Aalto University
School of Electrical
Engineering

# Comparison of three classic convolutional neural networks: LeNet, AlexNet, VGG-16

Kaisa Ryynänen

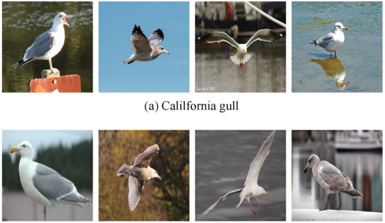
Department of Electronics and Nanoengineering
Aalto University, School of Electrical Engineering
kaisa.ryynanen@aalto.fi

1. Introduction
2. Gradient-Based Learning Applied to Document Recognition (LeNet)
3. ImageNet Classification with Deep Convolutional Neural Networks (AlexNet)
4. A Survey on Deep Learning-Based Fine-Grained Object Classification and Semantic Segmentation (AlexNet)
5. Very Deep Convolutional Networks for Large-Scale Image Recognition (VGG-16)
6. Comparison, Conclusions & Homework

# Introduction

- Convolutional neural networks (CNN) are a type of artificial neural networks
- Used e.g. in 2D structures: image classification
  - Text analysis
  - Object recognition
- Good performance in large-scale visual recognition
- Different architectures
- Classic:
  - LeNet
  - AlexNet
  - VGG-16

# Basics



(a) Calilfornia gull

(b) Glaucous gull

[3]

- Image classification is a demanding application
- Object recognition, e.g. bird species
  - High intra-class variance, small inter-class variance
  - Position and scale of the object (bird)
  - Background

# Basics

- Analysis on handwritten text
  - Large variance on handwriting styles → the same letter can look very different
  - Variance of the text position
  - Distortion, "noise", and unusual characters

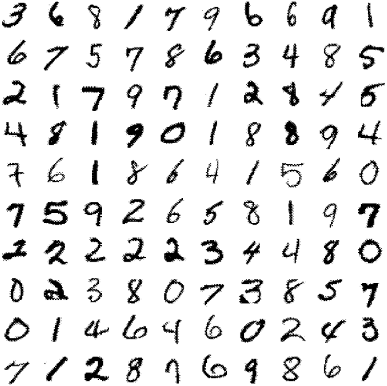


**Fig. 4.** Size-normalized examples from the MNIST database.

[1]

# Basic building blocks

- Convolutional layer
  - Performs convolution on the input data by one section at a time
  - Sum of element-wise multiplication
  - Consists of trainable filters (kernels)
  - Can e.g. detect edges in the image
  - The core of CNNs
- Pooling layer
  - Often the exact location of a feature is irrelevant
  - Groups nearby kernels together to one value
  - Max pooling
  - Average pooling
- Fully connected layers
  - A unit is connected to every unit in the next layer

# Gradient-Based Learning Applied to Document Recognition (LeNet)

Yann Lecun, Léon Bottou, Yoshua Bengio, Patrick Haffner
Proceedings of the IEEE, Vol. 86, No. 11, November 1998

# Introduction

- Typical implementation of pattern recognition, e.g. text analysis:
  - Manually written feature extractor
  - Trainable classifier
- Fully trainable module → CNN LeNet
- Graph transformer network -based check reading system with LeNet core

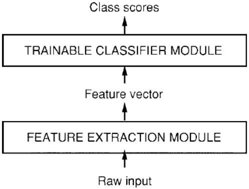


**Fig. 1.** Traditional pattern recognition is performed with two modules: a fixed feature extractor and a trainable classifier.

[1]

# Structure of LeNet-5

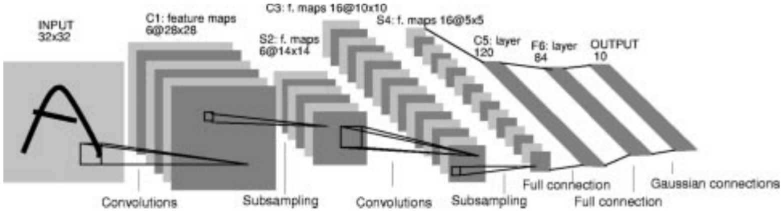


**Fig. 2.** Architecture of LeNet-5, a convolutional NN, here used for digits recognition. Each plane is a feature map, i.e., a set of units whose weights are constrained to be identical.

