ITMO

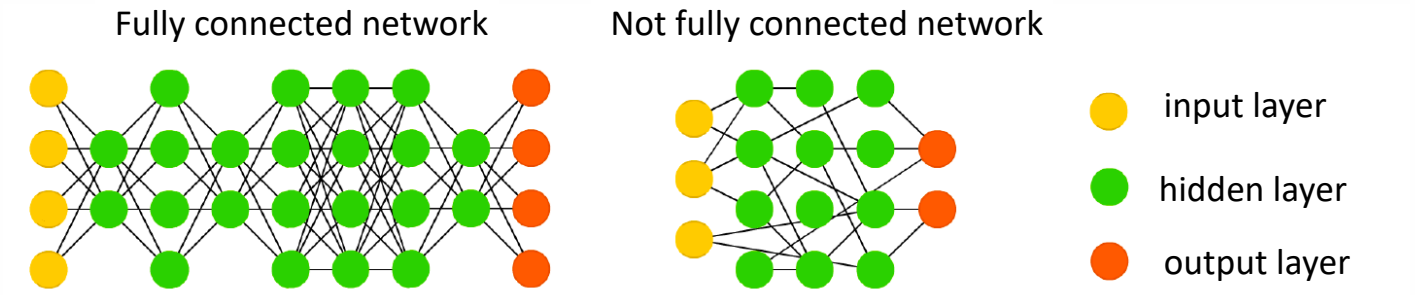
Convolutional Neural Networks

Computer Vision

Convolutional Neural Networks (CNN)

Deep Neural Networks (DNN)

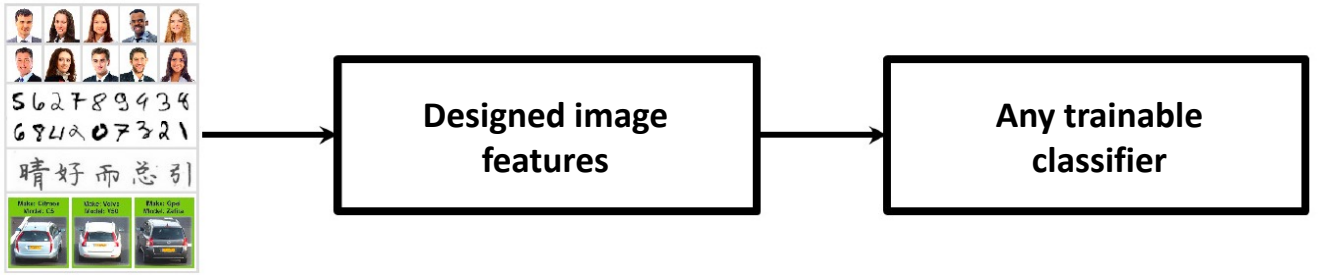
- 1965: first deep neural networks
- 2012: convolutional network for image classification AlexNet



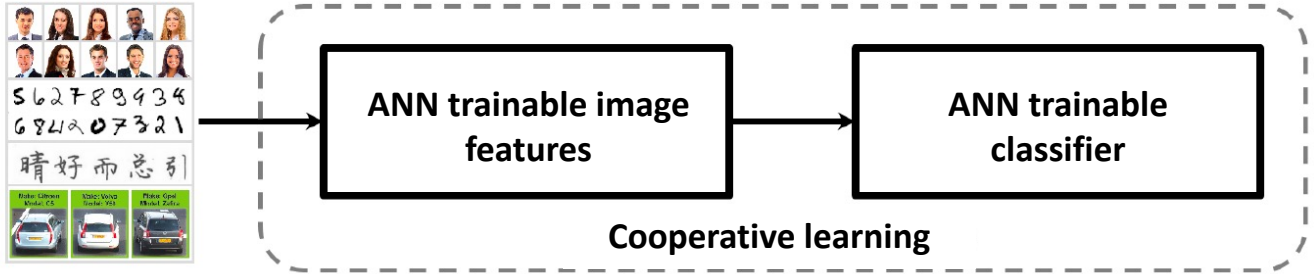
- Network architecture is a structure of connections between neurons that allows you to endow the DNN with the necessary properties.
- DNNs allow you to take as input and generate complexly structured data as output.

Deep Neural Networks (DNN)

- Classical approach to image recognition:

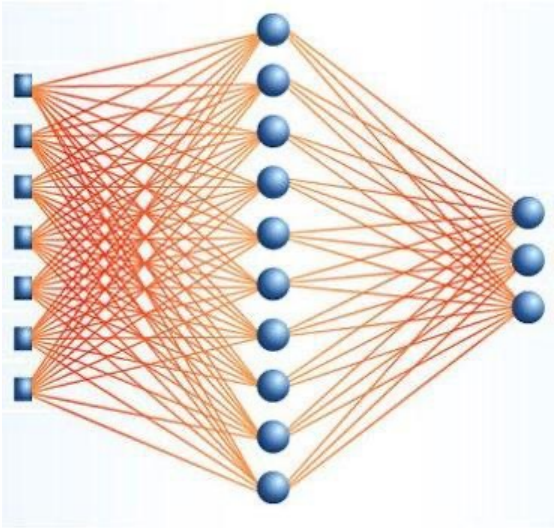


- Modern Approach – end-to-end deep learning:



Fully Connected Neural Networks

- Sometimes referred to as «classical» neural networks.



Fully Connected Neural Networks

ITMO

- Disadvantages of a fully connected multilayer network for image analysis:
  - Lots of weights for training.
  - The image is presented as a flat array – topology information is lost.
- Principles of Convolutional Neural Networks:
  - Local Perception.
  - Shared weights.
  - Dimensionality reduction.

NN vs. CNN

ITMO

- Convolutional Neural Networks are very similar to regular Neural Networks:
  - They are made up of neurons that have trainable weights and biases.
  - Each neuron receives some input, performs a dot product, and optionally accompanies it with a non-linearity.
  - The entire network still expresses a single differentiable evaluation function: from raw image pixels at one end to class scores at the other.
  - And they still have a loss function (like SVM/Softmax) on the last (fully connected) layer.
  - All the tips/tricks we covered for training regular neural networks still apply.

What are Different?

- CNN architectures were designed specifically for images.
- This allows us to code certain properties in the architecture.
- Through some transformations, convolutional networks can significantly reduce the number of network parameters.

Recall the Architecture of a Neural Network

- Neural networks take input (column vector) and transform it through a series of hidden layers.
- Each hidden layer consists of a set of neurons, where each neuron is fully connected to all the neurons of the previous layer.
- The neurons of one layer function completely independently and have no common connections.
- The last fully connected layer is called the «output layer» and in classification systems represents the class scores.

What problems do neural networks have with images?

- Poorly scaled to full images.
- In CIFAR-10, images are 32 × 32 × 3 (32 wide, 32 high, 3 color channels), so one fully connected neuron in the first hidden layer of a normal neural network would have $32 * 32 * 3 = 3072$ weights.
- This amount still seems to be acceptable, but this fully connected structure is not suitable for large images.
- For example, a larger image, such as 200 x 200 x 3, would result in neurons having $200 * 200 * 3 = 120{,}000$ weights. Moreover, we will almost certainly want to have several of these neurons, so the parameters will increase rapidly!
- Obviously, such full connectivity is wasteful, and a huge number of parameters will quickly lead to overfitting.

