Foundations of Deep Learning, Winter Term 2021/22

Week 6: Convolutional Neural Networks

Convolutional Neural Networks

Frank Hutter Abhinav Valada

University of Freiburg

Overview of Week 6

Lecture Overview

Foundations of Deep Learning, Winter Term 2021/22

Week 6: Convolutional Neural Networks

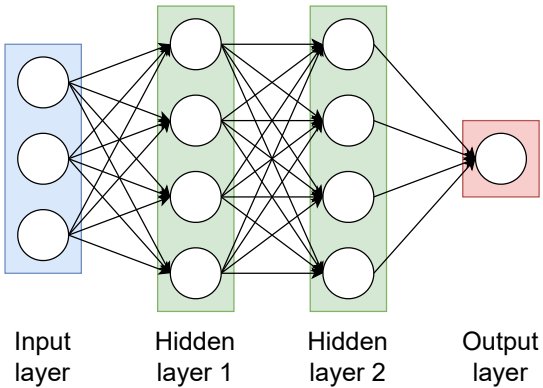
Introduction

Frank Hutter Abhinav Valada

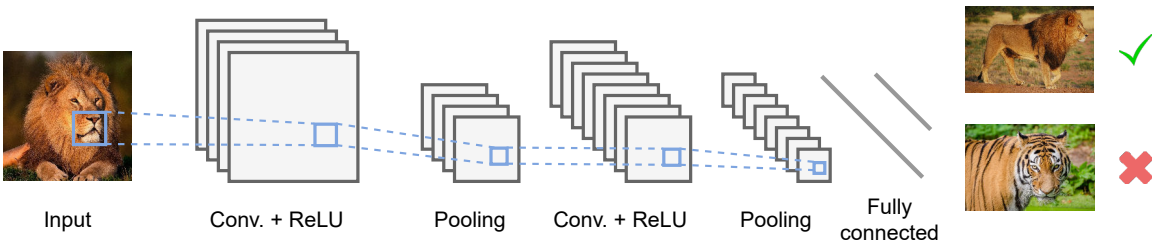
University of Freiburg

Recall: Structure of a Multilayer Perceptron

- Cascade neurons together
- Output from one layer is the input to the next
- Each layer has its own set of weights

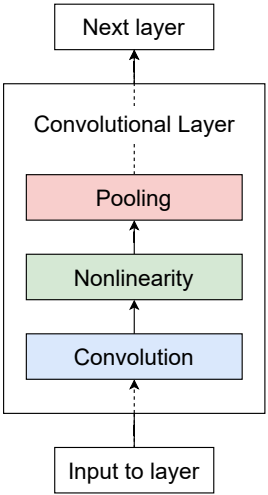


Structure of a Convolutional Neural Network



Convolutional Neural Networks

- Convolutional Neural Networks (ConvNets, CNNs) are specially structured neural networks.
- Very well suited to process data with a grid-like structure, e.g., images.
- Use convolution operation in at least one layer but usually use a stack of convolutional layers (with optional pooling).
- Tremendously successful in practical applications.



Today: ConvNets Applications Everywhere

Image classification and object recognition via deep convolutional networks

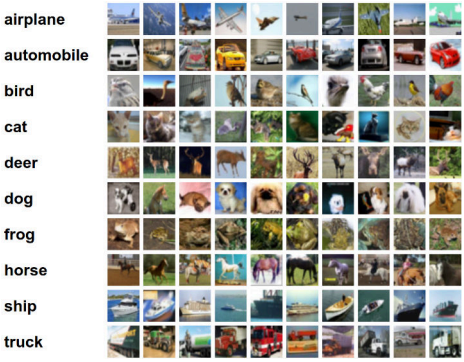


Figure: The CIFAR-10 dataset.

[Image source: https://www.cs.toronto.edu/~kriz/cifar.html]

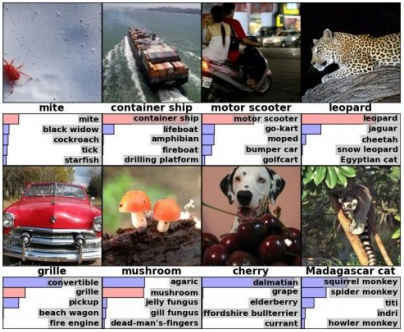


Figure: Results on ILSVRC-2010 test images from AlexNet. [Krizhevsky et al., 2012]

Today: ConvNets Applications Everywhere

Object detection and segmentation with ConvNets

Figure: Results on PASCAL VOC 2007 test from Faster R-CNN. [Ren et al., 2016]

Figure: Results on PASCAL VOC 2012 from DeepLabv3+. [Chen et al., 2018]

Today: ConvNets Applications Everywhere

Research in Freiburg: simultaneous prediction of semantic segmentation and instance masks → panoptic segmentation

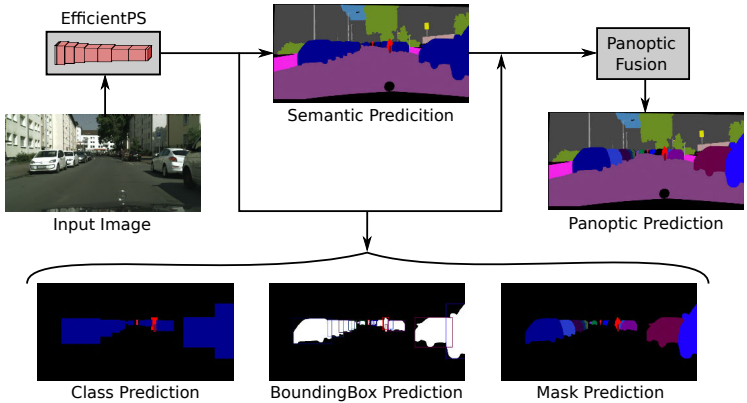


Figure: Results on Cityscapes from EfficientPS. [Mohan and Valada, 2021]

Today: ConvNets Applications Everywhere

ConvNets are used in various stages of autonomous driving software stacks, e.g.

Figure: Object detection.

[Qi et al., 2018]

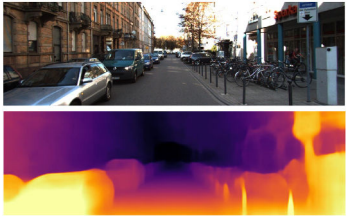


Figure: Depth estimation.

[Watson et al., 2021]

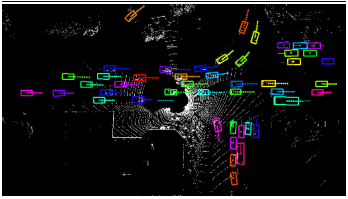


Figure: Motion prediction.

[Luo et al., 2018]

Today: ConvNets Applications Everywhere

Image captioning connecting computer vision and natural language processing

LSTM: a bear is standing on a rock in a zoo
CNN: two bears are walking on a rock in the zoo
GT: two bears touching noses standing on rocks

LSTM: a box of donuts with a variety of toppings
CNN: a box of doughnuts with sprinkles and a sign
GT:A bunch of doughnuts with sprinkles on them

Figure: Results from [Aneja et al., 2018].

Figure: Results from [Vinyals et al., 2015].