[1]

- Input character
- Units in one plane have the same weights, units perform the same operation in different parts of the input
- Convolutional layer consists of several feature maps with different weights → Different features are extracted

# Structure of LeNet-5

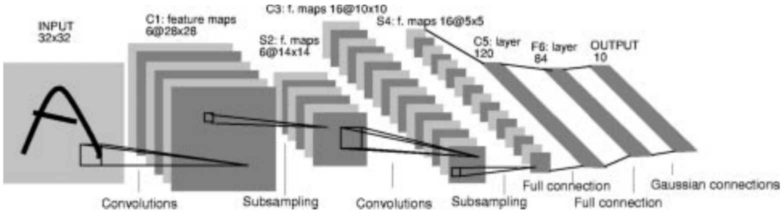


**Fig. 2.** Architecture of LeNet-5, a convolutional NN, here used for digits recognition. Each plane is a feature map, i.e., a set of units whose weights are constrained to be identical.

[1]

- Convolutions with 5x5 filters, subsampling with 2x2 windows
- Subsampling (pooling): local averaging → reduces the sensitivity to e.g. distortions
- Fully connected layers

# Structure of LeNet-5

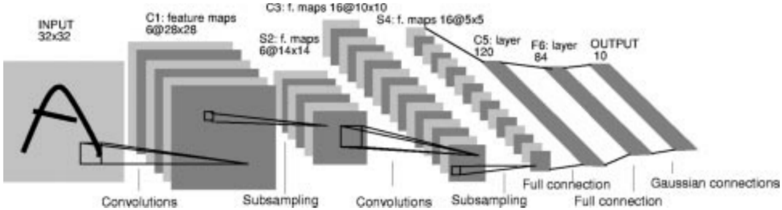


**Fig. 2.** Architecture of LeNet-5, a convolutional NN, here used for digits recognition. Each plane is a feature map, i.e., a set of units whose weights are constrained to be identical.

[1]

- Output layer with Euclidean radial basis function units

$$y_i = \sum_j (x_j - w_{ij})^2 \tag{1}$$

- Measures the fit between the input pattern and the class associated with the RBF

# Structure of LeNet-5

- Gradient-based learning
  - Loss function tries to find as close F6 configuration as possible to the RBF parameter vector of the corresponding desired class of the pattern
  - Back propagation
    $\rightarrow$ Computes the loss function gradient with respect to the weights in the network layers from the output to the input
- About 60 000 trainable parameters
  - Weight sharing

# Check reading system

- GTN based
  - Consists of different modules
  - Graphs as inputs and outputs
- Recognition transformer uses LeNet-5
- Finds most likely fields for the check amount, reads and analyzes them
  → Recognizes the check amount

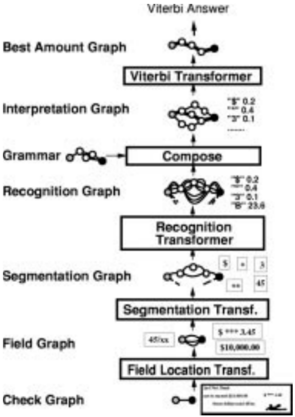


**Fig. 33.** A complete check amount reader implemented as a single cascade of GT modules. Successive graph transformations progressively extract higher level information.

[1]

# Check reading system

- Field location transformer
  - Extracts the fields which may contain the check amount
  - Penalty term: close to zero for likely candidates
- Segmentation transformer
  - Cuts each field into segments containing a whole or a part of a character
  - Penalty term: probability that the segment contains a character

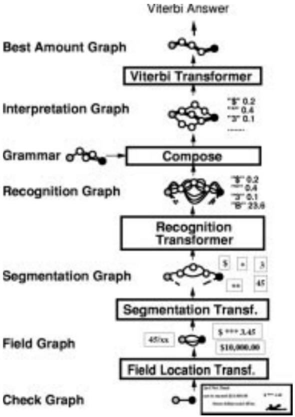


**Fig. 33.** A complete check amount reader implemented as a single cascade of GT modules. Successive graph transformations progressively extract higher level information.

