Convolutional Neural Networks and Deep Neural Networks

CS 6140

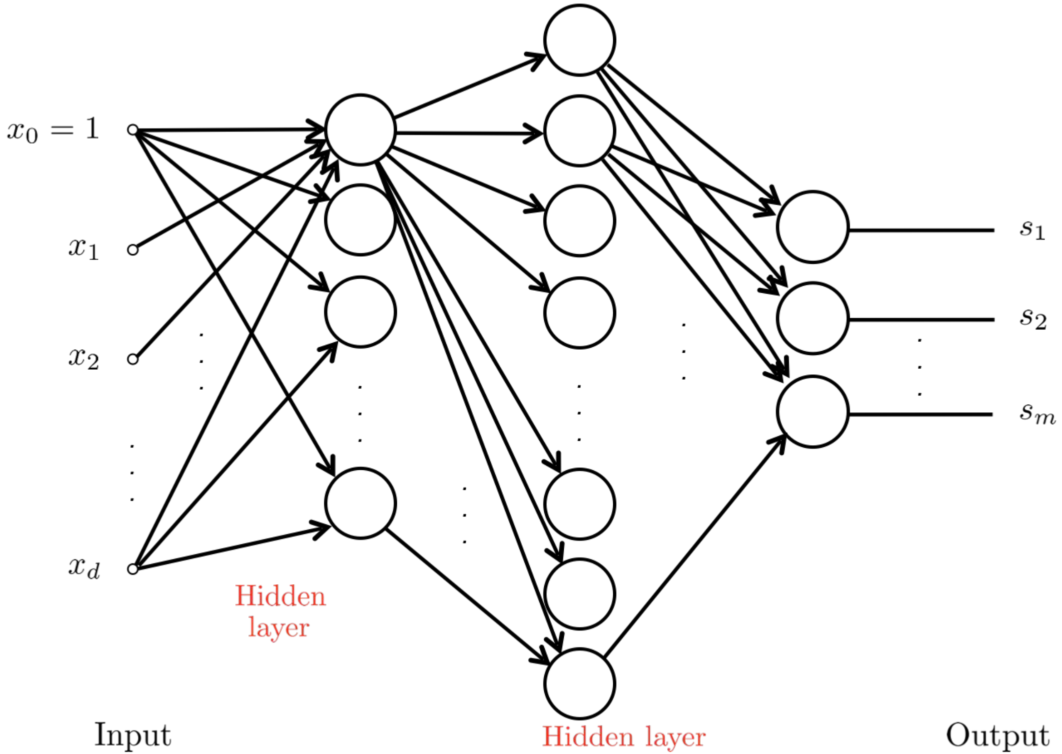
Clara De Paolis Kaluza
Khoury College of Computer Sciences
Northeastern University
Fall 2024

Big ideas

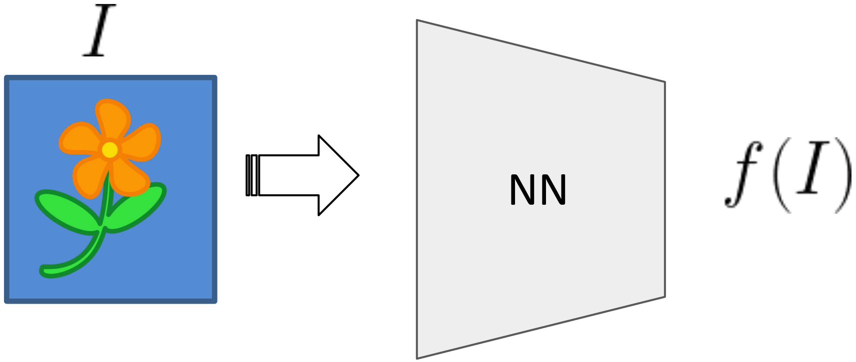
- Deep neural networks (DNNs) are neural networks with “many” hidden layers
- Convolutional Neural Networks (CNNs) are a kind of neural network with a topology that *exploits structure* in the input data to make learning easier/faster
- CNNs use the convolution operation
- CNNs are popular and ubiquitous

Feed-Forward Neural Network

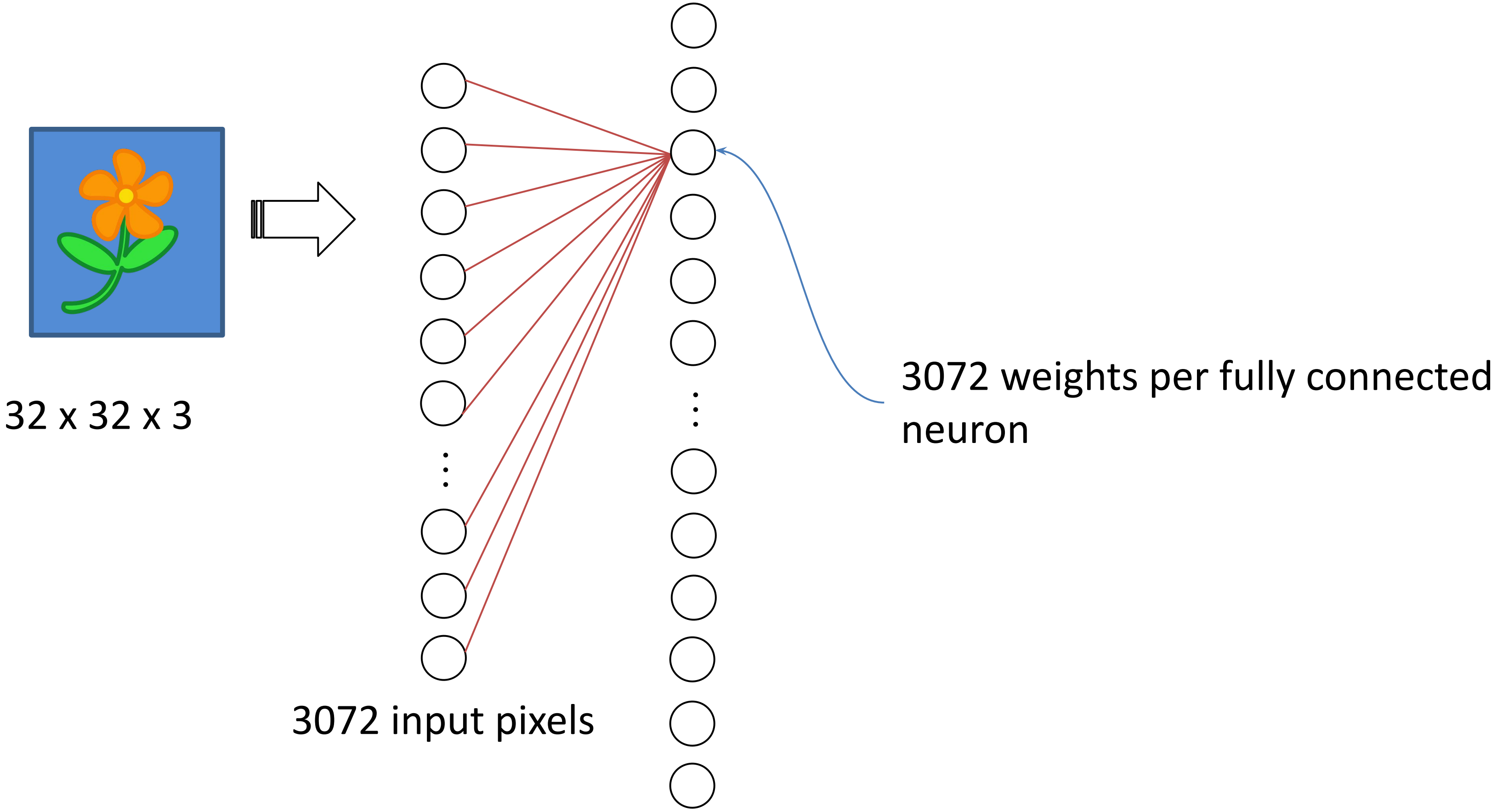
Universal approximator



Why Convolutional Neural Networks?

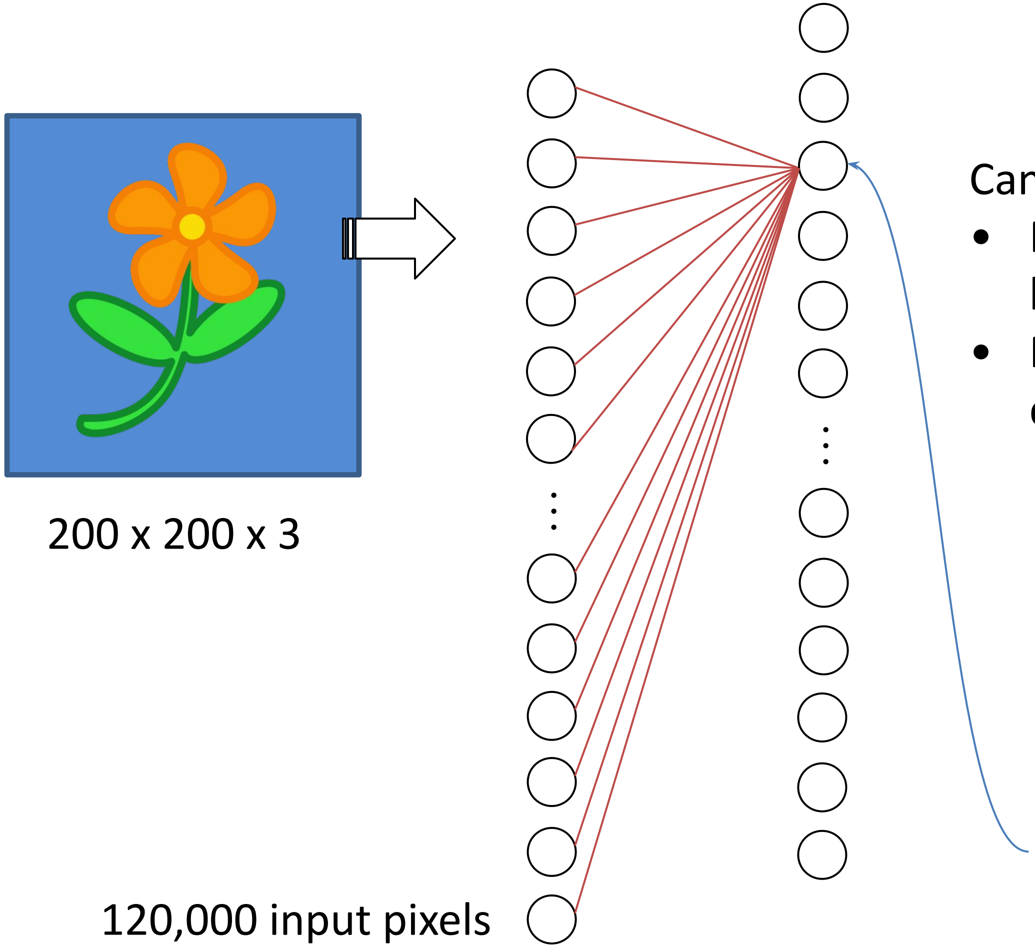


Why Convolutional Neural Networks?



