.
- [7] S. McCanne and V. Jacobson, “The bsd packet filter: A new architecture for user-level packet capture.” in *USENIX winter*, vol. 46, pp. 259–270, 1993. https://vodun.org/papers/net-papers/van_jacobson_the_bpf_packet_filter.pdf.
- [8] J. B. (jackap), “Seminar topic on ebpf setup.” <https://github.com/jackap/ebpf-example-repo>, 2023.

Prospects of WebAssembly to address performance and security challenges beyond the browser

Roni Kirla

roni.kirla@aalto.fi

Tutor: Hannu Flinck

Abstract

WebAssembly (WASM) is a portable bytecode language designed to address the performance issues of JavaScript. With its near-native speed, portability and security features, WASM has a lot of desirable traits for non-web applications too. This paper explores the prospects of WASM outside the web.

The main use cases can be identified as desktop and mobile applications, embedded systems and IoT-devices as well as server and cloud environments. Various runtimes already exist for running WASM on the desktop, while on mobile the selection is more limited. For embedded systems and IoT-devices, the conclusion is that the current tools and libraries are insufficient to create a full WASM-based stack, but the technology can still be applied to individual modules. In server and cloud environments, the most significant benefits can be gained by replacing containers in simple serverless functions with WASM to reduce cold-boot times.

While WASM is currently not widely used either inside or outside of the web, the technology shows promise and is bound to improve in the future. Even in its current state, it is a worthwhile alternative, and a significant part of its unpopularity can be attributed the dominant position of its competitors.

KEYWORDS: WebAssembly, bytecode, security, portability, performance

1 Introduction

WebAssembly (WASM) is a relatively recent bytecode format developed for the web to address some of the shortcomings of JavaScript [1]. The use of JavaScript as the de facto language of the web has made it outgrow its original design intent. As a result, it is not surprising that the language has significant issues in several areas, such as performance. There have been several attempts at creating a more efficient, lower-level alternative to JavaScript, but none have gained as much traction as WebAssembly has in recent years [2, 3]. It maintains the cross-platform compatibility and safety of JavaScript while providing a performance boost close to that of lower-level languages [1], which makes it seem like an ideal language for the future of the web.

As a bytecode format, WASM shares the same basic principles with the likes of Java [4] and Common Intermediate Language [5] (used for languages like C#). This means that WASM has potential for use in applications other than just the web browser. This could be beneficial, since the problems WASM aims to solve are fairly ubiquitous across environments. The purpose of this paper is to study current and potential future applications of WASM outside the browser.

We begin with a general overview of WASM in section 2 to provide a sufficient background of its security and performance benefits. Then, in section 3 we go through the individual use cases one by one. We then discuss and compare the utility and prospects of these use cases in section 4 and finally summarize our findings in section 5.

2 Overview of WebAssembly

In WASM, code from a higher-level programming language is compiled into a binary format [1]. This compiled bytecode then gets executed by a virtual stack machine. The stack machine, in contrast to a register machine, operates by pushing the operands of instructions to a stack instead of registers and then popping them from the stack and pushing the result when executing the instruction.

In terms of the web, this bytecode-based approach has two main advantages over the interpreted JavaScript: the pre-compiled code is faster to execute and more compact as the compilation phase performs various optimizations, removing the overhead of the interpretation process [1]. An

additional two properties present in JavaScript are also desirable: Cross-platform compatibility and memory safety. As a bytecode format, WASM is naturally cross-platform compatible, but the safety implementation is more novel, as it does not rely on a trust mechanism, such as ActiveX [6], or a garbage collector, such as Java [4]. Instead, it uses a linear memory model, where the memory of a program is separated from its code and other modules, even when sharing the address space, using in-process isolation [1]. Logically, the memory is just a continuous array of bytes.

This memory model, however, does not come without its security flaws. Despite being able to provide protection against memory corruption between modules in the same process address space, it cannot enforce strong memory safety within the sandbox [7]. Therefore, it is susceptible to the traditional memory safety vulnerabilities that are present in the language that gets compiled to WASM [8]. In other words, the isolation is sufficient to prevent bad actors from taking over the system through memory exploitation, but insufficient from the application developer's point of view, because the application itself can still be exploited to behave in undesired ways. Using a memory-safe language like Rust [9], would overcome this. Alternatively, a memory-safe extension, MS-WASM, has been proposed in [7].

Several studies have been conducted on the performance of WASM. The current consensus is that WASM is generally faster than JavaScript, but still considerably slower than native code written in C [10, 2]. On top of performance, [10, 11] show that these results extend to energy efficiency. The performance gap between WASM and JavaScript shrinks as the input size of the program grows, to the point where at large input sizes the differences become indistinguishable in a large proportion of the test cases [12, 10, 11]. The stated reason for this is the lacking optimization of the novel WASM compared to the significantly more mature JavaScript. This means that there is a lot of remaining potential in the technology, and the results will improve in the future as advancements are being made in the compilers and runtimes. Additionally, [11] finds that a significant part of the less convincing results were micro-benchmarks – heavy computational tasks that tend to be quite repetitive. These types of computations can often be heavily optimized using techniques such as Just-In-Time compilation in interpreted languages. The study finds that real world applications have more noticeable gains when using WASM.

3 Use Cases Outside the Browser

As stated, WASM aims to solve three major challenges. First, it has to provide a safe isolated environment in which it can execute untrusted code. Second, it has to provide one compilation target that works across different platforms and programming languages. Lastly, it should offer performance that is comparable to native code.

Looking at these properties, it is clear that their benefits are not simply limited to the web. Spies and Mock [3] provide an overview of the current use cases of WASM outside the web environment. Additionally, various works exist that explore the use cases of WASM in different environments. This section is divided into 4 subsections, each giving an overview of the prospects of WASM in a specific non-web environment.

Before looking into these use cases, it makes sense to investigate how WASM is run outside the browser in the first place. While it can run in virtually any environment, it cannot do anything meaningful, unless it can interface with the system resources, such as the file system and devices [3, 13]. Requiring the compilation target to be platform-independent leaves it up to the target itself provide this access. In the web space, the browser takes care of this. But trying to take the WASM virtual machine out of the browser and running it as a standalone poses a bit of a challenge. Instead of relying on the runtime of another language, a more elegant solution is the WebAssembly System Interface (WASI) [13]. This interface acts between the virtual machine and the operating system of the physical machine, providing the required system call implementations. It is also modular, with the core module only containing the most essential features for every program. More features can be added when relevant to the platform, or fewer modules can be used to keep the system more secure.

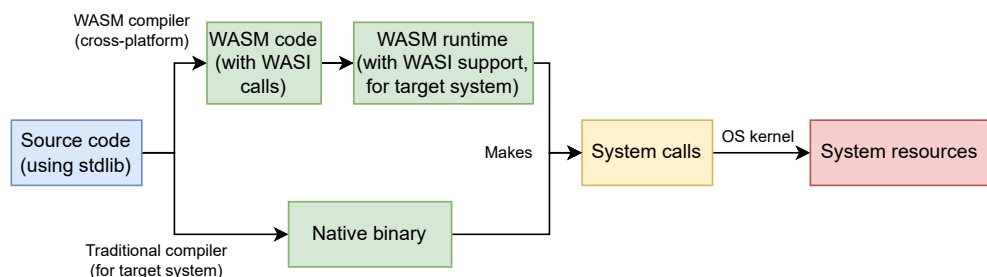


Figure 1. Difference between WASM with WASI [13] and native code in terms of compilation and execution on the target platform as well as accessing system resources.

3.1 Desktop and Mobile Applications

The first use case we explore is perhaps the most obvious one when considering application development. These are the applications that run on our everyday devices, such as desktop computers and mobile devices. Typically, cross-platform compatibility is highly desired in these systems and JavaScript frameworks are already commonly used [3]. WASM runtimes, such as Wasmtime [14], WAVM [15] and Wasmer [16], aim to offer a higher performance alternative to these. These runtimes provide a concrete implementation for the aforementioned WASI, allowing WASM applications to be run as standalone. Out of these, Spies and Mock [3] find WAVM to be the fastest, due to it taking more time optimizing the Just-In-Time (JIT) compilation from the WASM bytecode to the machine code of the target platform.

WAVM currently only supports the three major desktop operating systems, Windows, MacOS and Linux. This means that it cannot be used to run WASM applications on mobile devices. Wasmer supports Android though, and a dedicated Android runtime WasmAndroid has been presented as a prototype [17]. It features an Ahead-of-Time (AoT) compiler, that aims to create an optimized native binary for the platform before running. The study shows promising results, but the technology currently remains more of a proof of concept, as only a prototype is implemented, and no further studies could be found at the time of writing. When it comes to Apple's iOS, generating executable code at runtime is not permitted, so a slower interpreted approach is required, such as Wasm3 [18].

3.2 Embedded systems and Internet of Things devices

Characterized by their varying power, performance and security requirements, embedded systems and Internet of Things (IoT) devices are becoming increasingly common. As they range from simple sensors to critical security systems, WASM sounds like a perfect candidate when it comes to performance, security and compatibility. Kotilainen et al. [19] assess the prospects of WASM in IoT, while Wallentowitz and Kersting [20] analyze the topic in the broader space of embedded devices.

The main advantage of using WASM in embedded systems lies in portability and security [20]. These devices tend to be less standardized, with various operating systems and instruction sets in use. In order to reach sufficient performance, some of these operating systems might not

necessarily have the same level of security features as conventional ones, and unsafe languages like C are often used. Using WASM in this context would definitely pose challenges in terms of performance, but it may be a worthwhile trade-off if the WASM implementations can deliver performance close enough to native code.

The results in [19] show that at its current state, the available tools are not sufficient to create a full IoT technology stack. The absence of available libraries and lacking interfaces with WASI are the main reason for this. As a result, libraries using other languages need to be included, which not only defeats the purpose of a full WASM stack but is also impractical as a substantial amount of boilerplate code is required. Usage of WASM for individual modules is however found to be useful in the same sense that WASM is currently used in the web for performance-critical routines.

Wen and Weber [21] propose a different approach to WASM in IoT, with a full focus on performance. Instead of using an existing runtime for WASM, they propose a complete operating system kernel designed for it. Applications are compiled AoT from WASM and run natively. The compiler translates WASI calls to direct system calls supported by the OS, saving a lot of the overhead that normally happens in the calls. The efficiency is further optimized by taking advantage of the sandboxed nature of WASM, allowing all applications to run as isolated kernel threads. Preliminary results are indeed really promising, showing performance improvements of up to 11% over a traditional Linux kernel. Nevertheless, this is yet another WASM implementation still in the prototype phase.

3.3 Server and Cloud Environments

Traditionally, developers have preferred using the same programming language both on the server and the client. This has led to the popularity of Node.js [22], a JavaScript runtime for non-browser environments, such as the cloud. This would lead to the natural conclusion that a WASM runtime for the cloud would be desirable as the usage of the bytecode format grows in the browser.

However, there is a perhaps more notable application of WASM in the cloud. Kjorveziroski and Filiposka [23] explore the ability to use WASM in serverless functions and evaluate its advantages over a more traditional container-based approach.

While the key properties of WASM, such as portability and modularity

are important in the serverless space, one of the key issues of the normal containerized approach is the long cold start times [23]. As the goal of serverless is to provide seamless execution of functions by abstracting everything about deployment and maintenance away, it is of paramount importance that there is not a substantial amount of delay between a function call and its actual execution. This is hard to provide, though, as without a dedicated server for an application that is always running, some overhead is bound to happen on startup. With containers, this overhead is quite significant. WASM offers a more lightweight runtime, which would alleviate this issue.

The results support this notion [23]. Using a WASM runtime instead of containers cuts the startup times significantly. In terms of actual execution time, WASM is still slower than native, though. As a result, it ends up being optimal to use WASM for smaller lightweight functions, leaving the heavy computing tasks to containers.

A lot of attention is also spent on the various modes of compilation. These are, again, AoT, JIT and interpretation. As expected, the study finds AoT to be the fastest, followed by JIT, with interpretation being the slowest [23]. A previously undiscussed runtime, WasmEdge [24], focuses on the AoT compilation, leaving support for JIT out completely. The inclusion of AoT compilation in all the other WASM runtimes is not something that was brought to attention previously, as AoT was only used in the prototypes for WasmAndroid [17] and Wasmachine [21]. Perhaps AoT is not really treated as an option for most platforms, because it is hardly different from just compiling directly from a programming language to native machine code, producing a platform-specific binary. The isolation features and inclusion of a common system call interface in WASI are still desirable, though. A remaining challenge in a serverless WASM deployment is the current lack of orchestration implementations, something that normally goes hand-in-hand with container technology [23].

3.4 Other Uses

Spies and Mock [3] identify two miscellaneous use cases for WASM outside the browser: smart contracts and polyglot programming. The smart contracts of Ethereum are already executed by a virtual machine, and it is being proposed to replace this with the more efficient WASM VM. Polyglot programming, is the act of using multiple programming languages in a single application. This can be advantageous, since each part of the

program can be written in the most fitting language. WASM already naturally supports multiple languages, so it lends itself well to this.

4 Discussion

As it often is with new technologies, adaptation takes time due to the dominant position of existing technologies. For something new to be adapted, it has to be substantially better to provide sufficient incentive to overcome the hurdles of switching technologies.

Despite the possibility to create high performance cross-platform applications being there, the usage of WASM outside the web currently remains low. This can indeed be attributed to the technology being very new and programmers already being used to other workflows offering similar results, such as .NET and JVM. In theory, the linear memory model should be more efficient than a garbage collector, but with today's hardware the performance gains are often negligible compared to the extra development overhead that porting existing applications to a new technology would cause.

Even in the wider scope, it is well-known that development time is often considered more crucial than run-time. This manifests in the popularity of interpreted languages, such as Python and JavaScript. Only in performance-critical applications or constrained hardware does run-time typically become a concern with modern hardware. Even in those situations, however, WASM struggles to provide a competitive alternative, as it still suffers a performance hit compared to running a language like C, C++ or Rust on bare metal without any virtualization layer in the middle.

As a result, in its normal bytecode form, WASM is in a narrow gap in the trade-off between performance and development time. It is faster and more complicated than interpreted languages, but easier and slower than native code. Ease here refers to the discussed desirable properties, such as cross-platform compatibility and safety guarantees. In this space, WASM seems to have an edge with its linear memory and sandboxing models when it comes to security and performance but has yet to gain the momentum to overcome the popularity of competitors, such as Java or C#. Perhaps comparing WASM to its closest alternatives could be an interesting field of further research. Most published research so far seems to focus on comparing WASM to interpreted languages or native code, which are fundamentally different approaches to execution.

There could still be ways to increase the popularity of WASM outside the aforementioned domain. An advantage to WASM is how it is not bound to a specific programming language, but rather supports a wide variety of ones. This on paper should mean that there is a low barrier to entry for new developers. As such, most of the problems that WASM encounters are currently related to lacking support in the environment. This could mean that once these problems are sorted out, the popularity of WASM could increase rapidly despite its current low rate of adoption. This is especially true if the popular interpreted languages could reliably be compiled to WASM, offering the same ease of development while also bringing the speed benefits of WASM.

The prospects of AoT-compiled WASM on the other hand show promise when it comes to the performance-critical applications. This would offer the best of both worlds, as it would bring the features of WASM to native applications. However, out of the analyzed options these seem to be in the earliest stage of the implementations. This makes sense, as it is a difficult task that involves writing compatible compilers for various architectures and operating systems, maintaining the WASI compatibility while targeting the exact hardware.

One way for WASM to become more popular outside the browser would be for it to become more popular inside the browser first. As one of the dominant code execution platforms, the web defines a lot about the space of software development. This can be seen in the popularity of Node.js. If WASM were to become widely used on the web, it would increase the number of developers familiar with the technology, making it easier for them to develop non-web applications using it as well.

As it stands, the most practical use case of WASM is optimizing routines with heavy computations in applications made in high level languages. These routines are usually the defining factor when it comes to the run-time of an application, making the other parts negligible to the point where it does not matter even if a slower language is used. The routines typically require minimal interfacing with the other components of the software. This makes it suitable for WASM, which at its current state suffers from the lacking compatibility in its system interface. The high-level components of the software can handle those parts, and then provide the required data to the WASM component to perform the computations.

Perhaps the most promising short-term advancement would be the wide-spread use of WASM as a replacement for containers in small server-

less functions. With serverless gaining popularity in web-development as a way to abstract even more of the maintenance overhead away, the solution that WASM offers to the long cold-boot times tackles one of the major challenges that the paradigm faces.

5 Summary and conclusions

This paper has explored the usage of WebAssembly outside the web browser by reviewing its various use cases and the technological solutions that they utilize. It has also discussed the practicality of these use cases alongside other factors that affect the adoption of the technology.

The main use cases for WASM outside the browser were identified as desktop and mobile applications, embedded systems and IoT-devices as well as server and cloud environments. While the results show promise, the technology is currently more at the level of theory and early implementations: various solutions exist for running WASM outside the browser, but they suffer from issues with interfacing with the operating system and have not gained mainstream traction yet. This in turn leads to a lack of support in third-party libraries, making the situation even more difficult.

Despite its current low rate of adoption, the potential in WASM should not be understated. What gives WASM an edge is the flexibility that the technology provides in terms of programming languages and cross-platform compatibility alongside the competitive speed and security. A gradual increase in awareness and traction alongside improving implementations can be projected to lead to an ever-increasing rate of adoption, making the future of WASM look quite promising.

References

- [1] A. Haas, A. Rossberg, D. L. Schuff, B. L. Titzer, M. Holman, D. Gohman, L. Wagner, A. Zakai, and J. Bastien, “Bringing the web up to speed with webassembly,” in *Proceedings of the 38th ACM SIGPLAN Conference on Programming Language Design and Implementation*, pp. 185–200, 2017. doi: 10.1145/3062341.3062363.
- [2] A. Jangda, B. Powers, E. D. Berger, and A. Guha, “Not so fast: Analyzing the performance of {WebAssembly} vs. native code,” in *2019 USENIX Annual Technical Conference (USENIX ATC 19)*, pp. 107–120, 2019.
- [3] B. Spies and M. Mock, “An evaluation of webassembly in non-web environments,” in *2021 XLVII Latin American Computing Conference (CLEI)*, pp. 1–10, IEEE, 2021. doi: 10.1109/CLEI53233.2021.9640153.

- [4] T. Lindholm, F. Yellin, G. Bracha, and A. Buckley, *The Java virtual machine specification*. Addison-wesley, 2013.
- [5] G. C. Necula, S. McPeak, S. P. Rahul, and W. Weimer, “Cil: Intermediate language and tools for analysis and transformation of c programs,” in *International Conference on Compiler Construction*, pp. 213–228, Springer, 2002. doi: 10.1007/3-540-45937-5_16.
- [6] “Activex controls.” [https://msdn.microsoft.com/en-us/library/aa751968\(v=vs.85\).aspx](https://msdn.microsoft.com/en-us/library/aa751968(v=vs.85).aspx). Accessed: September 29, 2023.
- [7] C. Disselkoe, J. Renner, C. Watt, T. Garfinkel, A. Levy, and D. Stefan, “Position paper: Progressive memory safety for webassembly,” in *Proceedings of the 8th International Workshop on Hardware and Architectural Support for Security and Privacy*, pp. 1–8, 2019. doi: 10.1145/3337167.3337171.
- [8] B. McFadden, T. Lukasiewicz, J. Dileo, and J. Engler, “Webassembly: A new world of native exploits on the browser,” *Blackhat briefings*, vol. 2018, 2018.
- [9] N. D. Matsakis and F. S. Klock, “The rust language,” *ACM SIGAda Ada Letters*, vol. 34, no. 3, pp. 103–104, 2014. doi: 10.1145/2692956.2663188.
- [10] J. De Macedo, R. Abreu, R. Pereira, and J. Saraiva, “On the runtime and energy performance of webassembly: Is webassembly superior to javascript yet?,” in *2021 36th IEEE/ACM International Conference on Automated Software Engineering (ASE)*, (Melbourne, Australia), pp. 255–262, 2021. doi: 10.1109/ASEW52652.2021.00056.
- [11] J. De Macedo, R. Abreu, R. Pereira, and J. Saraiva, “Webassembly versus javascript: Energy and runtime performance,” in *2022 International Conference on ICT for Sustainability (ICT4S)*, (Plovdiv, Bulgaria), pp. 24–34, 2022. doi: 10.1109/ICT4S55073.2022.00014.
- [12] W. Wang, “Empowering web applications with webassembly: Are we there yet?,” in *2021 36th IEEE/ACM International Conference on Automated Software Engineering (ASE)*, (Melbourne, Australia), pp. 1301–1305, 2021. doi: 10.1109/ASE51524.2021.9678831.
- [13] L. Clark, “Standardizing wasi: A system interface to run webassembly outside the web.” <https://hacks.mozilla.org/2019/03/standardizing-wasi-a-webassembly-system-interface/>, March 2019. Accessed: October 13, 2023.
- [14] “Wasmtime.” <https://github.com/bytedcodealliance/wasmtime>. Accessed: October 13, 2023.
- [15] “Wavm.” <https://github.com/WAVM/WAVM>. Accessed: October 13, 2023.
- [16] “Wasmer.” <https://wasmer.io/>. Accessed: September 29, 2023.
- [17] E. Wen, G. Weber, and S. Nanayakkara, “Wasmandroid: A cross-platform runtime for native programming languages on android,” *ACM Transactions on Embedded Computing Systems*, vol. 22, no. 1, pp. 1–19, 2022. doi: 10.1145/3530286.
- [18] “Wasm3.” <https://github.com/wasm3/wasm3>. Accessed: October 13, 2023.

- [19] P. Kotilainen, V. Järvinen, J. Tarkkanen, T. Autto, T. Das, M. Waseem, and T. Mikkonen, “Webassembly in iot: Beyond toy examples,” in *International Conference on Web Engineering*, pp. 93–100, Springer, 2023. doi: 10.1007/978-3-031-34444-2_7.
- [20] S. Wallentowitz, B. Kersting, and D. M. Dumitriu, “Potential of webassembly for embedded systems,” in *2022 11th Mediterranean Conference on Embedded Computing (MECO)*, pp. 1–4, IEEE, 2022. doi: 10.1109/MECO55406.2022.9797106.
- [21] E. Wen and G. Weber, “Wasmachine: Bring iot up to speed with a webassembly os,” in *2020 IEEE International Conference on Pervasive Computing and Communications Workshops (PerCom Workshops)*, pp. 1–4, IEEE, 2020. doi: 10.1109/PerComWorkshops48775.2020.9156135.
- [22] “Node.js.” <https://nodejs.org/>. Accessed: September 28, 2023.
- [23] V. Kjorveziroski and S. Filiposka, “Webassembly as an enabler for next generation serverless computing,” *Journal of Grid Computing*, vol. 21, no. 3, p. 34, 2023. doi: 10.1007/s10723-023-09669-8.
- [24] “Wasmedge.” <https://wasmedge.org>. Accessed: October 20, 2023.

Migrating from monolithic architecture to microservices: A comparative analysis of migration approaches

Samuel Ashraff

samuel.ashraff@aalto.fi

Tutor: Antti Ylä-Jääski

Abstract

This paper navigates the migration from monolithic to microservices architecture through a comparative analysis of seminal studies. Firstly, it introduces the primary concepts, highlighting the autonomy and scalability of microservices. The review section surveys multiple migration approaches, including semi-automatic, and case study-driven strategies [1, 2, 3]. The conclusion summarizes findings, equipping practitioners with a comprehensive understanding of this architectural transition.

KEYWORDS: Microservices, Microservice Architecture, Migration

1 Introduction

In recent years, the architectural landscape of software systems has experienced a shift in direction, driven forward by the growing demand for greater scalability, flexibility, and maintainability [3]. This change is best exemplified by the transition from the conventional monolithic architecture to the innovative approach of microservices. The pursuit of this architectural change is fueled by the promise of improved agility, faster development cycles, and enhanced resource utilization [4].

As organizations aim to embrace this new architectural paradigm, the

migration process from monolithic systems to microservices becomes a pivotal endeavor. This paper explores this transition, exploring the multifaceted aspects of migrating from monolithic to microservices architecture. Drawing on studies [3, 1, 2], each contributing a unique perspective and approach to this migration, we conduct a comparative analysis. The insights obtained from these studies will hopefully serve as guideposts for organizations undertaking similar transitions.

The following section introduces the primary concepts, highlighting the autonomy and scalability microservices provide. Section 3 analyzes distinct approaches to the migration, each offering its own set of methodologies, tools, and strategies. The final section concludes this paper by summarizing our findings, thereby equipping practitioners with a comprehensive understanding of this architectural transition.