[1]

# Check reading system

- Recognition transformer
  - LeNet-5 based
  - Recognizes and classifies the segments into characters
  - Outputs the recognized classes
  - Penalty term: sum of the previous transformer term and the probability of the character belonging in the recognized class

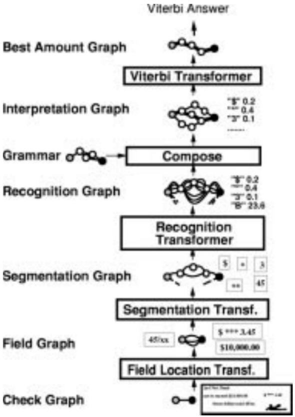


**Fig. 33.** A complete check amount reader implemented as a single cascade of GT modules. Successive graph transformations progressively extract higher level information.

[1]

# Check reading system

- Composition transformer
  - Grammar graph includes all possible options for check amounts
  - Selects the likely candidates for the check amount
  - Penalty term: the sum of the recognizer penalty and the arc penalty in the grammar graph
- Viterbi transformer
  - Selects the lowest penalty → The most probable candidate for the check amount

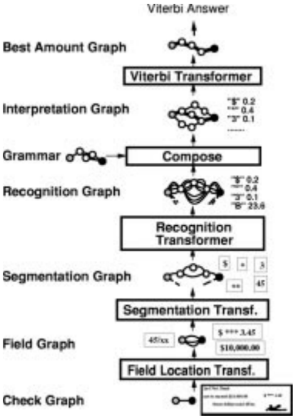


**Fig. 33.** A complete check amount reader implemented as a single cascade of GT modules. Successive graph transformations progressively extract higher level information.

[1]

# Results

- System was tested on machine-printed business checks
- The classifier was first trained on 500 000 character images with both handwritten and machine-printed characters
- From 646 business checks:
  - 82% correctly recognized
  - 1% errors
  - 17% rejected

# ImageNet Classification with Deep Convolutional Neural Networks (AlexNet)

Alex Krizhevsky, Ilya Sutskever, Geoffrey E. Hinton
NIPS 2012

# Introduction

- Object recognition with CNNs
- Large set of training data requires large learning capacity
- CNN tested with ImageNet dataset containing over 15 million high-resolution images
- Images have variable resolutions → down-sampling

# Architecture

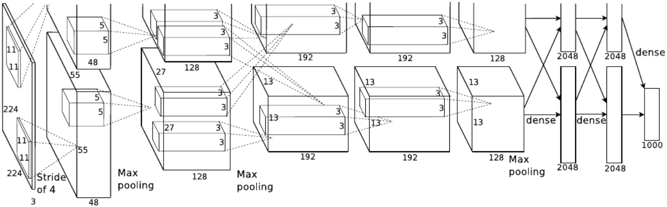


Figure 2: An illustration of the architecture of our CNN, explicitly showing the delineation of responsibilities between the two GPUs. One GPU runs the layer-parts at the top of the figure while the other runs the layer-parts at the bottom. The GPUs communicate only at certain layers. The network's input is 150,528-dimensional, and the number of neurons in the network's remaining layers is given by 253,440–186,624–64,896–64,896–43,264–4096–4096–1000.

[2]

- Two GPUs
- Eight trainable layers
- Five convolutional, three fully-connected
- 60 million parameters

# Overfitting

- Generally means that the training dataset accuracy is larger than the testing dataset accuracy
- Data augmentation
  → Enlarge the training dataset artificially
- Dropout
  → Set the output of a neuron to zero with a probability of 0.5

# Results

- Tested with ImageNet data
- A subset of about 1000 images in 1000 categories
- Entered to ILSVRC-2010 and -2012 competitions
- Top-1 error rate: wrong classification
- Top-5 error rate: the correct answer is not included in the top-5 class guesses
- 2010:
  - Top-1 error rate 37.5%
  - Top-5 error rate 17.0%
- 2012:
  - Top-1 40.7%
  - Top-5 18.2%

# A Survey on Deep Learning-Based Fine-Grained Object Classification and Semantic Segmentation (AlexNet)

Bo Zhao, Jiashi Feng, Xiao Wu, Shuicheng Yan
International Journal of Automation and Computing, April 2017

