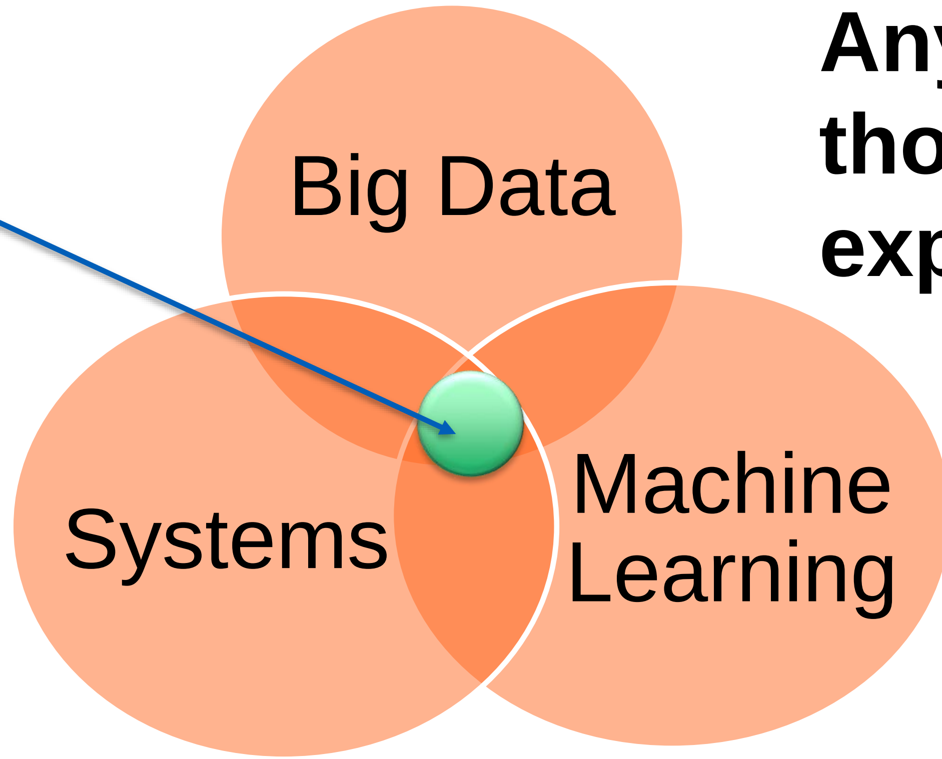
Aalto University
School of Science

Robustness, Reliability, Resilience and Elasticity (R3E) for Big Data/Machine Learning Systems

Hong-Linh Truong
Department of Computer Science
linh.truong@aalto.fi, https://rdsea.github.io

Our focus in this course

The focus



Any idea, thought, expectation?

Learning objectives

- **Identify commonality and complexity in end-to-end Big Data/ML systems**
- **Understand design goals and concerns for robustness, reliability, resilience and elasticity of Big Data/ML systems**
- **Learn an elasticity-based approach for R3E**