Today: ConvNets Applications Everywhere

Human pose estimation with ConvNets

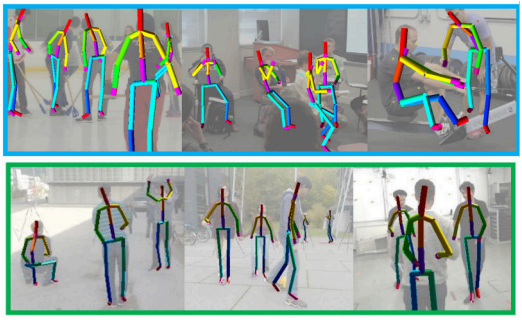


Figure: 3D pose estimation from mono RGB.

[Mehta et al., 2018]

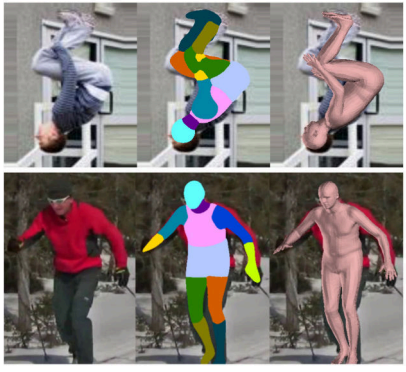


Figure: Fitting a 3D body model. [Omran et al., 2018]

Today: ConvNets Applications Everywhere

ConvNet-based weather forecasts

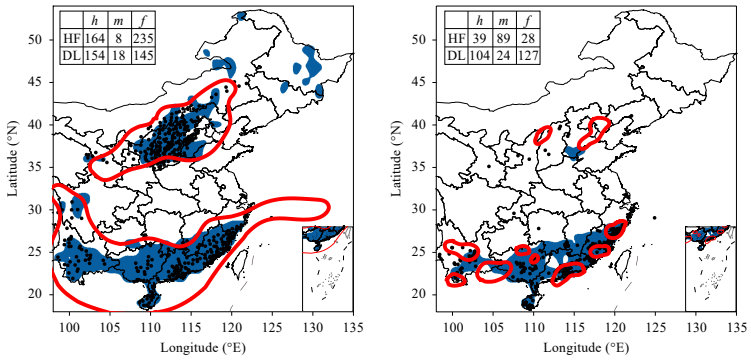


Figure: CNN-based forecasts of thunderstorm (left) and heavy rain (right) illustrated in blue. Real observations are shown by black dots. Red lines depict forecasts by meteorologists. [Zhuo et al., 2019]

Today: ConvNets Applications Everywhere

Learning to play games with ConvNets in a reinforcement learning setting

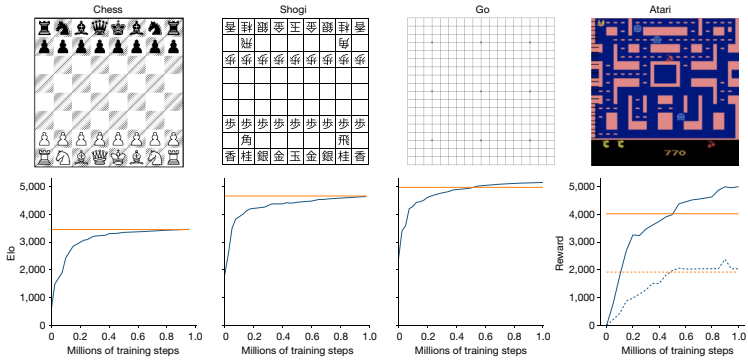


Figure: Improvement of MuZero while training (blue). AlphaZero's performance is shown in orange.

[Schrittwieser et al., 2020]

Today: ConvNets Applications Everywhere

Research in Freiburg: using reinforcement learning to teach robots by demonstration

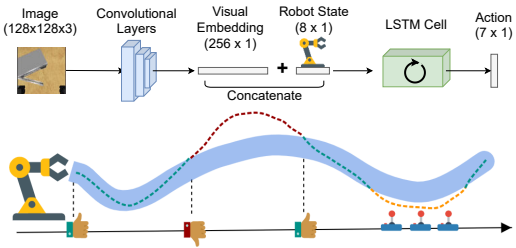


Figure: (top) Model architecture. (bottom) The robot receives positive or negative feedback by a human teacher correcting the trajectory.

[Chisari et al., 2021]

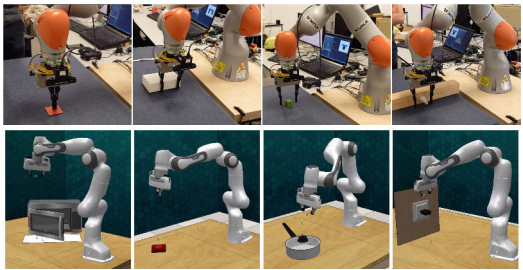


Figure: Overview of the tasks applied both in real world (top) and simulation (bottom).

[Chisari et al., 2021]

Today: ConvNets Applications Everywhere

Research in Freiburg: real-time optical flow estimation

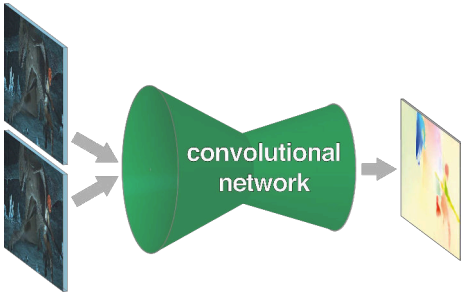


Figure: Architecture of FlowNet. [Dosovitskiy et al., 2015]

Figure: Results on Sintel from FlowNet2.

[Ilg et al., 2017]

Questions to Answer for Yourself / Discuss with Friends

- Repetition: Are CNNs only effective for learning from images or can they also learn directly from other modalities, e.g., raw sound waveforms, that do not have a natural spatial structure?

Lecture Overview

Foundations of Deep Learning, Winter Term 2021/22

Week 6: Convolutional Neural Networks

Historical Context

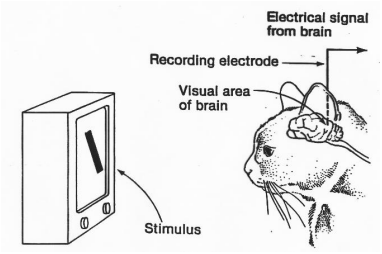
Frank Hutter Abhinav Valada

University of Freiburg

Historical Context and Inspiration from Neuroscience

Hubel and Wiesel (Nobel prize 1981) found in several studies in the 1950s and 1960s that:

- The visual cortex has feature detectors (among others, cells with preference for edges with specific orientation).
- Simple cells act as local feature detectors.
- Complex cells pool the responses of simple cells.
- There is a feature hierarchy.