Why Convolutional Neural Networks?

Efficiency



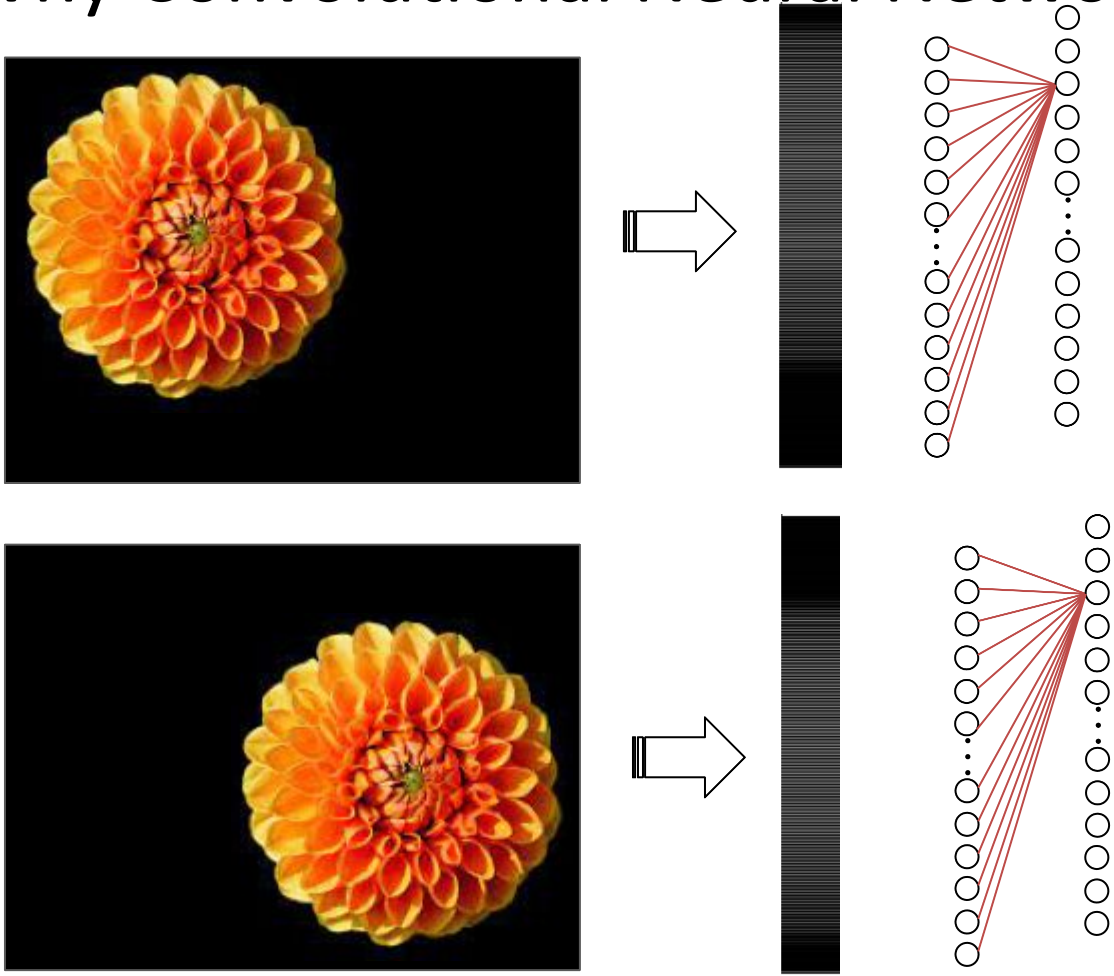
Cannot scale to realistic input sizes:

- More parameters → overfitting, unless you have A LOT of data
- More parameters → computationally difficult or intractable

120,000 weights per fully connected neuron

Why Convolutional Neural Networks?

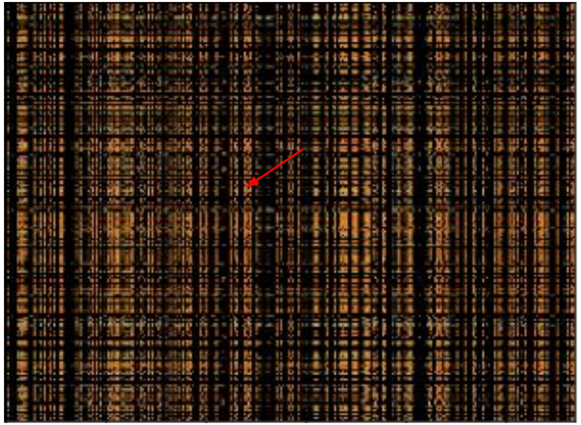
Structure



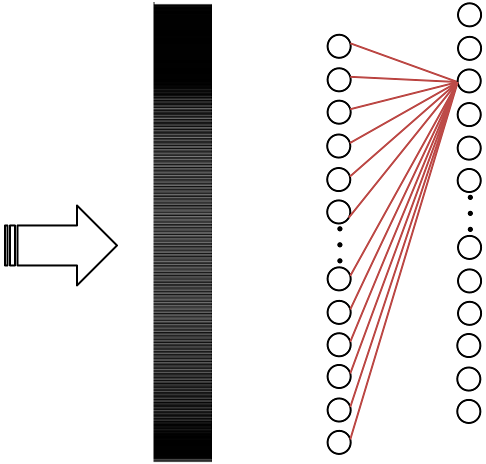
Weights depend on location

Why Convolutional Neural Networks?

Structure



Weights depend on location



Why Convolutional Neural Networks?

- **Efficiency**: Fully connected neural networks require too many parameters for large inputs
- **Structure**: FC NNs make no assumptions on the structure of the inputs.
    - Any underlying structure must be learned by the parameters
    - Parameters are independent and (possibly) redundant
    - We are not exploiting what we know about the problem

Inductive bias

Convolution

- Convolution is an operation on two functions which returns a function

$$f * g(x, y) = \sum_i \sum_j f(x + i, y + j) g(i, j)$$

- It is a measure of the interaction between the two input functions over eg time or space. The result of “filtering” one function with the other

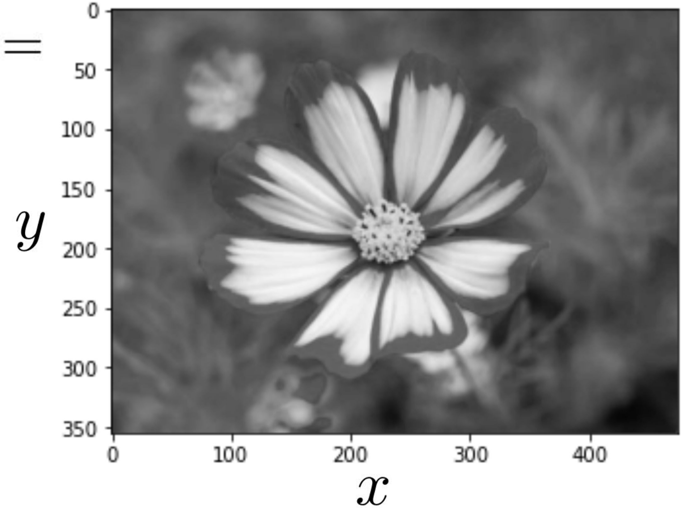
Convolution

- Convolution is an operation on two functions which returns a function

$$f * g(x, y) = \sum_i \sum_j f(x+i, y+j) g(i, j)$$

- It is a measure of the interaction between the two input functions over eg time or space. The result of “filtering” one function with the other

$f(x, y) =$



$$g(x, y) = 1/9 \times \begin{array}{|c|c|c|} \hline 1 & 1 & 1 \\ \hline 1 & 1 & 1 \\ \hline 1 & 1 & 1 \\ \hline \end{array}$$

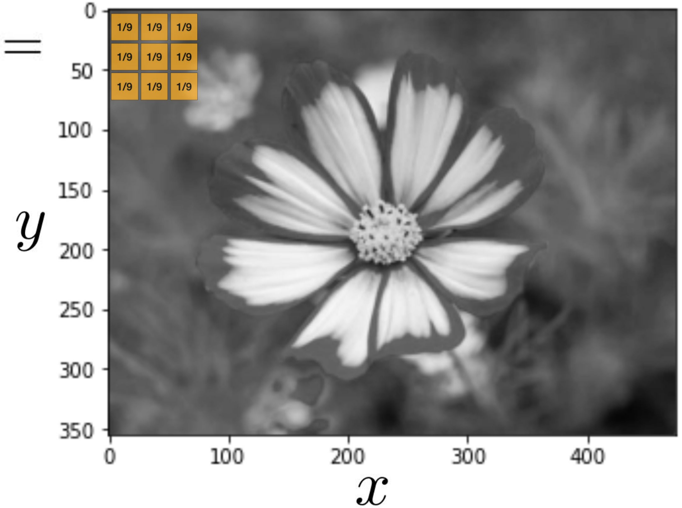
Convolution

- Convolution is an operation on two functions which returns a function

$$f * g(x,y) = \sum_i \sum_j f(x+i, y+j)g(i,j)$$

- It is a measure of the interaction between the two input functions over eg time or space. The result of "filtering" one function with the other

$$f(x,y) =$$



$$g(x,y) = 1/9 \times \begin{array}{|c|c|c|} \hline 1 & 1 & 1 \\ \hline 1 & 1 & 1 \\ \hline 1 & 1 & 1 \\ \hline \end{array}$$