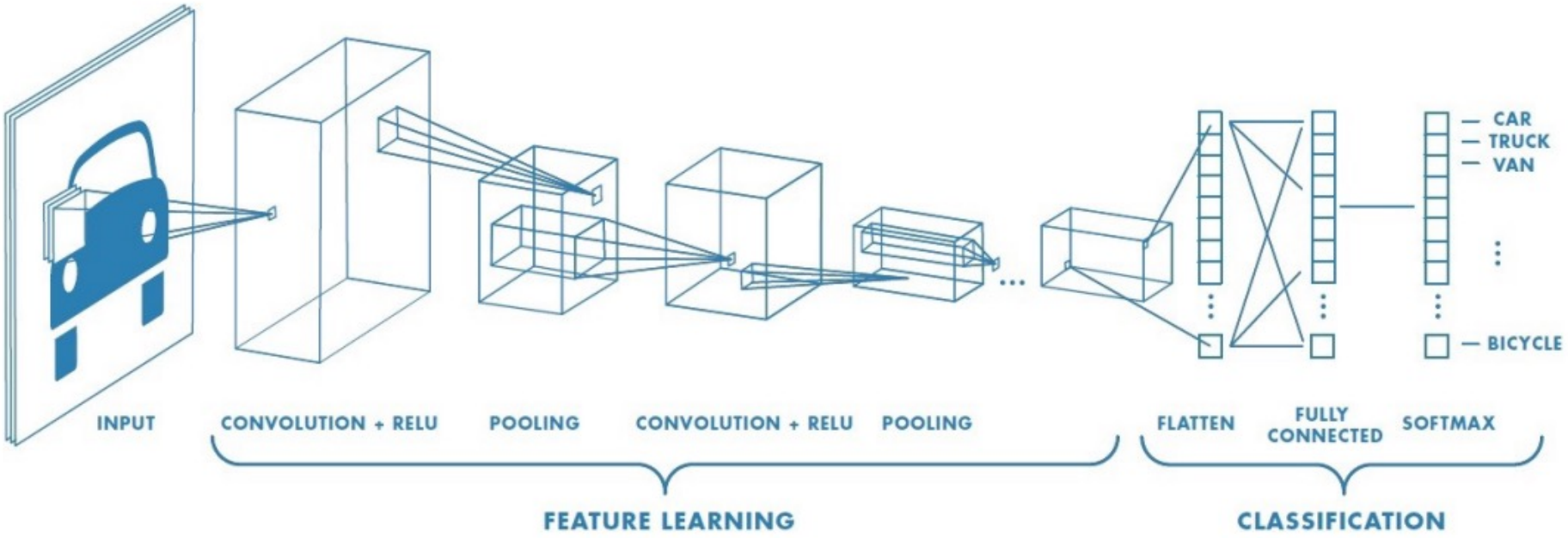
Convolutional Neural Networks (CNN)

- They take advantage of the fact that the input consists of images and constrain the architecture in a more reasonable way.
- Unlike a «classical» neural network, CNN layers have neurons arranged in three dimensions: width, height, depth.
- For example, the input images in CIFAR-10 are the input volume of activations, and this volume has dimensions of 32×32×3 (width, height, depth, respectively).
- As we'll see shortly, the neurons in a layer will only be connected to a small area of the previous layer, not all the neurons.
- Moreover, the final output layer for CIFAR-10 will be 1 × 1 × 10 because by the end of the CNN architecture we will reduce the entire image to a single vector of class scores along the depth dimension.

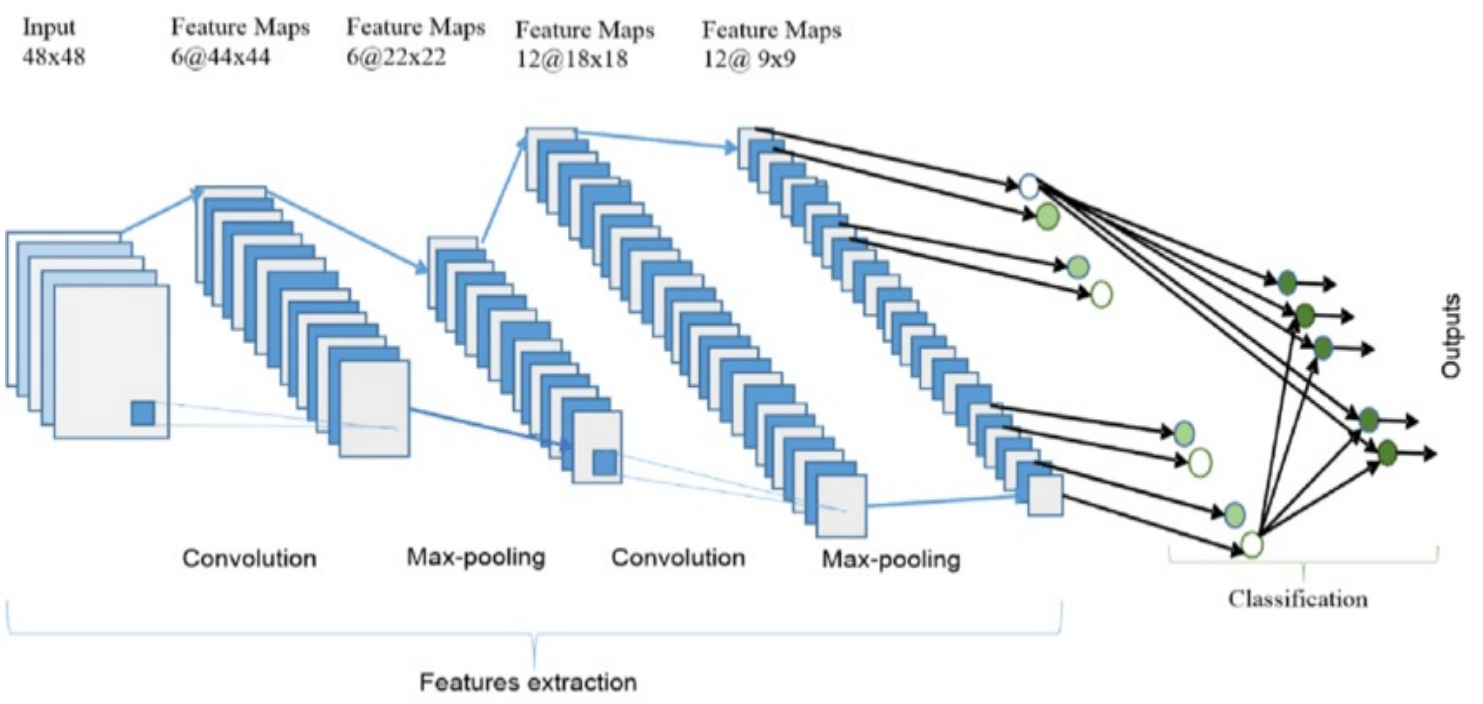
Convolutional Neural Networks (CNN)

- A simple CNN is a sequence of layers, and each layer of the CNN transforms one volume of activations into another one using a differentiable function.
- We use three main types of layers to build CNN architectures:
  - Convolutional layer,
  - Pooling layer,
  - Fully connected layer.
- We will add these layers to form the complete CNN architecture.