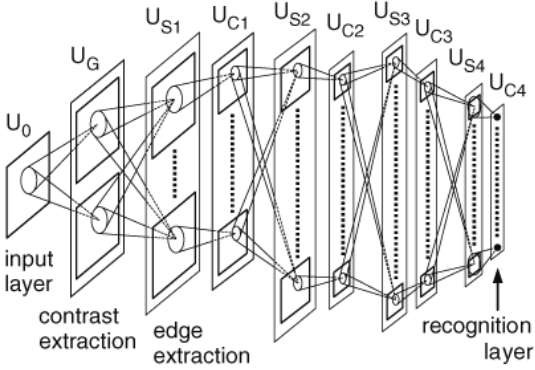


[Image source: Nguyen et al., 2019]

Fukushima's Neocognitron

Kunihiko Fukushima (1980) built a hierarchical computational model based on insights by Hubel and Wiesel:

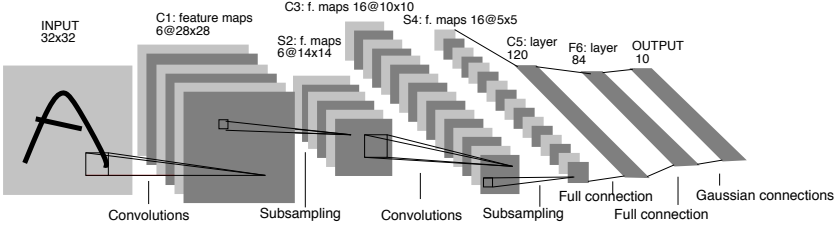
- Simple cells in layers U_S have adjustable parameters.
- Complex cells in layers U_C perform pooling.



[Image source: http://www.scholarpedia.org/article/Neocognitron]

LeNet-5

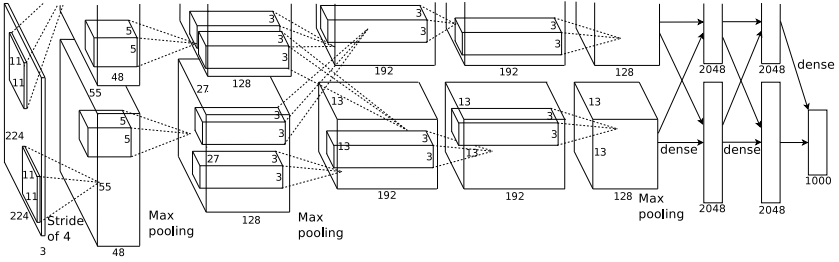
- Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner presented LeNet-5 in 1998.
- Built on Fukushima's work but used backpropagation to train the model.
- Very successfully used in practical application reading digits from handwritten checks in the US.



[Image source: LeCun et al., 1998]

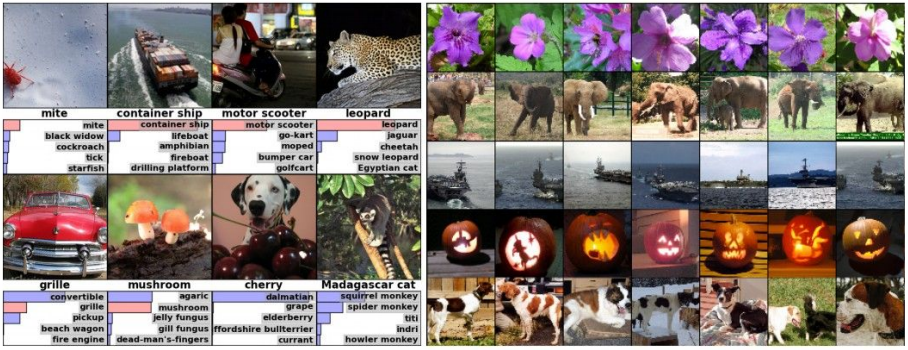
AlexNet

- In 2012, a ConvNet model by A. Krizhevsky, I. Sutskever, and G. Hinton outperformed all other models on the challenging ImageNet benchmark (1.2M images, 1K classes).
- Scaled up version of LeNet 5.
- Was trained on 2 GPUs using ReLUs, dropout regularization, and data augmentation.
- Led to a revolution in the field of computer vision!



[Image source: Krizhevsky et al., 2012]

AlexNet – Results



[Image source: Krizhevsky et al., 2012]

Questions to Answer for Yourself / Discuss with Friends

- Repetition: What was the first neural network to use convolutional layers?
- Repetition: What was the first 2D convolutional neural network that was trained using backpropagation?

Lecture Overview

Foundations of Deep Learning, Winter Term 2021/22

Week 6: Convolutional Neural Networks

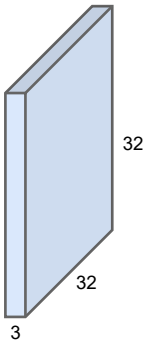
Convolutions

Frank Hutter Abhinav Valada

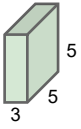
University of Freiburg

Convolutions Illustrated

- Convolution of the input using the filter, i.e., slide over the image spatially by computing dot products.
- The filters always match the depth channel of the input data.



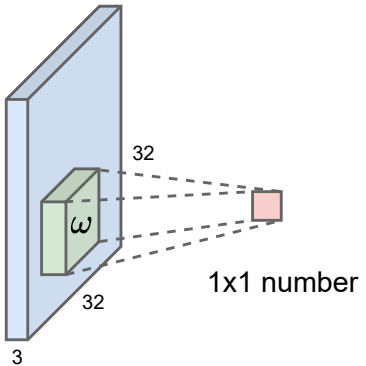
32x32x3 input



5x5x3 filter

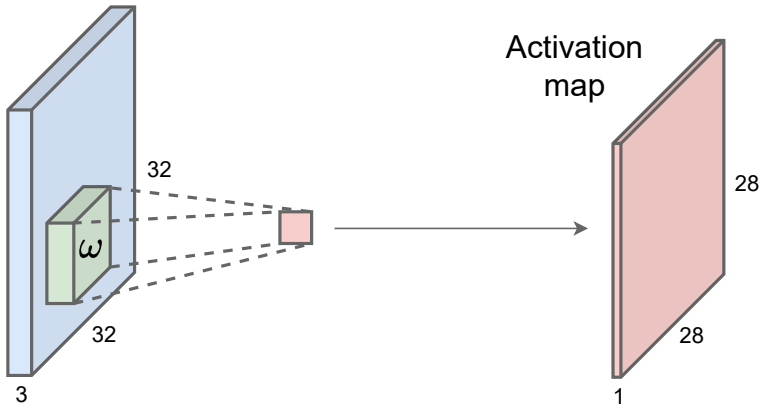
Convolutions Illustrated (cont.)

- Compute the dot product between the filter ω and a chunk of the input of the same size.
- Dimensions: $5 \times 5 \times 3 \Rightarrow 75$-dimensional dot product $+$ bias: $\omega^T x + b$.



Convolutions Illustrated (cont.)

- Convolution over all spatial locations of the input creates the activation map.



Convolution vs. Cross-Correlation

- Convolutions are defined as:

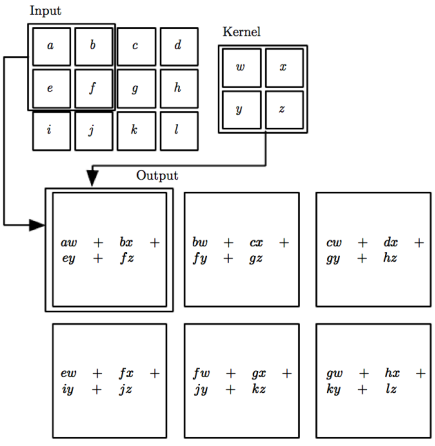
$$S(i,j) = (K * I)(i,j) = \sum_m \sum_n I(i-m, j-n) K(m,n)$$

- Often, people implement these as cross-correlation:

$$S(i,j) = (I * K)(i,j) = \sum_m \sum_n I(i+m, j+n) K(m,n)$$

- In convolutions, the kernel is flipped (rotated 180 deg) relative to the input.
- For the learning algorithm, this does not matter, since it can adapt to flipped/non-flipped kernel but mathematically there is a difference.
- Flipping kernel first + cross-correlation is equal to convolution.