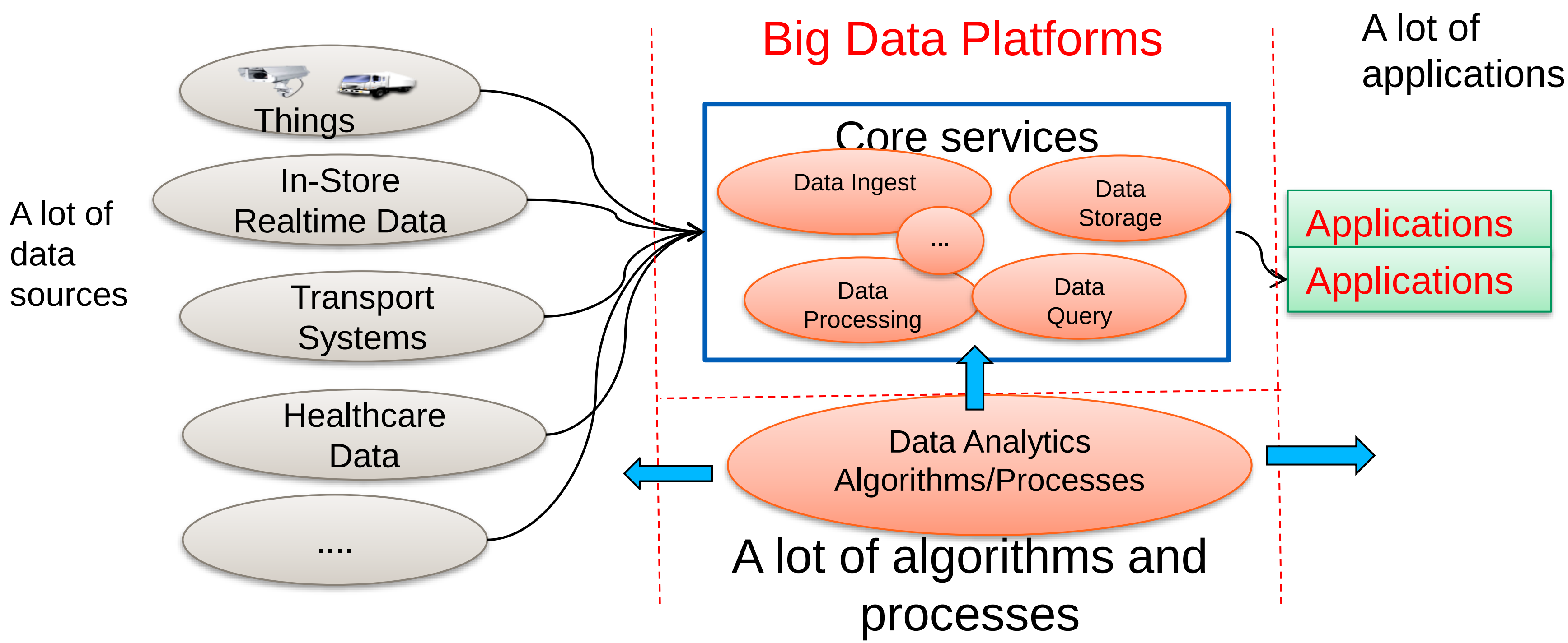
Commonality and complexity in

Big Data and Machine Learning systems

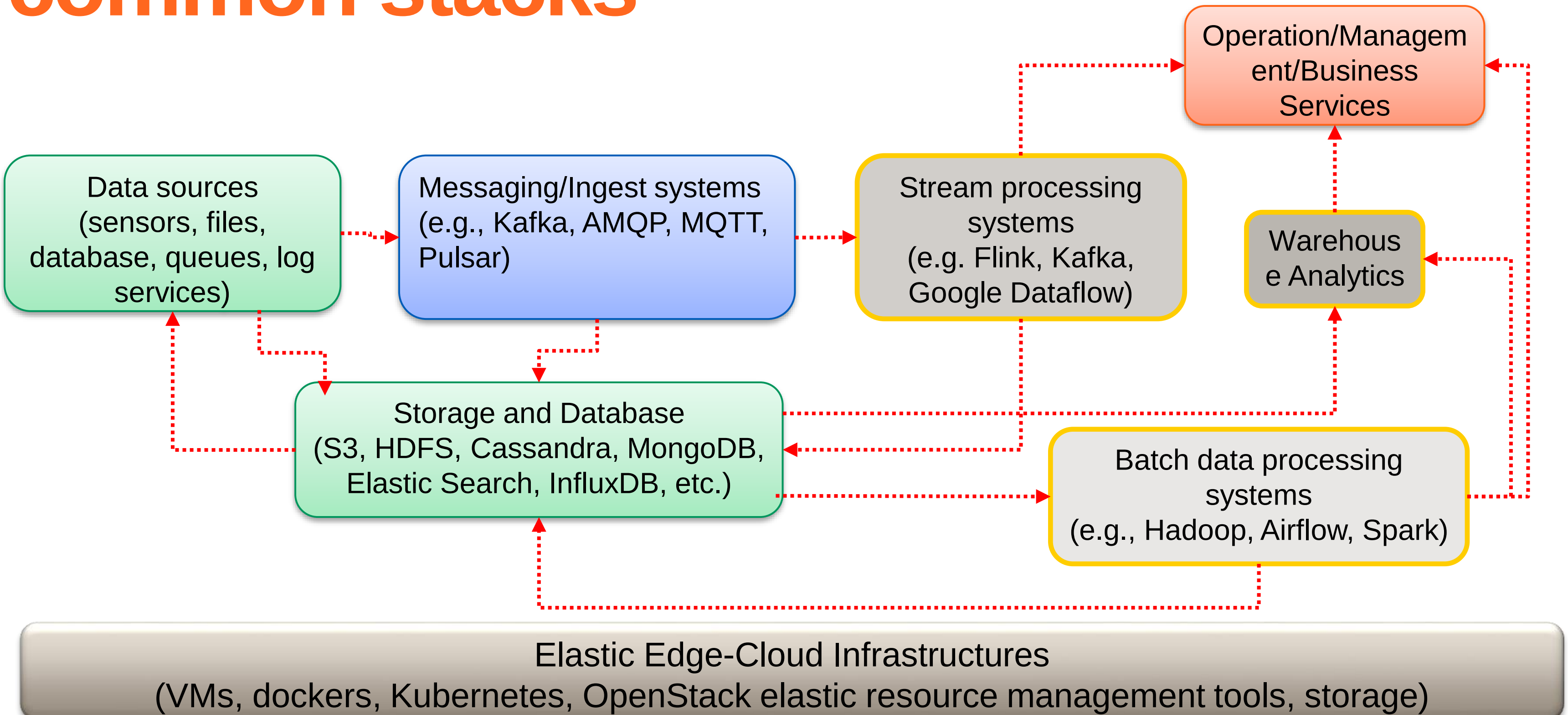
Big data with V*

- **Volume:**
  - big size, large data set, massive of small data
- **Variety:**
  - complexity of different formats and types of data
- **Velocity:**
  - generating speed, data movement speed
- **Veracity:**
  - quality is very different (timeliness, accuracy, etc.)

A bird view of big data platforms



Big data at large-scale: example of common stacks



Examples from Big Data Platforms

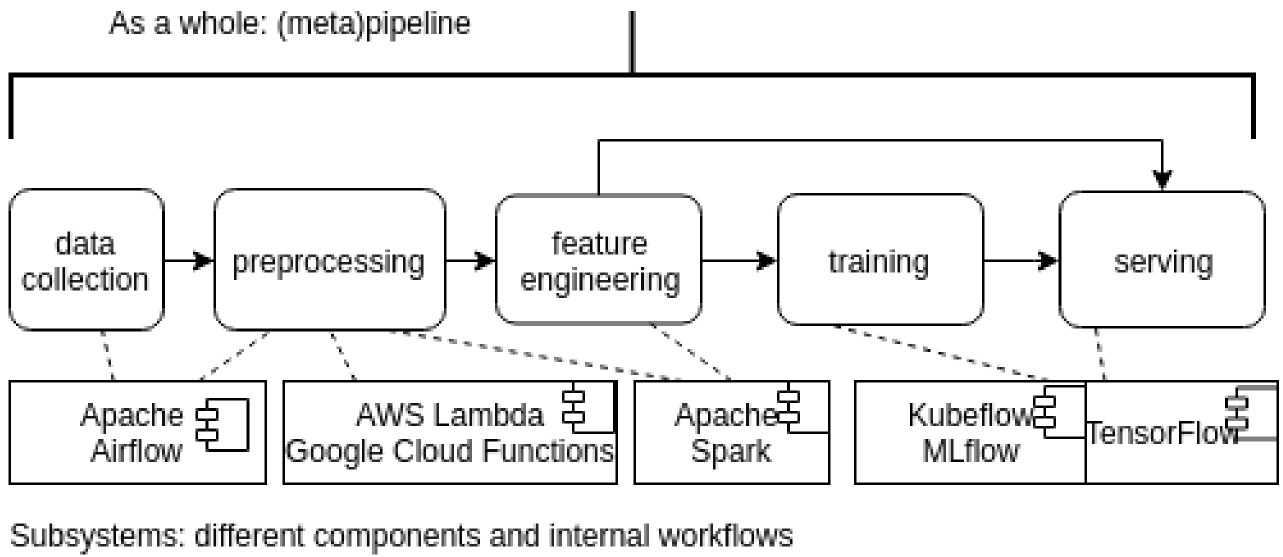
https://version.aalto.fi/gitlab/bigdataplatforms/cs-e4640

ML systems

- **Components in machine learning**
  - machine learning algorithms is a kind of "data processing"
  - there are many other components for data-preparation, data management, experiment management
- **Machine learning pipelines**
  - complex structured components, (meta)workflows
- **Data**
  - training/validation/test data, and data to be inferenced
  - models and parameters, ML experiment settings and data
  - from the big data platforms viewpoint: they are all data!

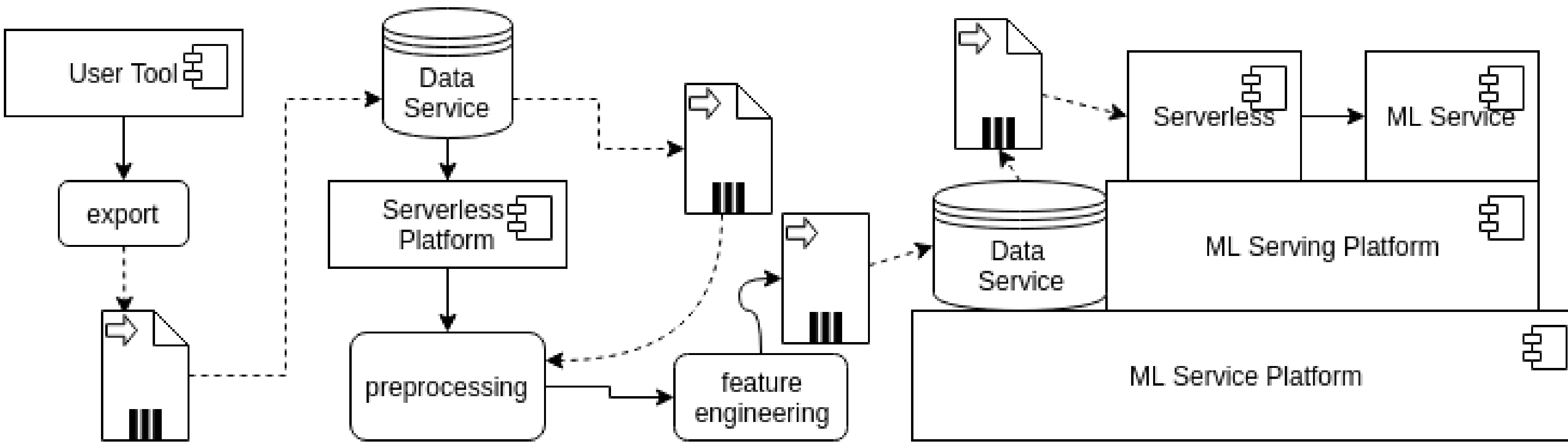
ML workflows

- **Two possible levels:**
  - meta-workflow or pipeline
  - inside each phase: pipeline/workflow or other types of programs



An example

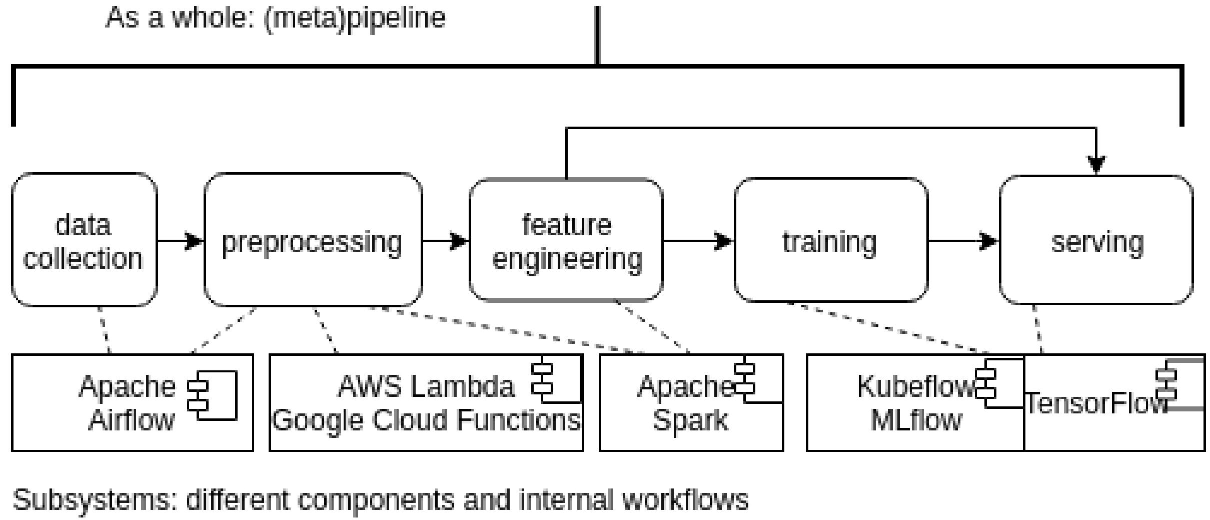
Classifying objects in Building Information Model (BIM) in Architecture, Construction and Engineering



System view: common characteristics of big data and ML systems?