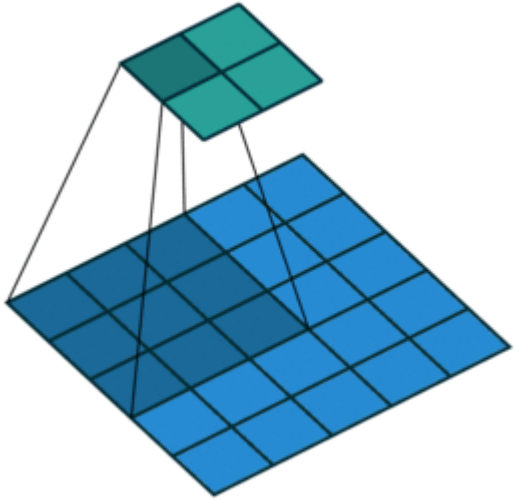
General Structure of CNN

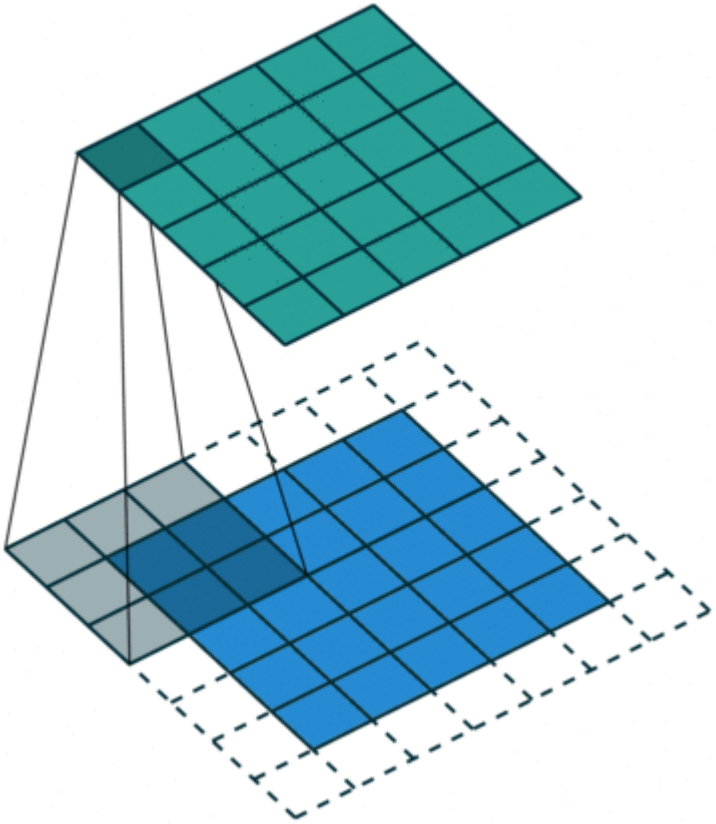


General Structure of CNN



Local Perception Principle





Simple CNN for CIFAR-10 Classification Structure

- INPUT $[32 \times 32 \times 3]$ will contain the pixels of an image, in this case an image 32 wide, 32 high, with three R,G,B color channels.
- The CONV layer will calculate the output of the neurons. The result can be a volume like $[32 \times 32 \times 12]$.
- The RELU layer will apply an element-wise activation function, such as $\max(0, x)$. This leaves the size of the volume unchanged ($[32 \times 32 \times 12]$).
- The POOL layer will perform a downsampling operation on the spatial dimensions (width, height), resulting in a volume like $[16 \times 16 \times 12]$.
- The FC (fully connected) layer computes class scores, resulting in a volume of size $[1 \times 1 \times 10]$, where each of the 10 numbers corresponds to a class score, for example, among 10 CIFAR-10 categories.

Convolutional Neural Networks (CNN)

- CNNs transform the original image layer by layer from the original pixel values to the final class scores.
- Note that some layers contain parameters while others do not.
- In particular, the CONV/FC layers perform transformations that are a function not only of activations in the input volume, but also of parameters (weights and biases of neurons).
- On the other hand, RELU/POOL layers implement a fixed function.
- The parameters of the CONV/FC layers will be trained using gradient descent so that the class scores that the CNN computes correspond to the labels in the training set for each image.

Convolutional Neural Networks (CNN)

ITMO

- The CNN architecture in the simplest case is a list of layers that transform the image volume into an output volume (e.g. containing class estimates).
- There are several different types of layers (e.g. CONV/FC/RELU/POOL are the most popular).
- Each layer takes an input 3D volume and converts it to an output 3D volume using a differentiable function.
- Each layer may or may not have parameters (e.g. CONV/FC does, RELU/POOL does not).
- Each layer may or may not have additional hyperparameters (e.g. CONV/FC/POOL has, RELU does not).

Convolutional Layer

Convolutional Layer

ITMO

- The CONV layer (C-layer) is the most important building block of a CNN and does most of the computational work.
- The CONV layer parameters consist of a set of trainable filters.
- Each filter is small spatially (width and height), but extends to the entire depth of the input volume:
  - For example, a typical filter on the first layer of a CNN might be 5x5x3 (i.e. 5 pixels in width and height, and 3 because images have a depth of 3, color channels).
  - During the forward pass, we shift (more precisely, collapse) each filter by the width and height of the input volume and calculate the dot products between the filter and input elements at any position.

Image Convolutional Algorithm

- Let an image be given in the format of a matrix of numbers X and a square matrix *W*:

$$X = \begin{bmatrix} 1 & 1 & 1 & 0 & 0 \\ 0 & 1 & 1 & 1 & 0 \\ 0 & 0 & 1 & 1 & 1 \\ 0 & 0 & 1 & 1 & 0 \\ 0 & 1 & 1 & 0 & 0 \end{bmatrix}$$

$$W = \begin{bmatrix} 1 & 0 & 1 \\ 0 & 1 & 0 \\ 1 & 0 & 1 \end{bmatrix}$$

Anchor

Kernel

- Then the convolution operation is defined by the following expression:

$$y_{i,j} = \sum_{s=1}^{K} \sum_{t=1}^{K} \left(w_{s,t} x_{((i-1)+s,(j-1)+t)} \right),$$

where $w_{s,t}$ is the value of the convolution kernel element at position *(s,t)*,

$y_{i,j}$ is the pixel value of the output image,

$x_{((i-1)+s,(j-1)+t)}$ is the pixel value of the source image,

K is the size of the convolution kernel.