Cross-Correlation Example



[Image source: Goodfellow et al., 2016, p. 330, Fig 9.1]

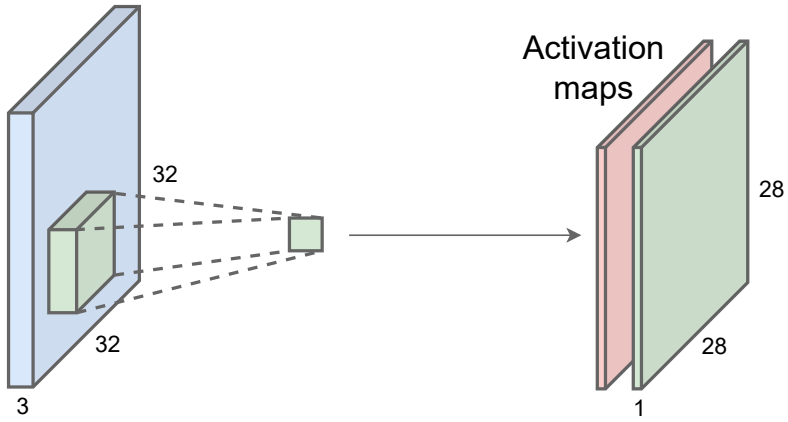
Cross-Correlation Animation

Animated demo for cross-correlation operation with multiple channels and padded borders:

http://cs231n.github.io/convolutional-networks/#conv

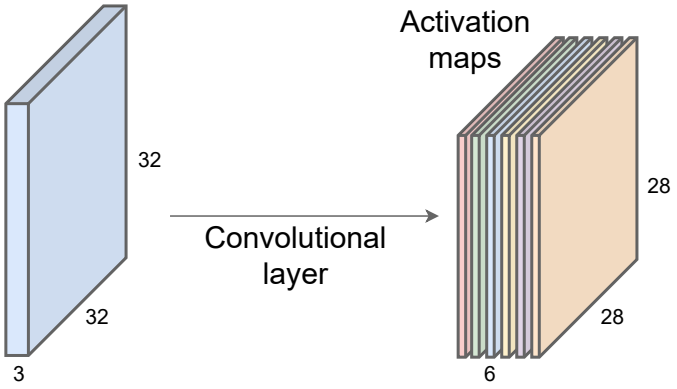
Convolutions – Several Filters

- Consider a second filter (green) of the same size.



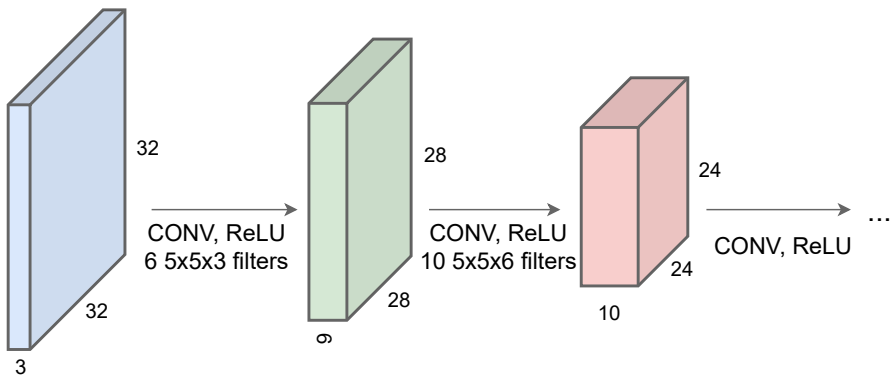
Convolutions – Several Filters (cont.)

- Using a convolutional layer with 6 filters of size 5×5 results in 6 separate activation maps.
- Stacking them creates a *new* image of size $28 \times 28 \times 3$.



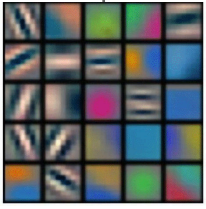
Stacking Several Convolutional Layers

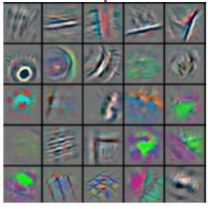
- Convolutional layers stacked in a ConvNet.



Learned Feature Hierarchy

Low-level features → Mid-level features → High-level features → Trainable network → ...

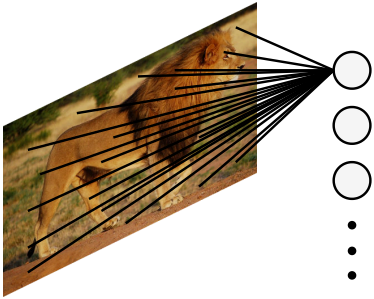


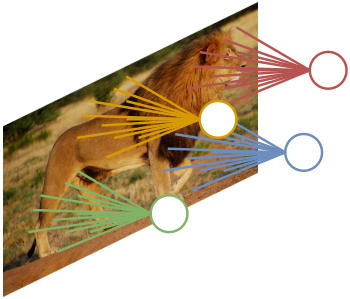


[Image (bottom) source: Zeiler and Fergus, 2014]

Weight Sharing

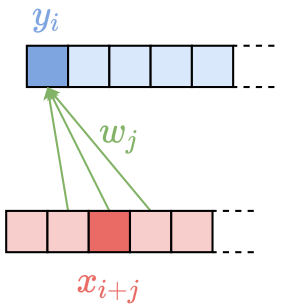
- Tremendous reduction in weights due to the local connectivity of convolutions.
- Example: 200×200 grayscale image
  - Fully connected: $400{,}000$ hidden units $\Rightarrow$ approx. $16 * 10^9$ parameters
  - Locally connected: $400{,}000$ hidden units with 10×10 fields $\Rightarrow$ approx. $40 * 10^6$ parameters





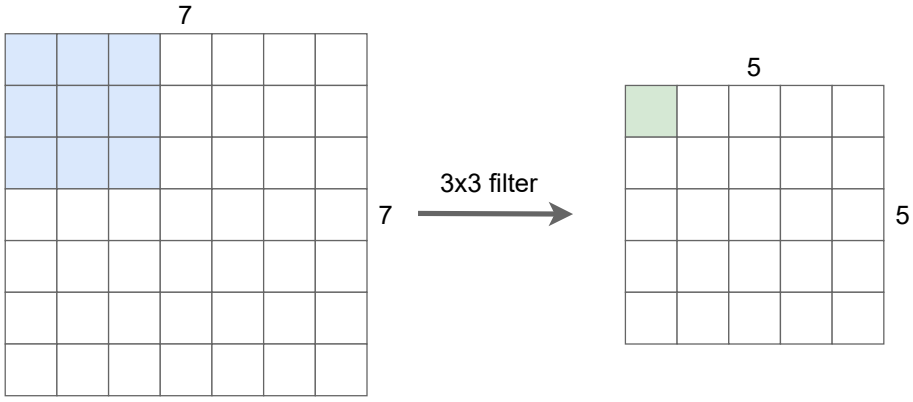
Backpropagation Through Convolutions

- Convolution: $y_i = \sum_j w_j x_{i+j}$ (cross-correlation)
- Backpropagation to input: $\frac{\partial C}{\partial x_j} = \sum_k w_k \frac{\partial C}{\partial y_{j-k}}$
- Backpropagation to weights: $\frac{\partial C}{\partial w_j} = \sum_i \frac{\partial C}{\partial y_i} x_{i+j}$