- **(Static) system structures and functions**
  - include components, algorithms, input/output data
  - viewed as a whole, sub-systems, and individual parts
- **Computing and data infrastructures/platforms**
  - virtual machines/containers, brokers, storage, orchestration
- **Runtime quality/capability**
  - fault-tolerance, high-performance, high availability, secure, etc.

Examples of common components in big data and ML systems



Big data storage/ingestion

Big data processing

Resource management, workflow execution, data management tools, etc.

Computing and data infrastructures

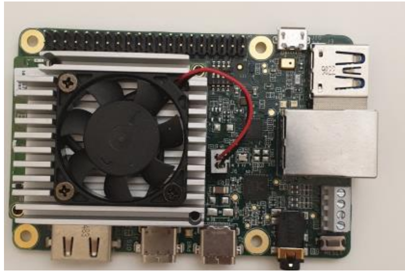
Cloud/HPC

- **Clusters of VMs/containers**
  - e.g., in Aalto we use CSC (https://www.csc.fi/)
- **High performance systems**
- **Known accelerators**
  - GPU and FPGA
- **New AI Accelerators/Processing Units**
  - TPU (Tensor Processing Unit)
  - Neutral Network Processor (NNP)
  - Vision Processor Unit (VPU)
  - IPU(Intelligent Processing Unit)

Edge systems

New types of edge and edge-cloud

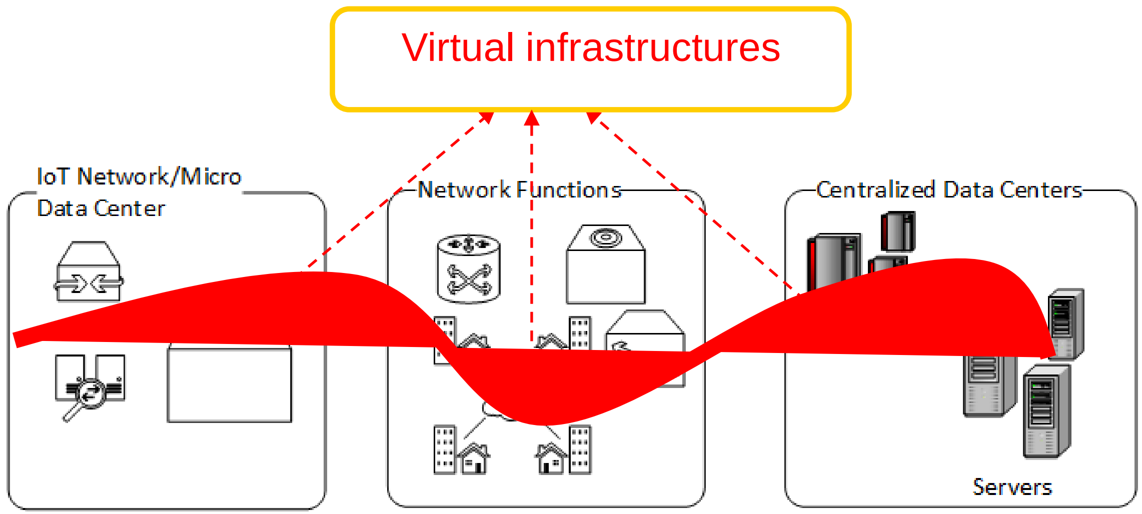
Coral with Edge TPU System-on-Module, Google Edge TPU ML accelerator coprocessor



Jetson NVIDIA (GPU+CPU)

Harnessing and orchestrating end-to-end resources

End-to-end Edge-Cloud Resources



New quantum computing for ML?

Home / Services / Azure Quantum

Azure Quantum PREVIEW

Experience quantum impact today on Azure

Become an early adopter · Get started with the Quantum Development Kit >

Amazon Braket

Explore and experiment with quantum computing

Sign up for the preview

Quantum Computing Playground

quantumplayground.net

Quantum Inspire

IBM Quantum Experience is quantum on the cloud

Accelerate your research and applications with the next generation of the leading quantum cloud services and software platform.

Try it out now

Atos Quantum Learning Machine. Photo: Atos.

Kvasi — CSC acquires quantum computing simulator

https://www.csc.fi/en/-/kvasi-csc-acquires-quantum-computing-simulator

Examples of common infrastructural/platform components

- **Data collection, ingestion, verification**
  - also data versioning management
- **Algorithms and serving components**
  - serving platforms and infrastructures
- **Configuration and workflow execution management**
- **Observability, monitoring and analysis**
- **Resource management and orchestration**

Runtime abilities/capabilities

Can you name some runtime abilities/capabilities that are important for your big data/ML systems?

Examples

- **Performance**
- **Accuracy**
- **Cost**
- **Scalability**
- **Failures handle/incidents management**
- **Site Reliability Engineering (SRE) concepts:**
  - Service level agreement (SLA), service level objective (SLO) and service level indicator (SLI)
    - *https://landing.google.com/sre/sre-book/toc/index.html*

Robustness, Reliability, Resilience and Elasticity (R3E)

Our objectives for end-to-end Big Data/ML systems engineering

- **Deal with end-to-end aspects that the real world requires**
  - e.g., not just ML models and their optimization
- **Reduce software and data engineering effort**
- **Scale our systems**
  - big data, large-scale infrastructures and high number of customers
- **Optimize the system under various constraints**
- **Offer a production-level “reliable service” for customers**

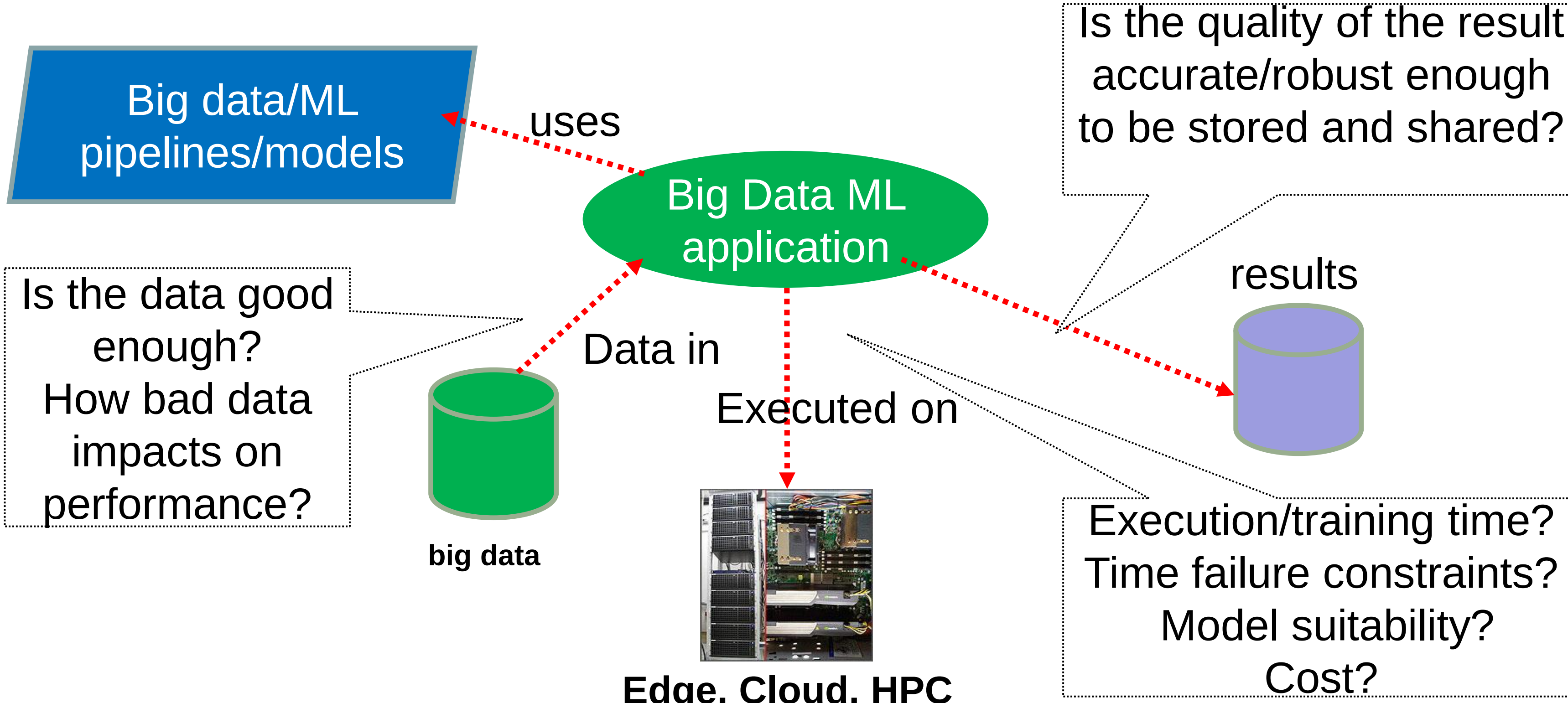
The complexity of end-to-end view

- **Engineering, optimizing and operating big data/ML systems**
  - which are key abilities that we should define, design, monitor, and measure?
  - how do we manage software artefacts, data, configuration, …?
  - how to enable flexibility and execution management?
  - how to prepare for "future"/"emerging" infrastructures?
  - which are tools and frameworks that help reducing engineering complexity?