Image Convolutional Algorithm

$X[i,j]$ – initial features, pixels of $n \times m$ image,

w_{ab} – kernel, $a = -A, \dots, +A$, $b = -B, \dots, +B$

$$(x * w)[i,j] = \sum_{a=-A}^{A} \sum_{b=-B}^{B} w_{ab} \times [i+a, j+b]$$

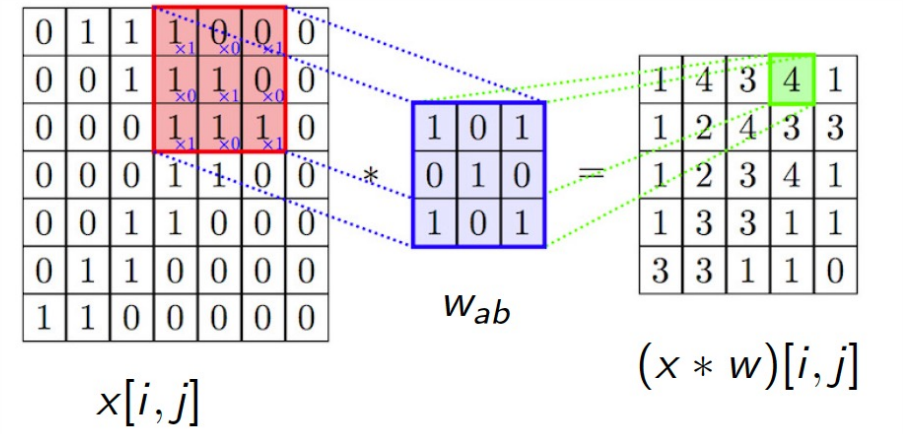


Image Convolutional Algorithm

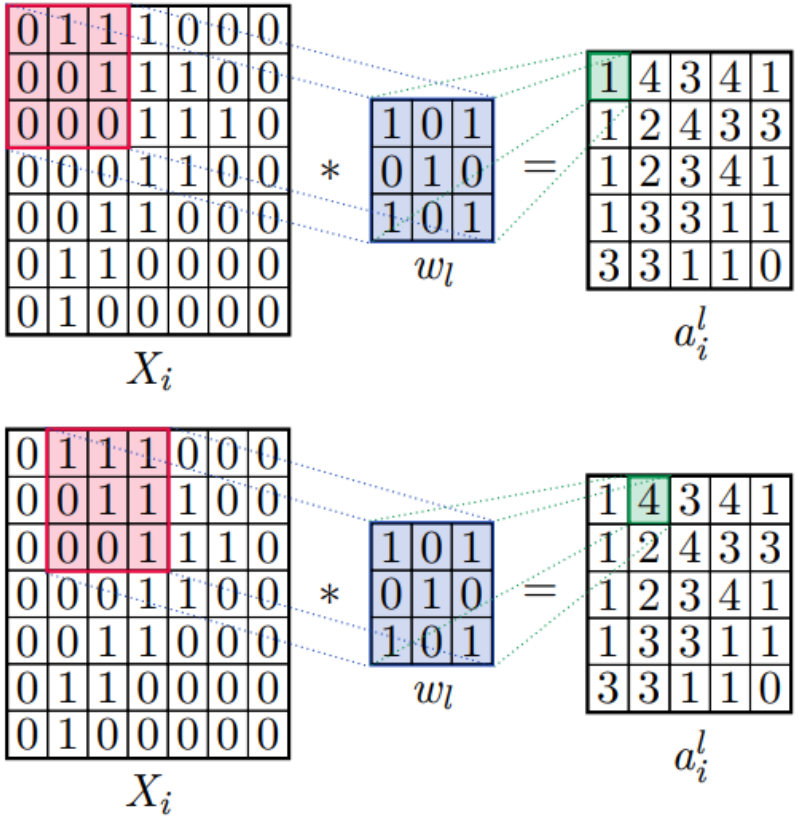
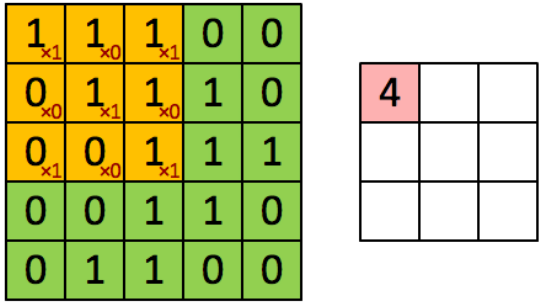


Image Convolutional Algorithm

- Let an image be given in the format of a matrix of numbers X and a square matrix *W*:



- Then the convolution operation is defined by the following expression:

$$y_{i,j} = \sum_{s=1}^{K} \sum_{t=1}^{K} \left(w_{s,t} x_{((i-1)+s,(j-1)+t)} \right),$$

where $w_{s,t}$ is the value of the convolution kernel element at position *(s,t)*,

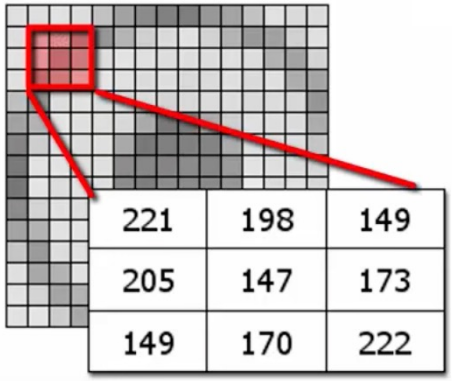
$y_{i,j}$ is the pixel value of the output image,

$x_{((i-1)+s,(j-1)+t)}$ is the pixel value of the source image,

K is the size of the convolution kernel.

Image Convolutional Algorithm

İTMO



Convolutional Kernel

-1	0	1
-2	0	2
-1	0	1

$$N(x,y) = 221 * (-1) + 198 * 0 + 149 * 1 + 205 * (-2) + 147 * 0 + 173 * 2 + 149 * (-1) + 170 * 0 + 222 * 1 = -63$$