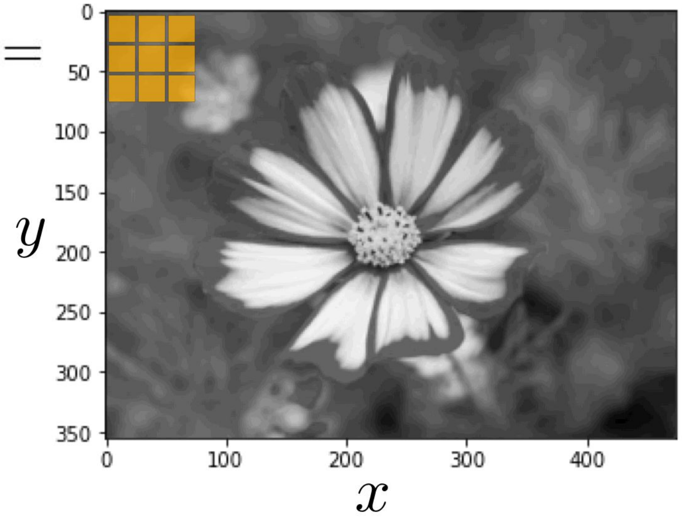
Convolution

- Convolution is an operation on two functions which returns a function

$$f * g(x, y) = \sum_i \sum_j f(x + i, y + j)g(i, j)$$

- It is a measure of the interaction between the two input functions over eg time or space. The result of “filtering” one function with the other

$f(x, y) =$



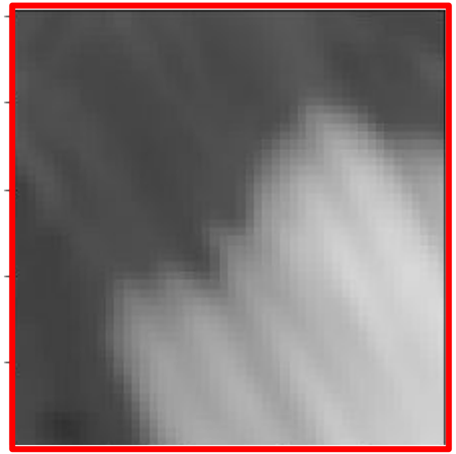
$$g(x, y) = 1/9 \times \begin{array}{|c|c|c|} \hline 1 & 1 & 1 \\ \hline 1 & 1 & 1 \\ \hline 1 & 1 & 1 \\ \hline \end{array}$$

Convolution

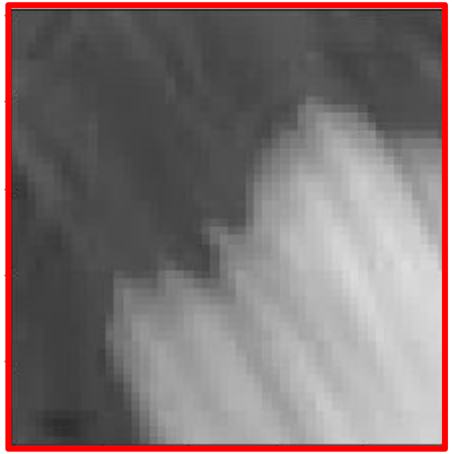
- Convolution is an operation on two functions which returns a function
- It is a measure of the interaction between the two input functions over eg time or space

$$f * g(x, y) =$$

$$f * g(x, y)$$



$$f(x, y)$$



$$f * g(x, y) = \sum_i \sum_j f(x + i, y + j) g(i, j)$$

Image kernels

original

Also known as “convolution matrix”
or “convolution filter”

Edge detection

$$\begin{bmatrix} -1 & -1 & -1 \\ -1 & 8 & -1 \\ -1 & -1 & -1 \end{bmatrix}$$

Sharpen

$$\begin{bmatrix} 0 & -1 & 0 \\ -1 & 5 & -1 \\ 0 & -1 & 0 \end{bmatrix}$$

Gaussian blur

$$\frac{1}{16}\begin{bmatrix} 1 & 2 & 1 \\ 2 & 4 & 2 \\ 1 & 2 & 1 \end{bmatrix}$$

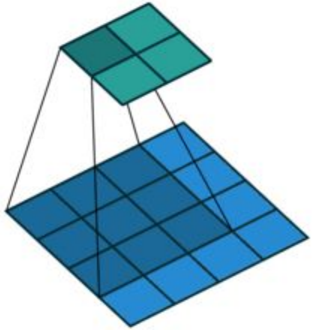
Convolution

$$f * g(x,y) = \sum_i \sum_j f(x+i, y+j) g(i,j)$$

Feature map

Kernel (filter)

$$g(x,y) = \begin{bmatrix} 0 & 1 & 2 \\ 2 & 2 & 0 \\ 0 & 1 & 2 \end{bmatrix}$$



$f(x,y) =$

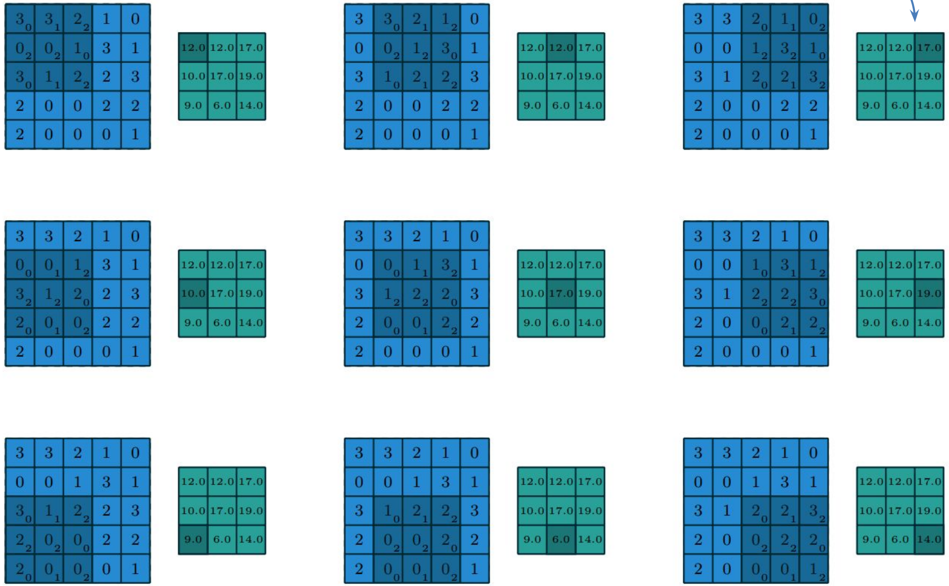


Image source: Dumoulin "A guide to convolution arithmetic for deep learning"

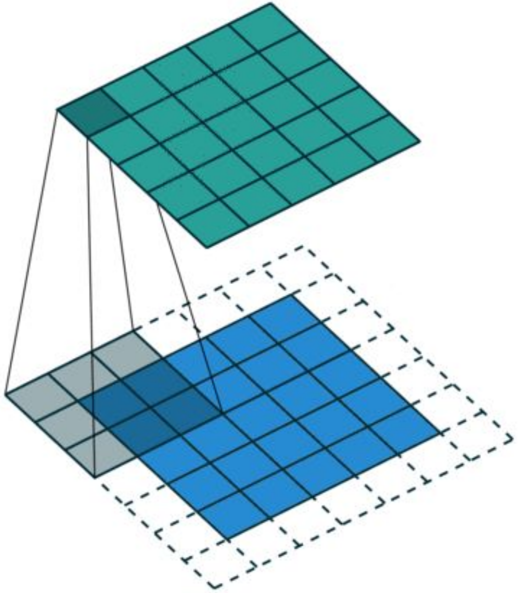
Convolution: Practical Considerations

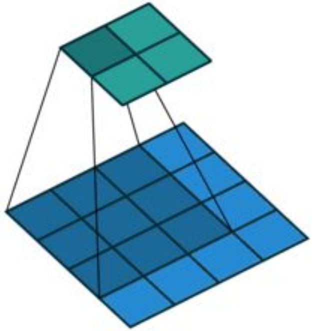
Padding:

- Avoid shrinking inputs
- Use edge information

Strides:

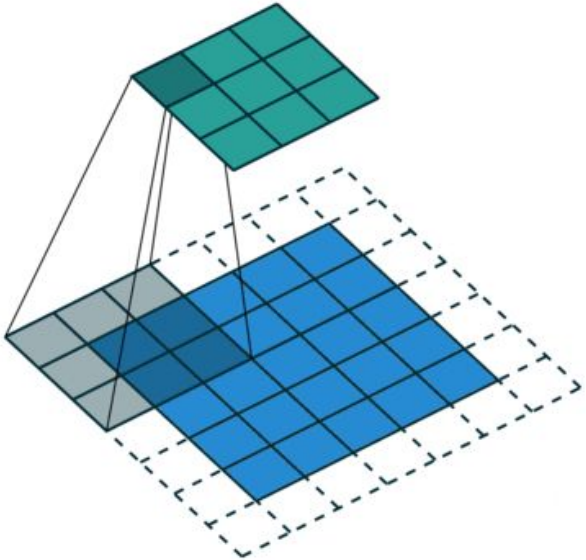
- Downsample input





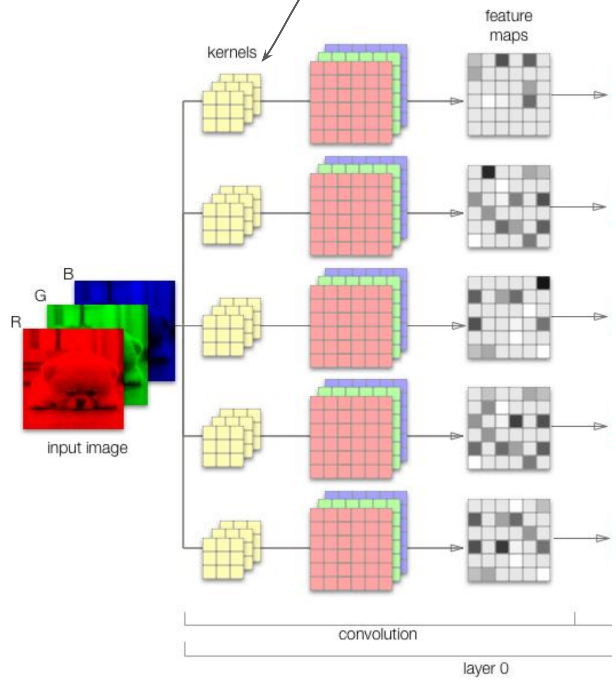
No padding

No strides



Learning

$$\begin{bmatrix} w_{00} & w_{01} & w_{02} \\ w_{10} & w_{11} & w_{12} \\ w_{20} & w_{21} & w_{22} \end{bmatrix}$$