2 Microservices and Microservice Architecture

This section provides an overview of microservices and their architectural counterpart, exploring their definitions, characteristics, and distinctions. This section begins by examining the essence of microservices, autonomous units of functionality that operate independently, before discussing Microservice architecture (MSA), an innovative architectural design approach that has redefined the way complex systems are constructed [4]. Lastly, this section explores the traditional monolithic system architecture, setting the stage for the comparative analysis.

2.1 Microservices: Autonomous Functional Units

Microservices are self-contained and independent components that encompass a specific function within a software system. They operate autonomously within a system, separate from other services, while communicating with each other through light-weight protocols, such as REST (Representational State Transfer) and HTTP [4]. This allows developers the flexibility to construct, deploy and expand these components independently, thus accelerating development even when system complexity increases [5].

2.2 Microservice Architecture: Modular Scalability

MSA encapsulates a software design approach where an application is structured as a collection of loosely coupled services, each responsible for a specific aspect of the application's functionality [4]. Unlike traditional monolithic architectures, where the entire application is developed as a single, tightly integrated unit, MSA decompose the system into smaller, independently deployable services. Each microservice can be developed, deployed, and maintained separately, resulting in increased development agility.

2.3 Monolithic Systems: Integrated Entities

In contrast, monolithic systems represent a traditional architectural style where all components of an application are tightly integrated and deployed as a single unit [5]. In such systems, any changes or updates to one part of the application necessitate the deployment of the entire system. This approach, while simple to develop and deploy initially, can become difficult to manage and hinder scalability as the application grows in complexity. Monolithic architectures often face challenges in accommodating evolving business requirements, as making alterations to one part of the application may impact the entire system.

3 Literature review: Approaches to Monolithic to Microservices Migration

There are two main steps when migrating a system from monolithic to microservices architecture [1]:

1. Identify how to separate the existing monolithic system into separate, independent entities (referred to later as the identification step).
2. Transform the separate parts of the monolithic application into independent microservices (referred to later as the transformation step).

This section conducts a survey of notable studies, each contributing unique perspectives and methodologies to this important transition. Each selected paper provides a unique approach, shedding light on the multi-faceted nature of monolithic to microservices migration.

3.1 Semi-Automatic migration approach: Selmadji A. et al. 2020

The first step (the identification step) in this migration, however, has three main limitations [1]. Firstly, this identification procedure isn't properly standardized and mostly follows informal guidelines due to the inherent uniqueness and variability of any given monolithic architecture. Secondly, the identification does not account for all the microservices characteristics. Thirdly, it may rely on experts, which can be limiting: these experts are required for the approach to be applied, they need a profound knowledge of the monolith, and they often require intensive intervention for certain steps of the approach.

Selmadji A. et al. [1] have proposed an approach to overcome these limitations. This approach identifies microservices from monolithic Object-Oriented (OO) applications based on a quality function that measures the quality (relevance) of each part of the applications code. This approach, unlike existing ones, has been defined after an analysis of microservice characteristics to ensure that the produced results are relevant and reflect the semantics of the concept microservices. Furthermore, their approach is able to utilize architect recommendations to guide the identification process.

The novelty of this approach lies in how this quality function measures the relevance of a monoliths source code. This quality measurement is inspired by the ISO/IEC 25010:2011 model and divides the monolithic source code into contextually relevant parts required for the migration to MSA.[1].

This quality function has the following key characteristics [1]:

1. Granularity
2. Autonomous
3. Technology-neutral
4. Automatically deployed

Additionally, this quality function aims to identify microservices with maximized quality function values, resulting in a systematic and metric-driven approach to the identification step, but without the three limitations men-

tioned earlier.

3.2 Industrial survey insights: Di Francesco et al. 2018

Di Francesco et al. [2] explore the migration practices towards microservices architecture by employing a dual-phase research methodology involving exploratory interviews and a comprehensive online survey, gathering insights from 18 practitioners across various IT companies and professional backgrounds by means of interviews and questionnaires. Through this survey, Di Francesco et al. aimed to highlight both the activities and challenges associated with this migration. This study provides quantitative data, analyzes practitioner perspectives on migration activities and challenges, and offers a foundation for future research directions.

The study summarizes its findings by gathering answers fed to it by an online survey and grouping them by the survey questions. These summaries provide the following key points when considering the migration from a monolith to microservices architecture:

1. During the migration, the information about the pre-existing system, predominantly the source code, is crucial and can be used for several purposes, most commonly for understanding the system.
2. The most significant actions when designing a new microservice architecture based on a pre-existing monolithic application are domain decomposition (divide domain into smaller subdomains), service identification in the new architecture, application of domain-driven design (DDD), and system decomposition (dividing the pre-existing system into smaller components).
3. When beginning with system decomposition, choose components that have the least dependencies in the pre-existing system.

As for the challenges gathered by this questionnaire, the data provided by this survey clearly shows that the most common challenge is the high-level coupling a pre-existing monolithic system may have as this makes it very difficult to separate functionalities into autonomous microservices [2] (Figure 2). Additionally, once the identification step has been completed, the transformation step brings its own set of challenges as well, especially for developers [2]. This is due to the shear differential nature

TABLE XII
MAIN CHALLENGES - ARCHITECTURE TRANSFORMATION

Challenges	Occurrences
High coupling among parts of the pre-existing system	9
Identification of the boundaries of each service	7
Decomposition of the pre-existing system	6
Automation support for testing	6
Reduce coupling among services in the new architecture	6
Finding the best Business-IT alignment	5
Decomposition of the domain	4
Finding the proper service granularity	4
Lack of proper documentation of the system	3
Bring domain experts into the process of designing the new system	3
Undocumented/uncommented code	2
Other	6

Figure 1.

Table 12 summarizing responses gathered from the survey regarding the main challenges associated with the migration from monolithic to MSA [2]

of MSA compared to monolithic architectures. In monolithic architectures a developer would normally house all of the data in one database, but in the case of MSA this is not possible due to the autonomy of each microservice. This leads to developers having to adopt a completely different mindset during the second step of this migration, which according to the study's findings, was also a major challenge.

3.3 A rapid review: Ponce et al. 2019

Ponce et al. [3] review the challenges inherent in this migration, particularly in the context of systems with complex dependencies. Additionally, it gathers, organizes, and analyzes 20 migration techniques from existing literature based on key research questions to aid practitioners navigate the migration from monolithic to microservices architecture. The study is divided into 4 research topics: different migration techniques, the types of systems these techniques have been applied to, validation the authors of these techniques use, and the challenges associated with the migration.

The first research question explores different migration techniques and summarized its findings as follows:

1. Model-driven (MD), an approach that use design elements as input

2. Static analysis (SA), an approach that requires the source code as input
3. Dynamic analysis (DA), an approach that uses a Graph-based microservice clustering approach using static and evolutionary coupling between software classes [3].

These methods are not mutually exclusive and can be implemented into the migration with each other. However, the most common approach is the Model-driven approach [3].

For the purpose of this paper, we neglect the second and third research topics of this study, and move immediately to the challenges this paper gathered through its findings. The challenges gathered by this study are the following [3]:

1. Database migration
2. Divide business capabilities in sub-business capabilities.
3. Candidate microservices could still need expert judgment before being developed in practice.
4. Distributed system development needs skilled developers.
5. Resources management.

4 Conclusion

It is evident that MSA brings many benefits from many different aspects, such as development and business. However, the migration discussed in this paper is seldom straightforward and with complex monolithic applications can be a great challenge, thereby incurring great costs.

One prominent challenge highlighted in [1] is the identification and separation of entities within the existing monolithic system. Traditional approaches to this task often rely on ad-hoc heuristics, which, as revealed by Selmadji et al., may lack standardization and specificity to the concept of microservices. The limitations of these heuristic-based approaches include their informality, the omission of certain microservices character-

istics, and the reliance on expert knowledge, introducing potential bottlenecks and knowledge dependencies.

To address these limitations, Selmadji et al. [1] propose a semi-automatic migration approach that introduces a quality function to systematically measure the relevance of each part of the monolithic application's source code. This innovative approach aims to overcome the challenges associated with the identification step by providing a metric-driven method inspired by the ISO/IEC 25010:2011 model. The quality function, characterized by granularity, autonomy, technology-neutrality, and automatic deployment, offers a more systematic and standardized means of identifying microservices.

Furthermore, insights from an industrial survey conducted by Di Francesco et al. [2] shed light on the practical aspects and challenges faced by practitioners during the migration. The survey emphasizes the significance of understanding the pre-existing system, employing domain-driven design (DDD), and carefully decomposing the system into smaller, independently deployable components. Challenges identified include the high-level coupling in monolithic systems and the mindset shift required from developers during the transformation step to microservices architecture.

Ponce et al.'s rapid review adds depth to the discussion by categorizing migration techniques into model-driven, static analysis, and dynamic analysis approaches. The prevalence of model-driven techniques is notable, reflecting a common preference for utilizing design elements as input for migration.

In conclusion, while the benefits of transitioning to MSA are substantial [4], the migration journey demands careful consideration, strategic planning, and the application of innovative approaches to reduce the complexity and cost of this migration. The integration of systematic, metric-driven identification methods, insights from industrial practices, and a comprehensive understanding of various migration techniques collectively contribute to navigating the complexities of migrating from monolithic to microservices architecture.

References

- [1] A. Selmadji, A.-D. Seriai, H. L. Bouziane, R. Oumarou Mahamane, P. Zaragoza, and C. Dony, "From monolithic architecture style to microservice one based on a semi-automatic approach," in *2020 IEEE International Conference on Software Architecture (ICSA)*, pp. 157–168, 2020.

<https://doi.org/10.1109/ICSA47634.2020.00023>.

- [2] P. Di Francesco, P. Lago, and I. Malavolta, “Migrating towards microservice architectures: An industrial survey,” in *2018 IEEE International Conference on Software Architecture (ICSA)*, pp. 29–2909, 2018. <https://doi.org/10.1109/ICSA.2018.00012>.
- [3] F. Ponce, G. Márquez, and H. Astudillo, “Migrating from monolithic architecture to microservices: A rapid review,” in *2019 38th International Conference of the Chilean Computer Science Society (SCCC)*, pp. 1–7, 2019. <https://doi.org/10.1109/SCCC49216.2019.8966423>.
- [4] P. Jamshidi, C. Pahl, N. C. Mendonça, J. Lewis, and S. Tilkov, “Microservices: The journey so far and challenges ahead,” *IEEE Software*, 2018. <https://doi.org/10.1109/MS.2018.2141039>.
- [5] N. Dragoni, S. Giallorenzo, A. L. Lafuente, M. Mazzara, F. Montesi, R. Mustafin, and L. Safina, *Microservices: Yesterday, Today, and Tomorrow*, pp. 195–216. Cham: Springer International Publishing, 2017. https://doi.org/10.1007/978-3-319-67425-4_12.

A Comprehensive Overview of Goal-Conditioned Hierarchical Reinforcement Learning: Algorithms, Challenges, and Future Directions

Sergio Hernández

sergio.hernandezgutierrez@aalto.fi

Tutor: Dr. Vivienne Wang

Abstract

Hierarchical reinforcement learning (HRL) methods have recently enabled higher sample efficiency in high-dimensional and long reinforcement learning (RL) problems. Goal-conditioned HRL (GCHRL) approaches concretize these hierarchical ideas by providing reachable sub-goals and considering a chain of policies that model the actions required to reach them, which are either less abstract sub-goals or the agent's native actions. This paper analyses and compares the current state-of-the-art GCHRL methods. Additionally, it discusses the current and future key challenges of the area, including efficient state space exploration, meaningful sub-goal generation and representation, the non-stationarity of policies and the transfer of skills learnt for one problem to solve another. Finally, it contributes to the current discussion on future directions and key focus points within the field of GCHRL.

KEYWORDS: *goal-conditioned, hierarchical, reinforcement learning, machine learning, artificial intelligence*

1 Introduction

Reinforcement learning (RL) has been proven a successful paradigm to solve a wide range of decision-making problems, such as robotics tasks [1, 2], navigation [3, 4] and games [5, 6, 7]. Hierarchical approaches strive to solve longer and higher-dimensional problems in a divide-and-conquer manner by abstracting the time space of the environment [8, 9, 10, 11]. Goal-conditioned methods have been developed to construct such hierarchies [12, 13, 14, 15]; these methods generate sub-goals in higher levels of the hierarchy as the milestones to achieve for lower levels [16, 17, 18]. Several challenges remain unsolved in the area, including ineffective and inefficient exploration, the non-stationarity and instability of low-level policies, producing transferable skills and the generation of diverse and meaningful sub-goals. However, solutions to these issues seem to be within theoretical bounds, and progress in this field could help to solve a new set of more complex real-world problems.

This paper serves as a survey into goal-conditioned hierarchical reinforcement learning (GCHRL) methods and the challenges faced by this area. Section 2 overviews the preliminary background required for the central topic of discussion. Section 3 reviews recently published goal-conditioned hierarchical reinforcement learning (GCHRL) methods and contrasts their differences, advantages and weaknesses. It also describes the current challenges addressed by these methods. Section 4 compares the performance of such methods under different tasks and baselines. Section 5 discusses the state-of-the-art and current lines of research, as well as general schemes to pursue new solutions for the aforementioned challenges. Finally, Section 6 summarizes the state of the GCHRL field at a higher-level and provides an outlook into future lines of work on the topic.

2 Background: reinforcement learning methods

Throughout this paper, we consider a reinforcement learning setting in which an agent interacts with an environment modelled as a Markov decision process (MDP) $\langle S, A, P, R \rangle$. At time t , the environment is in a state $s_t \in S$, and the agent performs an action $a_t \in A$ following a policy $\pi(a|s) \in [0, 1]$. Then, the agent receives a reward $R(s, a) \in \mathbb{R}$, and the environment transitions to a next state s_{t+1} sampled from the transition

function $P(s_{t+1}|s_t, a_t)$. In its deterministic form, the policy chooses the action yielding the highest probability; if acting non-deterministically, an action is sampled from the probability distribution produced by π instead.

The goal of reinforcement learning methods is to find a policy which maximizes the expected discounted rewards over the time horizon:

$$\mathbb{E} \left[\sum_{t=0}^{T-1} \gamma^t R(s_t, a_t) \right], \quad (1)$$

where γ is a discount factor hyperparameter. Nonetheless, calculating the expected discounted reward through this formulation requires knowing in advance the rewards obtained throughout the full trajectory $(s_0, s_1, \dots, s_{T-1})$. To avoid this, we can define a Q-value function $Q(s, a) \in \mathbb{R}$, which can be learnt to approximate the expected discounted reward for a state and action pair [19, 5]; it can be recursively expressed as

$$Q(s_t, a_t) = \sum_{s_{t+1}} P(s_{t+1}|s_t, a_t) \left[R(s_t, a_t) + \gamma \max_{a_{t+1}} Q(s_{t+1}, a_{t+1}) \right]. \quad (2)$$

Hierarchical reinforcement learning (HRL) is rooted at the idea of decomposing a RL task into smaller sub-tasks in a divide-and-conquer fashion [8, 10]. This division enables a temporal abstraction [9, 11]. Given the sequential nature of a RL setting, hierarchical approaches allow for the higher levels to consider actions executed over a larger number of time-steps. Furthermore, it is also appropriate to consider lower-dimensional action spaces in higher levels of the hierarchy [12, 14], since the information needed to operate at more symbolic levels can be summarized. These higher-level action spaces correspond to the sub-goal spaces in lower levels of the hierarchy, which can be either a subset of the state space or a learnt lower-dimensional latent space.

3 Goal-conditioned hierarchical reinforcement learning methods

As proposed in previous literature [9, 20, 12], we can consider a goal-conditioned HRL (GCHRL) setting in which we extend our MDP to be $\langle S, G, A, P, R \rangle$, where $\psi : S \rightarrow G$ is the mapping of states to goals, which constitute meaningful intermediate states for a given task. Without loss of generality, we can define a two-layer structure: a high-level policy $\pi^{hi}(g|s)$, which proposes a sub-goal $g \in G$ given a state $s \in S$; and a low-level policy $\pi^{lo}(a|s, g)$, which selects a native action to take for a given

combination of state s and sub-goal g . The high-level policy attempts to maximize the rewards R when proposing sub-goals, while the objective of the low-level policy is to minimize some distance $D : S \times G \mapsto \mathbb{R}$ between the sub-goal chosen by π^{hi} and the attained next state. The combination of sub-goals and chained policies enable a hierarchical structure, in which the high-level policy chooses a more reachable target for the low-level policy to achieve via native actions.

The following sections review the current state-of-the-art in the field of HRL, particularly focusing on goal-conditioned approaches. While generating and representing sub-goals along the path towards the main objective of a task enables its breakdown into smaller problems hierarchically, it also creates obstacles in the route to make GCHRL methods viable; Section 3.2, Section 3.1, Section 3.3 and Section 3.4 present the major challenges currently hindering progress in the field.

3.1 State space exploration

Most RL approaches encourage exploration using naive heuristics, such as a diminishing constant describing the percentage of instances in which the agent explores instead of exploiting the best found action. In contrast, goal-conditioned methods guide the agent to explore towards promising areas of the state space using more elaborate strategies. Other reinforcement learning paradigms do not offer a sense of exploration directionality.

A particular issue arising during exploration in goal-conditioned approaches is the reachability of sub-goals. To address it, Li et al. [14] propose a slowness regularization which incentivizes goal representations that minimize the distance between contiguous goals. To avoid trivial solutions, they balance the slowness objective with a contrastive objective which encourages ψ to maximize the distance between sub-goals appearing in the same trajectory. This balance is achieved by combining both objectives using a triplet loss [21], as shown in equation 3. Other recent methods apply a reachability regularization to the learning objective as well [18, 16]; reachability constraints improve exploration efficiency, as the higher-level policies only need to search for appropriate sub-goals in a subset of the goal space. Zhang et al. [16] suggest using an adjacency-constrained substitution technique for distant sub-goals: an adjacency network is used to model the distance (in terms of steps) between states and sub-goals, which is used to replace distant sub-goals with reachable ones that would direct the low-level policy towards the same direction.

$$\min_{\psi} \mathbb{E}_{(s_t, s_{t+1}, s_{t+c}) \sim \mathcal{D}} [\|\psi(s_t) - \psi(s_{t+1})\|_2 + \max(0, m - \|\psi(s_t) - \psi(s_{t+c})\|_2)] \quad (3)$$

Li et al. [14] also expose the importance of the reduction of dimensionality applied to the resulting latent sub-goal space, induced by ψ . Without doing this, the lower-level policies would need to explore in still large spaces, which would result in not leveraging the advantages that a hierarchical paradigm offers.

3.2 Sub-goal generation and representation

A major challenge currently obstructing the advancement of the GCHRL field is the process of generating and representing both diverse and meaningful sub-goals, an issue which is also related to effective exploration. Another desirable property of sub-goals is that they should be reachable by the lower-level policies. Existing literature attempts to formalize these requirements by providing quantifiable and unambiguous measures with which to evaluate the goodness of goals. Li et al. [18] propose two measures for the aforementioned purpose: a novelty measure and a potential measure. The novelty measure makes use of visit counts to assess the newness of a given goal; a discounted sum of the visit counts of future goals is added to the visit count of the current goal. In order to regularize the novelty measure (which is prone to mislead exploration, as the representation of goals ψ is non-stationary), the authors also suggest a potential measure, which encourages the minimization of the distance between the ending state and an imagined sub-goal. This imagined sub-goal is an extension in the direction of the proposed sub-goal with a magnitude approximated by the expected distance to the final state. By combining these two measures, we can obtain both novel and attainable sub-goals.

Kim et al. [17] propose the usage of landmarks to generate appropriate sub-goals. In this context, landmarks constitute "promising states to explore". With a similar intent as Li et al., they describe a novelty-based sampling process to obtain landmarks; however, in order to measure the novelty of a state, they employ Random Network Distillation (RND) [22] instead of a visit-counting-based method. RND approximates the novelty of a state using the prediction error given by a neural network. Additionally, Kim et al. integrate a second landmark sampling method to maximize coverage: they sample states maximizing the distance to each other

(in the optimal scenario, we would obtain a lattice of landmarks in the state space). The closest landmark to the current state along the shortest path to the final goal is selected for sub-goal generation. Then, a sub-goal is proposed in the direction of the chosen landmark and at a reachable distance from the current state.

3.3 Non-stationary policies

A major issue inherent to goal-conditioned hierarchical approaches is the non-stationarity of the actions taken by hierarchical policies during training. This happens because the higher-level policies are trained with the assumption of stable state distributions and transition functions, yet the concurrent training of the lower-level policies creates such instabilities (that is, distributional shifts). An additional cause of non-stationarity in high-level policies is the non-determinism of lower-level policies, which is allowed for exploration purposes. In other words, this phenomenon causes high-level policies to pursue a moving target. This problem has been addressed in recent literature.

Nachum et al. [12] devised HIERarchical Reinforcement learning with Off-policy correction (HIRO) to alleviate this issue. HIRO re-labels goals g_t in the replay buffer used for offline learning to a new goal \tilde{g}_t , which maximizes the log-likelihood of producing the same low-level action given the current low-level policy π^{lo} (in practice, the log-probability is evaluated for a small number of diverse candidate goals \tilde{g}_t). In contrast, Levy et al. [13] devised a greedier method known as Hierarchical Actor-Critic (HAC), which eliminates the computational cost of finding and evaluating candidate goals. They propose using hindsight action transitions, which replace the originally proposed sub-goal in each observation by the achieved sub-goal and reward the agent if the original sub-goal was achieved. Levy et al. also use two other types of transitions. Firstly, they employ hindsight goal transitions, which improve the performance of the system in sparse reward scenarios by replacing the goals in the observations with observed states that are reached later in the trajectory. This guarantees a dense updated reward. Secondly, they describe sub-goal testing transitions that, similarly to other methods mentioned in Section 3.1, reward the agent based on the reachability of the proposed sub-goals.

Lee et al. [15] propose Decoupling Horizons Using a Graph in Hierarchical Reinforcement Learning (DHRL), a method which maintains a graph structure of sub-goals. This graph is used to calculate the shortest

path between the projection of the current node $\psi(s_t)$ and the proposed sub-goal sg_t . The proposed sub-goal sg_t is then replaced by the first way-point $wp_{t,1}$ across the shortest path sequence (where, for a path sequence of length k , we consider that $wp_{t,0} = \psi(s_t)$ and $wp_{t,k-1} = sg_t$). DHRL also employs the hindsight action transitions introduced by HAC [13].

3.4 Transferable skills

Designing and engineering general-purpose intelligent systems is one of the ambitions that is common to all paradigms in artificial intelligence. GCHRL systems also share this ambition. However, transferring skills learned in one domain to a different domain is particularly difficult when the representation of state and action spaces are tailored to a given problem. GCHRL methods additionally have to succeed at representing goals generically. Mnih et al. [23] use raw pixels as the state representation in order to train an agent using RL to learn to play a wide range of games. Li et al. [14] show that their GCHRL method, LEarning Subgoal representations with SLOW dyNAMics (LESSON), is able to learn transferable skills between different tasks performed by the same robot. Their system displays such transferring capabilities thanks to the general nature of the latent sub-goal space, which in turn implies that the raw actions given by the low-level policy are also task-invariant.

4 Empirical comparison of current methods

This section compares the methods described throughout Section 3. The key metric to assess the performance of the algorithms is the rate at which the agent succeeds in a given task as a function of the number of environment steps taken during training. This metric provides an insight into the sample efficiency these methods are able to achieve, that is, the number of data points needed to converge to optimal behavior. The evaluation is performed under different MuJoCo tasks [24] (MuJoCo is a physical simulation engine thoroughly used in RL research). The experiments shown in this section have been collected from the works of the authors of such methods. For reproducibility purposes, one can refer to the original works, in which implementation details are included.