Popular Image Convolutional Kernels

İTMO

Blur

1/9	1/9	1/9
1/9	1/9	1/9
1/9	1/9	1/9

Edge detection

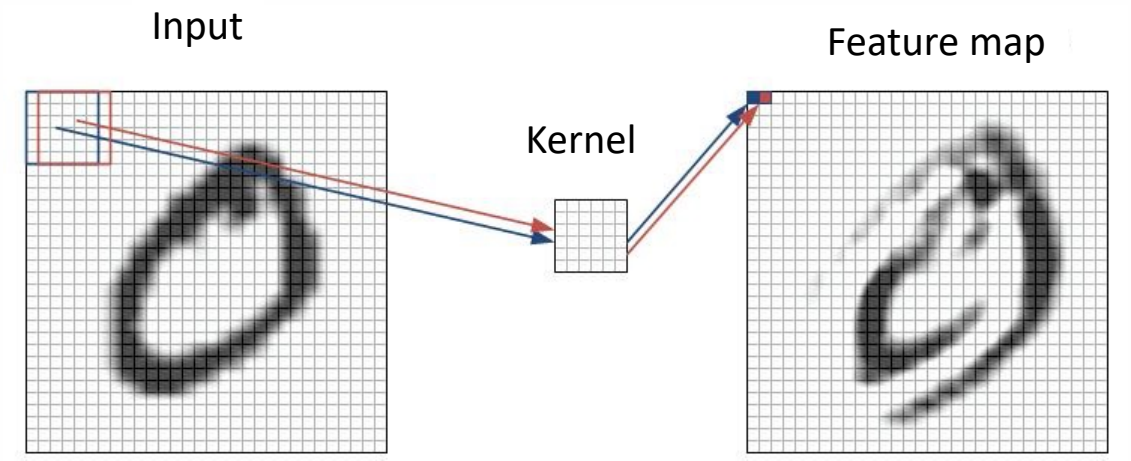
0	-1	0
-1	4	-1
0	-1	0

Sharpen enhancement

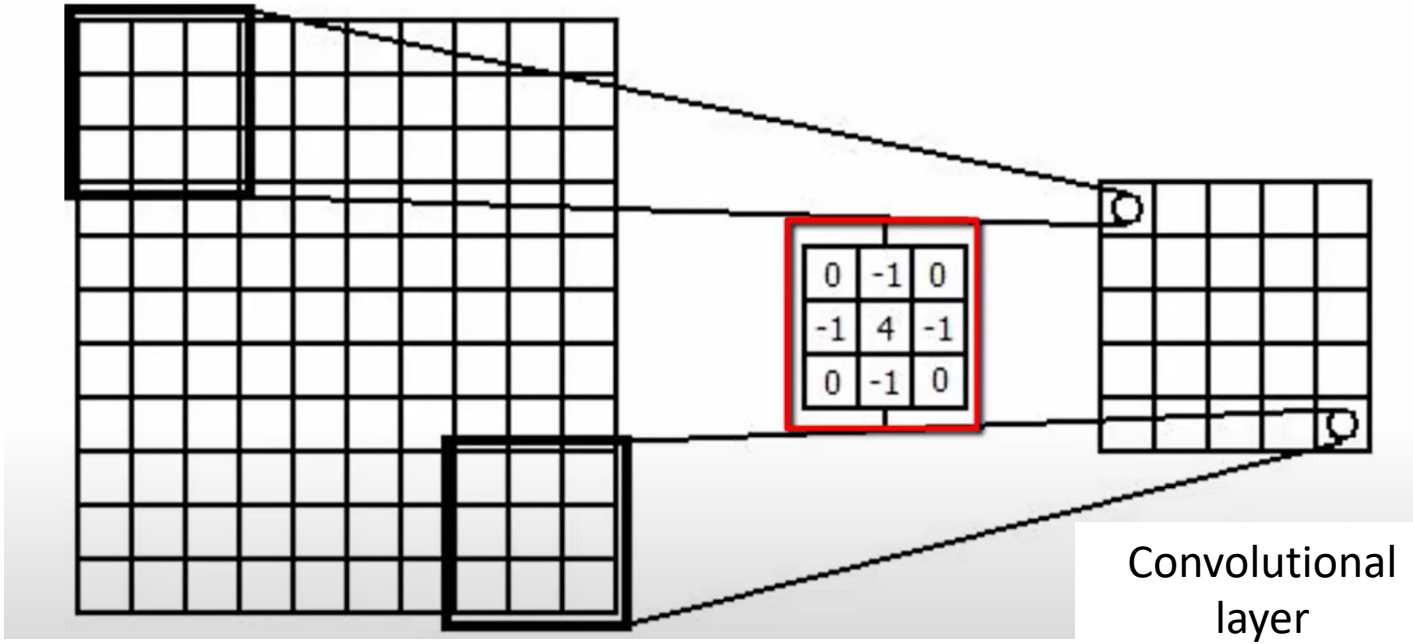
0	-1	0
-1	5	-1
0	-1	0

- In neural networks, convolution kernels are determined automatically during training!

Signal Propagation Process in Convolutional Layer



Separate Weights



Convolutional Layer

ITMO

- As the filter progresses across the width and height of the input volume, we will create a 2D activation map that gives the responses of this filter at each spatial location.
- The network will learn filters that are activated when they see some visual feature, such as an edge of a certain orientation or a patch of a certain color on the first layer, or eventually whole honeycombs or wheel-like patterns on the higher layers of the network.
- Now we will have a whole set of filters in each CONV layer (for example, 12 filters), and each of them will create a separate two-dimensional activation map.
- We will add these activation maps along the depth dimension and get the output volume.

Convolutional Layer

ITMO

- When dealing with high-dimensional inputs such as images, it is impractical to connect neurons to all neurons in the previous volume.
- Instead, we will only connect each neuron to a local area of the input volume.
- The spatial extent of this connection is a hyperparameter called the neuron's receptive field (equivalently, this is the filter size).
- The extent of the connection along the depth axis is always equal to the depth of the input volume.
- It is important to re-emphasize this asymmetry in how we treat spatial dimensions (width and height) and depth dimensions: relationships are local in 2D space (width and height), but always complete across the entire depth of the input volume.

Convolutional Layer

- **Example 1:**
  - Assume that the input volume is [32x32x3], (e.g. RGB CIFAR-10 image).
  - If the receptive field (or filter size) is 5x5, then each neuron in the CONV layer will have weights for the [5x5x3] region in the input volume, for a total of 5*5*3 = 75 weights (and +1 bias (stride) parameter).
  - Note that the degree of connectivity along the depth axis should be 3, as this is the depth of the input volume.
- **Example 2:**
  - Suppose the input volume is [16x16x20].
  - Then, using the example of a 3x3 receptive field, each neuron in the CONV layer will have a total of 3*3*20 = 180 connections to the input volume.
  - Note that, again, the relationship is local in 2D space (e.g. 3x3), but complete in the entry depth (20).

Convolutional Layer

ITMO

- Three hyperparameters control the size of the output volume: depth, width, and zero-padding:
  - The depth of the output volume corresponds to the number of filters that we would like to use, each of which learns to look for something different in the input.
    - For example, if the first CONV receives a raw image as input, then different depth-sensing neurons may fire in the presence of different oriented edges or color patches.
    - We will call the set of neurons that look at the same area of the input signal the depth column.
  - It is necessary to specify the step with which we will shift the filter.
    - If the stride is 1, then we move the filters one pixel at a time.
    - If the stride is 2 (or, in rare cases, 3 or more, although this rarely happens in practice), then the filters move 2 pixels at a time when we move them.
    - In this case, the volume of the output signal will be smaller in terms of space.

Convolutional Layer

ITMO

- Sometimes it is convenient to fill the input volume with zeros along the entire border (zero-padding).
  - The size of this zero pad is a hyperparameter.
  - A nice feature of zero padding is that it allows us to control the spatial size of the output volumes.

Convolutional Layer

- We can compute the spatial size of the output volume as a function of:
  - input volume size (W),
  - the size of the receptive field of neurons in the CONV layer (F),
  - the stride with which they are applied (S),
  - the number of zeros used (P) on the boundary.
- Formula for calculating how many neurons «fit»:

$$\frac{(W - F + 2P)}{S + 1}.$$

- For example, for a 7x7 input and a 3x3 filter with stride 1 without zero-padding, we get a 5x5 output.
- With stride 2 we will get a 3x3 output.

Convolutional Layer

- Using zero padding.
  - When we want to keep the output dimension equal to the input dimension, we can perform zero padding.
  - If our receptive fields were 3, stride 1, and we used a zero padding of 1, then for input 5, the output would be 5.
  - If there were no zero padding, then the output volume would have a spatial dimension of only 3, because that is how many neurons would “fit” in the original input.
  - In the general case, padding with zeros should be chosen from the relation:

$$P = \frac{F - 1}{2}.$$

  - With stride $S = 1$, this calculation ensures that the input and output volumes will have the same size in space.