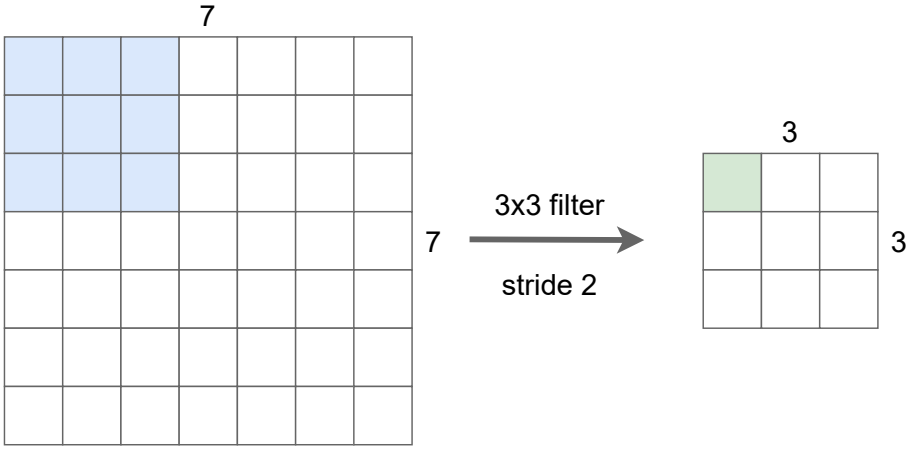
Convolutions Can Change Spatial Dimensions

- Applying a 3×3 filter to a 7×7 input generates a 5×5 output.

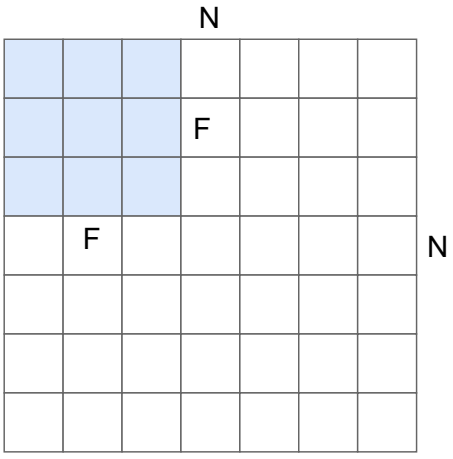


Convolutions Can Change Spatial Dimensions (cont.)

- Applying a 3×3 filter with stride 2 to a 7×7 input generates a 3×3 output.

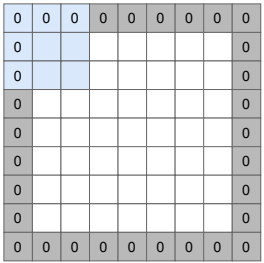


(Incomplete) Formula for Output Size



- Output size: $(N - F)/\text{stride} + 1$
- Examples for $N = 7$ and $F = 3$:
  - stride $1 \Rightarrow (7-3)/1 + 1 = 5$
  - stride $2 \Rightarrow (7-3)/2 + 1 = 3$
  - stride $3 \Rightarrow (7-3)/3 + 1 = 2.\overline{3}$ ☹

Zero Padding: Valid vs. Same Convolutions



- **What is the output?**
  - input 7×7
  - filter 3×3
  - stride 1
  - padding with pixel border 1
- Recall: $(N - F)/\text{stride} + 1$
- **Answer:** 7×7 output
- Commonly, we see convolutional layers with stride 1, $F \times F$ filters and zero padding with $(F - 1)/2$ to preserve the spatial size.
  - $F = 3 \Rightarrow$ zero padding with 1
  - $F = 5 \Rightarrow$ zero padding with 2
  - $F = 7 \Rightarrow$ zero padding with 3

- No padding and kernel required to be fully contained in image: valid convolution.
- With padding to keep image size constant through successive convolutions: same convolution.

Convolutions – Input / Output Summary

- Size of the input volume: $W_1 \times H_1 \times D_1$
- Specification of 4 hyperparameters:
  - number of filters K
  - their spatial extent F
  - stride S
  - amount of zero padding P
- Size of the output volume: $W_2 \times H_2 \times D_2$
  - $W_2 = (W_1 - F + 2P)/S + 1$
  - $H_2 = (H_1 - F + 2P)/S + 1$
  - $D_2 = K$
- Using filters of size $F \times F \times D_1$ and enabling parameter sharing, convolutions introduce $(F \cdot F \cdot D_1) \cdot K$ weights and K biases.
- The d-th depth slice ($W_2 \times H_2$) of the output results from performing a valid convolution of the d-th filter on the input with a stride of S and an offset of the d-th bias.

Questions to Answer for Yourself / Discuss with Friends

Input volume: $\mathbf{32} \times \mathbf{32} \times \mathbf{3}$
10 filters of size 5×5 with stride 1 and padding 2

- What is the output volume size?
- How many parameters are there in this layer?

Lecture Overview

Foundations of Deep Learning, Winter Term 2021/22

Week 6: Convolutional Neural Networks

Advantages of Convolutions

Frank Hutter Abhinav Valada

University of Freiburg

Advantages of Convolutions

Convolution leverages three important ideas that can help improve a machine learning system:

- Sparse Interactions
- Parameter Sharing
- Equivariant Representations

Sparse Interactions

- Traditional neural network layers use matrix multiplication by a matrix of parameters with a separate parameter describing the interaction between each input unit and each output unit. This means every output unit interacts with every input unit.
- Convolutional networks, however, typically have sparse interactions. This is accomplished by making the kernel smaller than the input.

Sparse Interactions

Connectivity in CNNs is sparse since convolutions only affect a subset of the inputs/outputs. In fully-connected networks (FCN), all inputs/outputs are affected.

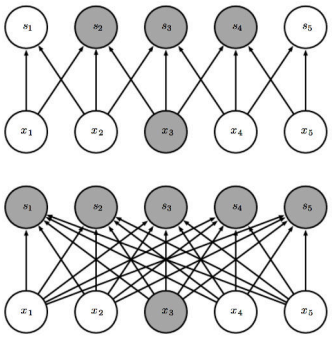


Figure: Sparse connectivity viewed from above.

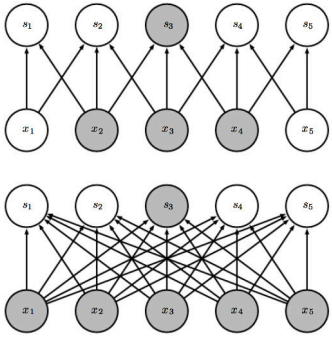


Figure: Sparse connectivity viewed from below.