# Introduction

- Comparison of AlexNet, VGGNet, and GoogLeNet using different methods
- Part detection and alignment
  - Identifies and isolates specific parts from the images
  - E.g. Identifying characteristics of a bird
- Ensemble of networks based approches
  - Divides the dataset to multiple similar subsets
  - Multiple neural networks
- Visual attention based approaches
  - Finds the essential parts of the dataset

# Results

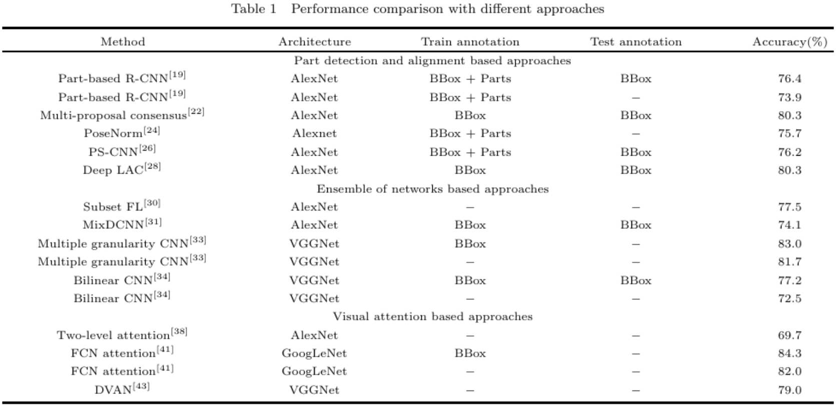


Table 1 Performance comparison with different approaches

| Method | Architecture | Train annotation | Test annotation | Accuracy(%) |
|---|---|---|---|---|
| | | Part detection and alignment based approaches | | |
| Part-based R-CNN[19] | AlexNet | BBox + Parts | BBox | 76.4 |
| Part-based R-CNN[19] | AlexNet | BBox + Parts | – | 73.9 |
| Multi-proposal consensus[22] | AlexNet | BBox | BBox | 80.3 |
| PoseNorm[24] | Alexnet | BBox + Parts | – | 75.7 |
| PS-CNN[26] | AlexNet | BBox + Parts | BBox | 76.2 |
| Deep LAC[28] | AlexNet | BBox | BBox | 80.3 |
| | | Ensemble of networks based approaches | | |
| Subset FL[30] | AlexNet | – | – | 77.5 |
| MixDCNN[31] | AlexNet | BBox | BBox | 74.1 |
| Multiple granularity CNN[33] | VGGNet | BBox | – | 83.0 |
| Multiple granularity CNN[33] | VGGNet | – | – | 81.7 |
| Bilinear CNN[34] | VGGNet | BBox | BBox | 77.2 |
| Bilinear CNN[34] | VGGNet | – | – | 72.5 |
| | | Visual attention based approaches | | |
| Two-level attention[38] | AlexNet | – | – | 69.7 |
| FCN attention[41] | GoogLeNet | BBox | – | 84.3 |
| FCN attention[41] | GoogLeNet | – | – | 82.0 |
| DVAN[43] | VGGNet | – | – | 79.0 |

[3]

# Very Deep Convolutional Networks for Large-Scale Image Recognition (VGG-16)

Karen Simonyan, Andrew Zisserman
ICLR 2015

# Introduction

- The architecture is based on VGG-16
- Focus on the architecture depth
- Increase the depth → Accuracy increases
- Used also for ILSVRC classification tasks

# Architecture

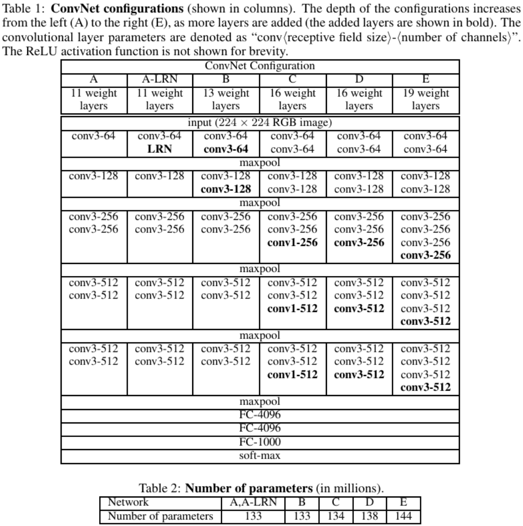


Table 1: **ConvNet configurations** (shown in columns). The depth of the configurations increases from the left (A) to the right (E), as more layers are added (the added layers are shown in bold). The convolutional layer parameters are denoted as "conv⟨receptive field size⟩-⟨number of channels⟩". The ReLU activation function is not shown for brevity.