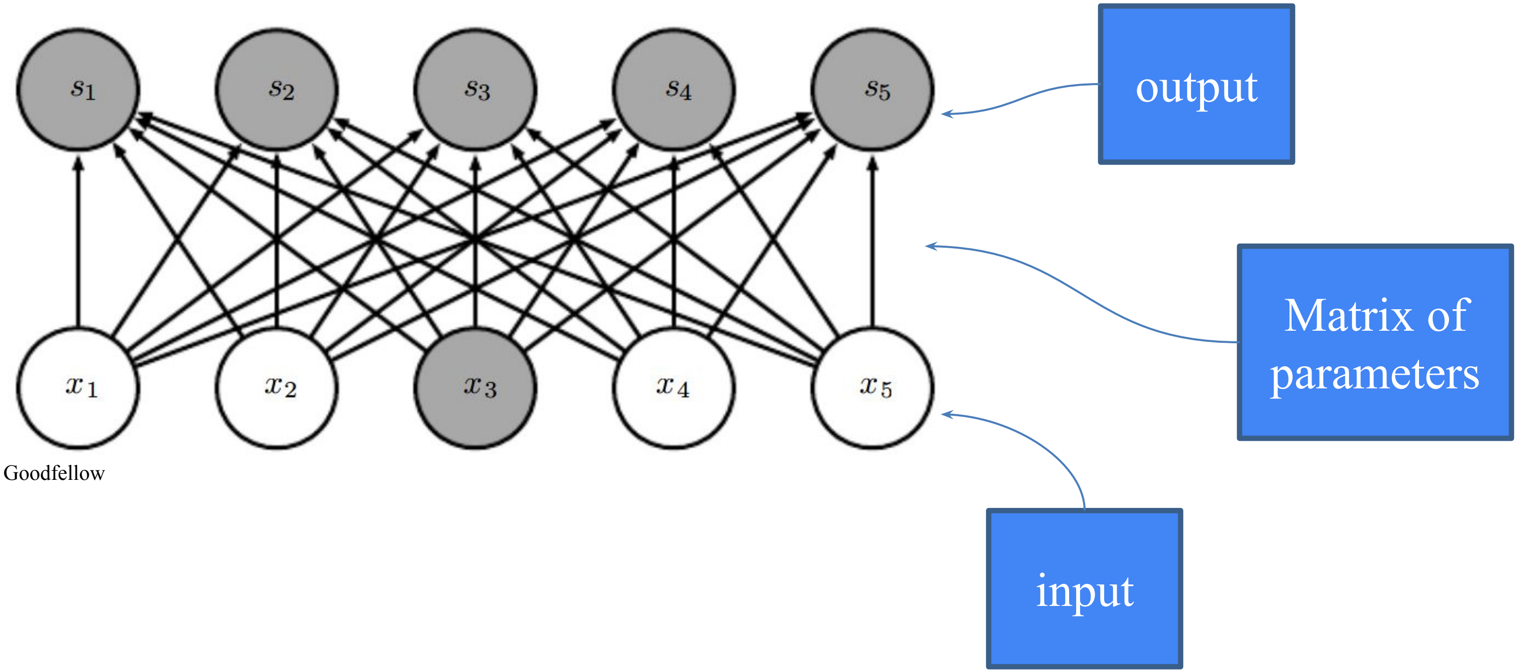


Learn the kernels instead of designing them

Image source:invasivecode.com

Sparse Interactions

m inputs and *n* outputs: *m* X *n* parameters



Source: Goodfellow

Sparse Interactions

Input size: 320 by 280

Kernel size: 2 by 1

Output size: 319 by 280

1	-1

Kernel

Source: Goodfellow

Sparse Interactions

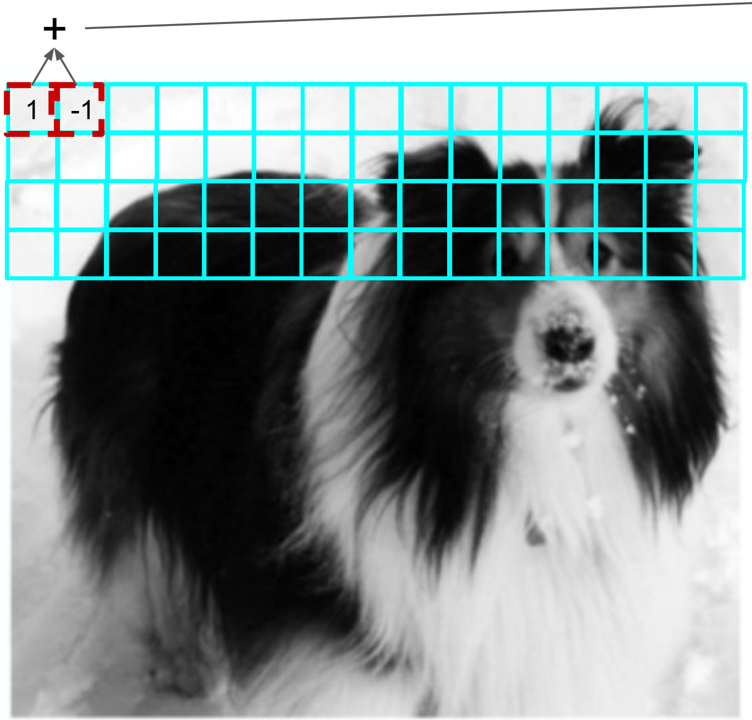
Input size: 320 by 280

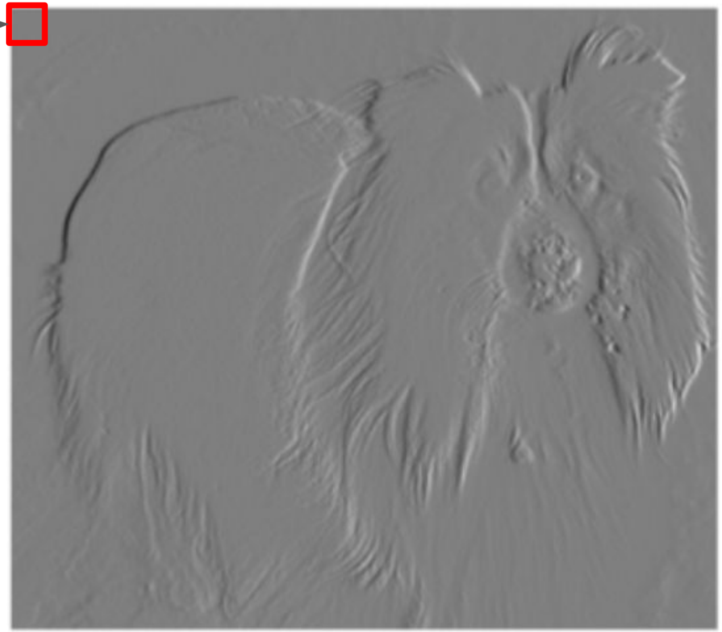
Kernel size: 2 by 1

Output size: 319 by 280

Two multiplications, one addition for each pixel

319 * (2+1) * 280 = 267,960





Output

Source: Goodfellow

Sparse Interactions

Input size: 320 by 280

Kernel size: 2 by 1

Output size: 319 by 280

vectorize

As a matrix multiplication:

320* 280

319* 280

```
[[-1, 0, 0, 0, 0, 0, 0, 0, 0, 0, ..., 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
 [1, -1, 0, 0, 0, 0, 0, 0, 0, 0, ..., 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
 [0, 1, -1, 0, 0, 0, 0, 0, 0, 0, ..., 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
 [0, 0, 1, -1, 0, 0, 0, 0, 0, 0, ..., 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
 [0, 0, 0, 1, -1, 0, 0, 0, 0, 0, ..., 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
 [0, 0, 0, 0, 1, -1, 0, 0, 0, 0, ..., 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
 [0, 0, 0, 0, 0, 1, -1, 0, 0, 0, ..., 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
 [0, 0, 0, 0, 0, 0, 1, -1, 0, 0, ..., 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
 [0, 0, 0, 0, 0, 0, 0, 1, -1, 0, ..., 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
 [0, 0, 0, 0, 0, 0, 0, 0, 1, -1, ..., 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
 ...,
 [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, ..., 1, -1, 0, 0, 0, 0, 0, 0, 0, 0],
 [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, ..., 0, 1, -1, 0, 0, 0, 0, 0, 0, 0],
 [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, ..., 0, 0, 1, -1, 0, 0, 0, 0, 0, 0],
 [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, ..., 0, 0, 0, 1, -1, 0, 0, 0, 0, 0],
 [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, ..., 0, 0, 0, 0, 1, -1, 0, 0, 0, 0],
 [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, ..., 0, 0, 0, 0, 0, 1, -1, 0, 0, 0],
 [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, ..., 0, 0, 0, 0, 0, 0, 1, -1, 0, 0],
 [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, ..., 0, 0, 0, 0, 0, 0, 0, 1, -1, 0],
 [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, ..., 0, 0, 0, 0, 0, 0, 0, 0, 1, -1],
 [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, ..., 0, 0, 0, 0, 0, 0, 0, 0, 0, 1]])
```

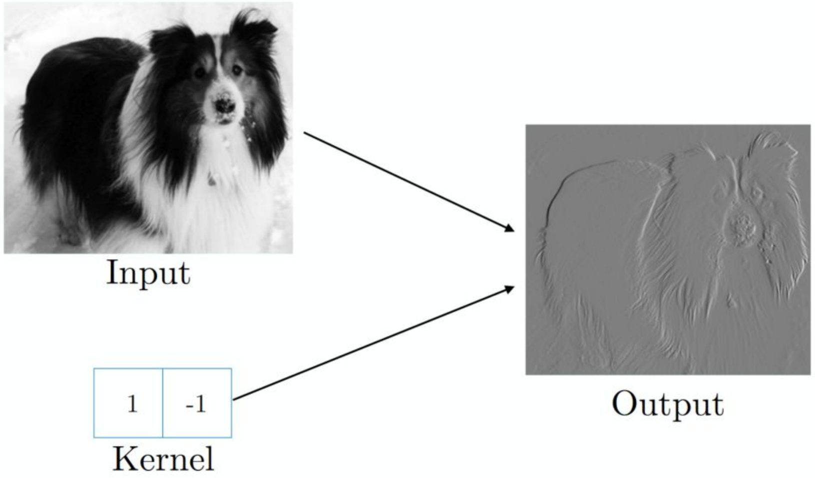
·

Sparse Interactions

Input size: 320 by 280

Kernel size: 2 by 1

Output size: 319 by 280



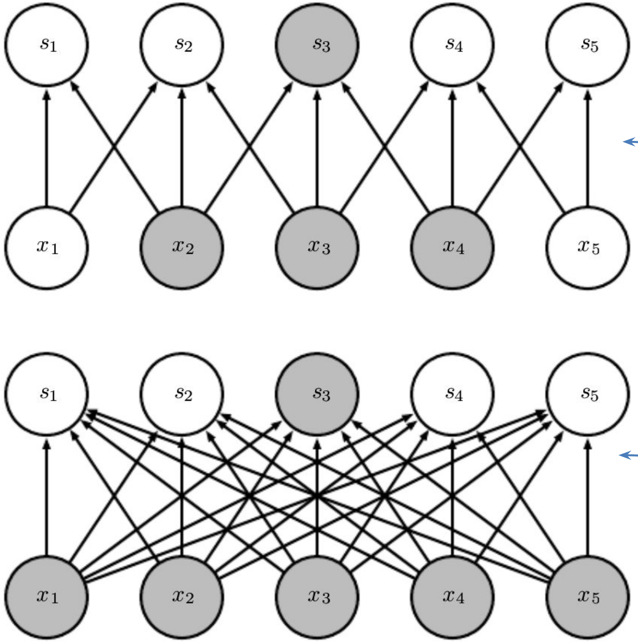
	Convolution	Dense matrix	Sparse matrix
Stored floats	2	319*280*320*280 $>$ 8e9	2*319*280 = 178,640
Float muls or adds	319*280*3 = 267,960	$>$ 16e9	Same as convolution (267,960)