Convolutional Layer

- Stride limits:
  - Spatial location hyperparameters have mutual constraints.
  - For example, when the input data has size W = 10 zero padding is not used $P = 0$ and filter size $F = 3$ then it will not be possible to use stride $S = 2$ because

$$\frac{\mathrm{W} - \mathrm{F} + 2\mathrm{P}}{\mathrm{S} + 1} = \frac{10 - 3 + 0}{2 + 1} = 4.5,$$

those not an integer, indicating that the neurons are not symmetrical at the input.

  - Therefore, this setting of hyperparameters is considered incorrect.
  - Sizing the CNN, so that all measurements "work" can be a real headache, which is greatly facilitated using zero pads and some design guidelines.

Convolutional Layer

- The architecture of Krizhevsky et al., which won the ImageNet competition in 2012, accepted [227x227x3] size images.
  - The first CONV used neurons with receptive field size $F = 11$, stride $S = 4$, and no zero padding $P = 0$.
  - Since $(227 - 11)/4 + 1 = 55$, and since the Conv layer has depth $K = 96$, the output volume of the Conv layer was [55x55x96].
  - Each of the 55*55*96 neurons in this volume was connected to a [11x11x3] area in the input volume.
  - Moreover, all 96 neurons in each depth column are connected to the same region [11x11x3] of the input volume, but with different weights.
  - If you read the article, it claims that the input images were 224x224, which is not true, because $(224 - 11)/4 + 1$ is clearly not an integer. This confused many, and little is known about what happened.

Convolutional Layer

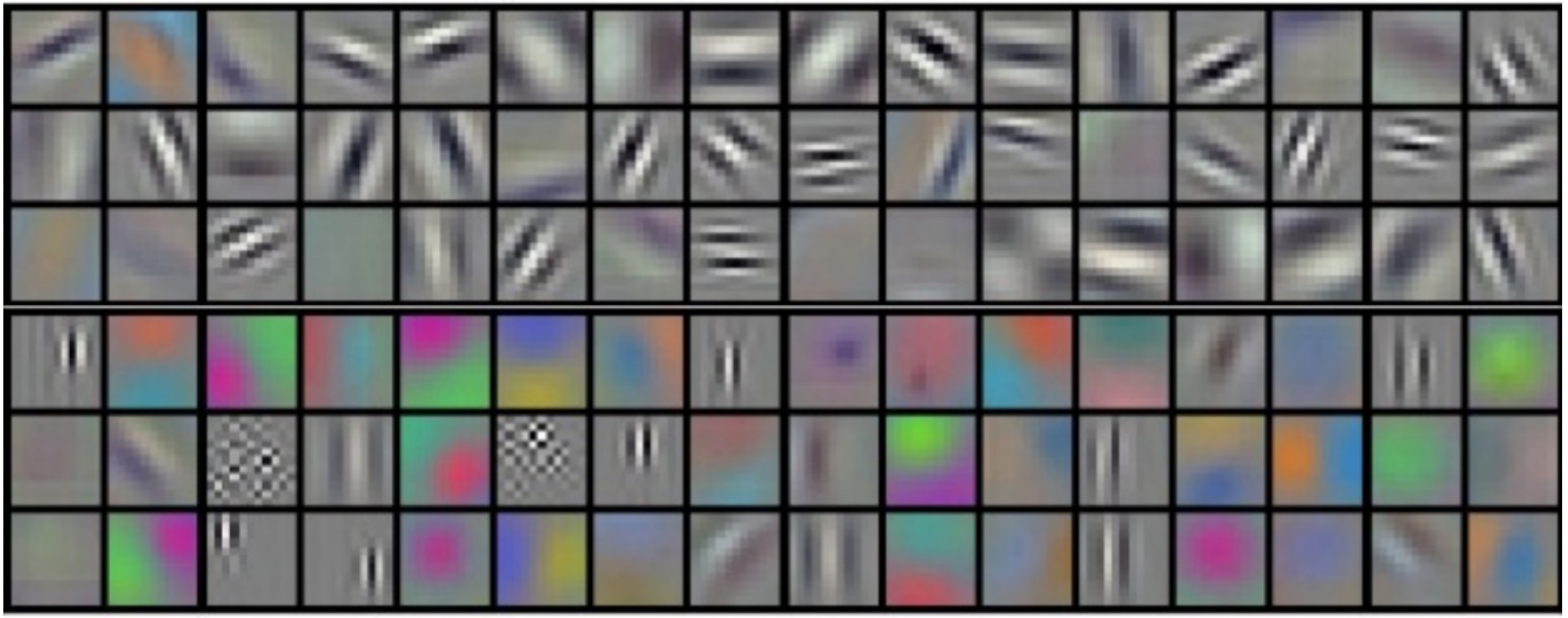
- A parameter separation scheme is used in CONV to control the number of parameters.
- In the example above, we see that the first CONV has 55*55*96 = 290,400 neurons, and each of them has 11*11*3 = 363 weights and 1 stride.
- All together, this gives 290,400 * 364 = 105,705,600 parameters on the first layer of the CNN alone. Obviously, this number is very large.
- It turns out that we can significantly reduce the number of parameters by making one reasonable assumption:
  - If it is useful to calculate one feature in a certain spatial position (x, y), then it is also useful to calculate it in another position (x_2, y_2).
  - In other words, by designating a single 2D depth slice as a depth slice (for example, a [55x55x96] volume has 96 depth slices, each [55x55]), we are going to restrict the neurons in each depth slice to use the same weights and strides.

Convolutional Layer

- With this parameter splitting scheme, the first CONV layer in our example will now only have 96 unique weight sets (one for each depth slice), for a total of 96*11*11*3 = 34,848 unique weights, or 34,944 parameters (+96 strides).
- Alternatively, all 55*55 neurons in each depth slice will now use the same parameters.
- In practice, with backpropagation, each neuron in the volume computes a gradient for its weights, but these gradients are summed over each depth slice and update only one set of weights per slice.

- Note that if all neurons in the same depth slice use the same weight vector, then the forward pass of the CONV layer in each depth slice can be computed as the convolution of the neuron weights with the input volume (hence the name: convolutional layer).
- That is why the sets of weights are usually called the filter (or kernel), which is convolved with the input signal.

Convolutional Layer



Examples of Learned Filters

Convolutional Layer

- Each of the 96 filters has a size of [11x11x3], and each of them is separated by 55*55 neurons in one depth slice.
- If horizontal edge detection is important at some point in the image, then intuitively it should be useful elsewhere due to the translation-invariant structure of images.
- Therefore, there is no need to re-learn how to define a horizontal edge at each of the 55*55 individual locations in the output volume of the CONV layer.

Convolutional Layer

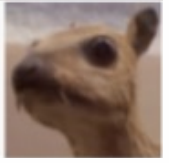
ITMO

- Sometimes the assumption of parameter separation may not make sense. This is especially true when the input images for the CNN have a certain centered structure, where we should expect, for example, that completely different features should be learned on one side of the image than on the other.
- One practical example is when the input is faces centered in the image.
- It can be expected that different eye- or hair-specific features can (and should) be learned at different spatial points.
- In this case, it is common to relax the parameter separation scheme and instead just call the layer a locally-coupled layer.