In Figure 1, we can observe that DHRL converges faster than other methods in all tasks. We can also observe that DHRL, HIGL and HRAC

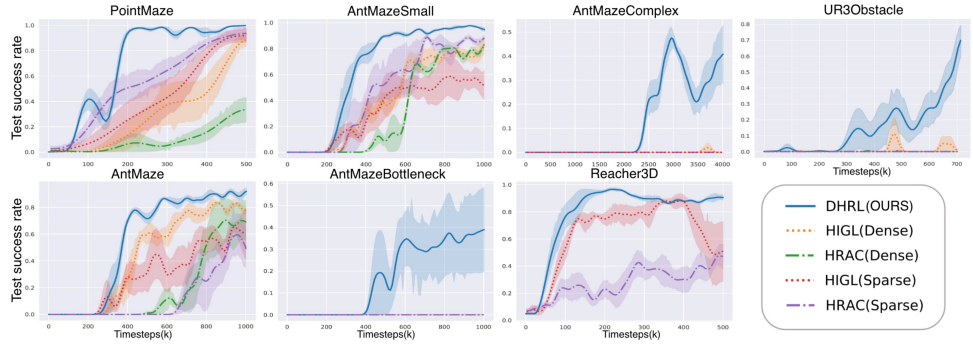


Figure 1. A comparison plot of test success rate as a function of environment time-steps for the methods DHRL [15], HIGL [17] and HRAC [16]. These experiments and summary graphics were created by Lee et al. [15].

display faster convergence to optimal behavior than the earlier methods HESS and LESSON, as observed in Figure 2 for tasks that are also reflected in Figure 1. We can also observe that DHRL can solve some problems that other methods cannot solve in any capacity under the given time-steps constraints, namely AntMazeComplex and UR3Obstacle; this is also the case for HESS in Ant FourRooms. All tasks are too complex for the non-hierarchical method (SAC) to solve them satisfactorily. Figure 2 also compares hierarchical variants of methods employing an Intrinsic Curiosity Module (H-ICM) [25] and successor representation bonus rewards (H-SR) [26]. These two methods improve exploration by employing the prediction error of states as an intrinsic reward (in the case of H-ICM) and formulating the successor representation norm as a reward that encourages exploration towards less visited states (in the case of H-SR).

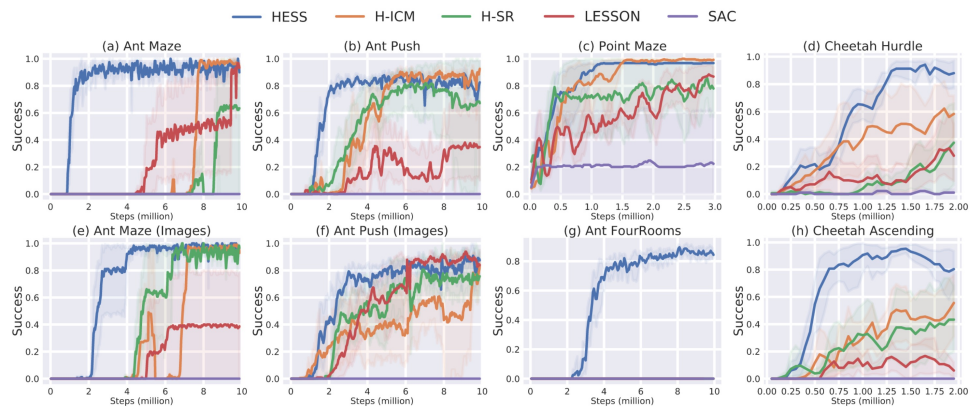


Figure 2. A comparison plot of test success rate as a function of environment time-steps for the methods HESS [18] and LESSON [14], as well as for a non-hierarchical RL approach based on Soft Actor-Critic (SAC) [27]. The methods H-ICM [25] and H-SR [26] are also included. The experiments and summary graphics were developed by Li et al. [18].

5 Discussion

State-of-the-art hierarchical methods have exhibited higher sample efficiency than non-hierarchical systems. For complex problems, the earlier methods provide solutions that the latter do not find, given reasonable constraints on time and resources. GCHRL methods provide a simple formulation based on goal states to enable a hierarchical decomposition of problems. A key drawback of GCHRL approaches is the scalability of the system: increasing the amount of levels of abstraction requires more policy networks, which exacerbate non-stationarity issues and strain computing and memory requirements. A second challenge is their dependency on well-defined goals; in many tasks, it is difficult to define goal states that represent successful trajectories.

Another limitation of most hierarchical approaches, including goal-conditioned methods, is the computational complexity of their algorithms. These techniques involve data structures that are computationally expensive to traverse and employ, such as graphs [16, 15], as well as inner loops and additional procedures, which may be performed during the processing of every batch of observations. Previous research on the GCHRL area emphasizes the ability of the proposed methods to increase sample efficiency, but often fails to report the time it takes for them to process the given data. However, the computational complexity of these systems and the empirical time needed to run them are valuable points for analysis; feasible run-times are a necessary condition for the successful applicability of these methods to real-world problems. A study of memory complexity can also be of value to assess the scalability of these systems. A key direction of work in this sense is the parallelization of data processing; particularly, parallelizing the training of each policy level. If different levels of the hierarchy can be trained independently in parallel and asynchronously, hierarchical methods can achieve computational complexities similar to those of regular RL systems. Furthermore, this type of parallelization could enable deeper hierarchies with a higher amount of policy levels, whose training could be performed under reasonable computational constraints.

While hierarchical ideas are applicable to any problem in nature, goal-conditioned approaches are suitable only for scenarios where goals are meaningful. Nonetheless, GCHRL methods are a sensible choice for these scenarios. Particularly, these methods should be considered for highly-dimensional and extended problems; for simpler tasks, non-hierarchical

methods can achieve optimal performance within reasonable time bounds. Goal-conditioned hierarchical methods can also yield superior performance in settings with sparse rewards: the directional guidance these approaches provide during exploration can be exploited particularly when high-reward areas are rare.

6 Conclusion

This paper has described the goal-conditioned approach to hierarchical RL, reviewed and compared the major state-of-the-art methods in the field, listed the key challenges faced and tackled by such methods and discussed the strengths, weaknesses and applicability of GCHRL systems. Overall, these systems offer a promising direction to solve reinforcement learning problems involving large state spaces and prolonged tasks. While several challenges remain open, recent progress in the area has demonstrated the potential and feasibility of these ideas.

Hereafter, this area of research can benefit from focusing on reducing the computational complexity attached to increasing the number of abstraction layers; this would enable GCHRL methods to tackle more complex real-world problems. An additional necessity on this front is to devise training techniques that mitigate the distributional shifts occurring due to the hierarchical structure of the policy networks. Model-based RL ideas could be explored: non-stationarity issues may be alleviated if policies at each level consider a more stable model of the environment. In systems of larger scales, we would also expect to observe a higher capability for generalization and, hence, diverse policies that are transferable from one task to another.

The hierarchical paradigm offers an avenue for the applicability of reinforcement learning systems to the real world, and goal-conditioned formulations are a robust and well-defined approach to materialize these ideas in suitable scenarios. With current and future research focusing on the challenges presented in this paper, the area of GCHRL provides promising ideas with which to tackle new sets of more complex and larger problems.

References

- [1] J. Schulman, S. Levine, P. Moritz, M. I. Jordan, and P. Abbeel, “Trust Region Policy Optimization,” Apr. 2017. doi: 10.48550/arXiv.1502.05477.
- [2] T. P. Lillicrap, J. J. Hunt, A. Pritzel, N. Heess, T. Erez, Y. Tassa, D. Silver, and D. Wierstra, “Continuous control with deep reinforcement learning,” July 2019. doi: 10.48550/arXiv.1509.02971.
- [3] Y. Zhu, R. Mottaghi, E. Kolve, J. J. Lim, A. Gupta, L. Fei-Fei, and A. Farhadi, “Target-driven Visual Navigation in Indoor Scenes using Deep Reinforcement Learning,” Sept. 2016. doi: 10.48550/arXiv.1609.05143.
- [4] P. Mirowski, R. Pascanu, F. Viola, H. Soyer, A. J. Ballard, A. Banino, M. Denil, R. Goroshin, L. Sifre, K. Kavukcuoglu, D. Kumaran, and R. Hadsell, “Learning to Navigate in Complex Environments,” Jan. 2017. doi: 10.48550/arXiv.1611.03673.
- [5] V. Mnih, K. Kavukcuoglu, D. Silver, A. A. Rusu, J. Veness, M. G. Bellemare, A. Graves, M. Riedmiller, A. K. Fidjeland, G. Ostrovski, S. Petersen, C. Beattie, A. Sadik, I. Antonoglou, H. King, D. Kumaran, D. Wierstra, S. Legg, and D. Hassabis, “Human-level control through deep reinforcement learning,” *Nature*, vol. 518, pp. 529–533, Feb. 2015. doi: 10.1038/nature14236.
- [6] D. Silver, A. Huang, C. J. Maddison, A. Guez, L. Sifre, G. van den Driessche, J. Schrittwieser, I. Antonoglou, V. Panneershelvam, M. Lanctot, S. Dieleman, D. Grewe, J. Nham, N. Kalchbrenner, I. Sutskever, T. Lillicrap, M. Leach, K. Kavukcuoglu, T. Graepel, and D. Hassabis, “Mastering the game of Go with deep neural networks and tree search,” *Nature*, vol. 529, pp. 484–489, Jan. 2016. doi: 10.1038/nature16961.
- [7] J. Schrittwieser, I. Antonoglou, T. Hubert, K. Simonyan, L. Sifre, S. Schmitt, A. Guez, E. Lockhart, D. Hassabis, T. Graepel, T. Lillicrap, and D. Silver, “Mastering Atari, Go, chess and shogi by planning with a learned model,” *Nature*, vol. 588, pp. 604–609, Dec. 2020. doi: 10.1038/s41586-020-03051-4.
- [8] E. D. Sacerdoti, “Planning in a hierarchy of abstraction spaces,” *Artificial Intelligence*, vol. 5, pp. 115–135, June 1974. doi: 10.1016/0004-3702(74)90026-5.
- [9] P. Dayan and G. E. Hinton, “Feudal Reinforcement Learning,” in *Advances in Neural Information Processing Systems*, vol. 5, Morgan-Kaufmann, 1992.
- [10] R. Parr and S. Russell, “Reinforcement Learning with Hierarchies of Machines,” in *Advances in Neural Information Processing Systems*, vol. 10, MIT Press, 1997.
- [11] R. S. Sutton, D. Precup, and S. Singh, “Between MDPs and semi-MDPs: A framework for temporal abstraction in reinforcement learning,” *Artificial Intelligence*, vol. 112, pp. 181–211, Aug. 1999. doi: 10.1016/S0004-3702(99)00052-1.
- [12] O. Nachum, S. Gu, H. Lee, and S. Levine, “Data-Efficient Hierarchical Reinforcement Learning,” Oct. 2018. doi: 10.48550/arXiv.1805.08296.
- [13] A. Levy, G. Konidaris, R. Platt, and K. Saenko, “Learning Multi-Level Hierarchies with Hindsight,” Sept. 2019. doi: 10.48550/arXiv.1712.00948.

- [14] S. Li, L. Zheng, J. Wang, and C. Zhang, “Learning Subgoal Representations with Slow Dynamics,” Oct. 2020. doi: [conf/iclr/LiZWZ21](https://arxiv.org/abs/2010.08111).
- [15] S. Lee, J. Kim, I. Jang, and H. J. Kim, “DHRL: A Graph-Based Approach for Long-Horizon and Sparse Hierarchical Reinforcement Learning,” Nov. 2022. doi: [10.48550/arXiv.2210.05150](https://arxiv.org/abs/2210.05150).
- [16] T. Zhang, S. Guo, T. Tan, X. Hu, and F. Chen, “Generating Adjacency-Constrained Subgoals in Hierarchical Reinforcement Learning,” Oct. 2020. doi: [10.48550/arXiv.2006.11485](https://arxiv.org/abs/2006.11485).
- [17] J. Kim, Y. Seo, and J. Shin, “Landmark-Guided Subgoal Generation in Hierarchical Reinforcement Learning,” Dec. 2021. doi: [10.48550/arXiv.2110.13625](https://arxiv.org/abs/2110.13625).
- [18] S. Li, J. Zhang, J. Wang, Y. Yu, and C. Zhang, “Active Hierarchical Exploration with Stable Subgoal Representation Learning,” Mar. 2022. doi: [10.48550/arXiv.2105.14750](https://arxiv.org/abs/2105.14750).
- [19] C. J. C. H. Watkins and P. Dayan, “Q-learning,” *Machine Learning*, vol. 8, pp. 279–292, May 1992. doi: [10.1007/BF00992698](https://doi.org/10.1007/BF00992698).
- [20] A. S. Vezhnevets, S. Osindero, T. Schaul, N. Heess, M. Jaderberg, D. Silver, and K. Kavukcuoglu, “FeUdal Networks for Hierarchical Reinforcement Learning,” Mar. 2017. doi: [10.48550/arXiv.1703.01161](https://arxiv.org/abs/1703.01161).
- [21] S. Chopra, R. Hadsell, and Y. LeCun, “Learning a similarity metric discriminatively, with application to face verification,” in *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR’05)*, vol. 1, pp. 539–546 vol. 1, June 2005. doi: [10.1109/CVPR.2005.202](https://doi.org/10.1109/CVPR.2005.202).
- [22] Y. Burda, H. Edwards, A. Storkey, and O. Klimov, “Exploration by Random Network Distillation,” Oct. 2018. doi: [10.48550/arXiv.1810.12894](https://arxiv.org/abs/1810.12894).
- [23] V. Mnih, K. Kavukcuoglu, D. Silver, A. Graves, I. Antonoglou, D. Wierstra, and M. Riedmiller, “Playing Atari with Deep Reinforcement Learning,” Dec. 2013. doi: [10.48550/arXiv.1312.5602](https://arxiv.org/abs/1312.5602).
- [24] E. Todorov, T. Erez, and Y. Tassa, “MuJoCo: A physics engine for model-based control,” in *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 5026–5033, Oct. 2012. doi: [10.1109/IROS.2012.6386109](https://doi.org/10.1109/IROS.2012.6386109).
- [25] D. Pathak, P. Agrawal, A. A. Efros, and T. Darrell, “Curiosity-driven Exploration by Self-supervised Prediction,” May 2017. doi: [10.48550/arXiv.1705.05363](https://arxiv.org/abs/1705.05363).
- [26] M. C. Machado, M. G. Bellemare, and M. Bowling, “Count-Based Exploration with the Successor Representation,” Nov. 2019. doi: [10.48550/arXiv.1807.11622](https://arxiv.org/abs/1807.11622).
- [27] T. Haarnoja, A. Zhou, K. Hartikainen, G. Tucker, S. Ha, J. Tan, V. Kumar, H. Zhu, A. Gupta, P. Abbeel, and S. Levine, “Soft Actor-Critic Algorithms and Applications,” Jan. 2019. doi: [10.48550/arXiv.1812.05905](https://arxiv.org/abs/1812.05905).

Query Formation: A Comparative Analysis of Search Engines and AI Models

Sneha Saj

sneha.saj@aalto.fi

Tutor: Rongjun Ma

Abstract

Understanding the query-creation process and bridging gaps is crucial for elevating user prompts and achieving desired outcomes. This research delves into user query formation and result correlation in search engines and AI models to enhance insights into user interactions. The study utilizes a comprehensive methodology, measuring key query formation metrics, and conducting semi-structured interviews to explore thoughts and strategies of participants. By addressing these aspects, the research aims to optimize user experiences related to queries and their outcomes in artificial intelligence (AI) model interactions.

***KEYWORDS:** Information Retrieval, Search Engines, (artificial intelligence) AI Language Models, Generative Pre-trained Transformer-3 (GPT-3), User-Centric Approach, Query Formation, Prompt Engineering, Query Formation, ChatGPT*

1 Introduction

The digital age has revolutionized information retrieval methods, bringing challenges that impact both conventional web search engines and advanced artificial intelligence(AI) - driven systems. Central to this trans-

formation is the influence on user experience and the efficacy of query creation. Proficient query formulation, critical for refining search engines and enhancing AI-driven models, relies on understanding user intent and their information needs [1].

The emergence of web search engines, AI-powered chatbots, and language models such as Generative Pre-trained Transformer-3 (GPT-3) has empowered users with a versatile toolkit for extracting information from the internet. Yet, this technological evolution has also given rise to 'information overload,' a phenomenon where users struggle with efficiently navigating vast volumes of online data [2]. While basic web browsing skills demand minimal training, achieving proficiency in query-based searching and inter-site navigation requires more substantial experience, highlighting the necessity of understanding user strategies [2].

To unravel the intricacies of information-seeking behavior, it is crucial to delve into the objectives and strategies users employ. Rosie Jones' study [1] provides insights into the nature of user interactions with search engines within web browsers, emphasizing the influence of users' goals on query formulation. Furthermore, the advent of AI models such as GPT-3 and AI-driven chatbots is expanding the dynamics of query formation beyond traditional web search engines. These AI models offer a unique perspective for examining the complexities of query formation and human-technology interaction due to their ability to understand and generate natural language with a high degree of contextuality, enabling conversational interactions between users and AI systems[2].

This paper aims to explore people's interactions with search engines and AI models such as GPT-3, with a primary focus on query formation in search. In the following sections, we will discuss query formation. Moreover, we will investigate areas for future research, including the creation of prompt generators guided by user behavior and preferences.

The paper is organized as follows. Section 2 explores the historical evolution of search engines and AI models, addressing their challenges and information retrieval methods, with a focus on query formation. Section 3 covers methodology and data analysis. Section 4 discusses the outcomes of the user study, exploring query formation and the correlation between queries and search results. Finally, Section 5 summarizes key findings and discusses their implications.

2 Background

2.1 Evolution of Search Engines and AI Language Models

The evolution from web search engines to AI language models reflects a shift driven by the changing needs and expectations of users in the digital landscape. Several key trends have pushed this transformation, addressing the limitations of conventional approaches and ushering in a new era of more personalized and efficient information retrieval.

S. Brin's work [3] outlines the historical development of web search engines, emphasizing the challenges users faced in obtaining relevant and contextually meaningful information. The transition towards AI language models has been motivated by a growing demand for enhanced user experiences. Users now seek more than just keyword-based search results; they crave natural and context-aware interactions with digital platforms.

GPT-3, an example of this evolution developed by OpenAI, addresses the desire for more human-like interactions with technology. Explored in OpenAI's work, GPT-3's natural language understanding and generation capabilities allow for a more intuitive and conversational interaction, transforming the way users engage with information retrieval systems. This shift is not just about finding information; it's about creating a dialogue between users and AI systems[4].

BERT (Bidirectional Encoder Representations from Transformers) has furthered this transformation by significantly improving natural language understanding, as documented in related research. Users today expect search engines and language models to comprehend complex queries, understand context, and provide responses. BERT's contributions address this need, ensuring that users receive more accurate and relevant information in response to their queries [5].

The shift from web engines to AI language models is driven by the user's demand for a more tailored and efficient information retrieval process. Search engines often struggled with understanding user intent and context, delivering results primarily based on keyword matching and lacking depth. The evolution to AI language models responds directly to expectations of users for personalized interactions with technology. In conclusion, this shift signifies a user-centric approach to information retrieval, where users now anticipate systems capable of understanding context, providing relevant information, and engaging in natural lan-

guage interactions.

2.2 Challenges in Search: Web Search Engine vs. AI-Driven Queries

Effectively formulating queries and retrieving relevant information poses significant challenges for both conventional web search engines and sophisticated AI-powered systems. These challenges directly impact user experience and the effectiveness of query crafting. Proficient query formulation relies on comprehending user intent and their specific information needs. This understanding is crucial for enhancing search engines and AI-driven models. D.E. Rose and D. Levinson [1] have delved into the complexities users face when expressing their information needs.

When examining challenges in web search [3], we gain a foundational understanding of web search engines, which includes the difficulties in relevance ranking and organizing vast web content. Furthermore, there has been significant progress in natural language understanding, exemplified by Devlin et al.'s work on pre-training deep bidirectional transformers for AI language models [4].

Challenges with AI queries involve the details of regular language and making sure the responses fit the context. While advanced models excel in natural language understanding and generation, they are not immune to limitations, such as language generation precision and the need for enhanced user intent comprehension, as outlined in [5]. These challenges are relevant to both conventional web search and AI-driven queries, highlighting the dynamic nature of information retrieval.

A comprehensive understanding of user intent, query formulation, and technological capabilities is essential for addressing these challenges and advancing the field of search. As we strive to address these challenges in both web search and AI-driven queries, it becomes evident that comprehending user intent and alleviating ambiguity are crucial for enhancing the overall search experience.

2.3 Web Browsing vs. AI Models: Contrasting Information Retrieval Approaches

To discern disparities between conventional web browsing and AI language models for information retrieval, illuminating the shifting landscape of search and the inherent dichotomies in these methodologies. In a comprehensive study [6], R. Xu directly compared ChatGPT and Google

in terms of search performance and user experience, revealing significant differences. ChatGPT excels in understanding user queries within a conversational context, providing responses tailored to the specific conversation. Google, on the other hand, demonstrates strength in delivering concise and well-structured information. These distinctions highlight the complementary nature of these methodologies, with ChatGPT flourishing in interactive and exploratory searches, while Google remains steadfast in providing straightforward facts.

Traditional AI systems operate on rule-based principles and excel in specific, well-defined tasks, whereas language models such as GPT-3 rely on extensive pre-training and adapt to a wide range of natural language understanding tasks. This differentiation underscores the adaptability of language models and their ability to handle a broader spectrum of information retrieval tasks, as observed in ([7]). Y. Wu et al., while primarily focused on machine translation, indirectly shed light on the technological underpinnings of search engines, as discussed in ([8]). Their work reveals the role of neural networks in search engines, such as Google, and how these networks contribute to the quality of translations and search results. This insight highlights the reliance of web search engines on neural network technologies to enhance information retrieval ([8]).

In conclusion, while web search engines excel at providing structured and factual information, AI language models shine in conversational and exploratory searches. Each approach possesses distinct strengths, and comprehending these strengths is crucial for optimizing information retrieval strategies and elevating the quality of user experiences.

2.4 Query Making in Search Engines and AI Models

This section delves into the process of query formation in information retrieval, examining user interactions with both web search engines, such as Google, and AI language models such as GPT.

2.4.1 How Users Form Queries in Search Engines

User query formation in search engines is a dynamic process influenced by individual preferences, behaviors, and evolving search habits. As users initiate searches, expressing their information needs in natural language, search engines encounter challenges in interpreting ambiguous terms and contextual details [9]. This difficulty highlights a critical as-

pect where the user's intent and the algorithm's ability to decipher query meaning may diverge.

The significance of context-aware search, which necessitates query rewriting to capture meanings, is underscored [10]. This is particularly relevant when dealing with words that have multiple interpretations, highlighting the need for search engines to bridge this contextual understanding gap for accurate interpretation of user intent. Moreover, research by White and Morris delves into the keyword-based query patterns of users, showcasing their proficiency in tailoring queries with specific keywords to refine search results [11]. This reveals the diversity in user search strategies and preferences, emphasizing a crucial area where users' approaches to query formation may significantly differ.

Furthermore, the impact of users on browsing behavior introduces another layer to the user-query interaction. Users often engage in iterative and exploratory search sessions, refining queries based on initial search results [11]. The challenge lies in search engines supporting and adapting to these iterative behaviors, ensuring a seamless experience for users refining their queries during the search process.

In conclusion, understanding and addressing challenges in user query formation, such as interpreting intent, overcoming vocabulary disparities, and adapting to diverse search strategies, are crucial for evolving search engines. An understanding of user behaviors, especially the iterative nature of query refinement, is essential for providing an optimal user experience that aligns with the dynamic ways users articulate their information needs in the digital space.

2.4.2 How users form queries in AI Models

The evolution of AI models has transformed our interaction with digital information, extending beyond mere query understanding. Marcus's insights into broader AI challenges underscore the significance of robustness, commonsense reasoning, and transparency in addressing the complexities of user queries [7]. This collaborative approach, merging BERT's linguistic advancements with Marcus's interdisciplinary vision, recognizes the limitations of language understanding and emphasizes the need for AI systems to expand their knowledge base across diverse contexts [4] [7]. Transparent AI models, advocated by Marcus, provide users with explanations for recommendations, fostering trust and understanding in user

interactions.

As users engage with AI models such as ChatGPT, their interactions are shaped by practical needs, expectations, and the tool's capabilities. Positive experiences often hinge on ChatGPT's proficiency in providing detailed, specific, and comprehensive information, enabling users to seek swift and precise solutions. However, challenges arise as users encounter difficulties in formulating queries that ChatGPT readily comprehends, necessitating rephrasing. This communication friction prompts the exploration of prompt-writing support as a potential area for future research, aiming to enhance the overall user experience [12].

Trust is paramount for user satisfaction in the AI querying landscape, where inaccuracies or biases can undermine trust and overall experiences. ChatGPT faces the challenge of consistently delivering accurate and reliable information, distinct from intent-based chatbots that may convincingly present misinformation. Addressing this challenge requires extensive research to understand how users navigate inaccurate output and develop effective preventive measures [12]. Additionally, as practical attributes take precedence, the inclusion of enjoyable elements significantly contributes to positive user experiences. Users express delight when ChatGPT exceeds expectations, creating a "wow" factor, particularly during initial interactions. Designing for both practical utility and user delight ensures a holistic and engaging user experience [12].