Source: Goodfellow

Sparse Interactions

convolution

matrix multiplication



Kernel matrix parameters

Matrix of parameters

Source: Goodfellow

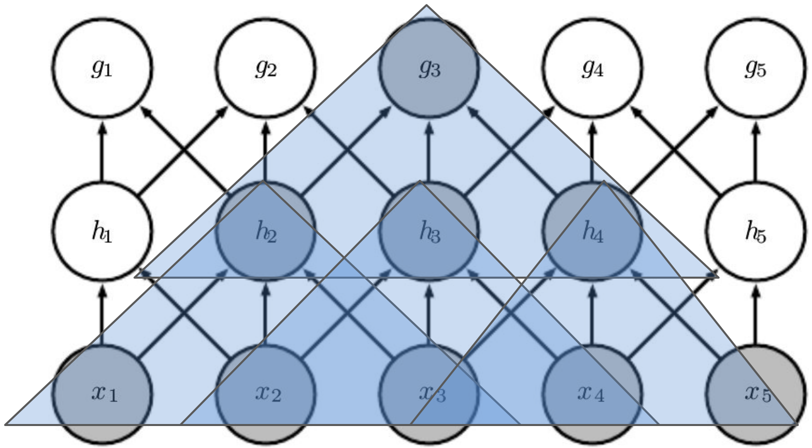
Sparse Interactions

Local receptive fields

g_3 depends only on h_2, h_3, and h_4

But h_2 depends on x_1, x_2, and x_3

Sparse model, but layers allow information to propagate "out"



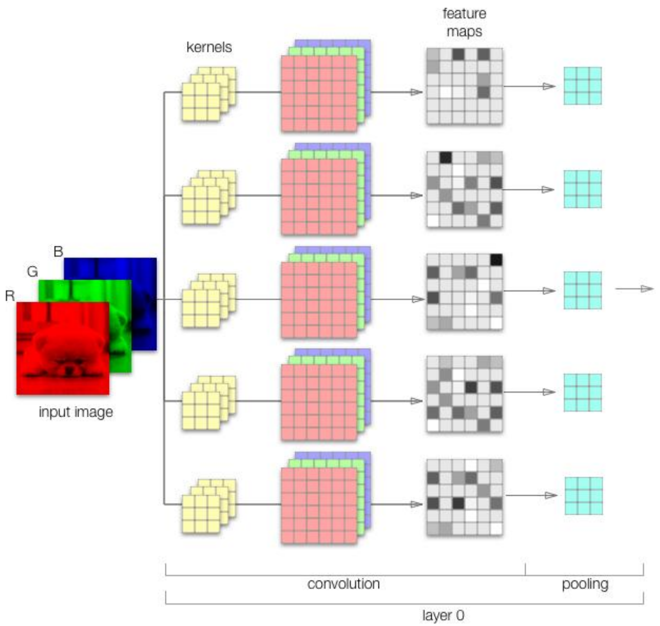
Source: Goodfellow

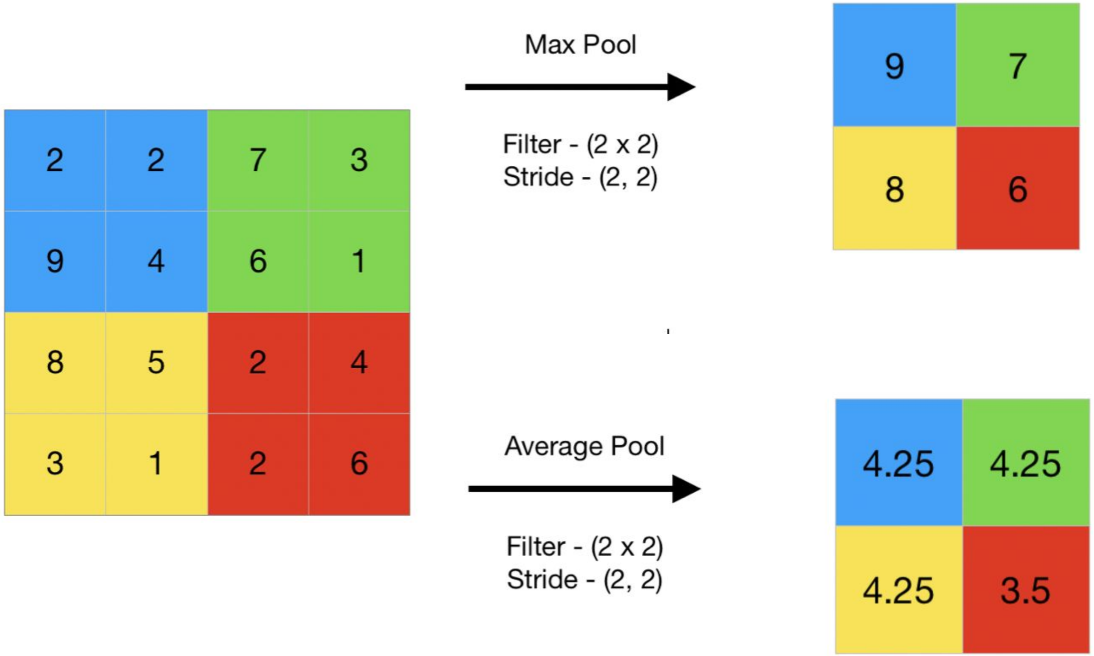
Pooling

Summarizes a region

Reduce representation size

Reduces needed parameters





Feature Maps

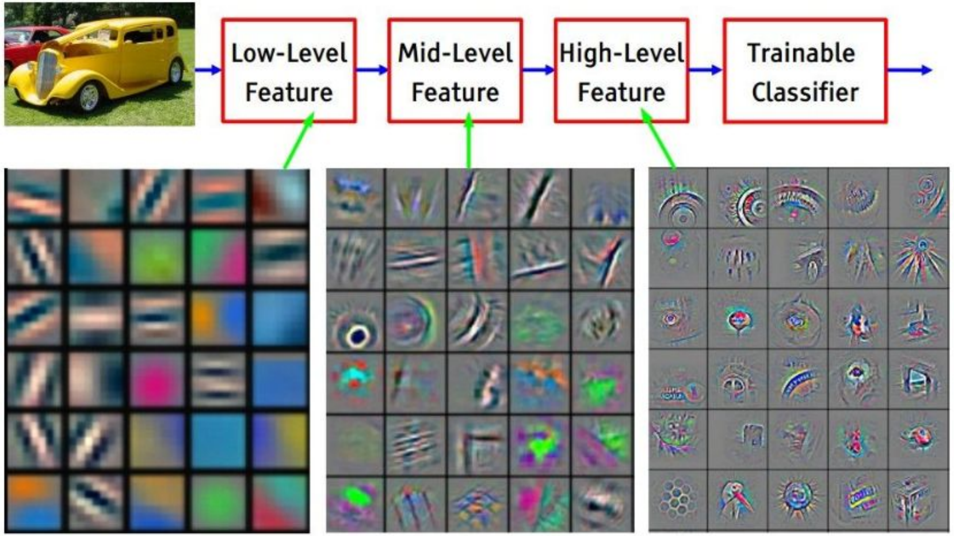
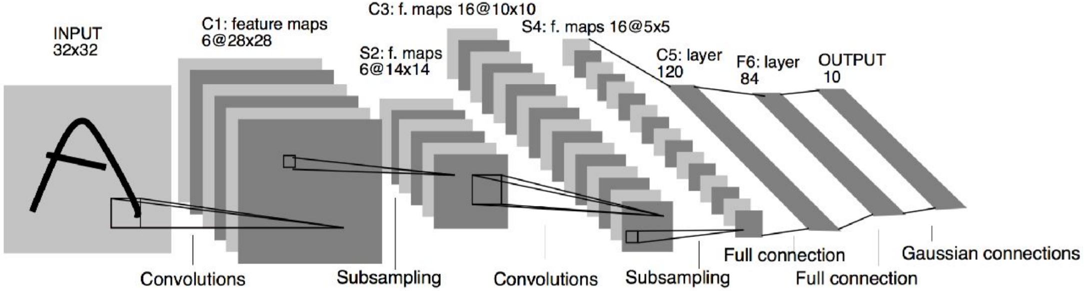


Image source: https://medium.com/@chriskevin_80184/feature-maps-ee8e11a71f9e

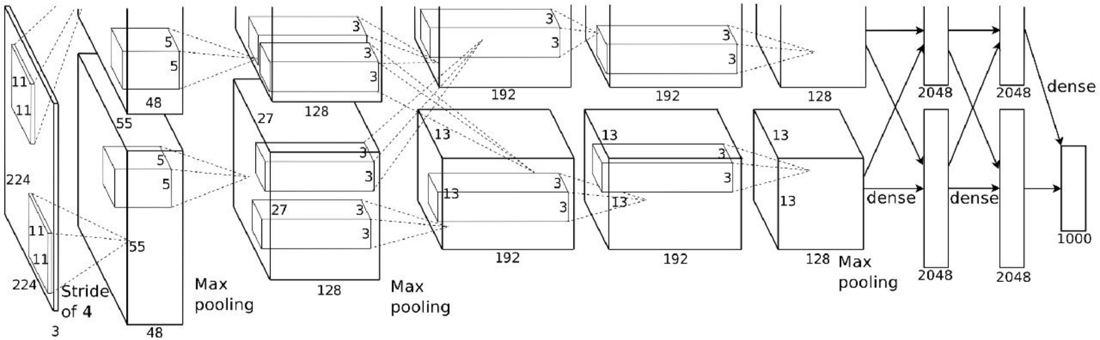
Architectures

LeNet 1998



LeCun, Y.; Bottou, L.; Bengio, Y. & Haffner, P. (1998). Gradient-based learning applied to document recognition. Proceedings of the IEEE. 86(11): 2278 - 2324