[Image sources: Goodfellow et al., 2016, p. 331, Fig. 9.2, p. 332, Fig. 9.3]

Sparse Interactions

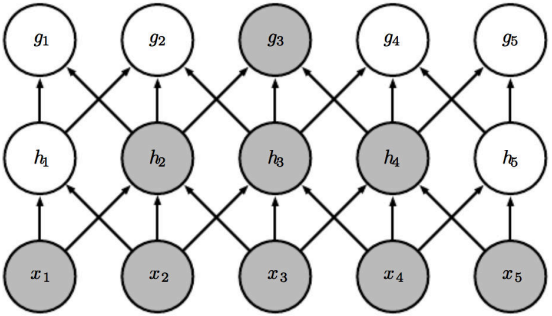


Figure: The receptive field of the units in the deeper layers of a convolutional network is larger than the receptive field of the units in the shallow layers.

[Image source: Goodfellow et al., 2016, p. 332, Fig. 9.4]

Sparse Interactions

- The local receptive fields are translated across the image to create a feature map from the input layers to the hidden layer units.
  One can use convolutions to implement this process efficiently.
- If there are m inputs and n outputs, then matrix multiplication requires $m \times n$ parameters and the algorithms used in practice have $O(m \times n)$ runtime (for example). If we limit the number of connections each output may have to k, then the sparsely connected approach requires only $k \times n$ parameters and yields $O(k \times n)$ runtime.

Parameter Sharing

- In a traditional neural net, each element of the weight matrix is used exactly once when computing the output of a layer. It is multiplied by one element of the input and then never revisited.
- In a convolutional neural net, each member of the kernel is used at every position of the input.

Parameter Sharing

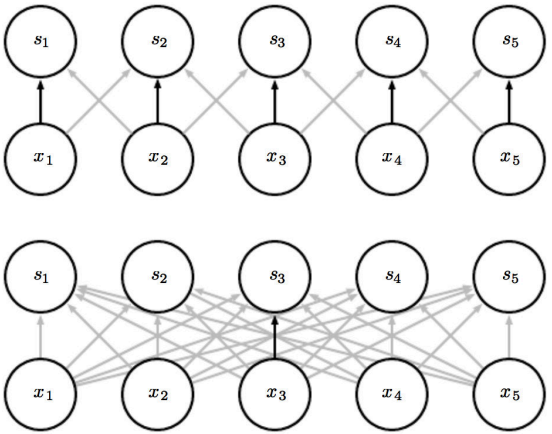


Figure: Black arrows indicate the connections that use a particular parameter in two different models. (top) CNN, (bottom) traditional neural network

[Image source: Goodfellow et al., 2016, p. 333, Fig 9.5]

Parameter Sharing

- Weights and biases are the same for all hidden neurons in the same layer. They are used to detect the same feature such as edge making it robust to translations of revisited objects.
- This does not affect the runtime of forward propagation. It is still $O(k \times n)$ but it does further reduce the storage requirement of the model to k parameters.
- Convolution is thus dramatically more efficient than dense matrix multiplication in terms of the memory requirements and statistical efficiency.

Equivariant Representation

- Saying a function is equivariant means that if the input changes, the output changes in the same way.
- Specifically, a function $f(x)$ is equivariant to a function g if $f(g(x)) = g(f(x))$.

Equivariant Representation – Example

- Let I be a function giving image brightness at integer coordinates.
- Let g be a function mapping one image function to another image function, such that $I' = g(I)$ is the image function with $I'(x, y) = I(x - 1, y)$. This shifts every pixel of I one unit to the right.
- If we apply this transformation to I, then apply convolution, the result will be the same as if we applied convolution to I, then applied the transformation g to the output.
- When processing time-series data, this means that convolution produces a sort of timeline that shows when different features appear in the input. If we move an event later in time in the input, the exact same representation of it will appear in the output, just later in time.

Questions to Answer for Yourself / Discuss with Friends

- Repetition: At which stage of a CNN is the receptive field the largest?

Lecture Overview

Foundations of Deep Learning, Winter Term 2021/22

Week 6: Convolutional Neural Networks

Miscellaneous Convolutions

Frank Hutter Abhinav Valada

University of Freiburg

3D Convolution

- 3D filter can move in all three direction (height, width, channel) in 3D convolutions, as opposed to 2D (height and width) in 2D convolutions.
- Filter depth should be smaller than the input layer depth, i.e., kernel size $<$ channel size.

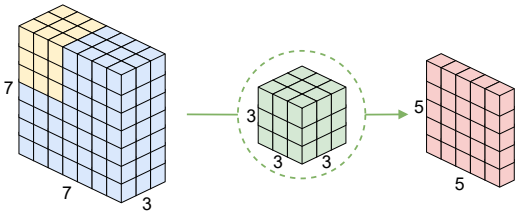


Figure: 2D convolution.

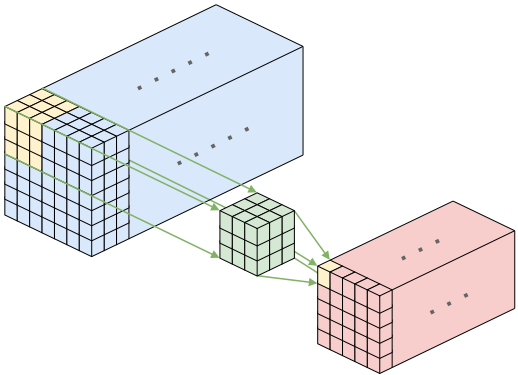


Figure: 3D convolution.

1×1 Convolution

- Dimensionality reduction for efficient computations.
- Efficient low dimensional embedding, or feature pooling.
- Applying non-linearity after 1×1 convolution enables the network to learn more complex functions.

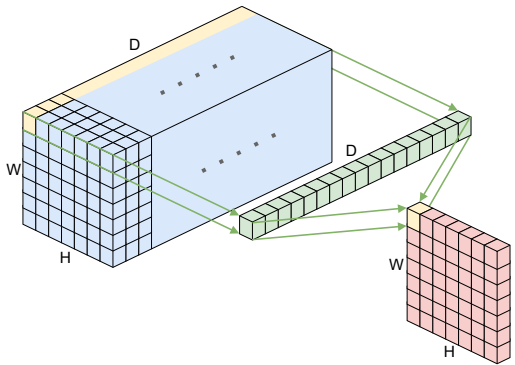


Figure: 1×1 Convolution.

Transposed Convolution

- Transposed convolution is also known as deconvolution (bad!), fractionally strided convolution, upconvolution, or backward strided convolution.
- A transposed convolution will reverse the spatial transformation of a regular convolution with the same parameters.

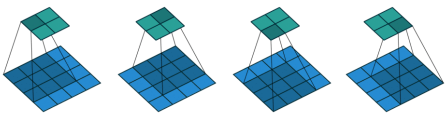


Figure: Standard 2D convolution with a 3×3 kernel over a 4×4 input, stride=1, and padding=0.

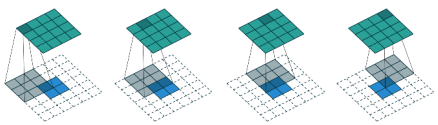


Figure: 2D transposed Convolution with a 3×3 kernel over a 2×2 input, stride=1, and padding=0.