Key areas in our concerns

- **Software development**
  - testing, experimenting, benchmark, optimization, cost management
- **Resource management**
  - execution atop multiple computing frameworks suitable for ML, such as Clouds, Supercomputing, edge, …
- **(Runtime) Ability/Quality Assurance**
  - specification, monitoring and assurance of performance, availability, costs, reliability, etc.

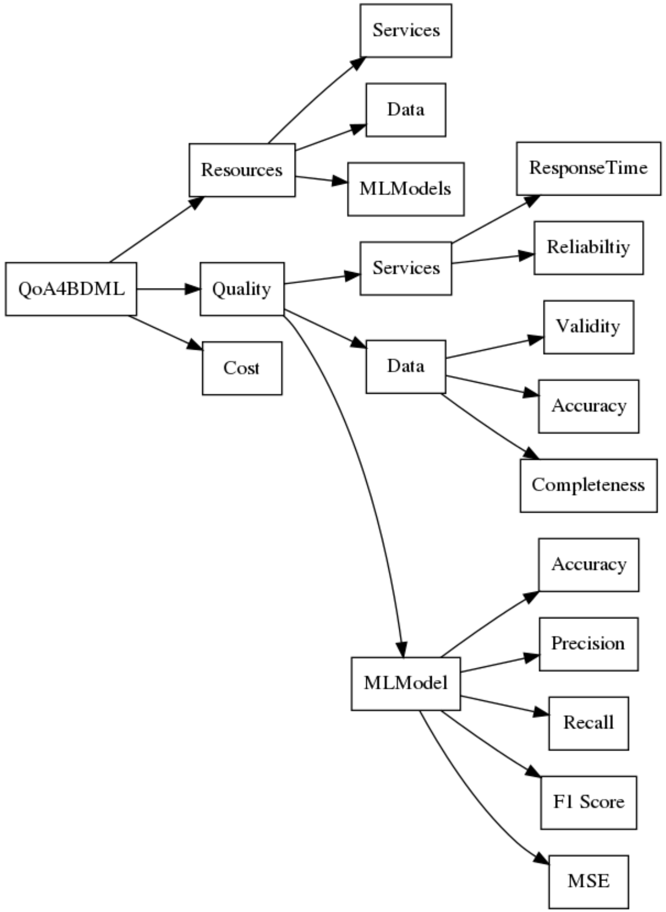
Quality of Analytics (QoA)



QoA = {quality of result, performance, cost}

Key attributes/indicators

Just example, can be more!



Our focus – R3E

- **Robustness**
  - ability to cope with errors
- **Reliability**
  - ability to function according to the indented specification (in a proper way)
- **Resilience**
  - “ability to provide the required capability in the face of adversity”(https://www.sebokwiki.org/wiki/System_Resilience)
- **Elasticity**
  - ability to stretch and return to normal forms (under external forces)

Robustness

- **In Machine Learning**
  - overfitting/underfitting
  - transfer learning
  - machine learning in an open-world
    - *how to deal with OOD (out-of-distribution) situations?*
  - when we can decide to stop training if performance/robustness does not improve?
- **In Big Data**
  - how to deal with erroneous and bad data?

Reliability

- **System reliability versus “reliable service” (from customer/business/production view)**
- **System reliability**
  - reliable infrastructures, components, networks, ...
- **“Reliable service” → reliable data analysis/inference**
  - without failure, with specified performance
- **Some hard problems**
  - have good and enough data, clean data
  - robust pipelines without degraded performance and accuracy

Resilience

- **Common issues in resilience**
  - distributed software and systems bugs
  - system attacks
- **Some specific issues in big data/ML systems**
  - bias in data
  - well-known problems in adversary attacks in ML phases

Elasticity

- **Add and remove resources**
  - CPUs, memory, data, networks, ...
- **Dynamic changes of algorithms**
- **Shift computation between edge and cloud infrastructures dynamically**
  - cloud data centers, edge systems and edge-cloud systems
- **Add/remove data to improve performance**
- **Hyperparameter tuning tradeoffs**

Short summary

Attributes	Cases from big data view	Cases from machine learning view
Robustness	deal with erroneous and bad data [45], data processing job robustness	dealing with imbalanced data, learning in an open-world (out of distribution) situations [23, 34, 35]
Reliability	reliable data sources, support of quality of data [28, 46], reliable data services [26], reliable data processing workflows/tasks [47]	reliable learning and reliable inference in terms of accuracy and reproducibility of ML models [22, 34]; uncertainties/-confidence in inferences; reliable ML service serving
Resilience	software bugs, infrastructural resource failures, fault-tolerance and replication for data services and processing [44]	bias in data, adversary attacks in ML [25], resilience learning [14], computational Byzantine failures [8]
Elasticity	utilizing different data resources, increasing and decreasing data usage w.r.t. volume, velocity, quality; elasticity of underling resources for data processing [42]	elasticity of resources for computing [19, 21, 24], elasticity of model parameters; performance loss versus model accuracy; elastic model services for performance

Table 1: R3E with big data and ML concerns

Source: https://www.researchgate.net/publication/341762862_R3E_-An_Approach_to_Robustness_Reliability_Resilience_and_Elasticity_Engineering_for_End-to-End_Machine_Learning_Systems

Do we need to treat

Robustness, Reliability, Resilience and Elasticity

equally in all your design?

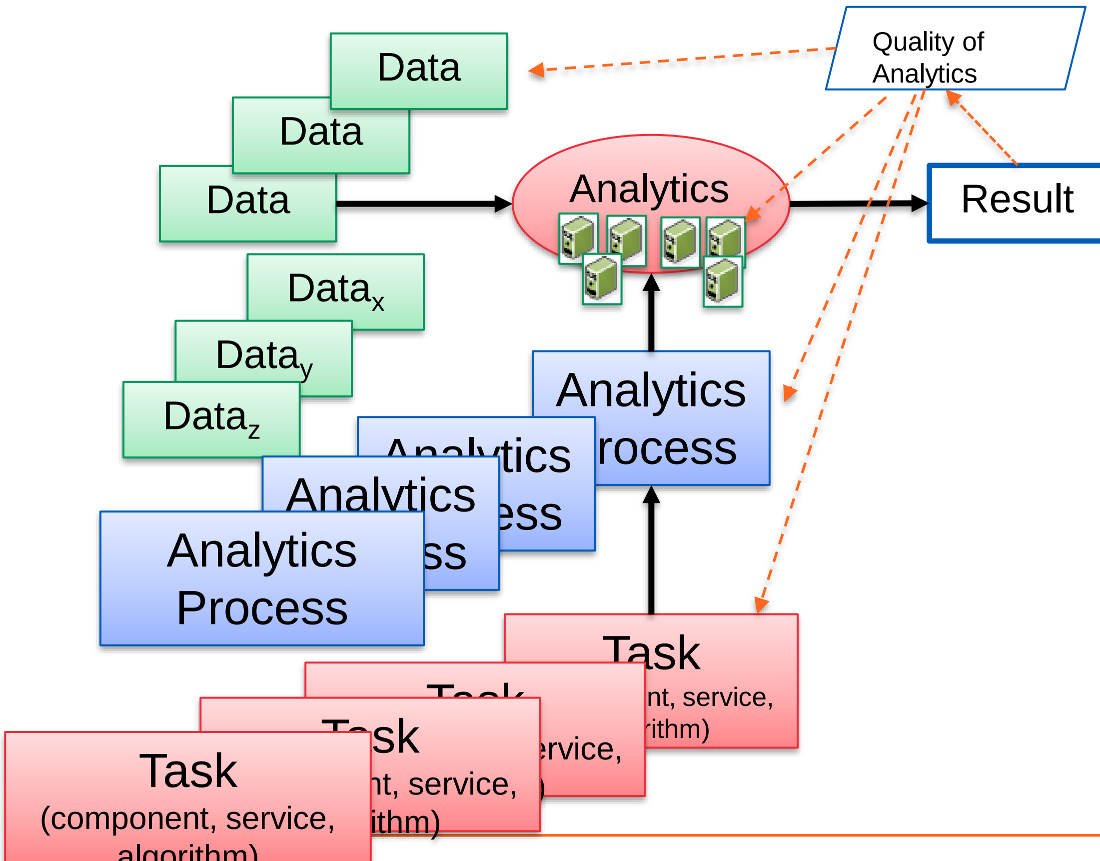
from which views?