AlexNet 2012



GPU implementation

Dropout regularization

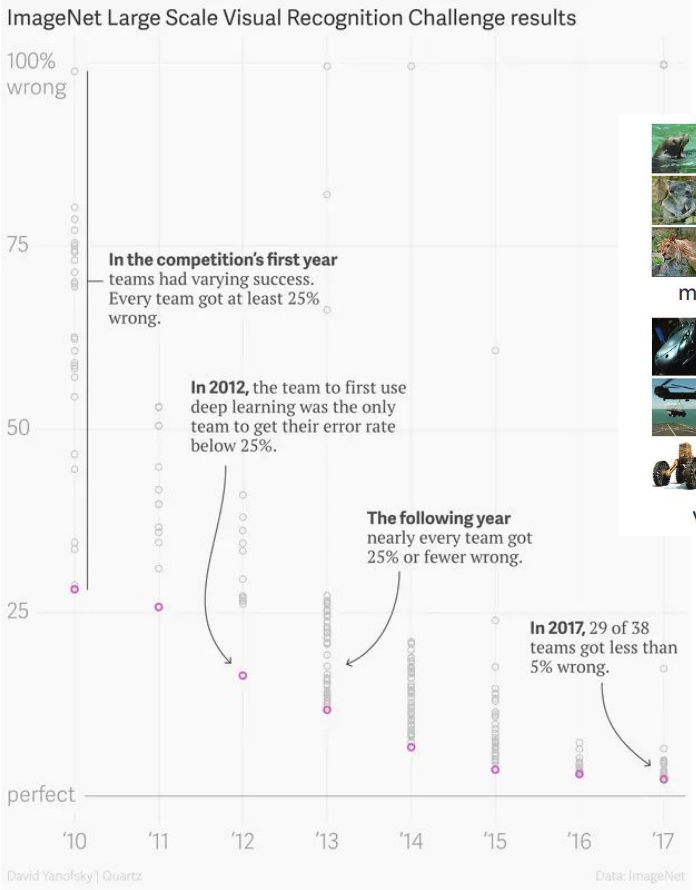
Data augmentation

ReLu

Overlapping Max Pooling

Alex Krizhevsky, Ilya Sutskever, Geoffrey E. Hinton “ImageNet Classification with Deep Convolutional Neural Networks” NeurIPS 2012

Datasets



ImageNet

Datasets

More than half of the labels in the people subtree were considered potentially harmful: **600,000 images were removed from ImageNet**.

Towards Fairer Datasets: Filtering and Balancing the Distribution of the People Subtree in the Imagenet Hierarchy

ACM Conference on Fairness, Accountability and Transparency (FAccT), January 2020

Kaiyu Yang, Klint Qinami, Li Fei-Fei,
Jia Deng, Olga Russakovsky

PROBLEM 1: STAGNANT CONCEPT VOCABULARY

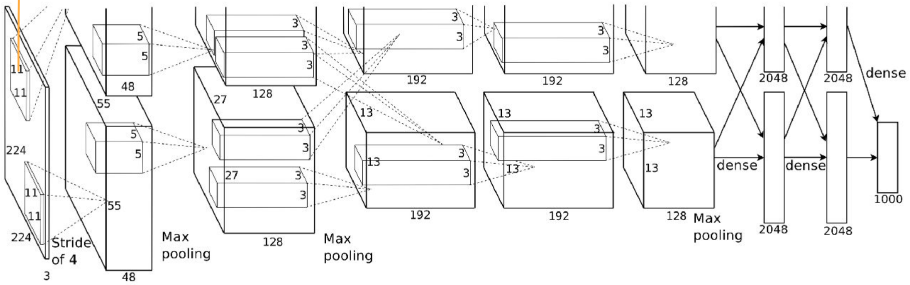
PROBLEM 2: NON-VISUAL CONCEPTS

PROBLEM 3: LACK OF IMAGE DIVERSITY

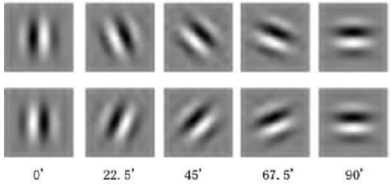
ains representative of only a few. People have
ing offensive prediction results and lower
sion models are typically developed using
the data and label distributions in these
amine ImageNet, a large-scale ontology of
uter vision methods. We consider three key factors within the "person" subtree of ImageNet that may lead to problematic behavior in downstream computer vision technology: (1) the stagnant concept vocabulary of WordNet, (2) the attempt at exhaustive illustration of all categories with images, and (3) the inequality of representation in the images within concepts. We seek to illuminate the root causes of these concerns and take the first steps to mitigate them constructively.

What Neural Network Learn

Figure 3: 96 convolutional kernels of size $11\times11\times3$ learned by the first convolutional layer on the $224\times224\times3$ input images. The top 48 kernels were learned on GPU 1 while



Gabor filters

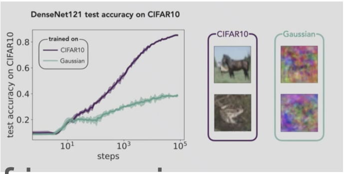


ImageNet training + AlexNet architecture + SDG optimization ≈ Gabor filters

But if you replace the first layer with the explicit mathematical expression for Gabor filters, performance decreases– Goldt 2023

Data-dependent features ≫ hand-crafted features

Why? Non-Gaussian fluctuations in the data are particularly important



NNs learn distributions of increasing complexity through training

Making it work

Depth

Regularization

Normalization

Residual connections

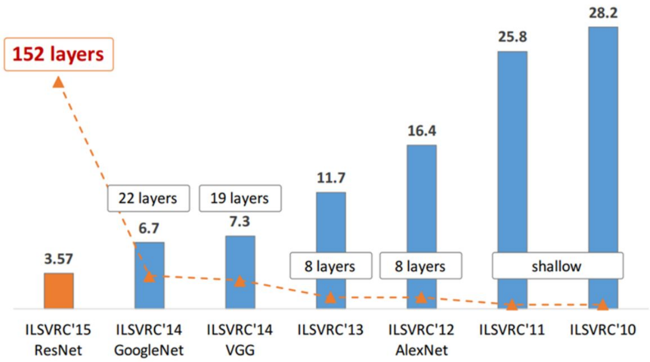
Activation function

Invariance: pooling and data augmentation

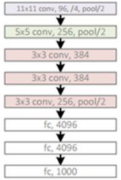
Optimization

Making it work

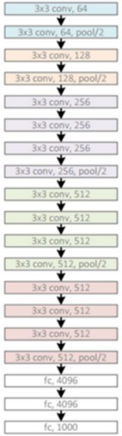
ImageNet classification top-5 error rate



AlexNet, 8 layers
(ILSVRC 2012)



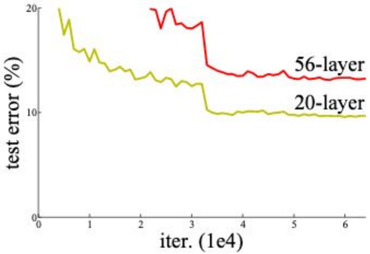
VGG, 19 layers
(ILSVRC 2014)

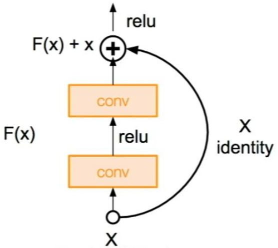


GoogleNet, 22 layers
(ILSVRC 2014)