In summary, users actively shape their interactions with AI models such as ChatGPT to fulfill practical needs, seeking immediate and effective solutions to real-world challenges. However, challenges in query formulation and the delicate balance between trust, accuracy, and delight highlight areas for improvement and further research in the ongoing evolution of AI-driven conversational interfaces.

3 User Study: User Query Formation and Result Correlation in Search Engines and AI Models

This study aims to explore user query formation, understand interactions with platforms, and examine the correlation between user queries and results in search engines and AI models.

3.1 Methodology

The methodology comprises two main components: the measurement of query formation metrics and a semi-structured interview.

3.1.1 Measurement of Query Formation Metrics

To quantify and analyze the structure and content of user generated queries, several metrics will be considered:

Query Length (QL): The length of queries will be measured to understand the users' preference for concise or detailed queries.

Complexity Index (CI): Low, Moderate or High.

- a. Low Complexity: Involves simple language without technical terms, easily understood by all.
- b. Moderate Complexity: Involves incorporating some technical terms with limited jargon while remaining broadly understandable.
- c. High Complexity: Involves extensive technical terms and complex jargon, requiring expertise for full understanding.

Relevance Score (RS): Participants will provide a numerical rating, ranging from 1 (very dissatisfied) to 5 (very satisfied), to assess their satisfaction with the relevance of search results or AI responses.

3.1.2 Semi-Structured Interview

A semi-structured interview approach will be employed to delve deeper into users' thoughts, strategies, and perceptions during the query formation process.

3.2 Data Collection and Analysis

The data collected for the study includes demographics, query lengths, relevance scores, and complexity indices, providing insights to analyze how users form query and interact with search engines and AI models.

Table 1. Demographic Information of Participants

Participant	Gender	Occupation	Education
P1	Male	Machine Learning student	Master's Degree
P2	Male	Industrial Engineering student	Master's Degree
P3	Male	Sustainable Industry Analyst	Master's Degree
P4	Female	Technical Consultant	Bachelor's Degree
P5	Male	Security and Cloud Computing student	Master's Degree

3.3 Query Length Analysis

The study measured query lengths to understand preferences for concise or detailed queries across various tasks. Table 2 displays the observed query lengths for both Google and ChatGPT tasks.

Participant	QL (Google-1)	QL (Google-2)	QL (ChatGPT-1)	QL (ChatGPT-2)
P1	5	8	31	23
P2	6	17	33	19
P3	6	20	50	38
P4	4	13	21	23
P5	9	6	20	18

Table 2. Query Length (QL)

- Participants tended to generate longer queries when interacting with ChatGPT than with Google, suggesting a preference for more detailed information or instructions in conversational settings.
- Query lengths varied among participants, with some favoring shorter queries overall, while others preferred longer queries for specific tasks.

3.4 Complexity Index Analysis

The complexity index was assessed based on the presence of technical terms or jargon in the queries. Table 3 categorizes the complexity indices for both Google and ChatGPT tasks.

Participant	CI (Google-1)	CI (Google-2)	CI (ChatGPT-1)	CI (ChatGPT-2)
P1	Low	Low	Moderate	Moderate
P2	Low	Low	Moderate	Moderate
P3	Low	Moderate	High	High
P4	Low	Moderate	Moderate	Moderate
P5	Low	Low	Moderate	Moderate

Table 3. Complexity Index (CI)

Participants generally maintained a consistent complexity level, with some opting for more complex queries to attain personalized results.

3.5 Relevance Score

The participants were instructed to rate the relevance of search results or AI responses using a Likert 5-point satisfaction scale (1 - very dissatisfied, 5 - very satisfied) as shown in Table 4.

Participant	RS (Google)	RS (ChatGPT)
P1	3	4
P2	4	3
P3	3	4
P4	4	3
P5	4	3

Table 4. Relevance Score (RS)

Majority of participants rated ChatGPT lower due to source trust concerns, despite valuing its detailed answers. ChatGPT's personalized queries simplify retrieval, but the limitation of a single answer is noted. Google offers reliable sources but requires navigating multiple links. The trade-offs users face involve source trust, result personalization, and the breadth of information paths.

4 Discussion and Implications

User interviews highlighted the strengths and limitations of search engines such as Google and AI models such as ChatGPT. Participants valued ChatGPT's word-centric query formation, which allows an exploration of concepts in a single query. The model's time efficiency, analyzing the whole part with one query, was seen as an advantage over Google's need for separate searches. Users adapted their queries in ChatGPT iteratively, showcasing a dynamic approach to information retrieval. A common strategy involved using Google for initial information gathering and turning to ChatGPT for more detailed insights. Google's proficiency in delivering quick and verified information, especially with concise queries, inspired trust. However, occasional reservations about ChatGPT's reliability surfaced, suggesting the need for improvements in accuracy.

Future development should optimize the strengths of Google and ChatGPT, addressing gaps identified by users. Enhancements for ChatGPT should focus on improving source trust by validating information and expanding response scopes. For Google, refining result presentation to

reduce the need for excessive link navigation is crucial. A user-centric approach, accommodating personalized queries in ChatGPT and exploring ways to introduce adaptability in Google, is essential. Integration of conversational features into search engines could enhance the user experience and mitigate identified gaps.

5 Conclusion

In conclusion, this research delves into user interactions with search engines and AI models, with a primary focus on query formation. Users appreciate ChatGPT's efficiency in handling detailed queries and trust Google for quick information retrieval. Notably, concerns regarding ChatGPT's reliability have surfaced. Users commonly adopt a dual-platform strategy, utilizing Google for initial information and turning to ChatGPT for in-depth insights. Future development should capitalize on the strengths of both platforms, addressing user-identified gaps. Recommendations include enhancing source trust for ChatGPT, refining Google's result presentation, and introducing adaptability features. The potential integration of conversational elements into search engines emerges as a promising avenue to enhance how users form query and get results.

References

- [1] D. E. Rose and D. Levinson, "Understanding user goals in web search," in *Proceedings of the 13th international conference on World Wide Web*, pp. 13–19, 2004. <https://doi.org/10.1145/988672.988675>.
- [2] C. Hölscher and G. Strube, "Web search behavior of internet experts and newbies," *Computer networks*, vol. 33, no. 1-6, pp. 337–346, 2000. [https://doi.org/10.1016/S1389-1286\(00\)00031-1](https://doi.org/10.1016/S1389-1286(00)00031-1).
- [3] S. Brin and L. Page, "The anatomy of a large-scale hypertextual web search engine," *Computer networks and ISDN systems*, vol. 30, no. 1-7, pp. 107–117, 1998. [https://doi.org/10.1016/S0169-7552\(98\)00110-X](https://doi.org/10.1016/S0169-7552(98)00110-X).
- [4] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "Bert: Pre-training of deep bidirectional transformers for language understanding," *arXiv preprint arXiv:1810.04805*, 2018. <https://doi.org/10.48550/arXiv.1810.04805>.
- [5] T. Brown, B. Mann, N. Ryder, M. Subbiah, J. D. Kaplan, P. Dhariwal, A. Neelakantan, P. Shyam, G. Sastry, A. Askell, *et al.*, "Language models are few-shot learners," *Advances in neural information processing systems*, vol. 33, pp. 1877–1901, 2020. <https://doi.org/10.48550/arXiv.2012.15723>.

- [6] R. Xu, Y. Feng, and H. Chen, "Chatgpt vs. google: A comparative study of search performance and user experience," *arXiv preprint arXiv:2307.01135*, 2023. <https://doi.org/10.48550/arXiv.2307.01135>.
- [7] G. Marcus, "The next decade in ai: four steps towards robust artificial intelligence," *arXiv preprint arXiv:2002.06177*, 2020. <https://doi.org/10.48550/arXiv.2002.06177>.
- [8] Y. Wu, M. Schuster, Z. Chen, Q. V. Le, M. Norouzi, W. Macherey, M. Krikun, Y. Cao, Q. Gao, K. Macherey, *et al.*, "Google's neural machine translation system: Bridging the gap between human and machine translation," *arXiv preprint arXiv:1609.08144*, 2016. <https://doi.org/10.48550/arXiv.1609.08144>.
- [9] A. Anand, V. Setty, A. Anand, *et al.*, "Context aware query rewriting for text rankers using llm," *arXiv preprint arXiv:2308.16753*, 2023. <https://doi.org/10.48550/arXiv.2308.16753>.
- [10] M. Drosou and E. Pitoura, "Search result diversification," *ACM SIGMOD Record*, vol. 39, no. 1, pp. 41–47, 2010. <https://doi.org/10.1145/1860702.1860709>.
- [11] R. W. White and D. Morris, "Investigating the querying and browsing behavior of advanced search engine users," in *Proceedings of the 30th annual international ACM SIGIR conference on Research and development in information retrieval*, pp. 255–262, 2007. <https://doi.org/10.1145/1277741.1277787>.
- [12] M. Skjuve, A. Følstad, and P. B. Brandtzaeg, "The user experience of chatgpt: Findings from a questionnaire study of early users," in *Proceedings of the 5th International Conference on Conversational User Interfaces*, pp. 1–10, 2023. <https://doi.org/10.1145/3571884.3597144>.

Pulsing Convex DDoS Attack in CDN Networks

Sonika Baniya

sonika.baniya@aalto.fi

Tutor: Jose Luis Martin Navarro

Abstract

CDN infrastructure plays a very crucial role in serving clients all over the world. This infrastructure can be exploited by deploying the edge servers to achieve DDoS attacks. While performing this attack can be considered to be having to deploy heavy servers in the edges of CDN networks, this paper has done a feasibility analysis on the pulsing convex idea of DDoS attack in CDN Networks where the adversary can achieve the DDoS from a single device with the time analysis of request and response time of the server.

KEYWORDS: CDN, DDoS, DNS

1 Introduction

A Content Delivery Network (CDN) is a network of distributed servers calculatedly placed in different geographical data centers around the globe. The purpose of CDN is to increase the speed, accessibility, and dependability of providing web content to end users. CDN attacks are a broad term for numerous security risks that target CDN infrastructure. CDN security challenges are categorized per CDN infrastructure components, discuss possible countermeasures and their effectiveness, and delineate

future research directions [1]. The material delivered by the CDN may be altered by adversaries. Injecting dangerous scripts, hacking websites, DoS, or changing content can all be examples of this.

CDN (Content Delivery Network) technology plays a crucial role in delivering content efficiently to users by distributing it across multiple servers. However, DDoS (Distributed Denial of Service) attacks can pose a significant threat to CDN infrastructure. DDoS attacks aim to overwhelm the target network with a massive volume of malicious traffic, rendering it inaccessible to legitimate users. These attacks can be implemented at different layers of the network, including the application layer, transport layer, and network layer. When a DDoS attack targets a CDN, it can disrupt the delivery of content to users, leading to service unavailability and degraded performance. The attack can overload the CDN servers, causing delays in content delivery and potentially impacting the user experience [4].

To mitigate the impact of DDoS attacks on CDNs, various defense mechanisms and detection algorithms have been developed [6]. Mainly to identify and filter out malicious traffic, allowing legitimate requests to reach the CDN servers. However, designing an effective real-time detector with low computational overhead remains a challenge. Additionally, the evaluation of new detection algorithms and techniques relies heavily on the availability of well-designed datasets [6]. There has been work going on developing comprehensive and reliable datasets to test and evaluate DDoS attack detection systems[5]. These datasets help in analyzing attack patterns, identifying new attack types, and improving defense mechanisms.

CDN-Convex Attack is a specific type of pulsing DDoS attack that takes advantage of the CDN infrastructure as a converging lens. By exploiting the various network latencies provided by the distributed CDN nodes, adversaries can schedule the arrival of requests in a way that they converge as a series of intermittent pulses at the victim, causing severe performance degradation and availability issues.

2 CDN-Convex Attack Strategy

To perform the CDN-Convex attack, the adversary needs to overcome several technical challenges. One crucial challenge is to craft and synchronize a maximum number of requests to arrive at the origin server

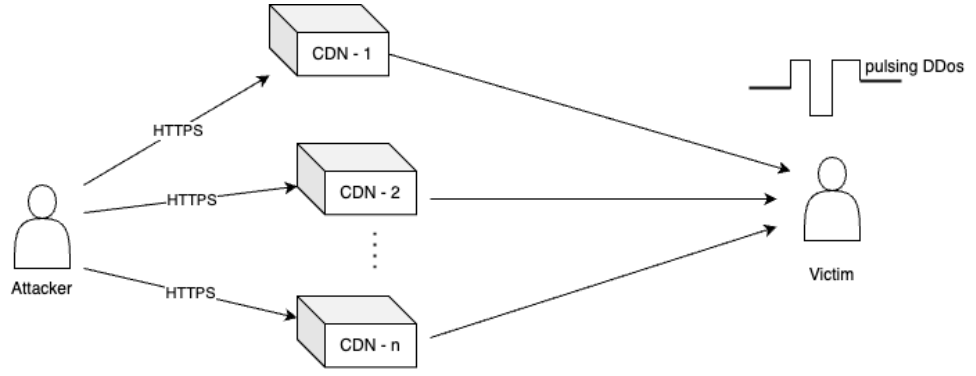


Figure 1. Sample example of Convex DDoS attack on CDN

simultaneously, achieving a high bandwidth concentration at the victim. The attack strategy involves four techniques: the basic CDN-Convex attack, CDN-Cascading Convex attack, DNS-hold on Convex attack, and Request-pending Convex attack. These techniques aim to extend the attack path, manipulate DNS resolution, leverage incomplete HTTP requests, and exploit IP fragmentation to increase the number of attacking requests and enhance the temporal convergence of the attack.

2.1 Basic CDN-Convex attack

As described in above introduction section the basic CDN-convex attack is a way where the destination server is overwhelmed by the multiple requests an adversary server is sending. Here in the analysis section, we will experiment with a time delay response in the destination server and see how it can conclude to DoS attack.

2.2 Environment of Attack

In this environment, we are considering $N_{adversary}$ to be the adversary node, $N_{destination}$ be the destination node. Let $m_1, m_2, m_3 \dots m_n$ be n requests that the adversary sends to the destination node.

So, we can achieve DoS if the destination node receives all n messages at the same time which means:

$$request(m_1) \tag{1}$$

$$sleep(t_1) \text{ seconds} \tag{2}$$

$$request(m_2) \tag{3}$$

$$sleep(t_2) \text{ seconds} \tag{4}$$

$$\dots \tag{5}$$

$$request(m_n) \tag{6}$$

$$sleep(t_n) \text{ seconds} \tag{7}$$

From above, we can see that analysis of $t_1, t_2, t_3 \dots t_n$ helps adversary to achieve Denial of Service attack.

2.3 Time Study to perform attack

By studying the server request and response time, the adversary can perform the attack. If we have n CDN multiplex that responds to the victim, the adversary can perform a DDoS attack by adjusting the correct sleep time. For example, we have three CDN networks that serve the destination node. CDN-1 node has a request time of 3 seconds and a response time of 10 seconds. Similarly, CDN-2 has 2 seconds and 3 seconds respectively. Also, CDN-3 has 4 seconds and 1 second. Now the above environment of attack becomes:

$$request(m_1) \tag{8}$$

$$sleep 0 \text{ second} \tag{9}$$

$$request(m_2) \tag{10}$$

$$sleep 4 \text{ seconds} \tag{11}$$

$$request(m_3) \tag{12}$$

$$sleep 5 \text{ seconds} \tag{13}$$

3 Time Analysis Experiment to achieve the attack

This analysis neglects the multiple buffer time it requires for the request to do intra-network travel and only considers the round trip time it requires for the request to travel to the CDN server and to come back. The experiment doesn't perform the DDoS attack but only reflects upon the time study that the adversary have to do to achieve it. It also assumes that the response trip and request time will be equal hence calculating the request/response time as:

$$(time_{request}) = (time_{response}) = (RoundTripTime)/2 \tag{14}$$

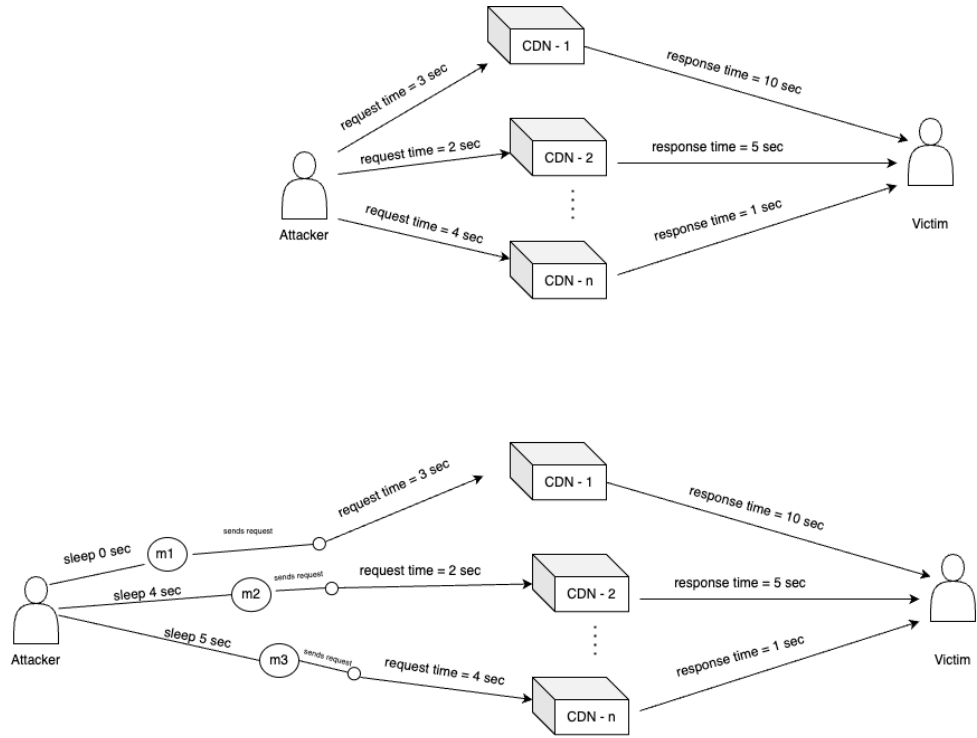


Figure 2. Time study to perform the DoS attack

IP address	Server Location	Round Trip Time	Request/Response Time
172.217.21.174	Stockholm	24.210	12.105
172.217.25.175	Osaka	306.962	153.481
172.217.27.175	Delhi	477.061	238.530
172.217.30.175	São Paulo	263.641	131.821
172.217.40.175	Las Vegas	167.102	83.551

Table 1. Time analysis in 5 geolocated CDN server of google

Table 1 presents the time analysis of above explained experiment. The setup of the experiment is provided as:

- i) Finding the destination node.
- ii) Locating the five CDN node multiplex
- iii) Time measurement.

This experiment is performed from Espoo, Uusima, Finland in 18th Nov 2023 and hence holds the data of the stated date and time.

4 Conclusion

This paper has done a feasibility analysis on the study of Convex attacks on CDN networks. Through above above-defined strategy we can conclude that the Convex idea of the attack is feasible and DDoS attack is now simpler than big infrastructures in edge servers.

5 Reference

- [1] M. Ghaznavi, E. Jalalpour, M. A. Salahuddin, R. Boutaba, D. Migault and S. Preda, "Content Delivery Network Security: A Survey," in *IEEE Communications Surveys Tutorials*, vol. 23, no. 4, pp. 2166-2190, Fourthquarter 2021, doi: 10.1109/COMST.2021.3093492.
- [2] Guo, Run, Jianjun Chen, Yihang Wang, Keran Mu, Baojun Liu, Xiang Li, Chao Zhang, Haixin Duan, and Jianping Wu. "Temporal CDN-Convex Lens: A CDN-Assisted Practical Pulsing DDoS Attack." In **32nd USENIX Security Symposium (USENIX Security 23)**, pp. 6185-6202
- [3] Li, Zihao, and Weizhi Meng. "Mind the amplification: cracking content delivery networks via DDoS attacks." In **Wireless Algorithms, Systems, and Applications: 16th International Conference, WASA 2021, Nanjing, China, June 25–27, 2021, Proceedings, Part II 16**, pp. 186-197. Springer International Publishing, 2021
- [4] Sharafaldin, Iman, Arash Habibi Lashkari, Saqib Hakak, and Ali A. Ghorbani. "Developing realistic distributed denial of service (DDoS) attack dataset and taxonomy." In **2019 International Carnahan Conference on Security Technology (ICCST)**, pp. 1-8. IEEE, 2019.
- [5] Q1'23 DDoS Threat Landscape Report, CloudFlare
- [6] How Modern Security Teams Fight Today's Cyber Threats, eBook Cisco Public© 2022

OT vs IT Cybersecurity Management

Venla Kuosa

venla.kuosa@aalto.fi

Tutor: Mikko Kiviharju

Abstract

Operational technology (OT) has unique characteristics in its cybersecurity management due to the interaction and effects OT can have on the physical world. This paper explores the differences in traditional information technology (IT) cybersecurity management to that of OT. Specifically, the cybersecurity aspects are reviewed by comparing CIA (confidentiality, integrity, availability) paradigm in IT to the corresponding paradigm, CAIC (control, availability, integrity, confidentiality), in OT, reflecting on the NIST Cybersecurity Framework and a supporting document for implementing the framework in an OT context, the NIST Guide to Operational Technology (OT) Security.

The main differences in OT and IT cybersecurity management relate to the control and availability principles. OT organizations need a heavier emphasis on risk management and protective measures against cyber attacks as their impact on the safety of physical devices and systems can be damaging. This combined with unexpected downtime in OT systems may have significant consequences to safety, society, the economy and the environment.

KEYWORDS: *operational technology, information technology, cybersecurity, safety, security*

1 Introduction

Cybersecurity has gained popularity as a topic as digitalization continues to expand. The discourse on cybersecurity has primarily revolved around traditional information technology (IT), focusing heavily on data breaches, ransomware and privacy issues. However, the threat profile of many critical systems differs from the topics in general discussion. Specifically, operational technology (OT) has dimensions often overlooked in discussions of securing critical infrastructure.

Operational technology refers to systems and devices which have an effect on the physical world [1]. These systems and devices can be found in most critical infrastructure sectors. Examples of OT systems include power grid supervisory control and data acquisition (SCADA) systems in the energy sector, safety systems in chemical manufacturing, and physical access control systems (PACS) in different facilities.

This paper reviews two documents by the National Institute of Standards and Technology (NIST) in the context of OT cybersecurity and outlines the unique characteristics of OT systems which should be regarded in an organization's cybersecurity management when OT is present.

This paper is organized as follows. Section 2 presents the latest version of the NIST Cybersecurity Framework (CSF). Section 3 describes the NIST Guide to Operational Technology (OT) Security and how the general-purpose Cybersecurity Framework can be applied in an OT context. Section 4 discusses how the CIA (confidentiality, integrity, availability) paradigm in IT compares to the corresponding paradigm, CAIC (control, availability, integrity, confidentiality), in OT, and how the differences can be observed in cybersecurity management. Finally, Section 5 concludes the observations.

2 NIST Cybersecurity Framework

This section briefly covers the history of the NIST Cybersecurity Framework and summarizes principles of the latest draft of the framework.

2.1 History

In 2013, Barack Obama, the president of the United States, issued Executive Order 13636, which assigned NIST to compose a framework for

improving the cybersecurity of critical services [2]. The initial draft was published in 2013, and the first version of the NIST Cybersecurity Framework was released a year later [3]. In 2018, the framework was updated to version 1.1, aiming to enhance user-friendliness and usability of the framework, and incorporating guidance on authentication and vulnerability disclosure [4].

The draft for Cybersecurity Framework 2.0 was published for public comments in 2022 [5]. The scope of the framework has been broadened to cover all organizations instead of targeting only organizations in critical sectors. Although the target scope has been expanded since the initial version, the framework has been presented as a model that should be modified and applied based on each organization's unique needs throughout its history; it was not intended to be a "one framework fits all". Thus, NIST has published several supporting documents for using the framework in different contexts [1] [6].

2.2 Content

The NIST Cybersecurity Framework 2.0 [5] is intended for all organizations for creating or improving the cybersecurity management processes in the organization. The framework presents three main tools: *core*, which consists of the main functions in cybersecurity management, *profiles* for identifying current and target states for the maturity of cybersecurity management, and *tiers* to help organizations understand the required maturity of cybersecurity management their type of organization might need.