| ConvNet Configuration | | | | | |
|---|---|---|---|---|---|
| A | A-LRN | B | C | D | E |
| 11 weight layers | 11 weight layers | 13 weight layers | 16 weight layers | 16 weight layers | 19 weight layers |
| input (224 × 224 RGB image) | | | | | |
| conv3-64 | conv3-64<br>**LRN** | conv3-64<br>**conv3-64** | conv3-64<br>conv3-64 | conv3-64<br>conv3-64 | conv3-64<br>conv3-64 |
| maxpool | | | | | |
| conv3-128 | conv3-128 | conv3-128<br>**conv3-128** | conv3-128<br>conv3-128 | conv3-128<br>conv3-128 | conv3-128<br>conv3-128 |
| maxpool | | | | | |
| conv3-256<br>conv3-256 | conv3-256<br>conv3-256 | conv3-256<br>conv3-256 | conv3-256<br>conv3-256<br>**conv1-256** | conv3-256<br>conv3-256<br>**conv3-256** | conv3-256<br>conv3-256<br>conv3-256<br>**conv3-256** |
| maxpool | | | | | |
| conv3-512<br>conv3-512 | conv3-512<br>conv3-512 | conv3-512<br>conv3-512 | conv3-512<br>conv3-512<br>**conv1-512** | conv3-512<br>conv3-512<br>**conv3-512** | conv3-512<br>conv3-512<br>conv3-512<br>**conv3-512** |
| maxpool | | | | | |
| conv3-512<br>conv3-512 | conv3-512<br>conv3-512 | conv3-512<br>conv3-512 | conv3-512<br>conv3-512<br>**conv1-512** | conv3-512<br>conv3-512<br>**conv3-512** | conv3-512<br>conv3-512<br>conv3-512<br>**conv3-512** |
| maxpool | | | | | |
| FC-4096 | | | | | |
| FC-4096 | | | | | |
| FC-1000 | | | | | |
| soft-max | | | | | |

Table 2: **Number of parameters** (in millions).

| Network | A,A-LRN | B | C | D | E |
|---|---|---|---|---|---|
| Number of parameters | 133 | 133 | 134 | 138 | 144 |

[4]

# Architecture

- ReLU activation function for all hidden layers → output is the input if it is positive
- 3x3 filters with a stride of 1
- Configuration C uses 1x1 convolutional layers
- Max pooling with 2x2 windows with a stride of 2
- Small filters → increased depth and smaller number of parameters

# Results

- ILSVRC-2012 dataset used for testing
- Tests performed with different settings
  - Single scale of the input image
  - Multiple scales of the input image
  - Different cropping of the input image
  - Combinations of different configurations

# Results: Single scale

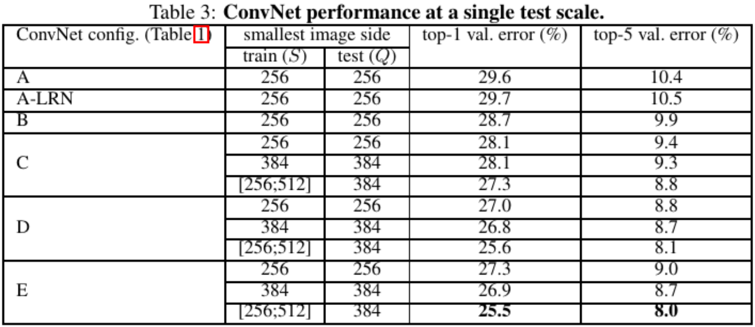


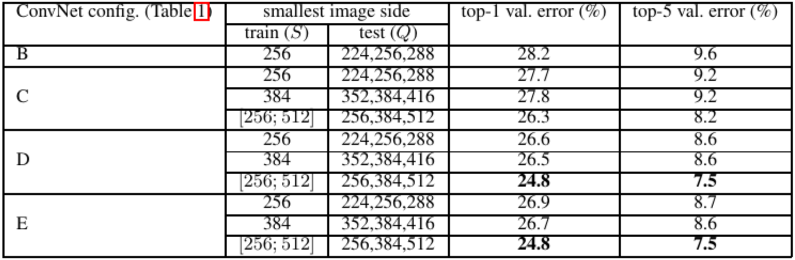
Table 3: **ConvNet performance at a single test scale.**