Image Convolutional Kernels

İTMO

Operation	Filter	Convolved Image
Identity	$\begin{bmatrix} 0 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix}$	
Edge detection	$\begin{bmatrix} 1 & 0 & -1 \\ 0 & 0 & 0 \\ -1 & 0 & 1 \end{bmatrix}$	
	$\begin{bmatrix} 0 & 1 & 0 \\ 1 & -4 & 1 \\ 0 & 1 & 0 \end{bmatrix}$	
	$\begin{bmatrix} -1 & -1 & -1 \\ -1 & 8 & -1 \\ -1 & -1 & -1 \end{bmatrix}$	

Sharpen	$\begin{bmatrix} 0 & -1 & 0 \\ -1 & 5 & -1 \\ 0 & -1 & 0 \end{bmatrix}$	
Box blur (normalized)	$\frac{1}{9}\begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix}$	
Gaussian blur (approximation)	$\frac{1}{16}\begin{bmatrix} 1 & 2 & 1 \\ 2 & 4 & 2 \\ 1 & 2 & 1 \end{bmatrix}$	

Pooling Layer

Pooling Layer

ITMO

- It is common in CNN architecture to periodically insert a pooling layer between successive CONV layers.
- Its function is to gradually reduce the spatial size of the representation in order to reduce the number of parameters and calculations in the network, and therefore to control overfitting.
- The pooling layer operates independently on each depth slice of the input signal and resizes its spatial size using the MAX or AVG operation.
- The most common form is a pooling layer with 2x2 filters applied in increments of 2 that reduces each input depth slice by 2 in width and height, discarding 75% of the activations.

Pooling Layer

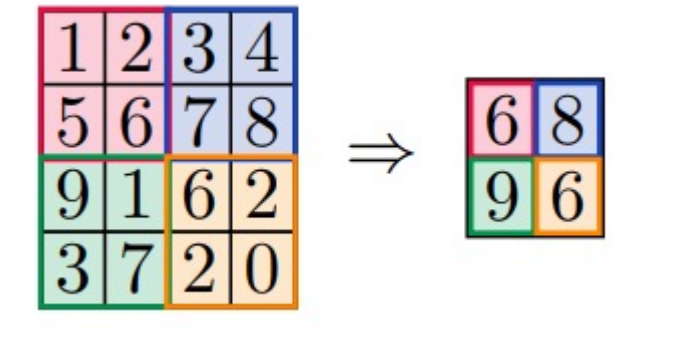
- Each MAX operation in this case will take at most over 4 numbers (a small 2x2 area in some depth slice).
- The depth dimension remains unchanged.
- More generally, a pooling layer:
  - Accepts $W_1 \times H_1 \times D_1$ volume.
  - Two hyperparameters are required:
    - spatial extent *F*,
    - stride *S*,
  - Produces volume $W_2 \times H_2 \times D_2$ where:

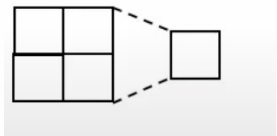
$$W_2 = \frac{(W_1 - F)}{S} + 1$$
$$H_2 = \frac{(H_1 - F)}{S} + 1$$
$$D_2 = D_1$$

  - Introduces zero parameters as it computes a fixed input function.

Pooling Layer

- It is not customary for pooling layers to pad the input with zeros.
- It is worth noting that in practice there are only two commonly encountered variations of the max pooling layer:
  - pooling layer with $F = 3$, $S = 2$ (also called overlapping pooling),
  - and more often $F = 2$, $S = 2$.
- Pool sizes with large receptive fields are too destructive.





Pooling Layer

- Let an image be given in the format of a matrix of numbers X and it is established that the size of the subsampling window is $k \times k$:

$$X = \begin{array}{|c|c|c|c|} \hline 1 & 3 & 2 & 0 \\ \hline 0 & 5 & 1 & 1 \\ \hline 3 & 0 & 1 & 4 \\ \hline 0 & 0 & 7 & 1 \\ \hline \end{array}$$

- Then the subsampling operation is defined by the expression:

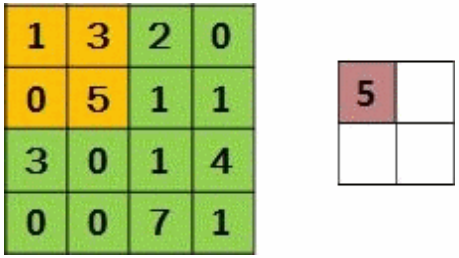
$$y_{(i,j)} = \max\left(x_{(ik+s,jk+t)}\right),$$

where $y_{(i,j)}$ is the pixel value of the output image,

$x_{(ik+s,jk+t)}$ is the pixel value of the source image.

Pooling Layer

- Let an image be given in the format of a matrix of numbers X and it is established that the size of the subsampling window is $k \times k$:



- Then the subsampling operation is defined by the expression:

$$y_{(i,j)} = \max\left(x_{(ik+s,jk+t)}\right),$$

where $y_{(i,j)}$ is the pixel value of the output image,

$x_{(ik+s,jk+t)}$ is the pixel value of the source image.

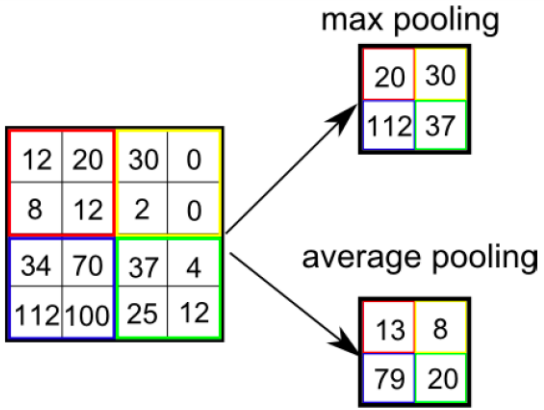
Pooling Layer

- A merging neuron is an untrainable convolution with a stride $h > 1$, aggregating the data of a rectangular area $h \times h$:

$$y[i,j] = F(x[hi, hj], \dots, x[hi + h - 1, hj + h - 1]),$$

where F is an aggregation function: max, average, etc.

- max-pooling allows you to detect an element in any of the cells.