The framework core outlines six functions in cybersecurity management. This includes IDENTIFY, PROTECT, DETECT, RESPOND and RECOVER, as well as a new addition since the previous version, GOVERN. GOVERN binds together the rest of the functions, describes their outcomes and defines actions needed to achieve them. The GOVERN function keeps evolving as the organization and its risk profile are changing. The GOVERN function is a part of the organization's risk management strategy, covering cybersecurity strategy and policies.

The rest of the functions give instructions for handling cyber threats that the organization may face. While some functions are defined for the purpose of acting on an actualized threat, others are continuously ongoing processes within the organization.

The IDENTIFY function stands for the phase of determining the orga-

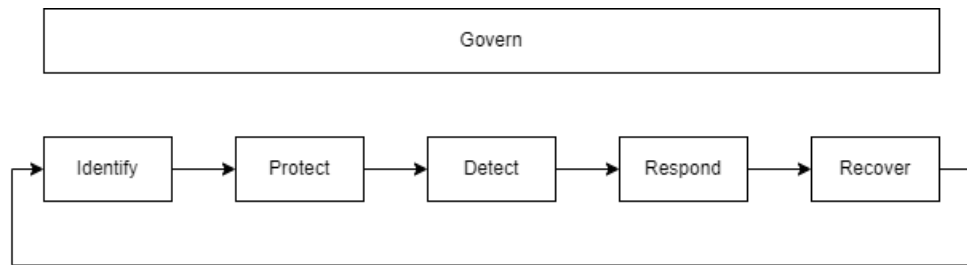


Figure 1. The NIST Cybersecurity Framework functions [5]

nization’s assets and possible risks related to them. This function also includes conducting risk assessments in cybersecurity aligned with the risk management strategy in the organization, while updated on new vulnerability discoveries as they arise.

PROTECT refers to the actions taken to prevent or decrease the risks outlined in the IDENTIFY function. This might include identity and access management, firewalls, network segmentation, patching or limiting physical access. Maintaining and improving these security measures is a continuous operation. Protective measures are often taken immediately after making new discoveries stated in IDENTIFY.

Protecting the systems will not be able to stop all cyber attacks. Thus, the next function, DETECT, aims to find all cyber attacks against the organization as soon as they appear. Detecting the attacks may involve monitoring of networks, physical premises and logs from data, software and other runtime operations. When an attack is detected, the organization starts an incident management process, which is described by the RESPOND function. The process includes inviting relevant people to figure out how to stop the attack and how to minimize its impact, as well as communication and analysis about the attack.

The final function, RECOVER, outlines the process of cleaning up after an attack that is successfully stopped and organizing ways to ensure business continuity, reducing the impact of the attack. Communication about the attack and its impact also continues in the recovery phase.

Each function can be divided into categories and their subcategories, which contain more detailed descriptions of specific cybersecurity operations. These elements are useful for determining profiles: simply put, a profile is a set of elements from each function. There are two types of profiles: current profile, which helps organizations gain an understanding of the current state of cybersecurity management, and target profile, which describes the target level of maturity in cybersecurity management

and the actions needed to achieve this level. NIST has developed several example profiles, which organizations can use as their target profiles, including profiles for smart grids [7], hybrid satellite networks [8] and manufacturing environments [9].

The level of maturity in cybersecurity can be further evaluated using the framework tiers. The framework [5] provides descriptions for different levels of cybersecurity management, from Tier 1 to Tier 4. Not every organization needs to aim for the highest tier – this depends on the risk management capabilities and decisions in the organization. Some organizations are able to take larger risks than others, and thus, cybersecurity operations do not need to be as strict in every organization. Tiers used in combination with the organization's current and target profiles may act as an indicator for which tier the organization should aim for.

3 NIST Guide to Operational Technology (OT) Security

The need for an OT specific document for cybersecurity management has been acknowledged by NIST, and the NIST Guide to Operational Technology (OT) Security was published September 2023 [1]. The guide is dedicated specifically to OT organizations, assisting them in establishing or enhancing their cybersecurity management. The preceding revision of the guide was targeted at industrial control systems [10], which is a subcategory of OT. The broadened scope led to major updates for most topics in the document, including OT-related risk management, threats and vulnerabilities, as well as security capabilities and tools for OT [1].

The updated document describes the architecture of OT security and offers a comprehensive guide for developing an OT cybersecurity program, including a description of its business benefits. The renewed risk management section is based on other NIST frameworks and guides for risk management, considering the OT context and giving OT-specific advice for managing security risk. From the perspective of this paper, it is most interesting that the new revision builds upon the existing Cybersecurity Framework, adding details and highlighting differences in traditional IT and OT systems.

As mentioned in the previous section, the GOVERN function was introduced in version 2.0 of the Cybersecurity Framework. Although the guide is based on CSF 1.1 instead of CSF 2.0, it does not lack support for the newly introduced GOVERN function: in earlier versions, most operations

related to GOVERN were categorized under the IDENTIFY function.

For the GOVERN function, the guide specifies that the OT context requires cybersecurity training and understanding of a broader area among all personnel: IT administrators and programmers in OT organizations are required to take into account the OT specific characteristics, and conversely, OT engineers and operators must comprehend IT cybersecurity principles. Moreover, the guide heavily emphasizes cybersecurity risk management, which is a part of the GOVERN function in CSF 2.0 [5]. The NIST guide to OT security [1] points out that in OT risk management, it is important to consider the organization's role in society affecting its risk tolerance and consider the organization's sector in the risk analysis. Supply chain risk management also falls under the GOVERN function. Hardware components must be purchased from an authorized source, and the guide recommends organizations to review the reliability of manufacturers and authorized third-party distributors.

Within OT organizations, asset management is one of the most important categories in the IDENTIFY function. This means having unique identifiers for each asset, and maintaining information about hardware, firmware, software, personnel and facilities. This becomes even more crucial for large or widely distributed systems, as it greatly improves vulnerability management by ensuring easy access to all necessary information.

In the context of the PROTECT function, NIST emphasizes that managing physical access to OT systems is important. OT facilities are encouraged to implement layered security measures in order to enhance the robustness of access controls and mitigating unauthorized entry. In addition, physical access control systems should log visits to the facility, and alarm systems should be used to monitor the premises. Access management may be challenging in OT systems due to the nature of OT, as access controls may be distributed across multiple systems, making it more prone to error when revoking or updating user's access.

In addition to physical access control, ensuring the security of components and tools in facilities is important. NIST suggests that wiring and cabling should be shielded or embedded in floors or walls. This helps avoid intended or unintended tampering of wires and cables. A lesson learned from Stuxnet, arguably the most famous cyber attack against an OT system, can also be found from the guide: all portable devices, such as laptops or USB drives, should be scanned for malware before attaching them to an OT system.

Overall, the guide emphasizes protective operations in OT systems. Other OT specific guidance concerns network configurations and communications, user authentication, configuration changes and maintenance, as well as backup procedures. The comprehensive advice related specifically to the PROTECT function may be due to the risk profile of OT organizations; protecting OT systems requires a large amount of resources and maintenance because it is highly important to avoid successful cyber attacks against the systems.

When moving on to detecting cybersecurity events, it is important to monitor the normal state in order to be able to distinguish anomalies from the data flow. Monitoring is very similar to that of IT organizations, however, NIST recommends OT organizations to consider alert thresholds in regard to their physical location – if a facility does not have personnel all the time, the alert threshold could be lowered to compensate for the time it takes to travel to the facility for a check-up.

For the RESPOND function, the guide gives some advice for the incident management process in OT organizations. OT organizations should choose and train personnel who are involved in managing certain types of incidents. This includes both internal personnel and external partners, such as vendors, distributors and possible customers. For incident management, the organization should have a crisis communication strategy in place for informing all necessary stakeholders about the situation on different levels and inviting necessary persons to participate in the incident management process, as well as communicate the incident to end customers in case of an outage or other issues in the services provided by the organization. In addition, NIST reminds that incident management for remote, non-staffed locations is challenging – good monitoring and remote control capabilities help to minimize the impact of the incident.

Recovering from an incident in OT organizations requires more people on-site, whereas in traditional IT organizations recovery actions can often be carried out remotely. This means that recovering from incidents may take more time in OT systems, especially if components or facilities are damaged in a way that requires a large maintenance operation involving the physical replacement of infrastructure and equipment, necessitating for example the ordering of spare parts.

4 OT specific security considerations

The IT industry has a widely known paradigm called CIA (confidentiality, integrity, availability), that describes three key elements, which together function as the foundation of security; confidentiality stands for sufficient protection against leaking information, integrity represents reliability of data in terms of tamper-proofness and storage solutions, and availability means that services are operational and accessible when needed.

Although exceptions exist, physical safety is rarely a major factor in traditional IT systems development. Consequently, it does not possess a significant role in cybersecurity management. However, the issue of physical safety is crucial when it comes to OT: a cyber attack against OT systems may result in significant financial losses, arising from damaged equipment or lost production, while concurrently posing a substantial threat to the environment and presenting the potential for severe harm to both OT personnel or the larger public through injuries or fatalities [1]. Depending on the system attacked, these effects may occur very rapidly.

As a result, priorities for security are different in OT, and thus, the OT industry has its own corresponding paradigm, CAIC (control, availability, integrity, confidentiality) [11]. The CAIC paradigm adds *control* as the highest priority and reverses the priority order for the principles in the CIA paradigm. The reason why control holds a higher priority than the other factors is the nature of OT and its safety perspective; all physical operations must remain under control to avoid unexpected events leading to possible emergencies. Thus, the physical functions require continuous monitoring and control and maintaining their operation under any conditions is a preeminent concern.

The physical aspect and the priority of control in OT systems affects the cybersecurity management in OT organizations. It can be noticed from the NIST Guide to Operational Technology (OT) Security [1] that the OT system's vulnerability to cyber attacks is very much dependent on the inherent physical security features and safety design of OT installations. On the other hand, OT systems may be geographically widely distributed so that physically reaching a part of the system for maintenance and repair may take a significant amount of time. This also links to the complexity and the varied composition of these systems. It must be remembered that the damage caused by attacks may take much longer to repair leading to unacceptable downtimes unless redundancy is ensured.

Sometimes, conflict between security controls and physical safety measures may arise. OT installations and infrastructures should be equipped with an array of safety-related systems that are designed to automatically ensure safety in case of failures and malfunctions. These provide safety from the most radical risks associated with cyber attacks on OT but may themselves also pose a potential attack vector for attackers that are able to penetrate and operate IT systems in the OT installation. It is thus quite relevant that the NIST guide to OT security [1] points out a very specific factor which requires considering the safety aspect in contrast to security: in an emergency situation, direct access to a human-machine interface (HMI) is essential. In this scenario, the HMI is used to force stop a system or a device which is malfunctioning or otherwise has a role in safety threat situation. The organizations must consider how they can authorize access to HMIs so that operating them in an emergency situation does not take too much time but are not too easily accessible so that they can be used for malicious intent.

One aspect where IT and OT systems differ is the prevalence and age of legacy systems. As an example, industrial installations, characterized by lifespans of several decades, often encounter increased complexity in ensuring compatibility and up-to-dateness of systems. Thus, latent problems and vulnerabilities become more challenging to resolve. A resolute attacker with intricate technical knowledge of the systems in question may be able to find innovative avenues of attack. On the other hand, depending on the type of system or installation, this may be mitigated by confidentiality and security procedures designed to limit the availability of critical information [12]. As the cost involved in replacing components outside their designed lifecycle may be prohibitive, additional protective measures and strategies might be needed to reach the desired level of security. This is why NIST emphasizes the importance of actions, such as network segmentation and asset management, for OT organizations [1], although these can be considered standard procedures in IT organizations as well.

Finally, within OT organizations, there is a heightened need to ensure that the personnel are aware and suitably proficient to execute the measures defined using the NIST Cybersecurity Framework [5]: depending on the type of organization, the knowledge of cybersecurity issues may be limited or obsolete and dismissive attitudes to cybersecurity practices may still persist. There may even be a false sense of security regarding

systems that have traditionally been considered highly safe, leading to organizational inertia in implementing cybersecurity measures. The importance continuous of education, training and relevant exercises to assess the impact of protective measures must be stressed. As traditional IT is often evolving much more rapidly than OT, this demands both a cultural shift and the allocation of enough resources to ensure security.

5 Conclusion

The differences in OT and IT cybersecurity management are heavily linked to the CIA and CAIC paradigms, the element of control creating the greatest difference between OT and traditional IT cybersecurity. Physical features being the defining characteristics of OT systems, the NIST Guide to Operational Technology (OT) Security [1] emphasizes risk management and risk tolerance regarding physical safety and effects of disruption against physical operations. Although availability is a common factor between IT and OT security, it has a larger emphasis in the CAIC paradigm, as unexpected downtime in OT systems may result in financial, physical, societal and environmental damage. To ensure availability, the NIST guide highlights protective measures in order to minimize the probabilities of successful cyber attacks. Confidentiality and integrity are more stressed in the CIA paradigm, however, their less prominent roles in the CAIC paradigm do not mean they are considered unimportant, but instead, the greatest risks in OT are likely to be found elsewhere.

The new NIST Guide to Operational Technology Security is a good starting point for OT organizations but only the practice of implementing it in a wide variety of OT organizations will show how efficient it is as a tool for improving security and safety. What is important, is however the fact that OT security is finally being considered in a more systematic manner in our ever-digitalizing world.

References

- [1] K. Stouffer, M. Pease, C. Tang, T. Zimmerman, V. Pillitteri, S. Lightman, A. Hahn, S. Saravia, and M. Thompson, “Guide to operational technology (OT) security,” NIST Special Publication (SP) 800-82, Rev. 3, Includes updates as of September 28, 2023, National Institute of Standards and Technology, Gaithersburg, MD, 2023. 10.6028/NIST.SP.800-82r3.
- [2] Executive Office of the President [Barack Obama], “Executive Order 13636 –Improving Critical Infrastructure Cybersecurity,” , United States, February 2013.
- [3] National Institute of Standards and Technology, “Updating the NIST Cybersecurity Framework – Journey To CSF 2.0,” May 2022. Available at: <https://www.nist.gov/cyberframework/updating-nist-cybersecurity-framework-journey-csf-20>. Accessed 15.11.2023.
- [4] National Institute of Standards and Technology, “Framework for improving critical infrastructure cybersecurity,” , National Institute of Standards and Technology, Gaithersburg, MD, April 2018. 10.6028/NIST.CSWP.6.
- [5] National Institute of Standards and Technology, “The NIST Cybersecurity Framework 2.0,” , National Institute of Standards and Technology, Gaithersburg, MD, August 2023. 10.6028/NIST.CSWP.29.ipd.
- [6] K. Stine, S. Quinn, G. Witte, and R. K. Gardner, “Integrating Cybersecurity and Enterprise Risk Management (ERM),” , National Institute of Standards and Technology, Gaithersburg, MD, October 2020. 10.6028/NIST.IR.8286.
- [7] J. Marron, A. Gopstein, N. Bartol, and V. Feldman, “Cybersecurity Framework Smart Grid Profile,” , National Institute of Standards and Technology, Gaithersburg, MD, July 2019. 10.6028/NIST.TN.2051.
- [8] J. McCarthy, D. Mamula, J. Brule, K. Meldorf, R. Jennings, J. Wiltberger, C. Thorpe, J. Dombrowski, and O. Lattin, “Cybersecurity Framework Profile for Hybrid Satellite Networks (HSN),” , National Institute of Standards and Technology, Gaithersburg, MD, June 2023. 10.6028/NIST.IR.8441.ipd.
- [9] K. Stouffer, T. Zimmerman, C. Tang, J. Lubell, J. Cichonski, and McCarthy, “Cybersecurity Framework Manufacturing Profile,” , National Institute of Standards and Technology, Gaithersburg, MD, September 2017. 10.6028/NIST.IR.8183.
- [10] K. Stouffer, V. Pillitteri, S. Lightman, M. Abrams, and A. Hahn, “Guide to Industrial Control Systems (ICS) Security,” , National Institute of Standards and Technology, Gaithersburg, MD, May 2015. 10.6028/NIST.SP.800-82r2.
- [11] S. Bhattacharjee, *Practical Industrial Internet of Things Security*. Packt Publishing, 2018.
- [12] S. Horsmanheimo, H. Kokkonieni-Tarkkanen, P. Kuusela, L. Tuomimäki, S. Puuska, and J. Vankka, “Situational awareness of critical infrastructure,” *Publications of the Government’s analysis, assessment and research activities*, no. 19, p. 56, 2017.

Amortized Meta Bayesian Optimization

Xinyu Zhang

xinyu.2.zhang@aalto.fi

Tutor: Daolang Huang

Abstract

Bayesian optimization (BO) has emerged as a popular method for optimizing high-cost, black-box objective functions. In recent year, several meta-BO architectures have been proposed, where neural networks learn surrogate models and acquisition functions across a set of related tasks. This article presents a comprehensive overview of meta-BO, delving into its recent advancements and summarising potential research directions within this evolving domain.

KEYWORDS: Bayesian Optimization, Meta-learning, Meta-Bayesian Optimization

1 Introduction

The global optimization of black-box functions is a ubiquitous challenges across scientific and industrial domains, encompassing tasks like hyperparameter tuning for machine learning models, drug design, and A/B testing. Bayesian Optimization (BO) [1] has emerged as a sample-efficient technique for optimizing such black-box objectives whose direct evaluations are costly. When learning a task, BO proceeds in two stages: first, fitting a probabilistic *surrogate model* of the objective function, then de-

ciding the next sampling point with an *acquisition function* (AF) based on the probabilistic predictions.

While effective, traditional BO tackles new tasks in isolation and thus often suffers slow convergence due to limited task-specific trials. To address this, researchers have infused BO with the essence of meta-learning, most commonly understood as *learning to learn*, which expedites new task modelling by leveraging training episodes across related source tasks and borrowing knowledge. By carefully crafting source tasks and training objectives, meta-BO also shows flexibility to address specific tasks where conventional choice of surrogate models and AFs may fall short.

This paper aims to elucidate the the principle of meta-BO and highlight recent application advancements. In addition, it discusses on the key components of the recent methods, offering insights into current progress and potential of this evolving field.

The subsequent sections are organized as follows: In Section 2, we first introduce the principle and high-level problem formulation of related concepts, including Bayesian Optimization and meta-learning paradigm. Then we survey recent application developments in Section 3. Finally, we investigate and discuss on promising directions for building new pipelines, together with a conclusion in the end.

2 Background

Given a black-box objective function $f : \mathcal{D} \rightarrow \mathbb{R}$, where $\mathcal{D} \subset \mathbb{R}^d$ denotes the d -dimensional input domain, the objective is to find the global optimum $x^* \in \operatorname{argmax}_{x \in \mathcal{D}} f(x)$ as rapidly as possible [2]. The information available about f is limited to the input x_i and the observations $y_i = f(x_i) + \epsilon$ at each optimization step $t \in \{1, 2, \dots, T\}$, forming the optimization history $\mathcal{H}_t \equiv \{x_i, y_i\}_{i=1}^{t-1}$. Here, $\epsilon \sim \mathcal{N}(0, \sigma^2)$ represents independently and identically distributed Gaussian noise, introducing randomness to the observations.

2.1 Bayesian Optimization

The Bayesian Optimization framework consists of two key components. The first is a probabilistic surrogate model that encapsulates our prior belief about f and updates the posterior based on the optimization history \mathcal{H}_t up to the current step t . Utilizing the prediction of surrogate model, an acquisition function $\alpha_t(\cdot | \mathcal{H}_t) : \mathcal{D} \rightarrow \mathbb{R}$ is employed to output values for

each point in \mathcal{D} , guiding the next iterate via $\operatorname{argmax}_{x \in \mathcal{D}} \alpha_t(\cdot | \mathcal{H}_t)$.

Surrogate Model

Gaussian Processes (GPs) [3] are commonly chosen as the surrogate model. GPs assume a multivariate normal distribution for the collection of x at any step t , leading to a cross-formed Gaussian posterior on f with mean $\mathbb{E}(f(x) | \mathcal{H}_t)$ and variance $\mathbb{V}(f(x) | \mathcal{H}_t)$. While Effective in training, GPs naturally suffer from cubic scaling inference cost. Even with the best-of-the-art approximation methods, the inference cost scales quadratically [4].

To address this computational challenge, Neural Processes (NPs) [5] have been proposed as one of the alternatives to GPs. NPs strike a balance between training and inference costs by combining neural networks and stochastic processes.

Acquisition Function

The acquisition function guides where to evaluate in the next iterate based on posterior provided by the surrogate model. Conditioned on posterior up to optimization step n , the acquisition function α_n produces a scalar value for any point in the searching space \mathcal{D} , so that the next query point x_{n+1} can be decided:

$$x_{n+1} = \operatorname{argmax}_{x \in \mathcal{D}} \alpha_n(x)$$

The main challenge for AFs is to balance exploration (sample from uncertain regions) and exploitation (sample in regions with potentially high objective values). There is a set of commonly employed acquisition functions, including Expectation Improvement (EI) [6], Thompson Sampling [7] and Gaussian Process Upper Confidence Bound (GP-UCB) [8].

2.2 Meta-Learning

In contrast to conventional machine learning paradigms, which primarily focus on refining model predictions with multiple data instances, meta-learning aims to enhance a learning algorithm through multiple learning episodes from a set of source tasks [9]. This can be seen as learning an inductive bias that effectively constraints the hypothesis space of model parameters θ for learning efficiency while avoiding overfitting.

Tasks within the meta-learning framework can be defined by a combination of a loss function and a dataset, denoted as $\mathcal{T} = (\mathcal{D}, \mathcal{L})$. The

learning to learn process is then formulated as

$$\min_{\omega} \mathbb{E}_{\mathcal{T} \sim p(\mathcal{T})} \mathcal{L}(\mathcal{D}; \omega) \quad (1)$$

Here, *across-task* knowledge ω is optimized to improve the model’s performance on a given dataset \mathcal{D} as measured by the loss function \mathcal{L} .

There is a wide range of choice for ω , such as parameter initialization [10], optimizer choice [11], and feed-forward model [12] which directly produces the parameters required by target tasks. Notably, the feed-forward model is often characterized as *amortized*, signifying that the training costs for unseen tasks are pre-paid during the meta-learning phase.

Hierarchical Optimization View

As suggested by Hospedales et al. [9], meta-training step can be interpreted as a hierarchical optimization problem, characterized by an outer and inner optimization loop formulated as follows:

$$\omega^* = \operatorname{argmin}_{\omega} \sum_{i=1}^M \mathcal{L}^{meta}(\theta^{*(i)}(\omega), \omega, \mathcal{D}_{source}^{val(i)}) \quad (2)$$

$$\text{s.t. } \theta^{x(i)}(\omega) = \operatorname{argmin}_{\theta} \mathcal{L}^{task}(\theta, \omega, \mathcal{D}_{source}^{train(i)}) \quad (3)$$

where L^{meta} and \mathcal{L}^{task} refers to the outer and inner loop objective respectively. The meta-training objective \mathcal{L}^{meta} is crucial for pointing out what to learn from the multiple training episodes.

3 Examples

In this section, we present two recently proposed amortized meta-Bayesian Optimization methods, serving as illustrative examples.

3.1 Path-wise Smooth Bayesian Optimization via Meta-learning

In certain scenarios, the expense associated with measurements extends beyond the measurement process itself to the cost of movement, which tends to escalate with the distance between successive measurements. In this case, the main optimization goal turns into minimizing cumulative moving cost

$$\sum_{t=0}^{T-1} \mathcal{C}(\mathbf{x}_{t-1}, \mathbf{x}_t),$$

where $\mathcal{C} : \mathcal{D} \times \mathcal{D} \rightarrow \mathbb{R}$ is the cost function for each movement from location x_{t-1} to x_t .

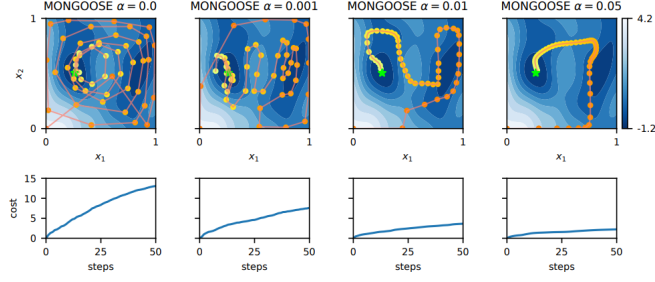


Figure 1. Trajectories generated by MONGOOSE and corresponding moving cost[17]

Essentially, the challenge is to traverse *smoothly* in small steps towards the *global* optima, which can be intuitively addressed by utilizing non-myopic AFs that look multi-step ahead. However, calculating these AFs that require nested optimizations can be computationally expensive as posited by González et al. [13]. This assertion has been substantiated by [14][15][16].