An Approach with Elasticity Principles for R3E

Elasticity

- **Demand elasticity**
  - elastic demands from consumers
- **Output elasticity**
  - multiple outputs with different price, quantity and quality
- **Input elasticity**
  - elastic data inputs, e.g., deal with increasing data sources
- **Elastic pricing and quality models associated resources**
  - CPU/GPU, memory/disk, networks, etc.

Elasticity in (big) data analytics

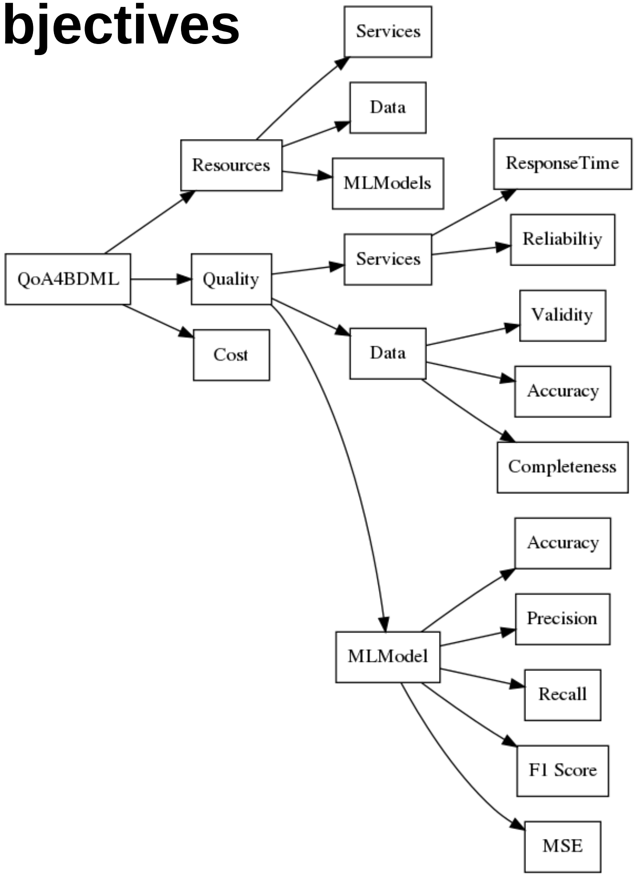


- **More data → more compute resources (e.g. more VMs)**
- **More types of data → more, different tasks → more analytics processes**
- **Change quality of analytics**
  - Change quality of data
  - Change response time
  - Change cost
  - Change types of result (form of the data output, e.g. tree, table, story)

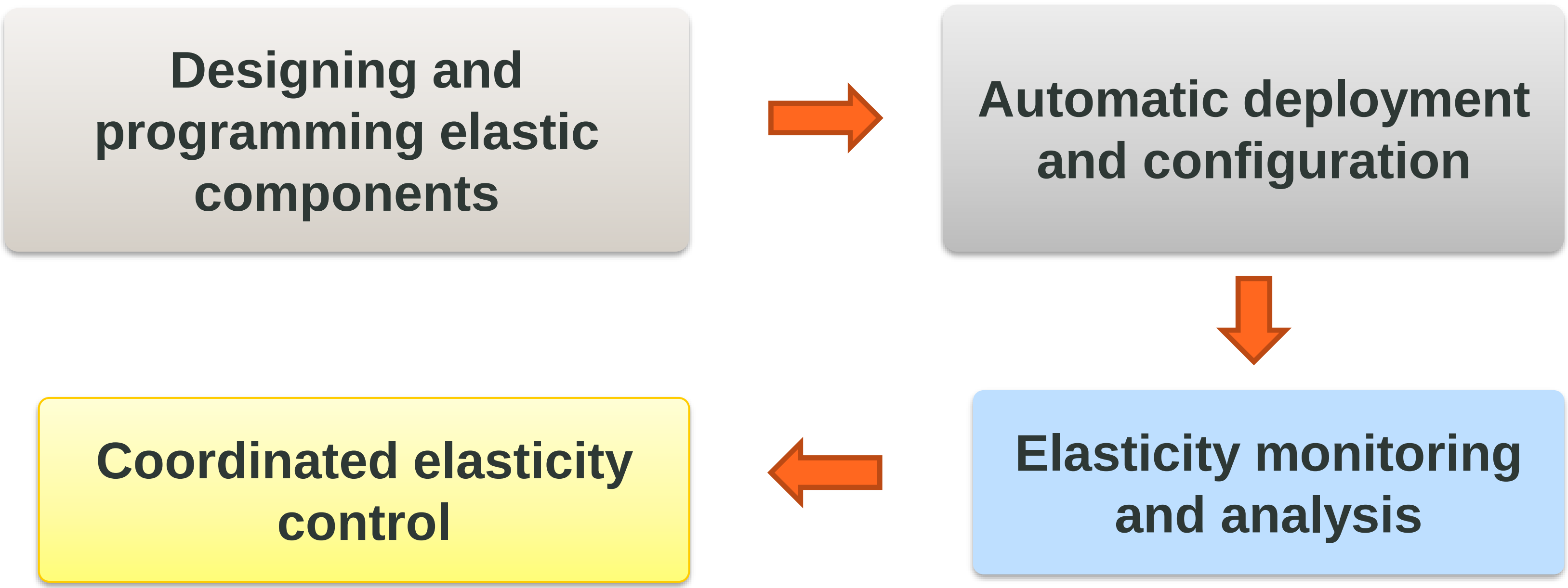
Establish quality of analytics for Big Data/ML

Possible indicators/objectives

- Have clear indicators/objectives so we can establish SLA for Quality of Analytics
- You can build your own dimensions