| ConvNet config. (Table 1) | smallest image side | | top-1 val. error (%) | top-5 val. error (%) |
|---|---|---|---|---|
| | train ($S$) | test ($Q$) | | |
| A | 256 | 256 | 29.6 | 10.4 |
| A-LRN | 256 | 256 | 29.7 | 10.5 |
| B | 256 | 256 | 28.7 | 9.9 |
| C | 256 | 256 | 28.1 | 9.4 |
| | 384 | 384 | 28.1 | 9.3 |
| | [256;512] | 384 | 27.3 | 8.8 |
| D | 256 | 256 | 27.0 | 8.8 |
| | 384 | 384 | 26.8 | 8.7 |
| | [256;512] | 384 | 25.6 | 8.1 |
| E | 256 | 256 | 27.3 | 9.0 |
| | 384 | 384 | 26.9 | 8.7 |
| | [256;512] | 384 | **25.5** | **8.0** |

[4]

# Results: Multiple scales

Table 4: **ConvNet performance at multiple test scales.**



| ConvNet config. (Table 1) | smallest image side | | top-1 val. error (%) | top-5 val. error (%) |
|---|---|---|---|---|
| | train ($S$) | test ($Q$) | | |
| B | 256 | 224,256,288 | 28.2 | 9.6 |
| C | 256 | 224,256,288 | 27.7 | 9.2 |
| | 384 | 352,384,416 | 27.8 | 9.2 |
| | [256; 512] | 256,384,512 | 26.3 | 8.2 |
| D | 256 | 224,256,288 | 26.6 | 8.6 |
| | 384 | 352,384,416 | 26.5 | 8.6 |
| | [256; 512] | 256,384,512 | **24.8** | **7.5** |
| E | 256 | 224,256,288 | 26.9 | 8.7 |
| | 384 | 352,384,416 | 26.7 | 8.6 |
| | [256; 512] | 256,384,512 | **24.8** | **7.5** |

[4]

# Comparison & Conclusions

# Comparison: Configuration

- LeNet-5
  - 5x5x6, 5x5x16, 2x2x6 and 2x2x16 kernels
  - 60 000 trainable parameters due to weight sharing
  - Average pooling
- AlexNet
  - 11x11x3, 5x5x48, 3x3x256, 3x3x192 kernels
  - 60 million parametes
  - Max pooling
- VGG-16
  - 3x3 and 2x2 kernels (and 1x1)
  - 133, 134, 138, and 144 million trainable parameters
  - Max pooling

# Comparison: Accuracy

- LeNet-5 82%
- AlexNet 62.5%, 69.7-80.3%
- VGG-16 74.5%
- Top-1 results, single scale for VGG-16
  $\rightarrow$ Comparable results
- Accuracy from the error rates: 100% - Error Rate

# Conclusions

- LeNet-5 tested in a different manner than AlexNet and VGG-16
- LeNet-5 has the best accuracy
- AlexNet ja VGG-16 have more trainable parameters
- VGG has largest depth
- Object recognition more challenging than text analysis?

# Homework Assignments

- What are the main building block of CNNs?
- What is pooling?
- What are the main differences between LeNet, AlexNet and VGG-16?

# References

1. Yann Lecun, Léon Bottou, Yoshua Bengio, Patrick Haffner, Gradient-Based Learning Applied to Document Recognition, Proceedings of the IEEE, Vol. 86, No. 11, November 1998
2. Alex Krizhevsky, Ilya Sutskever, Geoffrey E. Hinton, ImageNet Classification with Deep Convolutional Neural Networks, NIPS 2012
3. Bo Zhao, Jiashi Feng, Xiao Wu, Shuicheng Yan, A Survey on Deep Learning-Based Fine-Grained Object Classification and Semantic Segmentation, International Journal of Automation and Computing, April 2017
4. Karen Simonyan, Andrew Zisserman, Very Deep Convolutional Networks for Large-Scale Image Recognition, ICLR 2015