Recently, Yang et al. [17] have proposed a novel algorithm MONGOOSE that encourages smooth optimization trajectories, as shown in Figure 1. Sidestepping the need of non-myopic AFs, MONGOOSE learns a non-myopic meta-learned policy by leveraging memory-based optimization.

At the high level, MONGOOSE follows the principles of memory-based optimization elucidated by Chen et al. [11]. The paper proposes a Recurrent Neural Network (RNN)-based optimizer meta-trained on synthetic differentiable functions f generated by GPs. With the memory of previous queries and evaluations, the optimizer is allowed to capture the distribution of given functions $p(f)$ and generate new candidates by unrolling. The meta-training objectives can be described with expected observed improvements on functions, i.e.

$$\mathcal{L}_{memory}^{meta}(\theta) = \mathbb{E}_f[f(\mathbf{x}_1) - \min_{t=1, \dots, T} f(\mathbf{x}_t)] \quad (4)$$

MONGOOSE distinguishes itself through incorporating this memory-based objective with acquisition function Expected Improvement per unit (EIpu) that encourages smooth trajectories, i.e.

$$\mathcal{L}^{meta}(\theta) = \frac{\mathcal{L}_{memory}^{meta}(\theta)}{1 + \alpha \sum_1^{T-1} c(\mathbf{x}_t, \mathbf{x}_{t+1})} \quad (5)$$

Furthermore, this distinctive objective is accompanied by adding a quadratic bowl to the GPs prior to generate objective functions better aligned with real-world scenarios.

3.2 End-to-End Meta-BO Framework

Most meta-BO approaches follow the traditional two-step strategy, where the surrogate models and acquisition functions are separately implemented, leading to a potential overlook of benefits from the entire pipeline. Recently, Maraval et al.[18] have proposed an end-to-end differentiable meta-BO framework which jointly learns the surrogate and acquisition. As illustrated by the Equation (6), the model directly produces acquisition values for $x^{(\text{pred})}$ at locations where we would like to evaluate conditioned on optimization history \mathcal{H}_{obs} :

$$f_{\theta}(x^{\text{pred}}, \mathcal{H}_{obs}) = \alpha(x^{\text{pred}}) \quad (6)$$

Here f adopts a transformer-based NPs architecture.

Due to the lack of annotated acquisition data, the model is trained with reinforcement learning (RL), seeking a policy π_{θ} which performs well across K source tasks. Additionally, this paper observes a logarithmic sparsity pattern in trajectory lengths and augments the RL objective with an auxiliary objective under supervision.

4 Discussion and Conclusion

4.1 Discussion

From the examples presented above, we can see some promising directions of using meta-Bayesian optimization, including transfer learning, customizing for specific task scenarios, integration with RL, etc.

4.2 Conclusion

Bayesian Optimization has been a popular framework for solving black-box objective optimization for years, along with researchers' continuous efforts to tackle its limitations such as slow convergence and high computational complexity. Meta-learning has been recognized as a promising solution to benefit new BO task learning and amortize training costs across source tasks. In this article, we have introduced the related concepts and formulations of amortized meta-Bayesian Optimization, as well as highlight influential approaches or applications, including a efficient path-wise smooth BO algorithm and the first end-to-end meta-BO framework. We hope this survey will help beginners acquaintance themselves

with the principles and advancements of this emerging field, as well as reveal potential directions for future research.

References

- [1] B. Shahriari, K. Swersky, Z. Wang, R. P. Adams, and N. de Freitas, "Taking the human out of the loop: A review of bayesian optimization," *Proceedings of the IEEE*, vol. 104, pp. 148–175, 2016.
- [2] Y. Li, Y. Shen, W. Zhang, Y. Chen, H. Jiang, M. Liu, J. Jiang, J. Gao, W. Wu, Z. Yang, C. Zhang, and B. Cui, "Openbox: A generalized black-box optimization service," in *Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, KDD '21, ACM, Aug. 2021.
- [3] C. E. Rasmussen and C. K. I. Williams, *Gaussian processes for machine learning*. Adaptive computation and machine learning, MIT Press, 2006.
- [4] J. Quiñonero-Candela and C. E. Rasmussen, "A unifying view of sparse approximate gaussian process regression," *Journal of Machine Learning Research*, vol. 6, no. 65, pp. 1939–1959, 2005.
- [5] M. Garnelo, J. Schwarz, D. Rosenbaum, F. Viola, D. J. Rezende, S. M. A. Eslami, and Y. W. Teh, "Neural processes," 2018.
- [6] J. Mockus, V. Tiesis, and A. Zilinskas, "The application of Bayesian methods for seeking the extremum," *Towards Global Optimization*, vol. 2, no. 117-129, p. 2, 1978.
- [7] W. R. Thompson, "On the likelihood that one unknown probability exceeds another in view of the evidence of two samples," *Biometrika*, vol. 25, no. 3/4, pp. 285–294, 1933.
- [8] N. Srinivas, A. Krause, S. M. Kakade, and M. W. Seeger, "Information-theoretic regret bounds for gaussian process optimization in the bandit setting," *IEEE Transactions on Information Theory*, vol. 58, p. 3250–3265, May 2012.
- [9] T. Hospedales, A. Antoniou, P. Micaelli, and A. Storkey, "Meta-learning in neural networks: A survey," 2020.
- [10] C. Finn, P. Abbeel, and S. Levine, "Model-agnostic meta-learning for fast adaptation of deep networks," 2017.
- [11] Y. Chen, M. W. Hoffman, S. G. Colmenarejo, M. Denil, T. P. Lillicrap, M. Botvinick, and N. de Freitas, "Learning to learn without gradient descent by gradient descent," 2017.
- [12] J. Requeima, J. Gordon, J. Bronskill, S. Nowozin, and R. E. Turner, "Fast and flexible multi-task classification using conditional neural adaptive processes," 2020.
- [13] J. González, M. Osborne, and N. D. Lawrence, "Glasses: Relieving the myopia of bayesian optimisation," 2015.

- [14] S. Jiang, H. Chai, J. Gonzalez, and R. Garnett, “Binoculars for efficient, nonmyopic sequential experimental design,” 2020.
- [15] S. Jiang, D. R. Jiang, M. Balandat, B. Karrer, J. R. Gardner, and R. Garnett, “Efficient nonmyopic bayesian optimization via one-shot multi-step trees,” 2020.
- [16] E. H. Lee, D. Eriksson, V. Perrone, and M. Seeger, “A nonmyopic approach to cost-constrained bayesian optimization,” 2021.
- [17] A. X. Yang, L. Aitchison, and H. B. Moss, “Mongoose: Path-wise smooth bayesian optimisation via meta-learning,” 2023.
- [18] A. Maraval, M. Zimmer, A. Grosnit, and H. B. Ammar, “End-to-end meta-bayesian optimisation with transformer neural processes,” 2023.

Neural Radiance Fields (NeRF): Current State of Art and Ongoing Challenges

Yu Xu

yu.1.xu@aalto.fi

Tutor: Matti Siekkinen

Abstract

This paper provides a detailed review of the latest developments in Neural Radiance Fields (NeRF) and discusses the challenges currently faced in the field. NeRF, an innovative model for representing complex 3D scenes, relies solely on 2D images for supervision. Since its introduction in 2020 by Mildenhall and others, NeRF has been widely applied in multiple domains such as graphic software development, scene rendering, 3D surface extraction, and view synthesis.

The paper first begins by revisiting the background and theoretical aspects of NeRF, illustrating the volumetric rendering process based on Multi-Layer Perceptrons (MLP) and how NeRF learns to simulate various aspects of the 3D world by studying numerous 2D images taken from different camera angles and their corresponding poses.

Subsequently, the paper introduces several significant improvements in NeRF in recent years. Variants like Mip-NeRF and Ref-NeRF have made notable progress in the quality of synthetic views. For instance, Mip-NeRF employs cone tracing and integrated positional encoding (IPE) to enhance the quality of synthesized views; Ref-NeRF focuses on improving the simulation of reflective surfaces; and RegNeRF addresses the challenges of training with sparse input views by introducing additional depth and color normalization. MVSNeRF leverages pre-trained convolutional neural networks (CNNs) to extract image features, significantly reducing the

number of samples required for NeRF training. FastNeRF accelerates the inference process by decomposing the color function, achieving an increase in speed by over 3000 times.

Overall, despite significant progress in NeRF technology, challenges remain in handling extreme lighting conditions, high dynamic range scenarios, and large-scale data. Future research may focus on further enhancing the model's versatility, optimizing algorithms, and reducing reliance on high-quality training data.

KEYWORDS: *Neural Radiance Fields, Novel View Synthesis, Neural Rendering, Volume Rendering,*

1 Introduction

A Neural Radiance Field (NeRF) is an innovative model to represent complex 3D scenes using only 2D-posed images as supervision. It has been proposed to learn about the 3D structure of scenes by utilizing multi-view photos with a volume rendering based on Multi-Layer Perceptions (MLPs). This model is trained with extensive 2D images and their corresponding poses in different camera angles, enabling it to simulate all aspects of the 3D world, encompassing lighting, materials, and geometry. Mildenhall et al. [1] introduced NeRF in 2020, and it has inspired many subsequent works since then. In recent years, NeRF has been widely used in graphical software development, surround scene rendering, 3D surface extraction, and view compositions.

Compared to traditional methods, NeRF provides important advantages, such as a self-supervised model. It can be trained with only posed images in 2D to understand a scene. These posed images can also be estimated through colmap software [2] or packages, which was also demonstrated in the original NeRF paper. Additionally, the original NeRF model has also been demonstrated to be of superior visual quality compared to earlier novel view synthesis methods, such as [3], [4]. More recent models have further improved upon its performance.

Currently, the NeRF model has become increasingly popular in the field of computer vision, with a large number of derivative studies based on the baseline NeRF model continuing to appear on various open-source websites, many of which ultimately appear at top computer vision confer-

ences. However, with the large number of optimized NeRF models derived from current research progress, it is difficult for beginners to understand the overall development process of NeRF from scratch and clarify the remaining challenges.

To better assist beginners in understanding the overall development of the NeRF model, this paper reviews the latest developments in NeRF to assist computer vision practitioners in this emerging field. It also re-searches the identified challenges in the present landscape.

The structure of this paper is as follows. Section 2 presents the existing surveys on NeRF and briefly elaborates on the core theory behind NeRF. Section 3 introduces influential NeRF publications, mainly providing an introduction to the innovation of NeRF in recent years and the latest applications in various fields. Section 4 discusses open problems that have been identified and summarizes the current progress. Finally, Section 5 provides concluding remarks.

2 Background

2.1 Previous research on NeRF

In 2019, the concept of volume rendering was first introduced by Lombardi et al.[5], who developed regression equations for density and color in 3D volumes, albeit using a voxel-based representation. One of the main points of this paper is that neural volume rendering methods have the potential to be successful, partially explaining the subsequent achievements.

In 2020, NeRF was initially introduced by Mildenhall et al. [1] for generating synthetic views of complex scenes. Remarkably, even with its simple architecture, NeRF outperformed prior work quantitatively on datasets comprising both synthetic and real images. However, NeRF still presented opportunities for improvement, particularly in the realms of performance and generalization.

In the six months following the proposal, many researchers worked to address these challenges. Lindell et al. [6] improved the performance of NeRF by directly learning the volume integral, speeding up the rendering process. Park et al. [7] introduced a second MLP to apply deformation to each video frame, addressing the limitations of NeRF in handling only

static scenes. Then, Srinivasan et al. [8] successfully used the arbitrary ambient and indirect lighting with a second MLP, handling variations in image appearance due to different lighting conditions. Subsequent improvements branched into various aspects, such as photometric and geometric quality, composition, and pose estimation.

In May 2022, an updated and exhaustive report on neural rendering and NeRF models was published [9], focusing on methods for combining classical rendering theory with 3D scene representations. This comprehensive survey-style report summarized classic NeRF papers and other neural rendering papers from recent years. In contrast, our survey primarily concentrates on NeRF papers from 2021 to 2022, summarizing research progress during this period and discussing the remaining relevant issues to be resolved.

2.2 Theory of NeRF

The process of NeRF work can be divided into two parts: reconstruction of 3D scene and rendering. In the process of 3D reconstruction, 3D coordinates, and 2D viewpoints are converted into color and volume densities by a 5D vector-valued function

$$F(x, d) \rightarrow (c, \sigma) \quad (1)$$

where $x = (x, y, z)$ is a 3D location of the a point in the scene, $d = (\theta, \phi)$ is a 2D viewing direction of the image, $c = (r, g, b)$ represents the an emitted output color and σ represents the output the volume density. The 5D vector-valued function is approximated by an MLP network, which is sometimes denoted as F_θ and the weights θ were optimized to map from an input 5D vector to the corresponding directional calculated color and volume density. In the baseline NeRF model, to constrain the consistency of multiple-view, the neural network is limited to predict the volume density ϕ only through the function of position x , while allowing RGB color c to be predicted as a function of position and viewing direction. MLP requires two steps to achieve this, first processing the input 3D coordinate x as well as outputting it σ and a 256-dimensional feature vector. Then, the feature vector is connected to the viewing direction of the rays of a camera and transmitted to an additional fully connected layer that outputs RGB colors.

Generally speaking, to obtain a novel view synthesis based on images, we can use the NeRF model and proceed through the following steps:

- Emit camera rays from each pixel in the image to generate a set of sampling 3D points.
- For sampling points, the NeRF uses the 2D viewing direction $d = (\theta, \phi)$ and the 3D point location $x = (x, y, z)$ to output color and volume density.
- Render novel composite views based on the output color and volume density by volume rendering.

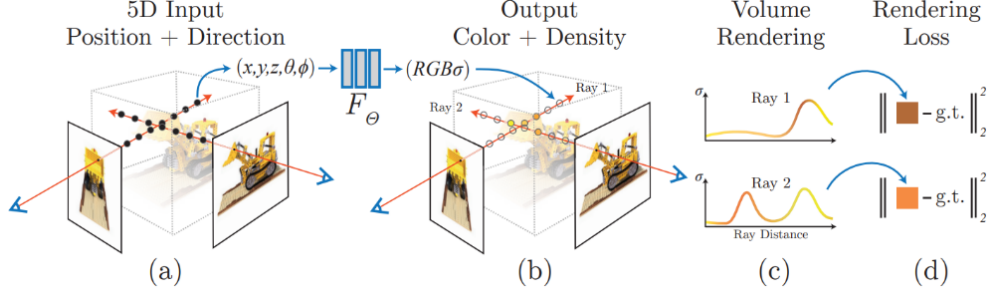


Figure 1. Volume rendering and training process of the NeRF. Image sourced (a) How camera rays project and generate 3D sampling points from pixel points on an image. (b) Use a NeRF model to generate the colors and densities at the specific sampling points. (c) explains the generation of a single image pixel along the camera ray based on the volume rendering. (d) is the optimization of the objective function, i.e., minimizing the residual between synthesized and observed images [3]

On the basis of body rendering, we can use integrals to describe the algebraic relationship between the expected color along the camera ray $r(t) = \mathbf{0} + t\mathbf{d}$, using

$$C(r) = \int_{t_n}^{t_f} T(t) \sigma(r(t)) c(r(t), d) dt \quad (2)$$

where $\int_{t_n}^{t_f}$ represents the near and far bounds of the camera rays, $\sigma(r(t))$ represents the volume density, $c(r(t), d)$ represents the emitted color and $r(t)$ represents the 3D point along the camera ray from d direction. $T(t)$ is a cumulative transmittance that represents the probability that a camera ray is not blocked by another object from t_n to t , given by

$$T(t) = \exp\left(-\int_{t_n}^t \sigma(r(u)) * du\right) \quad (3)$$

For each pixel, the MLP parameters are optimized using the squared error photometric loss. In the whole image, it is calculated as

$$L = \sum_{r \in R} \|\hat{C}(r) - C_{gt}(r)\|_2^2 \quad (4)$$

where $C_{gt}(r)$ is the ground true color of training color pixels associated with r , while R is a set of camera rays that would be synthesized associated with the image. Mildenhall et al. [3] also introduced a technique known as position encoding, which can greatly improve the reconstruction of details in synthesized views. In addition, they mention the use of some novel sampling methods in the original paper for more targeted sampling by dividing the network into two classes, coarse and fine [3]. As this paper is mainly a compendium of the recent advances in the NeRF model, we ignore these tricks here.

3 Latest Advances in NeRF

3.1 Improvements in Synthetic Views

Quality assessment is an important metric for view synthesis, numerous subsequent models have prioritized improving the quality of view synthesis. For instance, Mip-NeRF [10] diverged from the conventional ray tracing approach used in standard NeRF [1] by adopting cone tracing. This innovation was made possible through the incorporation of Integrated Location Encoding (IPE), as shown in Figure 2. In the process of generating a single pixel, a cone is projected from the camera’s central point in the direction of the viewing angle, passing through the representation point of the pixel. This cone is approximated through a multivariate Gaussian model, and its mean vector and variance matrix are determined based on relevant geometric structures, culminating in an integrated position encoding.

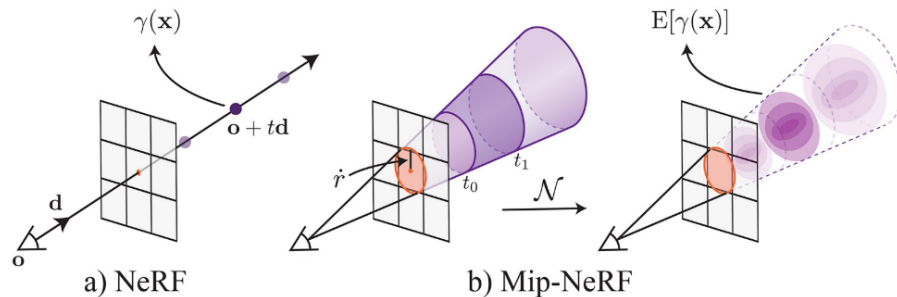


Figure 2. IPE plot for Mip-NeRF [10] a) Traditional point sampling method employed by NeRF based on ray tracing; b) Conic sampling method utilized by Mip-NeRF with the integration of IPE, where the conic curve is approximated using a multivariate Gaussian distribution.

Ref-NeRF [11] builds upon the foundation of Mip-NeRF with a specific

focus on enhancing the simulation of reflective surfaces. In Ref-NeRF, the radiance of NeRF is parameterized based on the reflection of local normal vectors as observed from a particular direction. To achieve this, several critical modifications are made. They transform the MLP into a non-orientation MLP. This modified MLP not only provides the input eigenvectors of the density and orientation MLPs but also includes additional information, such as the diffuse reflection color, specular reflection color, and surface normal. The final color representation is obtained by combining the diffuse and specular colors through a combination of multiplication and addition, in conjunction with the specular color generated by the orientation MLP. Ultimately, Ref-NeRF demonstrates superior performance compared to benchmark methods on various datasets, including the real-world scene capture data [12] and the original NeRF dataset [1].

RegNeRF, as presented in [13], aims to solve the challenge of training NeRF with sparse input views. In contrast to the majority of other approaches, which typically rely on image features from pre-trained networks to guide adjustments to NeRF’s volume rendering, RegNeRF takes a distinct approach by incorporating additional depth and color regularization. When evaluated on the DTU [14] and LLFF [3] datasets, RegNeRF demonstrated superior performance compared to the original NeRF models.

3.2 Reduction in Training Requirement

The essential NeRF framework depends on the availability of densely acquired multi-view images, each combined with well-defined camera poses for a given scene. A frequent challenge with the standard NeRF model is its vulnerability to variations in the number of training perspectives. When the training views exhibit insufficient variation, whether due to a limited number of training samples or minimal posture variations, the model can be prone to overfitting, resulting in the generation of unrealistic scene geometry. However, a series of NeRF models utilize pre-trained networks to extract image features, often based on pre-trained convolutional neural networks (CNNs), like those introduced in [15] and [16]. This incorporation of pre-trained networks substantially reduces the requisite number of training samples for NeRF training, enhancing its effectiveness.

MVSNeRF, as introduced in [17], employs pre-trained convolutional neural networks (CNNs) to extract the image feature vector. These ex-

tracted image features are further transformed into a 3D voxelated cost volume using planar scanning and cost calculations based on variance. Furthermore, the model incorporates pre-trained 3D CNNs for the extraction of a 3D neural encoding volume, which is subsequently applied to generate potential codes for individual points through interpolation. An MLP takes these potential features, point coordinates, and viewing directions as inputs to generate point density and color. The training process also includes the simultaneous optimization of the 3D feature volume. In evaluations conducted on the DTU dataset, MVSNerF exhibited the ability to achieve results not less than the standard NeRF baseline training, but notably accomplished this in a significantly shorter period, taking only 15 minutes instead of the usual several hours.

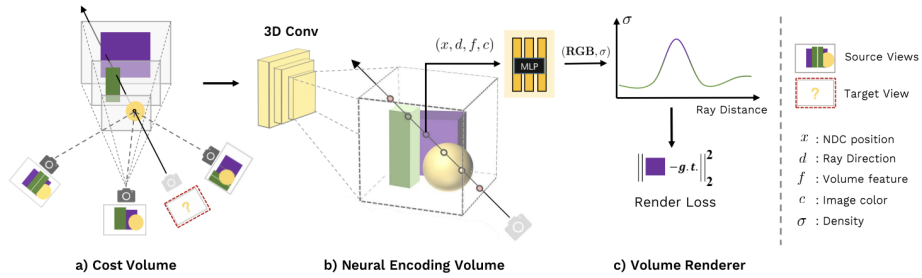


Figure 3. The overall process of MVSNerF [17] a) Creating a cost volume by projecting 2D image features onto a swept plane. b) Using a 3D CNN to generate a neural encoding volume, where each voxel is characterized by unique neural features. c) Utilizing a multi-layer perceptron (MLP) to estimate volume density and RGB radiance at a specified location, this process involves interpolating features from the encoding volume.

GeoNeRF [18] used a pre-trained feature pyramid network to extract features from each image view. This method used planar scanning to construct a cascaded 3D cost volume. From these two feature representations, GeoNeRF extracts a view-independent feature marker and multiple view-dependent feature markers for each of the N points along the ray. Next, the N view-independent markers undergo refinement using an AutoEncoder, yielding N density values along the ray. For the extraction of colors, N sets of tokens about the view are separately input into MLP. This methodology delivered impressive results when evaluated on datasets such as DTU [14], NeRF synthesis [1], and LLF forward [3].

3.3 Optimization of training efficiency

In the initial implementation by Mildenhall et al. [1], they introduced a layered rendering technique to enhance computational efficiency. Con-

ventional rendering would necessitate a dense evaluation of the MLP along every camera ray at all query points during numerical integration. Their inventive methodology involved the utilization of two separate networks to represent the scene: one designed to capture coarse details, and the other tailored for finer details. The output of the coarse network played a crucial role in selecting the sampling points for the fine network. This approach efficiently obviated the necessity for dense sampling at the fine level, resulting in a streamlined computational process that optimized efficiency.

In the FastNeRF model presented by Garbin et al [19], a remarkable innovation was the decomposition of the color function, which involved expressing a function as the combined output of a directional position-dependent MLP and the inner product of another MLP’s output, which is associated with the direction. This decomposition offered FastNeRF the capability to efficiently store and retrieve color and density evaluations within dense grids of the scene. Consequently, it resulted in a substantial improvement in inference speed, achieving a speedup of over 3000 times. Furthermore, the FastNeRF also implemented hardware-accelerated ray tracing, leveraging technology such as OptiX [20]. This approach allowed FastNeRF to terminate the inference process when the transmittance of the ray reached saturation, further enhancing efficiency.

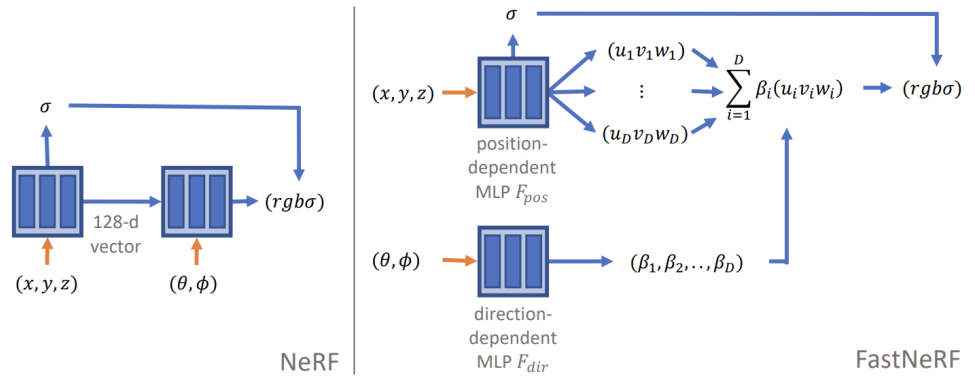


Figure 4. Differences between NeRF and FastNeRF [19]. Left: NeRF architecture. Right: FastNeRF architecture divides the task into two neural networks, both of which are suitable for caching. The first network, F_{pos} , is responsible for generating a deep radiance map (u, v, w) with D components. Concurrently, the second network, F_{dir} , is tasked with determining the weights for these components, taking into account the direction of an input ray.