[Image source: Dumoulin and Visin, 2016]

Dilated Convolutions

- A.k.a. atrous convolutions (fr. à trous): convolutions with holes.
- Inflates the kernel by inserting spaces between the kernel elements. Equivalent to dilating the filter by a dilation rate k before employing standard convolution.
- Increases the size of the receptive field relative to the kernel size.

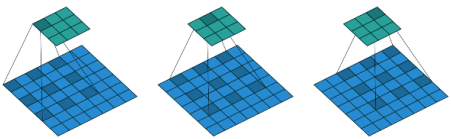


Figure: Dilated Convolution with a 3×3 kernel over a 7×7 input, stride=1, and padding=0.

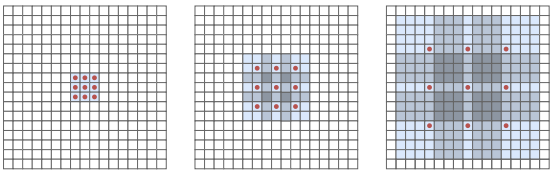


Figure: Exponential expansion of the receptive field with increasing dilation rates $k \in \{1, 2, 4\}$.

[Image (left) source: Dumoulin and Visin, 2016]

Grouped Convolutions

- Filters are first separated into different groups.
- Each group is then responsible for a standard 2D convolution with certain depth.
- Advantages: efficient training (convolutions are divided into several paths, each path can be handled by different GPUs) and efficient model (model parameters decrease as number of filter groups increases).

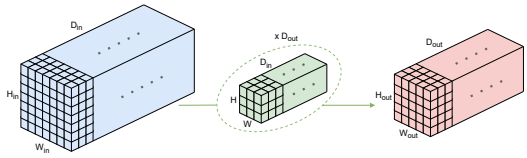


Figure: Standard 2D convolution.

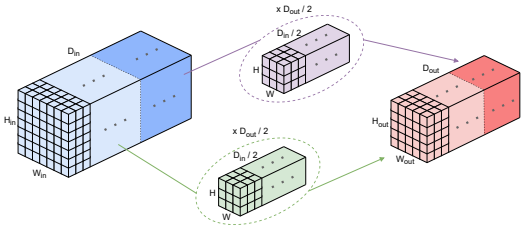


Figure: Grouped convolution with 2 filter groups.

Spatially Separable Convolutions

- Decomposes a convolution into two separate operations.
- Advantages: lesser parameters, lesser matrix multiplications.
- Disadvantage: training results may be suboptimal.

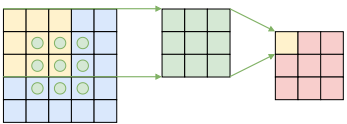


Figure: Standard convolution with 1 channel.

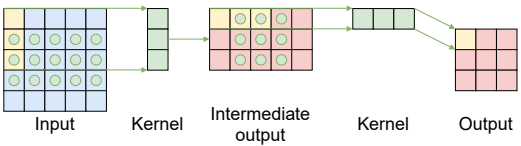


Figure: Spatially separable convolution with 1 channel.

Depthwise Separable Convolutions

- Consists of two consecutive convolution operations: first, a depth-wise convolution performing convolutions separately for each channel and, second, a 1×1 convolution.
- Advantages: lesser parameters, lesser matrix multiplications.
- Disadvantage: model may be suboptimal.

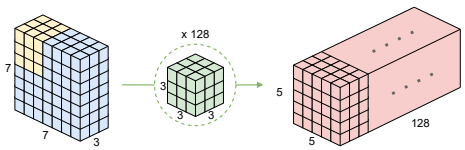


Figure: Standard 2D convolution using 128 filters.

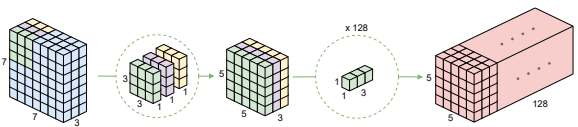


Figure: Depthwise separable convolution.

Questions to Answer for Yourself / Discuss with Friends

- Repetition: What is a 1×1 convolution helpful for?
- Repetition: What are the advantages of dilated convolutions?

Lecture Overview

Foundations of Deep Learning, Winter Term 2021/22

Week 6: Convolutional Neural Networks

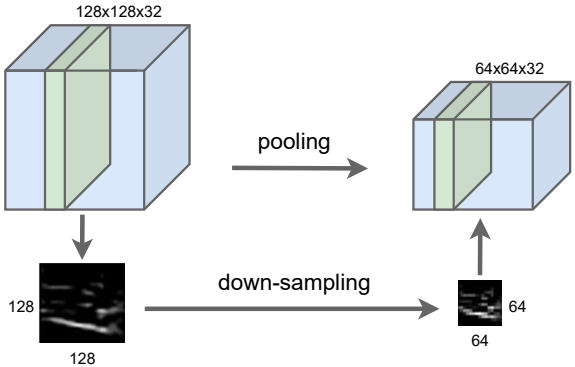
Pooling

Frank Hutter Abhinav Valada

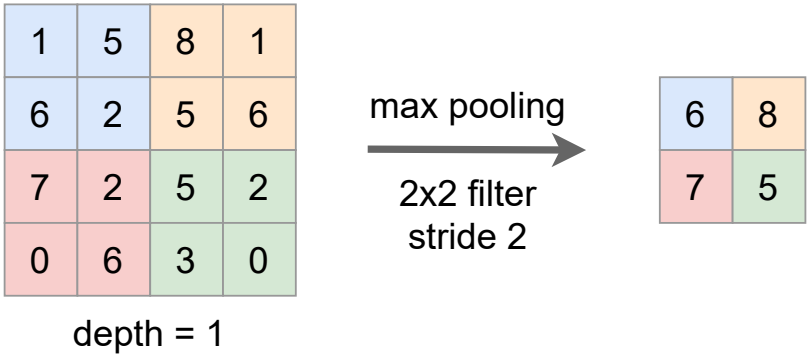
University of Freiburg

Pooling

- Reduces representation and make more manageable.
- Process each activation map separately.
- Makes representation approximately invariant to small translations in the input.



Max Pooling


1 5 8 1
6 2 5 6
7 2 5 2
0 6 3 0
depth = 1
max pooling
2x2 filter
stride 2
6 8
7 5

Pooling - Input / Output Summary

- Size of the input volume: $W_1 \times H_1 \times D_1$
- Specification of 3 hyperparameters:
  - their spatial extend F
  - stride S
- Size of the output volume: $W_2 \times H_2 \times D_2$ where:
  - $W_2 = (W_1 - F)/S + 1$
  - $H_2 = (H_1 - F)/S + 1$
  - $D_2 = D_1$
- No additional parameters as it computes a fixed function of the input.
- Usually, pooling layers do not use zero padding.

Questions to Answer for Yourself / Discuss with Friends

- Activation of what you just learned:
  How many learnable parameters does a pooling layer have?
- Repetition: What are the two most commonly used pooling layers?