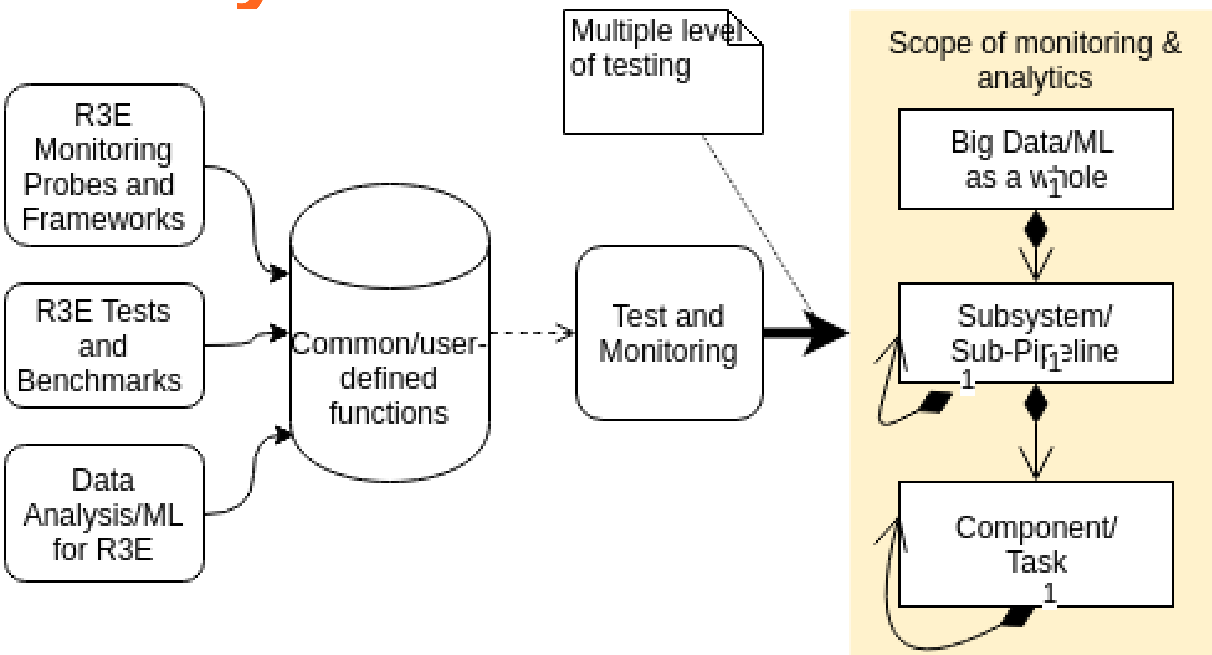
Elasticity engineering



Elasticity engineering for ML

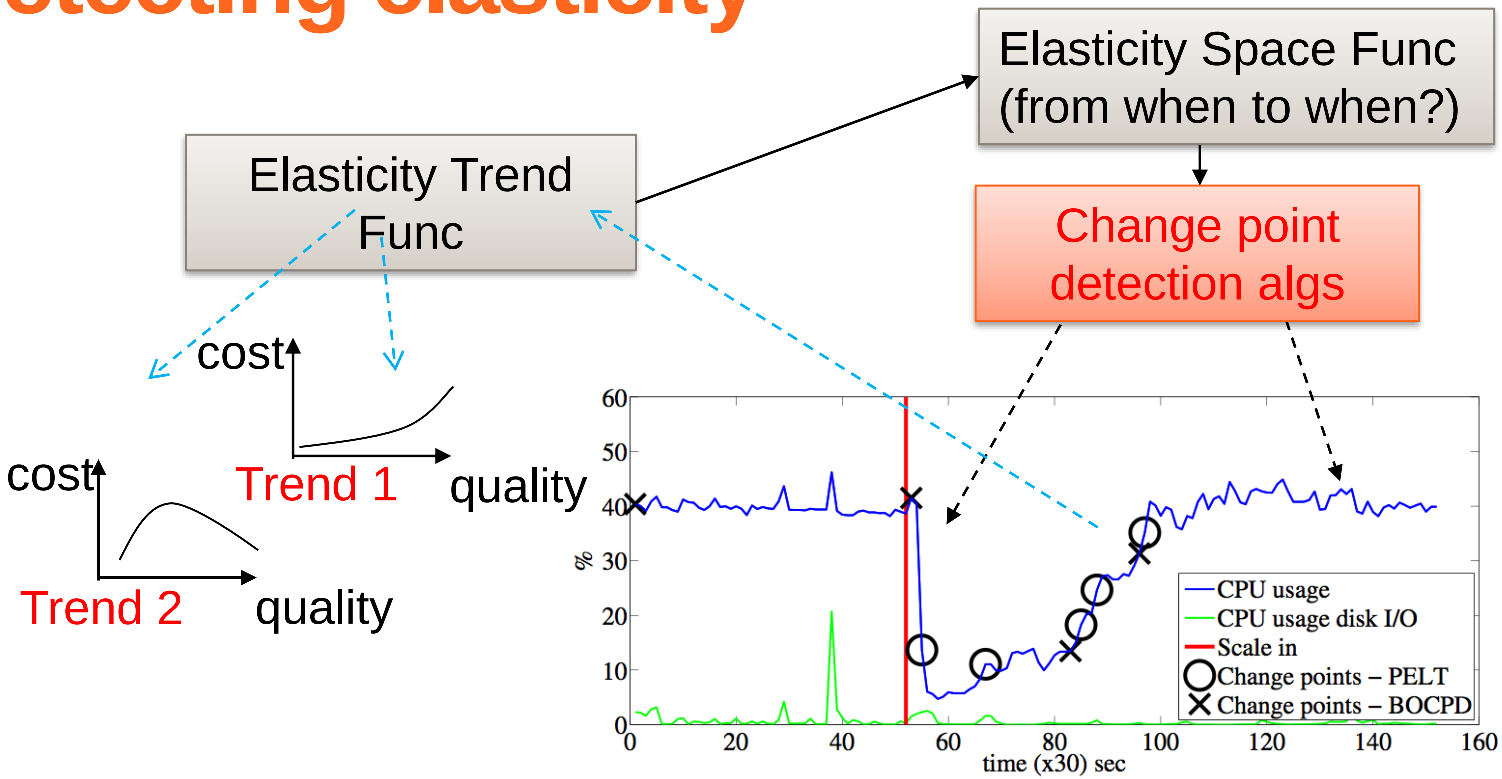
- **Conceptualizing and modeling elastic objects**
  - ML models, computing resources, data and QoA metrics
- **Defining and capturing elasticity primitive operations**
  - change resources, QoA metrics, model parameters, input data
- **Programming features for elastic objects**
  - with ML flows, coordinating QoA adjustment, dynamic serving models
- **Runtime deploying, control, and monitoring techniques for elastic objects**

Multi-level cross platforms monitoring and analysis



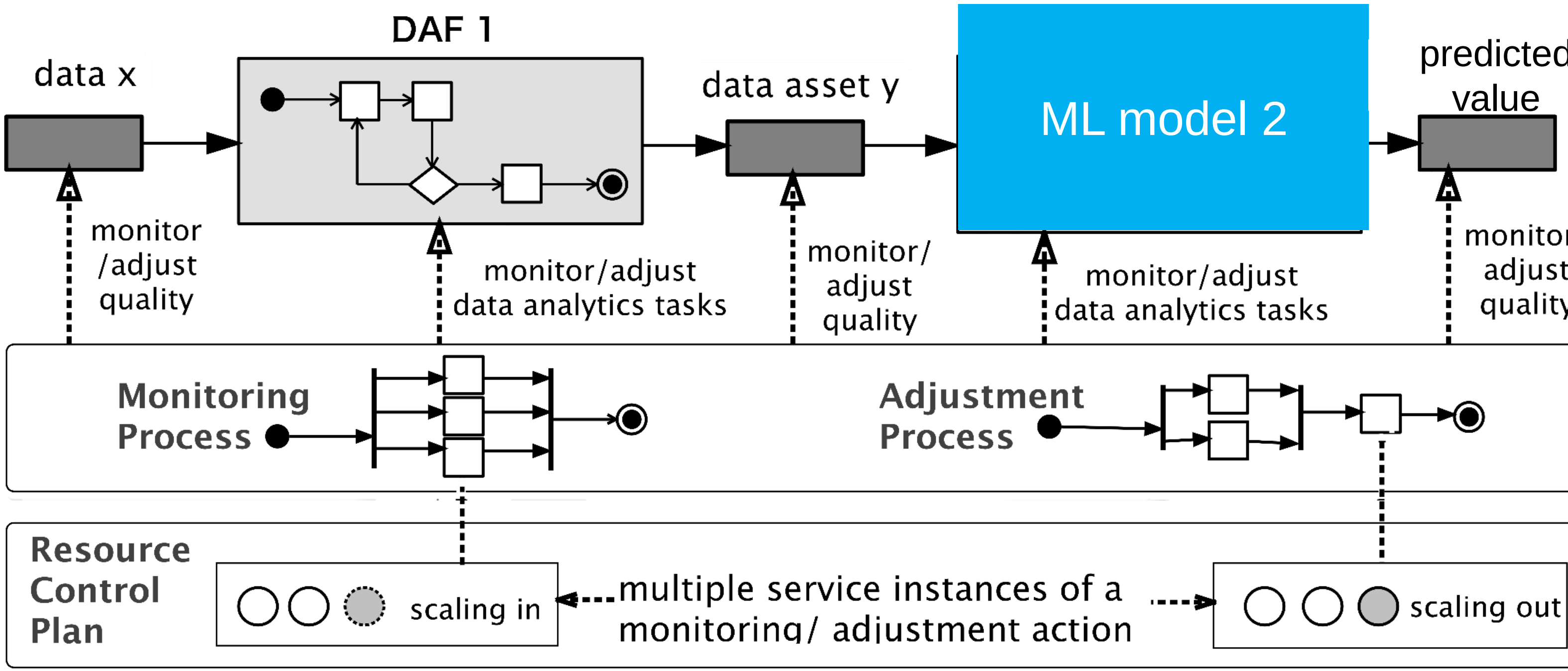
We will have a hands-on on observability and monitoring

Detecting elasticity



Alessio Gambi, Daniel Moldovan, Georgiana Copil, Hong Linh Truong, Schahram Dustdar: On estimating actuation delays in elastic computing systems. SEAMS 2013: 33-42

Using control process to ensure QoA



Will be covered in the hands-on on elastic ML serving

Some examples/results

With results from:

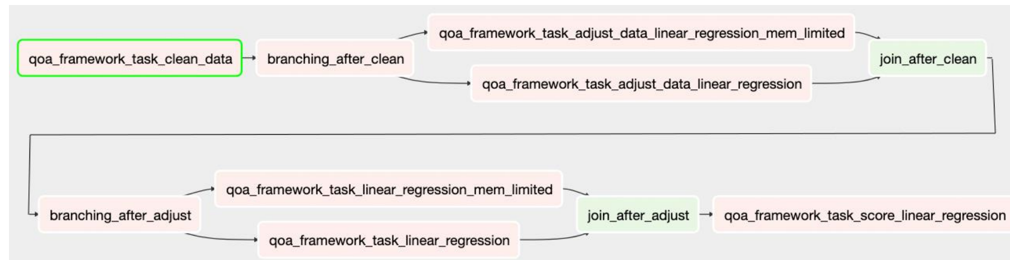
- Kreics Krists, „*Quality of analytics management of data pipelines for retail forecasting*,“, Aalto Master thesis, 2019, https://aaltodoc.aalto.fi/handle/123456789/39908
- Minjung Ryu, „*Machine Learning-based Classification System for Building Information Models* “, Aalto Master thesis, 2020
- Minjung Ryu, Linh Truong, Matti Kannala „*Understanding Quality of Analytics Tradeoffs in an End-to-End Machine Learning-based Classification System for Building Information Modeling*”, 2020, Working paper.
- Matt Baughman, Nifesh Chakubaji, Hong-Linh Truong, Krists Kreics, Kyle Chard, Ian Foster, *Measuring, Quantifying, and Predicting the Cost-Accuracy Tradeoff,* IEEE International Workshop on Benchmarking, Performance Tuning and Optimization for Big Data Applications, IEEE BigData 2019, https://research.aalto.fi/files/38801332/paper.pdf