EfficientNeRF, an extension of the PlenOctree framework, as introduced in [21], adopted several innovations. The most prominent among these was the significant acceleration of the training process. This acceleration was attained by leveraging momentum density voxel grids to store

the predicted density, updating them with exponential weighted averages. In the initial coarse sampling phase, these grids proved instrumental in eliminating sampling points with zero density, effectively optimizing the process. Furthermore, during the subsequent fine sampling phase, the incorporation of a pivot system also contributed to expediting the volume rendering process. These two enhancements collectively result in an eightfold reduction in training time when compared to the baseline NeRF [1]. Subsequently, the trained scene is cached within the NeRF tree, yielding rendering speeds that are on par with FastNeRF [19].

4 Conclusion

This paper introduces NeRF and its variants such as Mip-NeRF, MVS-NeRF, and FastNeRF, which have achieved significant advancements in the quality of synthetic views, training requirements, and training efficiency. These developments not only enhance the capability of NeRF models in processing complex scenes and diverse lighting conditions but also demonstrate their vast potential in applications such as graphic software development, scene rendering, and 3D surface extraction. Despite these advancements, and the demonstrated practicality and adaptability of NeRF models in various domains, challenges remain in handling extreme lighting conditions, high dynamic range scenarios, and large-scale data. Furthermore, future research should focus on improving the training and inference speed of the models, and optimizing them further to better cope with the complexities of the real world, making them more suitable for real-time applications. In summary, although there are still shortcomings, it is foreseeable that NeRF will play an increasingly significant role in fields such as virtual reality, augmented reality, film production, and game development. Additionally, further optimization of the algorithms will make NeRF easier to train and deploy, thus expanding its range of applications.

References

- [1] B. Mildenhall, P. P. Srinivasan, M. Tancik, J. T. Barron, R. Ramamoorthi, and R. Ng, "Nerf: Representing scenes as neural radiance fields for view synthesis," *Communications of the Acm*, vol. 65, no. 1, pp. 99–106, 2020. doi:10.1007/978-3-030-58452-8_24.

- [2] J. L. Schonberger and J.-M. Frahm, "Structure-from-motion revisited," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 4104–4113, 2016. doi:10.1109/cvpr.2016.445.
- [3] B. Mildenhall, P. P. Srinivasan, R. Ortiz-Cayon, N. K. Kalantari, R. Ramamoorthi, R. Ng, and A. Kar, "Local light field fusion: Practical view synthesis with prescriptive sampling guidelines," *ACM Transactions on Graphics (TOG)*, vol. 38, no. 4, pp. 1–14, 2019. doi:10.1145/3306346.3322980.
- [4] V. Sitzmann, M. Zollhöfer, and G. Wetzstein, "Scene representation networks: Continuous 3d-structure-aware neural scene representations," *Advances in Neural Information Processing Systems*, vol. 32, 2019. doi:10.48550/arXiv.1906.01618.
- [5] S. Lombardi, T. Simon, J. Saragih, G. Schwartz, A. Lehrmann, and Y. Sheikh, "Neural volumes: Learning dynamic renderable volumes from images," *arXiv preprint arXiv:1906.07751*, 2019. doi:10.1145/3306346.3323020.
- [6] D. B. Lindell, J. N. Martel, and G. Wetzstein, "Autoint: Automatic integration for fast neural volume rendering," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 14556–14565, 2021. doi:10.1109/cvpr46437.2021.01432.
- [7] K. Park, U. Sinha, J. T. Barron, S. Bouaziz, D. B. Goldman, S. M. Seitz, and R. Martin-Brualla, "Nerfies: Deformable neural radiance fields," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 5865–5874, 2021. doi:10.1109/iccv48922.2021.00581.
- [8] P. P. Srinivasan, B. Deng, X. Zhang, M. Tancik, B. Mildenhall, and J. T. Barron, "Nerv: Neural reflectance and visibility fields for relighting and view synthesis," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 7495–7504, 2021. doi:10.1109/cvpr46437.2021.00741.
- [9] A. Tewari, J. Thies, B. Mildenhall, P. Srinivasan, E. Tretschk, W. Yifan, C. Lassner, V. Sitzmann, R. MartinBrualla, and S. Lombardi, "Advances in neural rendering," in *Computer Graphics Forum*, vol. 41, pp. 703–735, Wiley Online Library, 2022. doi:10.1111/cgf.14507.
- [10] J. T. Barron, B. Mildenhall, M. Tancik, P. Hedman, R. Martin-Brualla, and P. P. Srinivasan, "Mip-nerf: A multiscale representation for anti-aliasing neural radiance fields," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 5855–5864, 2021. doi:10.1109/iccv48922.2021.00580.
- [11] D. Verbin, P. Hedman, B. Mildenhall, T. Zickler, J. T. Barron, and P. P. Srinivasan, "Ref-nerf: Structured view-dependent appearance for neural radiance fields. in 2022 ieee," in *CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 5481–5490, 2022. doi:10.1109/cvpr52688.2022.00541.
- [12] P. Hedman, P. P. Srinivasan, B. Mildenhall, J. T. Barron, and P. Debevec, "Baking neural radiance fields for real-time view synthesis," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 5875–5884, 2021. doi:10.1109/iccv48922.2021.00582.

- [13] M. Niemeyer, J. T. Barron, B. Mildenhall, M. S. Sajjadi, A. Geiger, and N. Radwan, “Regnerf: Regularizing neural radiance fields for view synthesis from sparse inputs,” in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 5480–5490, 2022. doi:10.1109/cvpr52688.2022.00540.
- [14] R. Jensen, A. Dahl, G. Vogiatzis, E. Tola, and H. Aanæs, “Large scale multi-view stereopsis evaluation,” in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 406–413, 2014. doi:10.14711/thesis-991012786067603412.
- [15] A. Krizhevsky, I. Sutskever, and G. E. Hinton, “Imagenet classification with deep convolutional neural networks,” *Advances in neural information processing systems*, vol. 25, 2012. doi:10.1145/3065386.
- [16] K. He, X. Zhang, S. Ren, and J. Sun, “Deep residual learning for image recognition,” in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 770–778, 2016. doi:10.1109/cvpr.2016.90.
- [17] A. Chen, Z. Xu, F. Zhao, X. Zhang, F. Xiang, J. Yu, and H. Su, “Mvsnerf: Fast generalizable radiance field reconstruction from multi-view stereo,” in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 14124–14133, 2021. doi:10.1109/iccv48922.2021.01386.
- [18] M. M. Johari, Y. Lepoittevin, and F. Fleuret, “Geonerf: Generalizing nerf with geometry priors,” in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 18365–18375, 2022. doi:10.1109/cvpr52688.2022.01782.
- [19] S. J. Garbin, M. Kowalski, M. Johnson, J. Shotton, and J. Valentin, “Fast-nerf: High-fidelity neural rendering at 200fps,” in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 14346–14355, 2021. doi:10.1109/iccv48922.2021.01408.
- [20] S. G. Parker, J. Bigler, A. Dietrich, H. Friedrich, J. Hoberock, D. Luebke, D. McAllister, M. McGuire, K. Morley, A. Robison, *et al.*, “Optix: a general purpose ray tracing engine,” *Acm transactions on graphics (tog)*, vol. 29, no. 4, pp. 1–13, 2010. doi:10.1145/1833349.1778803.
- [21] T. Hu, S. Liu, Y. Chen, T. Shen, and J. Jia, “Efficientnerf efficient neural radiance fields,” in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 12902–12911, 2022. doi:10.1109/cvpr52688.2022.01256.

Testing network policies with eBPF

Zehui Xu

zehui.xu@aalto.fi

Tutor: Jacopo Bufalino

Abstract

This paper investigates the application of Extended Berkeley Packet Filter (eBPF) for testing network policies on Linux hosts. eBPF, deeply integrated into the Linux kernel, offers a lightweight yet powerful tool for inspecting and manipulating packets directly within the kernel space. The primary objective is to leverage eBPF's kernel-hooking capabilities for real-time network monitoring, specifically to identify reasons for Transmission Control Protocol (TCP) and User Datagram Protocol (UDP) packet drops at the kernel level. The study involves a case study using eBPF for network monitoring and testing firewall policies on a Linux virtual machine. The findings demonstrate eBPF's effectiveness in accurately identifying and reporting packets dropped by firewall policies, highlighting its potential in network security and system administration.

KEYWORDS: eBPF, Network Monitoring, Firewall Policies, Linux Kernel, System Administration, Network Security

1 Introduction

In the field of modern computing, the ability to observe and monitor system resources and network activities is crucial. Currently, the industry is

increasingly adopting cloud-based solutions and container technologies, such as Docker and Kubernetes. These cloud-based solutions and technologies demand more advanced real-time monitoring tools because of their complexity. One tool that has risen to prominence is the Extended Berkeley Packet Filter (eBPF) [1].

eBPF is an extended version of the Berkeley Packet Filter, a technology initially designed for network traffic analysis. Over the years, it has evolved to offer a raw interface to different network layers and kernel process, enabling high-performance data packet capture and manipulation. eBPF extends these capabilities and is now used for a broad range of applications, most in observability, which includes network monitoring, system performance tracking, and more [2]. Many famous cloud-native applications achieve high performance through their utilization of eBPF technology [3, 4].

The primary objective of this paper is to survey the existing applications of eBPF in the field of network observability and packet manipulation. The paper aims to study how eBPF can be employed to monitor network activities, track network packets, and test network policies, thus offering a comprehensive observability solution.

This paper is organized as follows. Section 2 elaborates on the motivation and goals for this research. Section 3 provides the technical background relevant to the experiments. Section 4 presents a case study focusing on using eBPF for network monitoring and testing firewall policies of on Linux. Finally, Section 5 provides the concluding remarks and discussion on how the technology of eBPF can be further studied.

2 Motivation and Goals

In current cloud-native landscape, networking has become exceedingly complex. Organizations are increasingly adopting microservices architectures and orchestration tools, leading to an intricate mesh of network interactions. Traditional tools, such as iptables and tcpdump, have served developers well in the past, but they are limited when there is a need to debug intricate kernel-level network issues. Even when such traces look fine, network packets can mysteriously disappear, making diagnosis extremely difficult. The TRACE feature of the iptables diagnostic tool sometimes shows that packets are subjected to transformations and supposedly sent out. However, they never leave the host [5]. The issue seems

to lie in some space beyond what iptables could trace, perhaps in the network interface controller (NIC) driver or elsewhere in the kernel stack.

Given the shortcomings of existing methods, there is a clear need for tools that offer more granular insight into the Linux kernel's state. This becomes crucial for identifying not just the points of failure but also for understanding the behavior of various kernel components that interact with network packets [3, 5]. For this reason, we need an in-kernel understanding of a network packet. Ideally, a tool that offers insights into the decision-making processes affecting its routing is required. If a packet is dropped, understanding the specific reasons behind this action could be invaluable for debugging and future optimization.

Considering the complexities of modern networking and the limitations of existing diagnostic tools, such as iptables and tcpdump, this research aims to leverage eBPF's kernel-hooking capabilities for real-time network monitoring. The primary goal is to implement an initial feature that identifies the reasons for TCP and UDP packet drops at the kernel level, outputting these in a structured format that includes the protocol, source/destination IPs, and ports. This functionality will be validated for its utility within a Linux virtual machine environment. By achieving this, the research will contribute an eBPF tool for observing firewall behavior directly from the kernel state, enhancing debugging and optimization capacities in network configurations.

3 Background

This section aims to provide an understanding of eBPF technology and Linux network packet filtering and manipulation system. It also serves to bridge these technologies and provides a knowledge base that will be instrumental for the case study explored in subsequent section.

3.1 eBPF Fundamentals

Extended Berkeley Packet Filter (eBPF) functions as a virtual machine residing within the kernel space, enabling dynamic programmability of the Linux kernel without necessitating alterations to its source code or the loading of additional modules [2]. In contrast to conventional methods, including modifying kernel source code and employing kernel modules, eBPF offers a safer and more flexible approach that permits users

to inject custom code logic into the kernel. This is achieved by employing a Just-In-Time (JIT) compiler to optimize the execution of a set of highly constrained instructions, thereby upholding kernel security. Moreover, we cannot directly use kernel functions within eBPF. eBPF provides a set of helper functions, consisting of over 220 BPF Helper Calls [2], which facilitate communication with the kernel while adhering to security and validity constraints. These helper functions restrict the range of permissible operations and extend the capabilities of eBPF.

In practical, kernel-space eBPF programs, which are often written in languages such as C or Rust, are compiled into a universal eBPF ELF (Executable and Linkable Format) files. These files are then loaded into the kernel by user-space eBPF program.

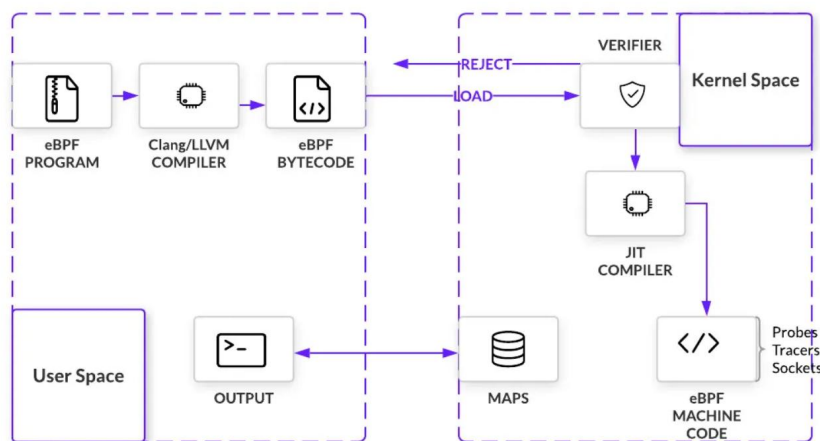


Figure 1. eBPF Architecture [6]

User-space programs are also aimed to process the results returned from the kernel. We can use eBPF Maps to facilitate data transfer between the kernel and user space. This structure supports extensive storage and the collection of key statistics [1]. These Maps enable efficient data exchange and storage management, playing a crucial role in various application scenarios, from packet filtering to performance tuning. Since user-space programs only inject kernel-space eBPF programs and process data sent back, they can be written in various programming languages [3].

Kernel-space eBPF programs must be compiled alongside kernel header files to generate the eBPF ELF file, which is subsequently injected into kernel hook points for executing custom logic. The data structures used in this hook context align with those used within the kernel. Through various hook points and eBPF Helper Calls, our kernel-space eBPF programs

can interact with various subsystems, covering a myriad of application scenarios from packet filtering to performance tuning.

3.2 Iptables, Netfilter and eBPF hook points

Iptables serves as a user-space utility program that facilitates the configuration of IP packet filter rules in the Linux kernel firewall. Developed by the Netfilter Core Team, iptables operates through different tables, each containing chains of rules that dictate how network traffic packets should be processed [7]. The underlying kernel feature that enables iptables to function is the netfilter framework. The Linux kernel's netfilter framework allows users to manipulate network packets via kernel modules, which register callback functions on a set of netfilter hooks [8].

In terms of architecture, iptables uses various tables, such as 'filter,' 'nat,' and 'mangle,' each serving a unique purpose. These tables contain chains like INPUT, OUTPUT, and FORWARD, which in turn hold rules specifying actions on packet matches. The system also utilizes hooks at specific points in the protocol stack, such as INPUT and OUTPUT, to trigger actions.

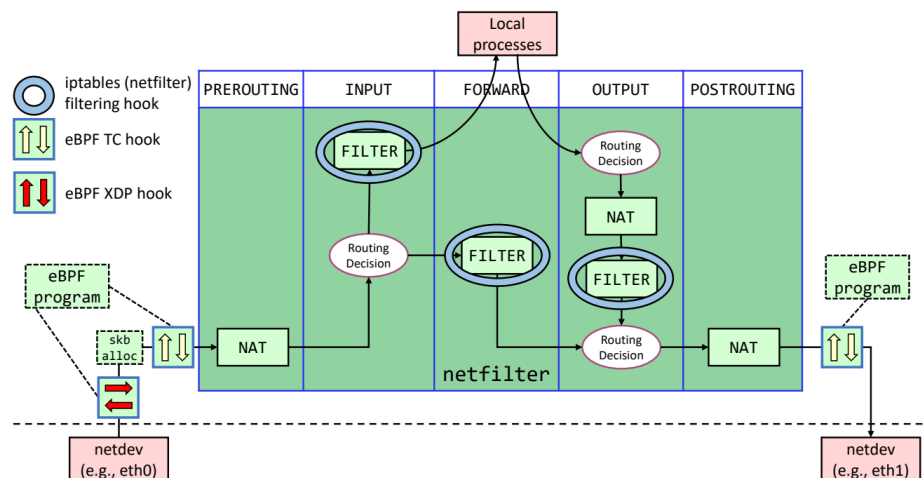


Figure 2. Location of netfilter and eBPF hooks [9]

These netfilter hooks are essentially invoked by a unified kernel function and return the results of the matching operations, essentially dictating the flow of the data packets. Two primary mechanisms exist for this purpose: the older k(ret)probe and the newer fentry/fexit mechanisms [8, 10, 11]. The k(ret)probe mechanism offers more versatility in terms of attachable functionalities; it allows attachment to any function within the

kernel. On the other hand, the fentry/fexit mechanism requires the availability of BPF Type Format (BTF) information for the functions to which they attach, posing a limitation especially for functions marked as static in the kernel [10, 9]. Given the extensive literature on k(ret)probe, this paper opts to use k(ret)probe to hook into key netfilter kernel functions to achieve the objective of monitoring firewall behavior within the kernel.

4 Case Study: Using eBPF to Test Firewall Policies on a Linux Host

In the experiment section of this study, the objective is to develop a program that leverages the capability of eBPF to hook into kernel functions. This program aims to accurately determine, from the kernel’s perspective, whether a set of given network quintuples—comprising the protocol number, source address, destination address, source port, and destination port—would be dropped by firewall policies. Essentially, the program inputs a list of these quintuples and outputs the firewall’s decision results.

4.1 Experiment Design

In actual design, to avoid altering the host system configuration and network environment, Vagrant is used for setting up a controlled environment within a VirtualBox virtual machine. The virtual machine runs on Ubuntu/Lunar64 with a 6.2 version kernel. The choice of a newer kernel version is driven by its generally improved support for eBPF and the availability of additional kernel hook points, which are commonly found in newer kernel releases [12].

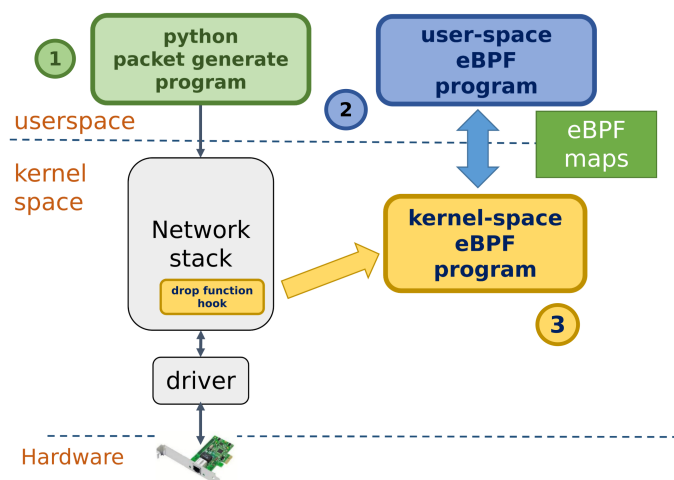


Figure 3. Program architecture

As shown in Figure 3, our program is divided into three main components. The first is a Python frontend application, which utilizes `hping3` to generate and send a series of network packets based on a custom-defined list of quintuples in a JSON file. The second and third components are eBPF programs, including both user-space and kernel-space elements. The kernel-space eBPF program is responsible for collecting the kernel's decision results and storing them in a map for retrieval by the user-space eBPF program. The user-space program then reads these results from the map and outputs them to a file.

4.2 Experiment Procedure

Rust Aya framework is chosen for the development of the eBPF program for its safety and simplicity. The primary functions of a user-space eBPF program include loading the eBPF code, deploying it into the kernel, and attaching it to a designated kernel function. We choose to hook into the `kfree_skb_reason` kernel function, which is called by netfilter when deciding to drop a packet [13]. This is illustrated in the following Rust code snippet.

Listing 1. Hooking into `kfree_skb_reason`

```
let program: &mut KProbe = bpf.program_mut("kfree_skb_reason_hook").
    unwrap().try_into()?;
program.load()?;
program.attach("kfree_skb_reason", 0)?;
```

Next, the program retrieves specific drop reasons from the `FLOW_TABLE` map, declared and filled by kernel eBPF program, and outputs them to a file (Listing 2).

Listing 2. Retrieving Drop Reasons from `FLOW_TABLE`

```
let flow_table: HashMap<_, IPTableFlow, u64> = HashMap::try_from(
    bpf_guard.map_mut("FLOW_TABLE_V4").unwrap()).unwrap();
```

For the kernel-space eBPF program attached to the `kfree_skb_reason` kernel function, the entry point is the context in which this kernel function is called. The signature of `kfree_skb_reason` in C language style in the Linux kernel is shown in Listing 3, indicating that the first parameter is `sk_buff`, and the second is the drop reason. In Listing 4, these parameters are used to construct the returned data in Rust code, specifically the firewall decision results corresponding to the quintuples.

Listing 3. Signature of `kfree_skb_reason`

```
kfree_skb_reason(struct sk_buff *skb, enum skb_drop_reason reason)
```

Listing 4. Put Firewall Decision Results to Map

```
let skb: *const sk_buff = ctx.arg(0).ok_or(1i64)?;
let reason = ctx.arg::
```

The implementation of the Python program for generating network packets is considerably simpler. Its primary function is to read the quintuples of flows from a JSON file and then use the `hping3` tool to send packets to the network stack. This process is demonstrated in Listing 5.

Listing 5. Sending Packets with `hping3`

```
sudo hping3 -c 1 {daddr} -p {dport} --spooft {saddr} -s {sport}
sudo hping3 --udp -c 1 {daddr} -p {dport} --spooft {saddr} -s {sport}
```

4.3 Experiment Results

In the testing phase, multiple drop rules are configured on the virtual machine using `iptables` commands, affecting both the `INPUT` and `OUTPUT` chains.

Listing 6. Configuring `iptables` Drop Rules

```
sudo iptables -A INPUT -s 10.10.0.10 -d 10.10.0.11 -p tcp --dport
    8080 -j DROP
.....
sudo iptables -A OUTPUT -s 10.10.0.6 -p udp --sport 67 -j DROP
sudo iptables -A OUTPUT -s 10.10.0.7 -d 10.10.0.17 -j DROP
```

Following the setup, users can initiate the eBPF program to monitor kernel drop events. After starting the eBPF program, users can run packets generation Python script to send packets to the network stack based

on the test cases defined in the JSON file. In our experiment, we observed that the eBPF program correctly identified and output the test case packets that are dropped by the firewall, along with their drop reason: SKB_DROP_REASON_NETFILTER_DROP.

Listing 7. Output of eBPF Program

```
[
  {
    "proto": "6",
    "saddr": "10.10.0.10",
    "daddr": "10.10.0.11",
    "sport": 1234,
    "dport": 8080,
    "reason": {
      "code": 8,
      "desc": SKB_DROP_REASON_NETFILTER_DROP
    }
  },
  .....,
  {
    "proto": "6",
    "saddr": "10.10.0.7",
    "daddr": "10.10.0.17",
    "sport": 1234,
    "dport": 80,
    "reason": {
      "code": 8,
      "desc": SKB_DROP_REASON_NETFILTER_DROP
    }
  },
  {
    "proto": "17",
    "saddr": "10.10.0.6",
    "daddr": "10.10.0.17",
    "sport": 67,
    "dport": 68,
    "reason": {
      "code": 8,
      "desc": SKB_DROP_REASON_NETFILTER_DROP
    }
  },
]
```

From the output file, users are able to easily and clearly determine whether our test cases are being dropped and unable to pass through the system firewall, as well as the specific reasons for their drop. This outcome

demonstrates the successful fulfillment of the experiment's objectives.