CIFAR-10

Test error





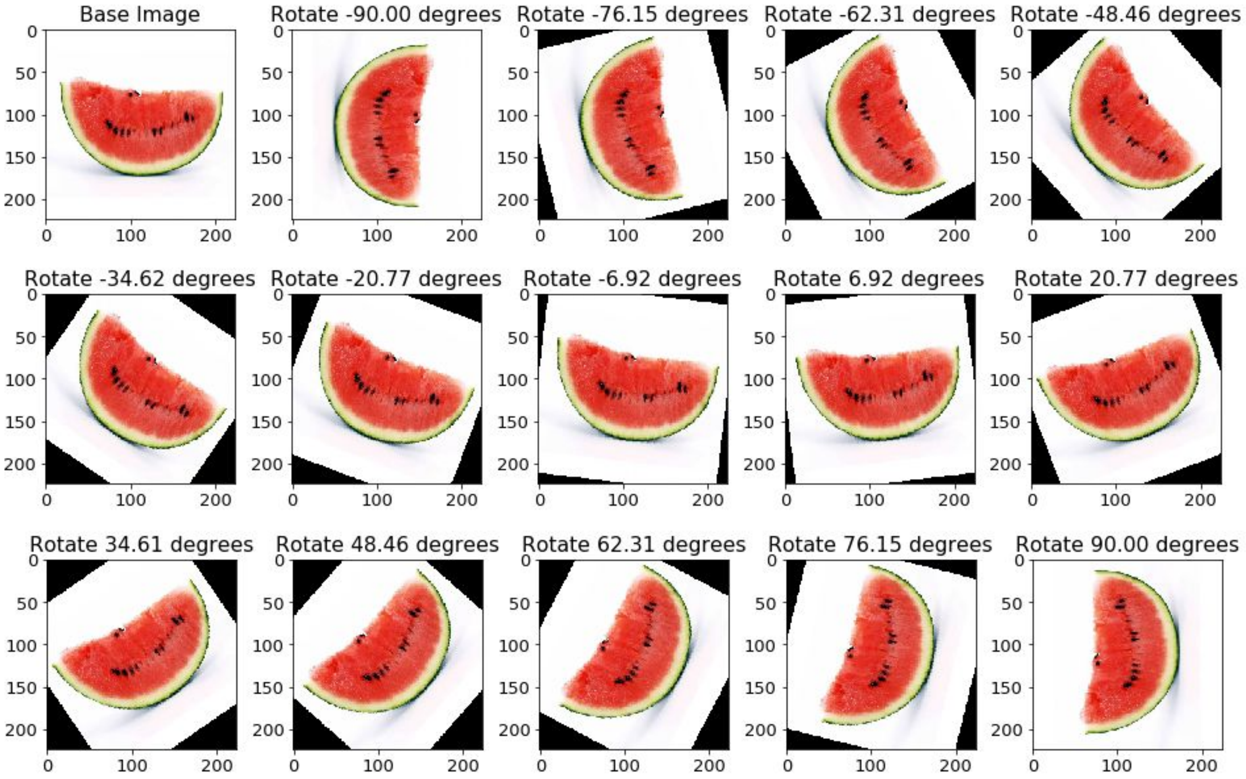
Residual block

It’s not overfitting- it’s optimization

He, Kaiming, et al. "Deep residual learning for image recognition." 2016

Making it work

Learned Invariance: Data augmentation



Rotation

Random cropping

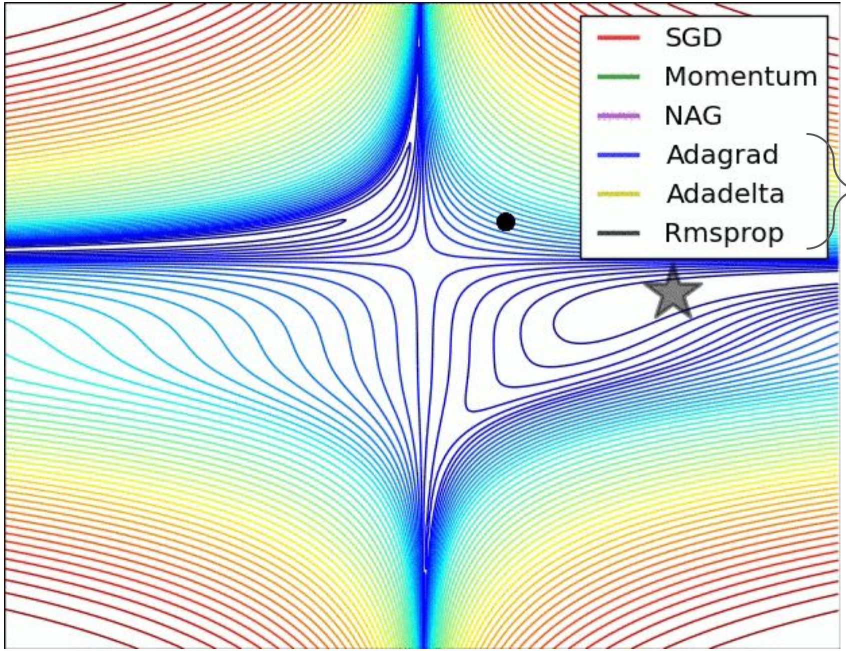
Mirroring

Color changes

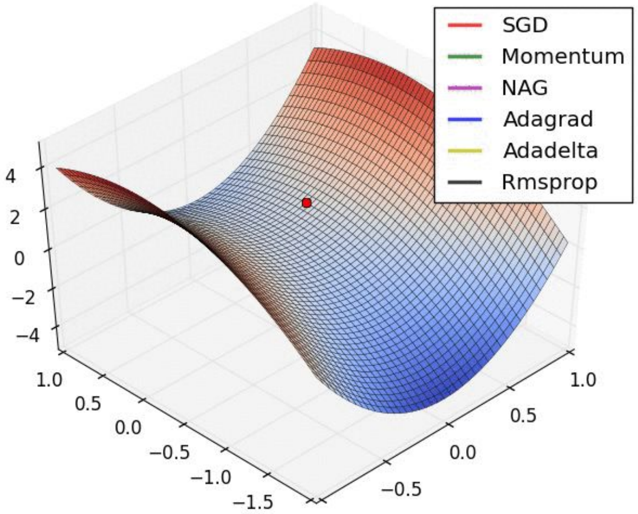
Noise

Making it work

Optimization



adaptive learning rate methods

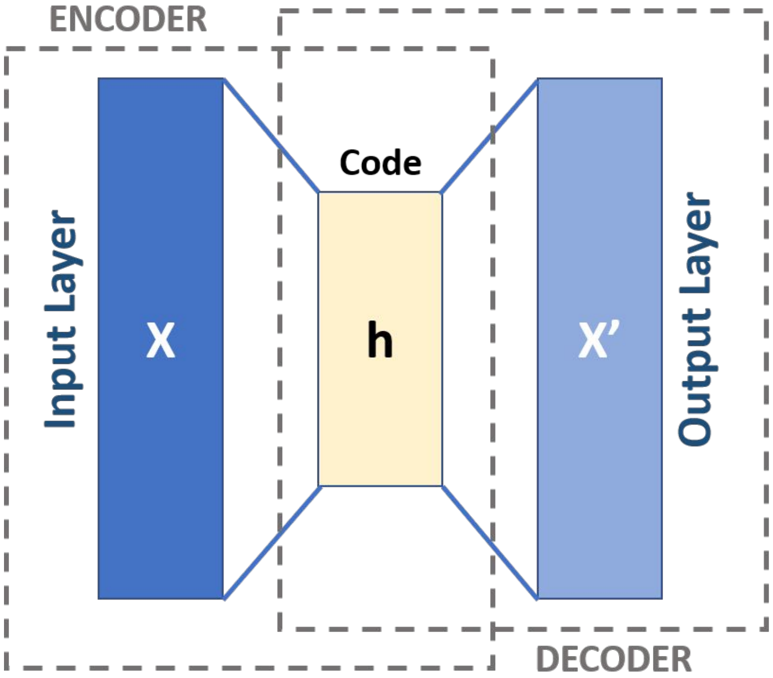


Adam: adds momentum to RMSprop
Kingma, D. P. and Ba, J. Adam: A method for stochastic optimization. arXiv preprint arXiv:1412.6980, 2014.

AdamW: changes to weight decay penalty
Loshchilov, I. and Hutter, F. Fixing weight decay regularization in adam. arXiv preprint arXiv:1711.05101, 2017

S. Ruder "An overview of gradient descent optimization algorithms"
https://ruder.io/optimizing-gradient-descent/index.html

Other structures: Autoencoder



- Dimensionality reduction
- Representation learning
- Denoising

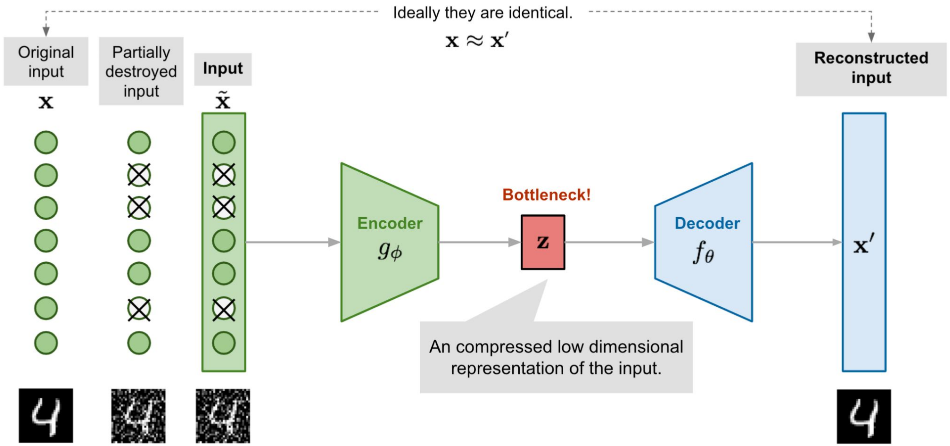
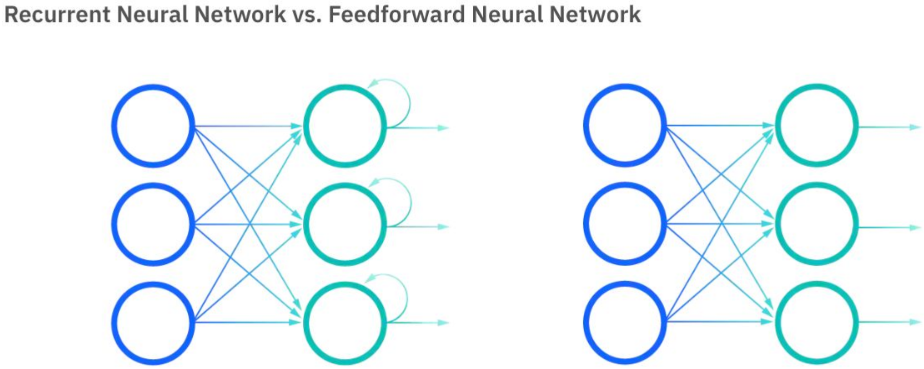


Image: https://en.wikipedia.org/wiki/Autoencoder

https://lilianweng.github.io/lil-log/2018/08/12/from-autoencoder-to-beta-vae.html

Other structures: Sequences

Recurrent Neural Networks



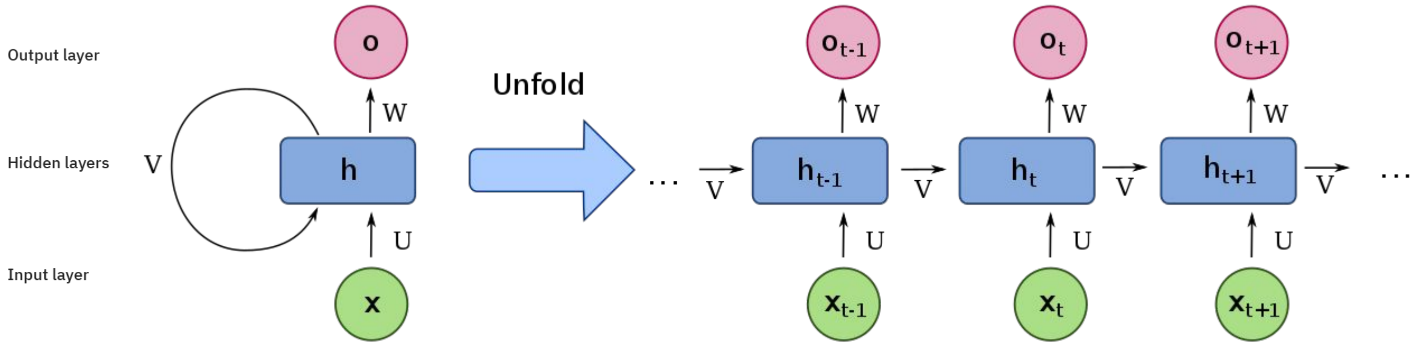
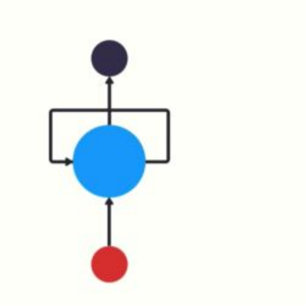
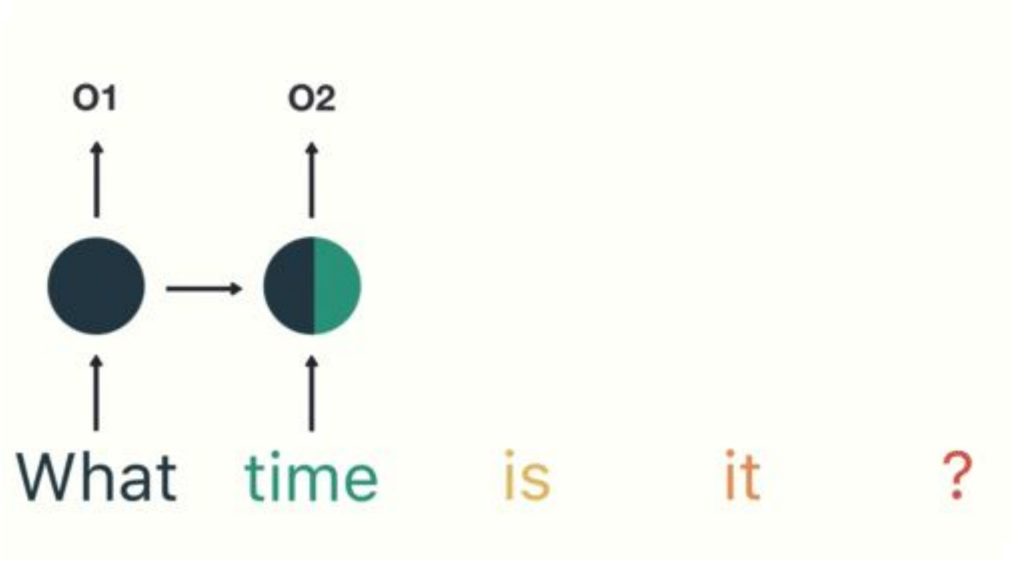