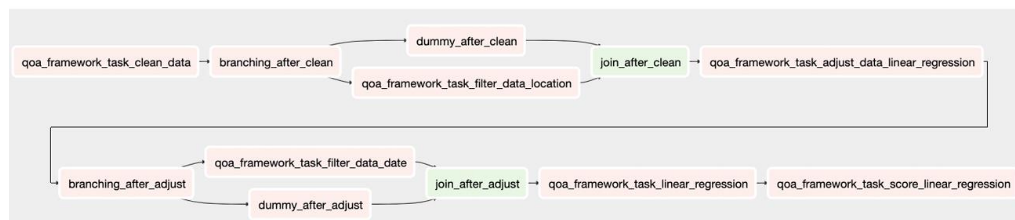
Industrial retail forecast (with Sellforte)

Forecast where to put marketing information, example of data

date	id	name	volume	price	cost	promo	category_net	margin	category1	category2	location	sales
07/01/2018	100	Chicken	38144.0	3.79	2.7	0	451692.0	0.25	Meat	Food	Helsinki	144565.76
14/01/2018	100	Chicken	36420.0	3.79	2.66	0	414342.0	0.25	Meat	Food	Helsinki	138031.8
21/01/2018	100	Chicken	35322.0	3.79	2.66	0	381854.0	0.25	Meat	Food	Helsinki	133870.38

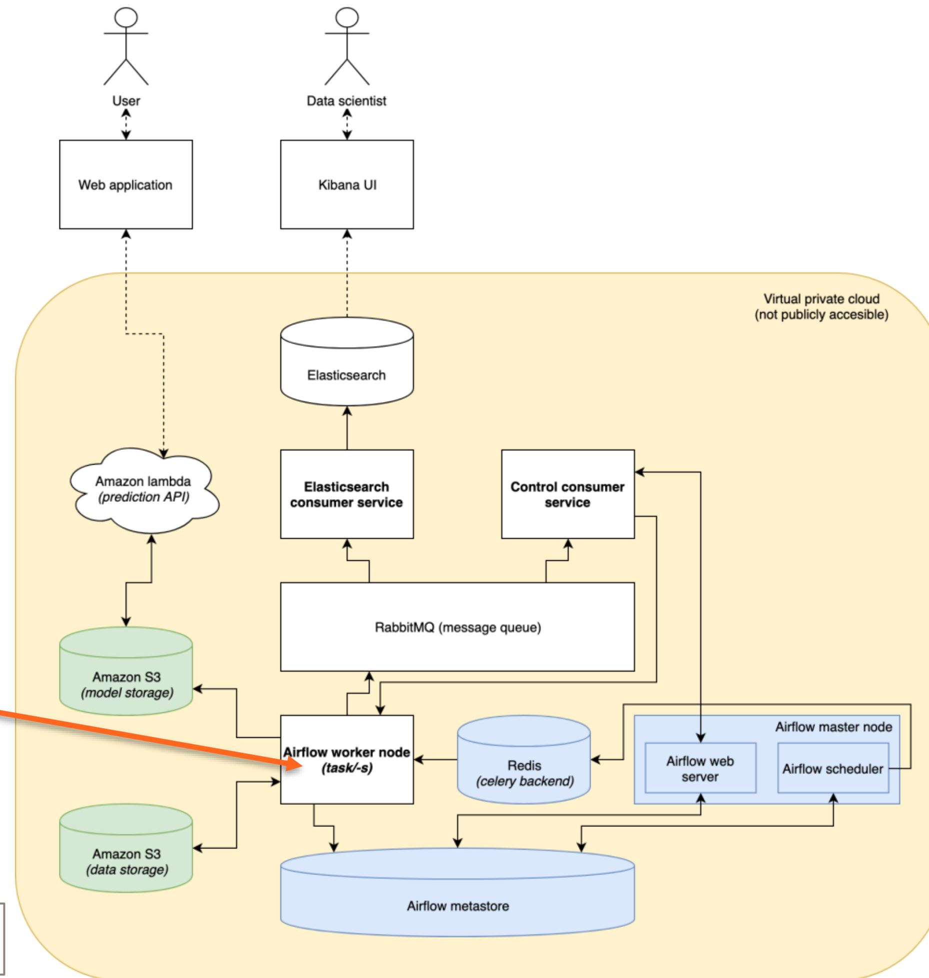
- **Metrics:**
  - data size, R square value, time, and cost
- **Pipelines**
  - tune pipelines with QoA primitive actions





Source: Kreics Krists, „Quality of analytics management of data pipelines for retail forecasting,“, Aalto CS Master thesis, 2019

Industrial retail forecast (with Sellforte)



Monitoring various indicators, including user-defined quality of data

Source: Kreics Krists, „Quality of analytics management of data pipelines for retail forecasting“, Aalto CS Master thesis, 2019

Initial results

Custom cost function

```python
def get_fargate_metrics_object(cpu, ram, elapsed_time, previous_result):
    # Fargate service cost per second
    FARGATE_CPU_COST = 0.04048 / 60 / 60
    FARGATE_RAM_COST = 0.004445 / 60 / 60
    if previous_result and 'cost_usd' in previous_result:
        cpu_cost = previous_result['cost_cpu'] + FARGATE_CPU_COST
        ram_cost = previous_result['cost_ram'] + (ram['used']/1024/1024/1024) * FARGATE_RAM_COST
    else:
        cpu_cost = FARGATE_CPU_COST
        ram_cost = (ram['used']/1024/1024/1024) * FARGATE_RAM_COST

    return { 'cost_cpu': cpu_cost, 'cost_ram': ram_cost, 'cost_usd': ram_cost + cpu_cost }
```

Custom instrumentation for model quality

```python
# model_score returns a dict -> { 'r2_squared': r2_squared_score }
model_score = score_model(store, model, data_path, preset)
pm.log_analytics_metric(model_score)
```

Source: Kreics Krists, „Quality of analytics management of data pipelines for retail forecasting“, Aalto CS Master thesis, 2019

Examples of actions in Elasticity Primitive Operations

```python
def default_get_control_action(body_dict):
    index = body_dict.pop('metric_type', None)
    print(body_dict, flush=True)
    try:
        if index == 'metrics':
            if body_dict['cost_usd'] > 1 or body_dict['time_elapsed'] > 500:
                return 'SOFT_STOP'
            elif body_dict['time_elapsed'] > 1000:
                return 'HARD_STOP'

        elif index == 'data_logs':
            if body_dict['task_name'] == 'clean_data':
                if body_dict['in']['train.csv'] / 2 > body_dict['out']['train.csv']:
                    return 'SOFT_STOP'

        elif index == 'analytics':
            if body_dict['payload']['r2_squared'] < 0.2:
                return 'SOFT_STOP'
        else:
            print('No valid index found!')
            return -1
    except KeyError:
        pass
```