5 Conclusion and Future Directions

This paper has explored the utilization of eBPF in creating a robust testing framework for firewall policies on a Linux host. eBPF, with its deep integration into the Linux kernel, offers a lightweight yet powerful tool, surpassing traditional methods in both efficiency and precision. This technology enables a detailed inspection and manipulation of packets directly within the kernel space, ensuring minimal performance overhead compared to more cumbersome approaches.

However, it is important to acknowledge the limitations of this study. This paper focuses solely on testing policies within the Linux netfilter firewall framework, without delving into other potential firewall implementations in Linux. Moreover, as most network devices may utilize Unix-based BSD systems [14], the implementation of firewalls in BSD significantly differs, and the applicability of eBPF in these environments remains to be verified.

It is our hope that the findings and methodologies presented in this paper will encourage further exploration and adoption of eBPF in the field of network security and system administration. The potential for eBPF to revolutionize network policy testing and enforcement is significant, offering a more streamlined and effective approach to managing complex network environments.

References

- [1] J. Schulist, D. Borkmann, and A. Starovoitov. (2019) Linux socket filtering aka berkeley packet filter (bpf). Retrieved October 10, 2023. [Online]. Available: <https://www.kernel.org/doc/Documentation/networking/filter.txt>
- [2] M. A. M. Vieira, M. S. Castanho, R. D. G. Pacífico, E. R. S. Santos, E. P. M. C. Júnior, and L. F. M. Vieira, "Fast packet processing with eBPF and xDP: Concepts, code, challenges, and applications," *ACM Comput. Surv.*, vol. 53, no. 1, feb 2020. [Online]. Available: <https://doi.org/10.1145/3371038>
- [3] S. A. Zadeh, A. Munir, M. M. Bahnasy, S. Ketabi, and Y. Ganjali, "On augmenting tcp/ip stack via eBPF," in *Proceedings of the 1st Workshop on EBPF and Kernel Extensions*, ser. eBPF '23. New York, NY, USA: Association for Computing Machinery, 2023, p. 15–20. [Online]. Available: <https://doi.org/10.1145/3609021.3609300>
- [4] T. Lévai, B. E. Kreith, and G. Rétvári, "Supercharge webRTC: Accelerate

- turn services with ebpf/xdp,” in *Proceedings of the 1st Workshop on EBPF and Kernel Extensions*, ser. eBPF '23. New York, NY, USA: Association for Computing Machinery, 2023, p. 70–76. [Online]. Available: <https://doi.org/10.1145/3609021.3609296>
- [5] M. Pumputis and B. Mulligan, “Going from packet where aren’t you to pwr,” <https://cilium.io/blog/2023/03/22/packet-where-are-you/>, Mar 2023, retrieved October 25, 2023.
- [6] V. Mody. (2021, May) A gentle introduction to ebpf. Retrieved October 25, 2023. [Online]. Available: <https://www.infoq.com/articles/gentle-linux-ebpf-introduction/>
- [7] W. contributors, “iptables — wikipedia, the free encyclopedia,” 2022, retrieved October 20, 2023. [Online]. Available: <https://en.wikipedia.org/wiki/Iptables>
- [8] —, “Netfilter — wikipedia, the free encyclopedia,” 2023, retrieved October 20, 2023. [Online]. Available: <https://en.wikipedia.org/wiki/Netfilter>
- [9] S. Miano, M. Bertrone, F. Risso, M. V. Bernal, Y. Lu, and J. Pi, “Securing linux with a faster and scalable iptables,” *SIGCOMM Comput. Commun. Rev.*, vol. 49, no. 3, p. 2–17, nov 2019. [Online]. Available: <https://doi.org/10.1145/3371927.3371929>
- [10] A. Starovoitov. (2019, Nov) [patch v4 bpf-next 00/20] introduce bpf trampoline. Retrieved October 25, 2023. [Online]. Available: <https://lore.kernel.org/bpf/20191114185720.1641606-1-ast@kernel.org/>
- [11] J. Keniston, P. S. Panchamukhi, and M. Hiramatsu. (2021, June) Kernel probes (kprobes). Retrieved October 25, 2023. [Online]. Available: <https://www.kernel.org/doc/html/latest/trace/kprobes.html>
- [12] L. Wüstrich, M. Schacherbauer, M. Budeus, D. Freiherr von Künßberg, S. Gallenmüller, M.-O. Pahl, and G. Carle, “Network profiles for detecting application-characteristic behavior using linux ebpf,” in *Proceedings of the 1st Workshop on EBPF and Kernel Extensions*, ser. eBPF '23. New York, NY, USA: Association for Computing Machinery, 2023, p. 8–14. [Online]. Available: <https://doi.org/10.1145/3609021.3609294>
- [13] D. Zhang, “[PATCH net-next v3 1/4] skbuff: introduce kfree_skb_list_reason(),” <https://lore.kernel.org/lkml/20220221053440.7320-2-dongli.zhang@oracle.com/>, Feb 2022, email to netdev@vger.kernel.org, bpf@vger.kernel.org, and others.
- [14] D. Spinellis and P. Avgeriou, “Evolution of the unix system architecture: An exploratory case study,” *IEEE Transactions on Software Engineering*, vol. 47, no. 6, pp. 1134–1163, 2021.

Decoding the Black-Box: A Review of Machine Learning Explainability Methods

Ziqi Wang

ziqi.wang@aalto.fi

Tutor: Alex Jung

Abstract

Explainability in machine learning has gained immense importance in today's data-driven decision-making landscape. The increasing complexity of machine learning models, especially those that achieve high accuracy, often leads to a 'black-box' scenario, where the reasoning behind decisions or predictions is not transparent. This paper presents a taxonomy and review of machine learning explainability methods, elucidating typical techniques employed to demystify these advanced models. The paper aims to enhance the understanding of explainability in machine learning, thus contributing to the development of more transparent and trustworthy AI systems.

KEYWORDS: machine learning, explainability, interpretability, black-box

1 Introduction

Machine Learning (ML) methods are integral to various decision-making processes in daily life. Achieving high accuracy is often given the highest priority when models are trained especially for business purposes because accuracy directly affects outcomes, decision-making, and profitability. As a result, ML models become increasingly complex because the complexity allows the model to react more flexibly to multidimensional input [1].

However, inferences made by such models after they receive considerable amount of data are usually obscure for humans and difficult to interpret [2]. The lack of interpretability and explainability in ML methods is likely to cause various problems, including decline in user trust and usability [3].

There are two major challenges in achieving the trade-off between explainability and accuracy in applying ML methods. First is the dilemma of determining the priority sequence explainability and accuracy. The second problem lies in the qualification of explainability in an ML method. This survey presents a taxonomy of state-of-the-art techniques that addresses the second problem and provides newcomers in this area with an update on current research progress. Section 2 provides a concise literature review, introducing some key terminologies in the paper and analyzing recent literature works in explainable ML models. A taxonomy of explainability methods is carried out in Section 3.

2 Literature review

This section provides an analysis of the different research focuses and outcomes in previous studies aimed at enhancing the explainability of ML methods, as well as three relevant definitions.

2.1 Terminology

Although the terms "explainability" and "interpretability" are often used interchangeably, they are not identical in the context of machine learning.

Interpretability of an ML model is the extent to which the model can explain its output to humans using intelligible terms [4]. Interpretation could be provided by possible mathematical or statistical knowledge.

Explainability of an ML model measures how accurately the model serves as a proxy between users and decision makers [5]. Methods to elevate explainability in machine learning models is the major topic of this paper

Accuracy is a variable that measures how well a model yields valid prediction results. For different ML problems, various benchmarks would be chosen based on specific requirements. For example, in supervised classification problems, the formula (1) calculates the accuracy of the model [6], with T and F standing for true and false prediction results and P and

N standing for positive and negative prediction.

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \quad (1)$$

Model-Specific and Model-Agnostic are used to describe explainability methods for machine learning models. Model-specific methods can be only applied to a specific family of algorithms, whereas model-agnostic methods can be applied to any possible algorithms.

2.2 Quantification of Explainability

Feature attribution is one of the key methods for explainability in machine learning models. It assigns a value to each input feature of the model, calculating the contribution from individual features. It illustrates the significance of each feature among input data points, determining which features were most and least influential in driving a particular outcome [7]. However, feature attribution often involves intensive computation, especially when researchers are dealing with deep neural networks or large data sets.

Jung [8] proposed a probabilistic model that can generate explanations of predictions based on user background. The model quantifies the quality of a explanation by shared information between explanation and prediction. Nevertheless, representing the user’s background simply as a summary of the features of a data point might be overly simplistic. Human backgrounds and prior knowledge are multifaceted and might not be adequately captured by such a summary.

In addition, explanation generation is another popular approach in yielding interpretable and understandable descriptions or reasons for the behavior or decision of a system. NEON [9], proposed in 2023, is a two-phase explanation generation framework which assists large pre-trained language models (PLMs) with explanation generation, revealing a false assertion in a previous research paper.

Preechakul [10] proposed that high-resolution heatmaps can be produced to illustrate which pixel points contribute most to the potential image classification. Compared to low-resolution methods, tiny pixels that used to be missed can be recognized when high-resolution heatmaps are applied.

2.3 Relation between Accuracy and Explainability

There are two major opinions towards the relation between accuracy and explainability of ML models. One perspective suggests that there is an inverse relationship between model accuracy and its explainability, prompting experts to lean towards black-box models for superior accuracy. Conversely, the opposing viewpoint argues that in real-world scenarios, this trade-off between accuracy and explainability is seldom seen, implying that clear, simpler models should be the default choice [11].

However, a shared belief between the two stances is that certain models, e.g, shallow decision trees and linear regression, offer more clarity, while others, including neural networks and random forests, tend to be more complex and less transparent.

Bell et al. [11] discovered that the interpretability of a model does not have a necessary relationship with explainability. However, the research has only covered two use-cases in laboratory conditions and not been carried out in a business environment or in real-world context, leaving space for future investigation.

3 Taxonomy of Explainability methods

Various perspectives arise when examining emerging development of interpretability ML methods, especially regarding the type of data they handle and the scope of the methods. As a result, the categorization of explainability methods should not be biased towards a single perspective. Linardatos divided all machine learning interpretability methods into 4 different classes based on four standards: local or global, purposes of interpretability, data types, and model-specific or model-agnostic [12]. This section will present a taxonomy of some examples of the state-of-the-art explainability methods based on different data types on which the model can be applied. Most common data types are tabulars and images. There are also some models for processing texts, but they won't be discussed in this paper. Examples of methods for ML models that are intended for processing tabulars and images are discussed in this section. In addition, suitable application scenarios and deficiencies of the methods are presented in this section.

3.1 Explainability Methods to explain models for image classification

The first category concerns the explainability methods for explainable machine learning models that are used to train and classify images. Popular models often involve deep neural networks, e.g., Convolutional Neural Network (CNN), which are usually used as black-box models [13]. Therefore, the requirement for explainability for image recognition has always been pursued by researchers. This section includes a review of three typical methods that achieved outstanding results and comparison between them.

Visualization is a standard practice for a better understanding of image-recognition networks, primarily focusing on the first layer where pixel space projections are feasible. For advanced layers, different techniques are required. A visualization method purposed by Zeiler addressed this problem by using a multi-layered Deconvolutional Network (DeconvNet) [14]. Initially, DeconvNets were introduced for unsupervised learning in [15]. However, they are not employed for learning purposes but serve as tools to interpret the workings of intermediate feature layers and information aspects within a pre-trained CNN. The effect of the DeconvNet is equivalent to applying a CNN to the original dataset but in a reverse way, and it can illustrate which part of the input originally caused a specific activation in the feature maps. The process involves connecting a DeconvNet to every layer of the CNN, ensuring a route back to the image pixels. Through occlusion experiments, they show that the model, while trained for classification, is highly sensitive to local structure in the image and is not just using broad scene context.

Class Activation Maps (CAM), first introduced in [16], is another deep learning interpretability method used for CNNs. The major effect of the method is to highlight the particular parts of an image that are used by a neural network to recognize the image. After a feature vector is created by calculating and combining the average activations of the convolutional feature maps positioned right before the last output layer, a weighted sum of this vector is fed to the final softmax loss layer. Using this particular method, unlike DeconvNet which generates activation points, CAM would localize specific image regions that trigger the recognition of the neural network. However, two major problems exist in CAM. The first is that localization can only be done when the final layer of the target neural net-

work follows a specific pattern. The second is that CAM can only explain the execution at the last stage of the neural network. Previous layers would remain unexplained after developers applied CAM.

Randomized Input Sampling for Explanation (RISE) algorithm, a novel method for explaining the predictions of deep neural networks, particularly those used in image classification[17], works by generating importance maps that visually indicate the significance of each pixel for the model's decision. The algorithm functions by masking the input image with randomly generated binary masks, with each mask revealing only a select portion of the pixels to the model. The response of the model to these masked inputs is then recorded. The final importance map is a linear combination of these random masks, where the combination weights are derived from the output probabilities predicted by the model for each masked input.

DeconvNet provides a comprehensive view of different response a CNN produces to specific input features. CAM enables object localization without requiring detailed annotations, hence reducing the annotation effort. It also elevates models explainability. However, CAM requires specific architectural changes, e.g, use of a global average pooling layer, which are not be feasible in every model. RISE operates autonomously from the internal mechanisms of the model, enabling its applicability across a broad spectrum of deep learning architectures. In addition, It provides pixel-wise importance maps, offering detailed insights into the parts of an image that influence decision-making process. Thus, RISE is the most powerful method among three methods regarding accuracy. RISE and CAM contribute to enhancing the interpretability of models, whereas DeconvNet primarily focuses on explaining the internal workings of the model. Additionally, CAM provides a straightforward application in weakly supervised localization, thereby facilitating performance enhancements in particular tasks.

As a result, DeconvNet is better when being applied to perform analysis of the structure of CNNs. CAM is preferred by light-weight supervised models, whereas RISE is more suitable for increasing explainability of complex black-box models.

Table 1. Explainability Methods to Explain Models for Image Recognition

Ref.	Name	Model Specific/Agnostic	Data Type	Application Area
[14]	DeconvNet	Specific	Image	Analyze CNN structures
[16]	CAM	Specific	Image	Supervised Learning
[17]	RISE	Specific	Image	Black-box Models

3.2 Explainability Methods to explain models for tabular data

The second category concerns machine learning models that are used to process only tabular data. ExplainD framework, a framework for visually explaining the decisions of machine-learned classifiers, first proposed by Poulin et al. [18] uses the concepts of additive models to measure the weight of the input features in the decision of the classifiers. Additive models refer to a class of statistical models in which the predictors contribute linearly to the outcome, but they do so in a way that allows for non-linear relationships between each predictor and the outcome. The main idea behind additive models is to decompose the response variable as a sum of functions of individual predictors. In additive models, the outcome (or response variable) is expressed as the sum of individual functions of predictors. Mathematically, for a response variable Y and predictors X_1, X_2, \dots, X_p , an additive model can be described as in (2):

$$Y = f_1(X_1) + f_2(X_2) + \dots + f_p(X_p) + \epsilon \quad (2)$$

f_i where represents the function (often non-linear) of the i^{th} predictor. ϵ is the error term, capturing the residual or unexplained variation in the response. This formulation allows each predictor to have its own specific relationship with the response variable, and these relationships can be non-linear, capturing complex patterns in the data. One limitation of ExplainD is that it only supports additive classifiers. Therefore, Robnik [19] proposed an extension of ExplainD which covers both additive classifiers and probabilistic models. This technique is capable of explaining the predictions of any classifier and works as follows.

A novel model-agnostic explainability technique was proposed by Ribeiro et al. [20] called Local Interpretable Model-agnostic Explanations (LIME). The explanation produced by LIME is obtained by solving optimization problem in (3):

$$\xi(x) = \arg \min_{g \in G} L(f, g, \pi_x) + \Omega(g) \quad (3)$$

where $L(f, g, \pi_x)$ is a measure of unfaithfulness (how inaccurately g approximates f in the locality defined by π_x), and $\Omega(g)$ is a measure of the complexity of the explanation g . LIME can be applied to any machine learning model, including linear, tree-based, or even deep learning models. It does this by approximating the model’s decision boundary locally with a simpler, interpretable model. However, the biggest drawback of LIME is that it can sometimes produce different explanations for the same prediction if the process is repeated. This variability can lead to a lack of trust in the explanations provided, as users might expect a consistent explanation for a model’s decision [21].

Disparate impact testing, as elucidated in [22], is a methodology designed to identify and mitigate unintentional biases in decision-making processes, particularly in algorithmic systems. The core of this method is the Balanced Error Rate (BER), defined as in (4):

$$\text{BER}(f(Y), X) = \frac{1}{2} (Pr[f(Y) = 0|X = 1] + Pr[f(Y) = 1|X = 0]) \quad (4)$$

where $f : Y \rightarrow X$ is a predictor function.

This form of testing hinges on the principle of disparate impact, which occurs when a seemingly neutral procedure leads to significantly different outcomes for various groups, especially protected classes like ethnicity or gender. The core of disparate impact testing lies in quantifying the extent to which the protected class can be predicted from other attributes. This involves assessing whether the decision outcomes disproportionately affect a protected group, typically using a statistical benchmark like the 80% rule. The rule posits that disparate impact is present if the selection rate for any protected group is less than 80% of the rate for the group with the highest selection rate. By applying this framework, Feldman et al. aim to not only identify instances of disparate impact but also propose methods to rectify data in a way that reduces or eliminates such biases while preserving the utility of the algorithm. This approach is pivotal in ensuring fairness in algorithmic decision-making, serving as a critical tool in addressing issues of equity and justice in automated systems.

ExplainD framework is versatile, compatible with various classifiers such as naive Bayes, SVMs, and logistic regression. It offers lucid, visual explanation of classification outcomes. ExplainD framework is particularly useful in business domains such as bioinformatics and healthcare, where understanding the rationale behind predictions is crucial for acceptance and decision-making. LIME is the most versatile framework

among three methods. However, if it was applied without due caution, its interpretability simplifications would result in inaccuracies in interpreting the logic of the model. LIME is particularly beneficial in contexts where discerning individual predictions is vital, such as in financial or legal environments, as well as in situations where users must determine the reliability of a specific model output. Disparate impact testing directly tackles the issue of algorithmic bias, making it essential for ensuring fairness in machine learning models. However, achieving an inherent trade-off between achieving fairness and maintaining data utility, especially in partial data repair, is challenging.

Table 2. Explainability Methods to Explain Models for Tabular Data

Ref.	Name	Model Specific/Agnostic	Data Type	Application Area
[18]	ExplainD	Agnostic	Tab.	Validate decisions
[20]	LIME	Agnostic	Tab.	Explain rationale
[22]	Impact testing	Agnostic	Tab.	Maintain fairness

4 Conclusion

The main contribution of this study is a taxonomy of typical machine learning explainability methods, facilitates a multi-faceted comparison between them. Two major categories for explainability methods were identified: methods for explaining image recognition models and methods for explaining tabular data processing models. This review offers insights into the strengths and limitations of different explainability approaches. However, the paper excludes models for processing text data, a critical area in machine learning applications like natural language processing. Future work should aim to include these models to present a more comprehensive overview of explainability methods across diverse data types.

References

- [1] R. Roscher, B. Bohn, M. F. Duarte, and J. Garcke, "Explainable machine learning for scientific insights and discoveries," *Ieee Access*, vol. 8, pp. 42200–42216, 2020.
- [2] D. Petkovic, R. Altman, M. Wong, and A. Vigil, "Improving the explainability of random forest classifier–user centered approach," in *Pacific symposium on biocomputing 2018: proceedings of the pacific symposium*, pp. 204–

- [3] G. Joshi, R. Walambe, and K. Kotecha, “A review on explainability in multi-modal deep neural nets,” *IEEE Access*, vol. 9, pp. 59800–59821, 2021.
- [4] A. B. Arrieta, N. Díaz-Rodríguez, J. Del Ser, A. Bennetot, S. Tabik, A. Barbado, S. García, S. Gil-López, D. Molina, R. Benjamins, *et al.*, “Explainable artificial intelligence (xai): Concepts, taxonomies, opportunities and challenges toward responsible ai,” *Information fusion*, vol. 58, pp. 82–115, 2020.
- [5] R. Guidotti, A. Monreale, S. Ruggieri, F. Turini, F. Giannotti, and D. Pedreschi, “A survey of methods for explaining black box models,” *ACM computing surveys (CSUR)*, vol. 51, no. 5, pp. 1–42, 2018. 10.1145/3236009.
- [6] Google, “Classification: Accuracy.” <https://developers.google.com/machine-learning/crash-course/classification/accuracy>. Accessed: 2023-10-02.
- [7] U. Schlegel and D. A. Keim, “Time series model attribution visualizations as explanations,” in *2021 IEEE Workshop on TRust and EXPertise in Visual Analytics (TRES)*, pp. 27–31, IEEE, 2021.
- [8] A. Jung and P. H. Nardelli, “An information-theoretic approach to personalized explainable machine learning,” *IEEE Signal Processing Letters*, vol. 27, pp. 825–829, 2020.
- [9] S. Cheng, Z. Wu, J. Chen, Z. Li, Y. Liu, and L. Kong, “Unsupervised explanation generation via correct instantiations,” in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 37, pp. 12700–12708, 2023.
- [10] K. Preechakul, S. Sriswasdi, B. Kijssirikul, and E. Chuangsuwanich, “Improved image classification explainability with high-accuracy heatmaps,” *Iscience*, vol. 25, no. 3, 2022.
- [11] A. Bell, I. Solano-Kamaiko, O. Nov, and J. Stoyanovich, “It’s just not that simple: an empirical study of the accuracy-explainability trade-off in machine learning for public policy,” in *Proceedings of the 2022 ACM Conference on Fairness, Accountability, and Transparency*, pp. 248–266, 2022. 10.1145/3531146.3533090.
- [12] P. Linardatos, V. Papastefanopoulos, and S. Kotsiantis, “Explainable ai: A review of machine learning interpretability methods,” *Entropy*, vol. 23, no. 1, p. 18, 2020.
- [13] M. Pak and S. Kim, “A review of deep learning in image recognition,” in *2017 4th international conference on computer applications and information processing technology (CAIPT)*, pp. 1–3, IEEE, 2017.
- [14] M. D. Zeiler and R. Fergus, “Visualizing and understanding convolutional networks,” in *Computer Vision—ECCV 2014: 13th European Conference, Zurich, Switzerland, September 6–12, 2014, Proceedings, Part I 13*, pp. 818–833, Springer, 2014.
- [15] M. D. Zeiler, G. W. Taylor, and R. Fergus, “Adaptive deconvolutional networks for mid and high level feature learning,” in *2011 international conference on computer vision*, pp. 2018–2025, IEEE, 2011.

- [16] B. Zhou, A. Khosla, A. Lapedriza, A. Oliva, and A. Torralba, "Learning deep features for discriminative localization," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 2921–2929, 2016.
- [17] V. Petsiuk, A. Das, and K. Saenko, "Rise: Randomized input sampling for explanation of black-box models," *arXiv preprint arXiv:1806.07421*, 2018. 1806.07421.
- [18] B. Poulin, R. Eisner, D. Szafron, P. Lu, R. Greiner, D. S. Wishart, A. Fyshe, B. Pearcy, C. MacDonell, and J. Anvik, "Visual explanation of evidence with additive classifiers," in *Proceedings of the national conference on artificial intelligence*, vol. 21, p. 1822, Menlo Park, CA; Cambridge, MA; London; AAI Press; MIT Press; 1999, 2006.
- [19] M. Robnik-Šikonja and I. Kononenko, "Explaining classifications for individual instances," *IEEE Transactions on Knowledge and Data Engineering*, vol. 20, no. 5, pp. 589–600, 2008.
- [20] M. T. Ribeiro, S. Singh, and C. Guestrin, "' why should i trust you?' explaining the predictions of any classifier," in *Proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining*, pp. 1135–1144, 2016.
- [21] M. Sahakyan, Z. Aung, and T. Rahwan, "Explainable artificial intelligence for tabular data: A survey," *IEEE access*, vol. 9, pp. 135392–135422, 2021.
- [22] M. Feldman, S. A. Friedler, J. Moeller, C. Scheidegger, and S. Venkatasubramanian, "Certifying and removing disparate impact," in *proceedings of the 21th ACM SIGKDD international conference on knowledge discovery and data mining*, pp. 259–268, 2015. 10.1145/2783258.2783311.