Image sources: ibm.com/cloud/learn/recurrent-neural-networks
Wikipedia "Recurrent Neural Network"

Other structures: Sequences

Recurrent Neural Networks



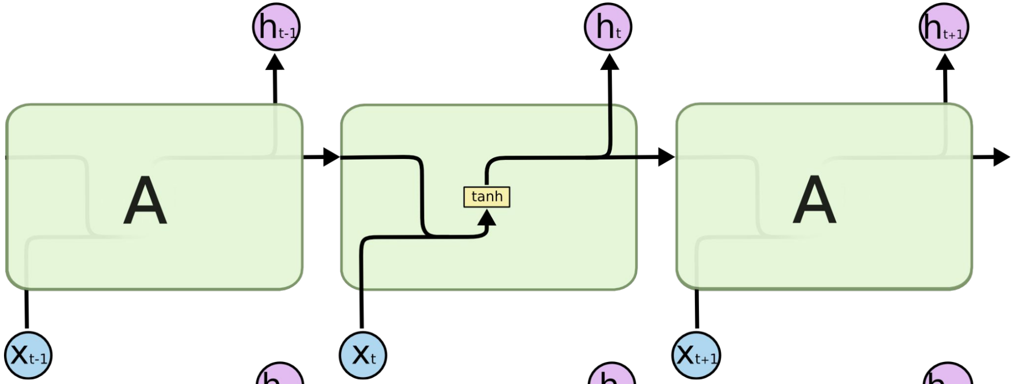


https://towardsdatascience.com/illustrated-guide-to-recurrent-neural-networks-79e5eb8049c9

Other structures: Sequences

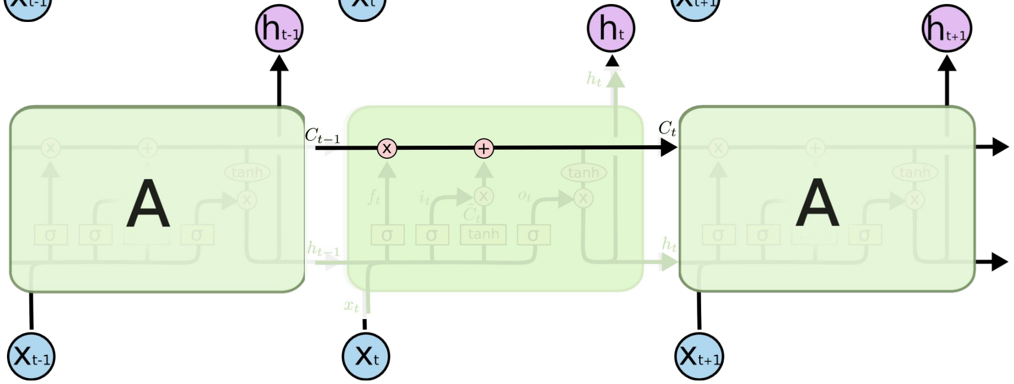
Recurrent Neural Networks

Original RNN



Long Short Term Memory

(LSTM) RNNs



https://colah.github.io/posts/2015-08-Understanding-LSTMs/

Other structures: Sequences

Recurrent Neural Networks

Bidirectional RNNs use the forward and reverse context of inputs

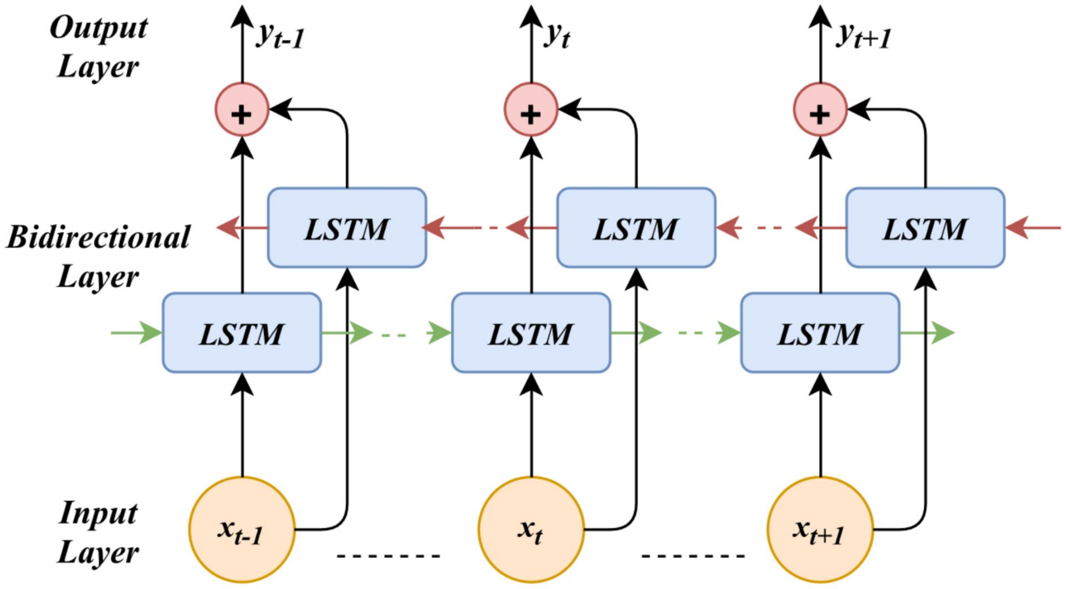
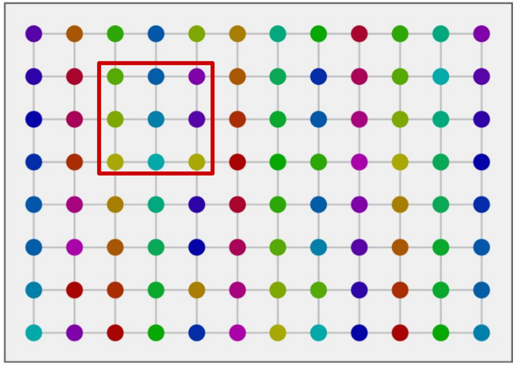
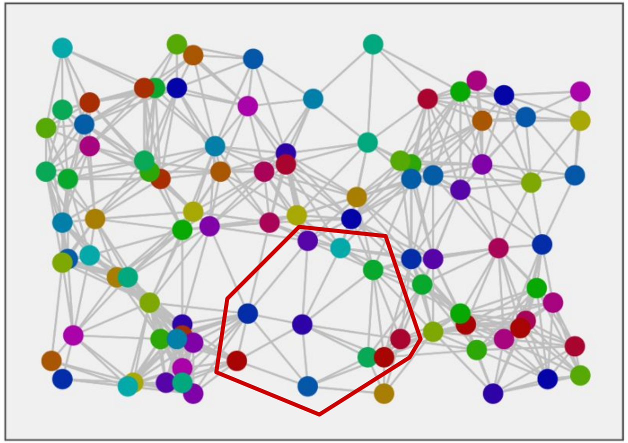


Image: I. K. Ihianle et al.: Deep Learning Approach for Human Activities Recognition From Multimodal Sensing Devices

Other structures

Graphs





Locality (neighborhood) varies for each node

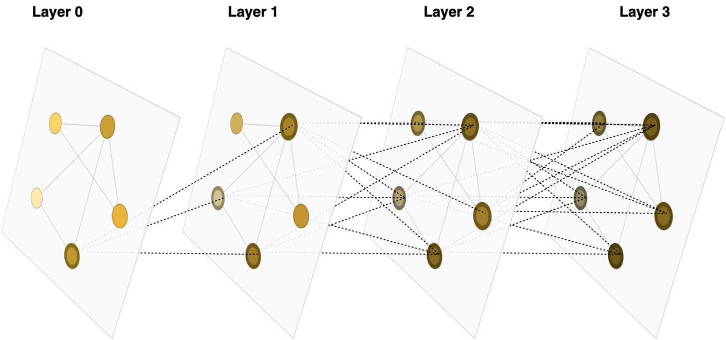


Image sources and further reading:
https://gnn.seas.upenn.edu
https://distill.pub/2021/gnn-intro/

Big ideas: CNNs

- Deep neural networks (DNNs) are neural networks with “many” hidden layers
- Convolutional Neural Networks (CNNs) are a kind of neural network with a topology that exploits structure in the input data to make learning easier/faster
  - Sparse interactions (“local receptive fields”)
  - Parameter/weight sharing
  - Translation invariance
- CNNs use the convolution operation
- CNNs are popular and ubiquitous
  - Datasets
  - Clearly defined tasks and evaluation
  - Computational tools
  - Methods and architectures

Big ideas: Universal

- Inductive bias limits your model hypothesis space and is a way to add what you know about the problem into the model
    - Structure can be a very useful inductive bias
- Implementation of successful models has “hidden” problems
- Many advances in ML driven by access to
    - Datasets
    - Clearly defined tasks and evaluation
    - Computational tools
    - Methods and architectures
    - Financial incentives
- Be mindful