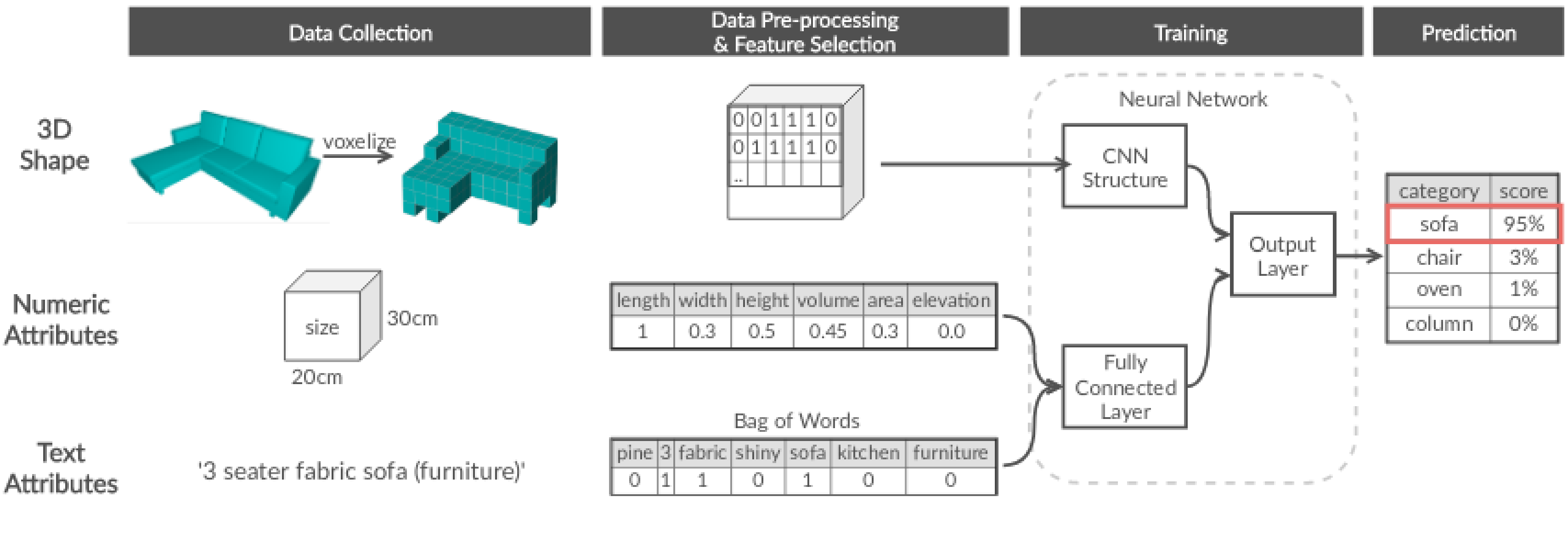
Initial results

- **Running with Airflows in Amazon EC2**
- **Apply different actions to change "store" (domain objects) and computing resources**
- **Real improvement (from the domain expert) with 1 million rows case**

  13.3% lower accuracy and 44% shorter time, R squared value was 9.5% lower → could good enough results for 50% of total store locations

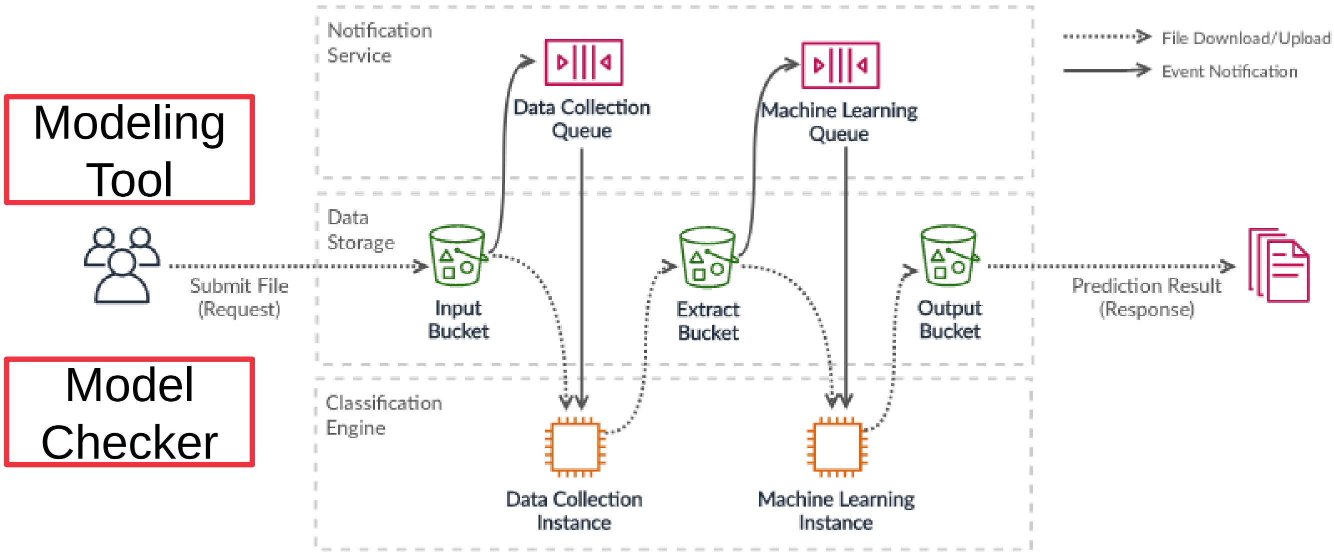
The application-aware data reduction strategy and cost-accuracy tradeoffs may be more intelligently made based on knowledge of the application domain.

ML classification for BIM (with Solibri data)



Source: Minjung Ryu, „Machine Learning-based Classification System for Building Information Models ", Aalto CS Master thesis, 2020

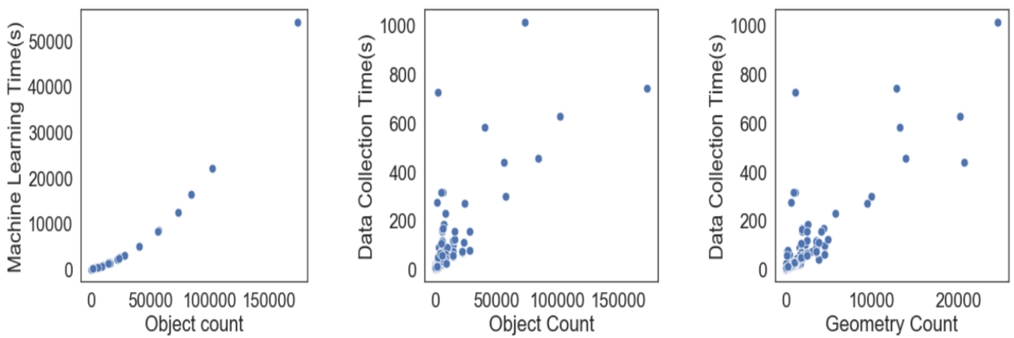
ML classification for BIM (with Solibri data)



Source: Minjung Ryu, „Machine Learning-based Classification System for Building Information Models “, Aalto CS Master thesis, 2020

Initial results

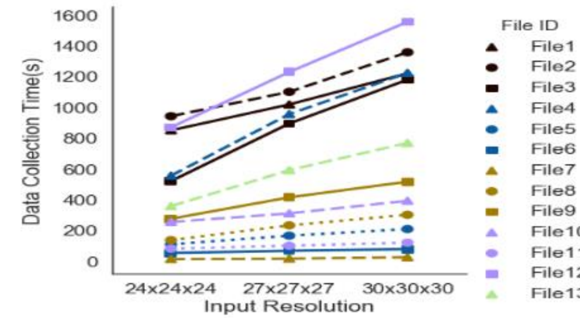
- Data set: 591 classification cases from 146 models
- Machines: AWS/Local with/out GPUs
- Different cases and settings



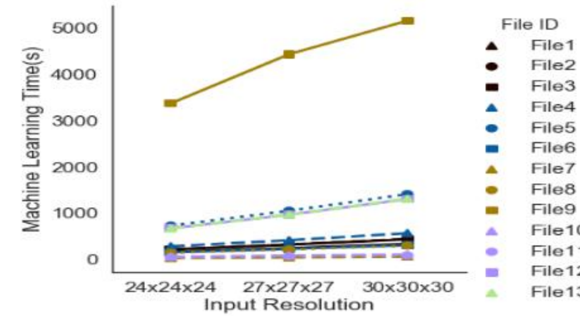
(a) Object count vs ML time (b) Object count vs DC time (c) Geometry count vs DC time

Figure 5: Impact of object counts on `DC time` and on `ML time`

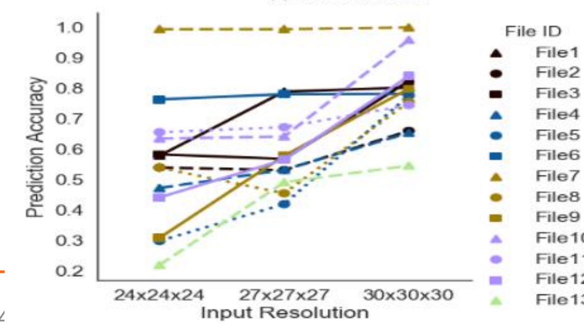
Reveal various relationships between types of data, extracting data resolution, machines and the accuracy of classifications



(a) Dim vs DC time



(b) Dim vs ML time



(c) Dim vs Accuracy

Study log for this week

Think about

- **What does it mean R3E for** *YOUR big data and machine learning systems***?**

Then

- **in your experience/work, which ones of R3E concern you most? Why? What would you do? What do you look for?**
- **~1 page – submit into the Mycourses for comments/feedback (keep it in your git)**

Thanks!

Hong-Linh Truong
Department of Computer Science

rdsea.github.io