Lecture Overview

Foundations of Deep Learning, Winter Term 2021/22

Week 6: Convolutional Neural Networks

Summary, Further Reading, References

Frank Hutter Abhinav Valada

University of Freiburg

Summary by Learning Goals

Having heard this lecture, you can now...

- explain what a ConvNet is.
- contrast a regular, fully connected MLP with a ConvNet of the same depth (architecture, number of parameters).
- explain why a ConvNet can learn more and more complex features.
- explain the difference between convolution and cross-correlation.
- calculate the values for convolution / cross-correlation and pooling operations.
- determine input/output sizes for convolutions and max pooling.
- explain the different types of miscellaneous convolutions.

Further Reading

- Deep Learning book: Chapter 9 [Goodfellow et al., 2016]
- Stanford Course CS231n: Convolutional Neural Networks for Visual Recognition

References

Aneja, J., Deshpande, A., Schwing, A. G. (2018)
Convolutional Image Captioning
Proceedings of the IEEE conference on computer vision and pattern recognition, 5561–5570
https://openaccess.thecvf.com/content_cvpr_2018/papers/Aneja_Convolutional_Image_Captioning_CVPR_2018_paper.pdf

Chen, L. C., Zhu, Y., Papandreou, G., Schroff, F., Adam, H. (2018)
Encoder-Decoder with Atrous Separable Convolution for Semantic Image Segmentation
Proceedings of the European conference on computer vision (ECCV), 801–818
https://arxiv.org/pdf/1802.02611.pdf

Chisari, E., Welschehold, T., Boedecker, J., Burgard, W., Valada, A. (2021)
Correct Me if I am Wrong: Interactive Learning for Robotic Manipulation
arXiv preprint arXiv:2110.03316
https://arxiv.org/pdf/2110.03316.pdf

Dosovitskiy, A., Fischer, P., Ilg, E., Häusser, P., Hazirbas, C., Golkov, V., Van Der Smagt, P., Cremers, D., Brox, T. (2015)
FlowNet: Learning Optical Flow with Convolutional Networks
Proceedings of the IEEE international conference on computer vision, 2758–2766
https://openaccess.thecvf.com/content_iccv_2015/papers/Dosovitskiy_FlowNet_Learning_Optical_ICCV_2015_paper.pdf

Dumoulin, V., Visin, F. (2016)
A guide to convolution arithmetic for deep learning
arXiv preprint arXiv:1603.07285
https://arxiv.org/pdf/1603.07285.pdf

Goodfellow, I., Bengio, Y., Courville, A. (2016)
Deep Learning
MIT Press
https://www.deeplearningbook.org

Ilg, E., Mayer, N., Saikia, T., Keuper, M., Dosovitskiy, A., Brox, T. (2017)
FlowNet 2.0: Evolution of Optical Flow Estimation with Deep Networks
Proceedings of the IEEE conference on computer vision and pattern recognition, 2462–2470
https://openaccess.thecvf.com/content_cvpr_2017/papers/Ilg_FlowNet_2.0_Evolution_CVPR_2017_paper.pdf

Krizhevsky, A., Sutskever, I., Hinton, G. (2012)
ImageNet Classification with Deep Convolutional Neural Networks
Advances in neural information processing systems 25, 1106–1114
https://dl.acm.org/doi/pdf/10.1145/3065386

LeCun, Y., Bottou, L., Bengio, Y., Haffner, P. (1998)
Gradient-Based Learning Applied to Document Recognition
Proceedings of the IEEE 86, 2278–2324
https://www.researchgate.net/publication/2985446_Gradient-Based_Learning_Applied_to_Document_Recognition

References

LeCun, Y. and Ranzato, M. (2013)
Deep Learning Tutorial
Tutorials in International Conference on Machine Learning (ICML'13) , 1–29
http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.366.4088&rep=rep1&type=pdf

Luo, W., Yang, B., Urtasun, R. (2018)
Fast and Furious: Real Time End-to-End 3D Detection, Tracking and Motion Forecasting with a Single Convolutional Net
Proceedings of the IEEE conference on Computer Vision and Pattern Recognition, 3569–3577
https://openaccess.thecvf.com/content_cvpr_2018/papers/Luo_Fast_and_Furious_CVPR_2018_paper.pdf

Mehta, D., Sotnychenko, O., Mueller, F., Xu, W., Sridhar, S., Pons-Moll, G., Theobalt, C. (2018)
Single-Shot Multi-Person 3D Pose Estimation From Monocular RGB
2018 International Conference on 3D Vision (3DV), 120–130
https://arxiv.org/pdf/1712.03453.pdf

Mohan, R., Valada, A., (2021)
EfficientPS: Efficient Panoptic Segmentation
International Journal of Computer Vision 129(5), 1551–1579
https://arxiv.org/pdf/2004.02307.pdf

Nguyen, A., Yosinski, J., Clune, J. (2019)
Understanding Neural Networks via Feature Visualization: A Survey
Explainable AI: interpreting, explaining and visualizing deep learning, 55–76, Springer
https://arxiv.org/pdf/1904.08939.pdf

Omran, M., Lassner, C., Pons-Moll, G., Gehler, P., Schiele, B. (2018)
Neural Body Fitting: Unifying Deep Learning and Model-Based Human Pose and Shape Estimation
2018 international conference on 3D vision (3DV), 484–494
https://arxiv.org/pdf/1808.05942.pdf

Qi, C. R., Liu, W., Wu, C., Su, H., Guibas, L. J. (2018)
Frustum PointNets for 3D Object Detection from RGB-D Data
Proceedings of the IEEE conference on computer vision and pattern recognition, 918–927
https://arxiv.org/pdf/1711.08488.pdf

Ren, S., He, K., Girshick, R., Sun, J. (2016)
Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks
IEEE transactions on pattern analysis and machine intelligence 39(6), 1137–1149
https://arxiv.org/pdf/1506.01497.pdf

Schrittwieser, J., Antonoglou, I., Hubert, T., Simonyan, K., Sifre, L., Schmitt, S., Guez, A., Lockhart, E., Hassabis, D., Graepel, T., Lillicrap, T. (2020)
Mastering Atari, Go, Chess and Shogi by Planning with a Learned Model
Nature 588(7839), 604–609
https://arxiv.org/pdf/1911.08265.pdf

Vinyals, O., Toshev, A., Bengio, S., Erhan, D. (2015)
Show and Tell: A Neural Image Caption Generator
Proceedings of the IEEE conference on computer vision and pattern recognition, 3156–3164
https://arxiv.org/pdf/1411.4555.pdf

References

Watson, J., Mac Aodha, O., Prisacariu, V., Brostow, G., Firman, M. (2021)
The Temporal Opportunist: Self-Supervised Multi-Frame Monocular Depth
Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 1164–1174
https://arxiv.org/pdf/2104.14540.pdf

Zeiler, M.D., Fergus, R. (2014)
Visualizing and Understanding Convolutional Networks
European conference on computer vision 2014, 818–833
https://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.683.4319&rep=rep1&type=pdf

Zhou, K., Zheng, Y., Li, B., Dong, W., Zhang, X. (2019)
Forecasting Different Types of Convective Weather: A Deep Learning Approach

Journal of Meteorological Research, 33(5), 797–809
https://link.springer.com/content/pdf/10.1007/s13351-019-8162-6.pdf