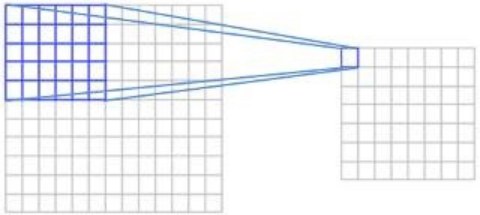


Architectures of CNN

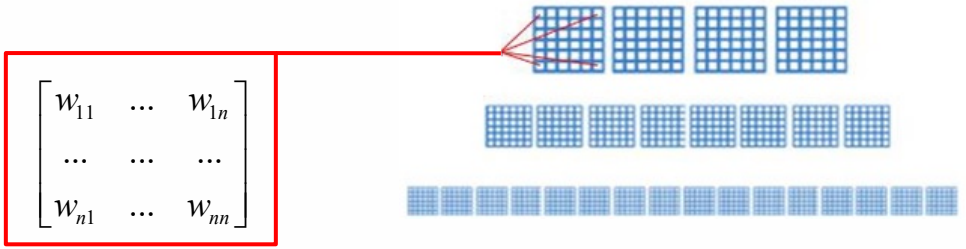
Paradigms of CNN

- Any CNN architecture includes 3 main paradigms:

  1. Local perception:

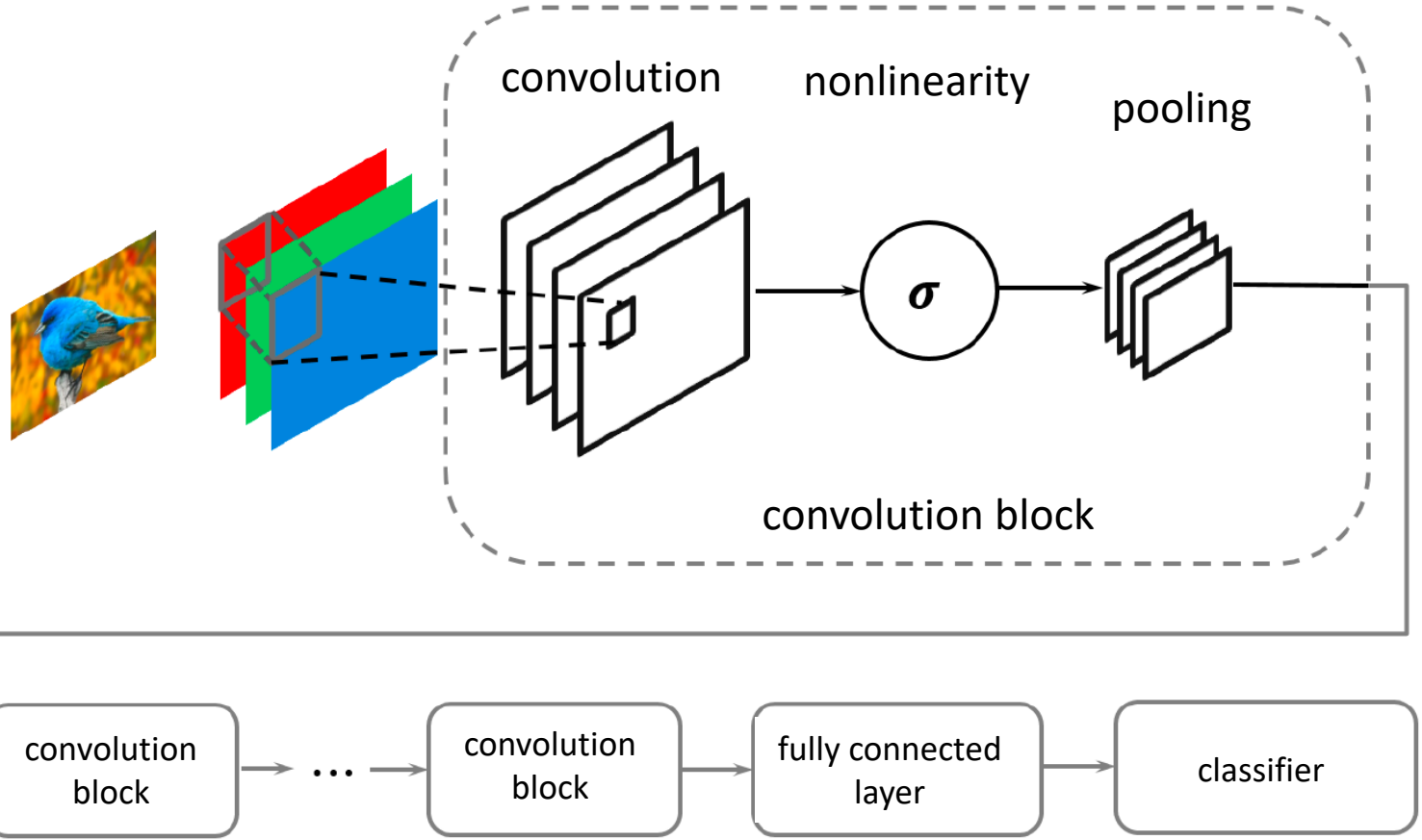


  2. Shared weights:



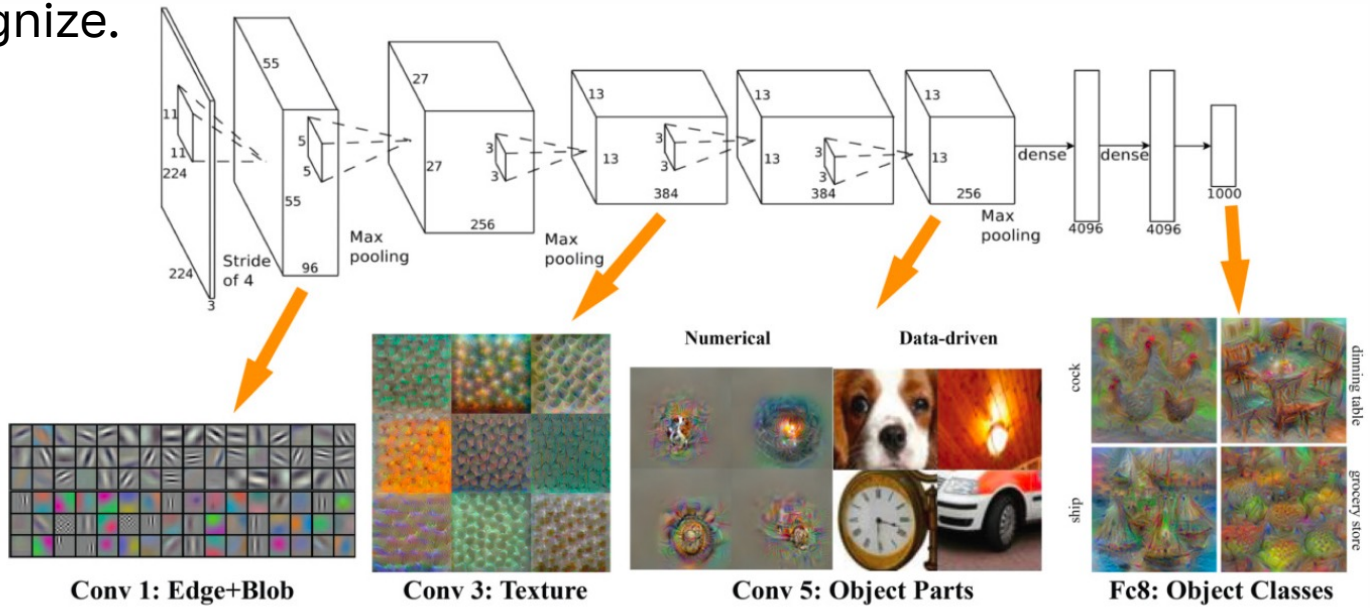
  3. Subsampling:
     - This increases the resistance of the input signal to displacements and slight deformations.

Classical CNN Diagram



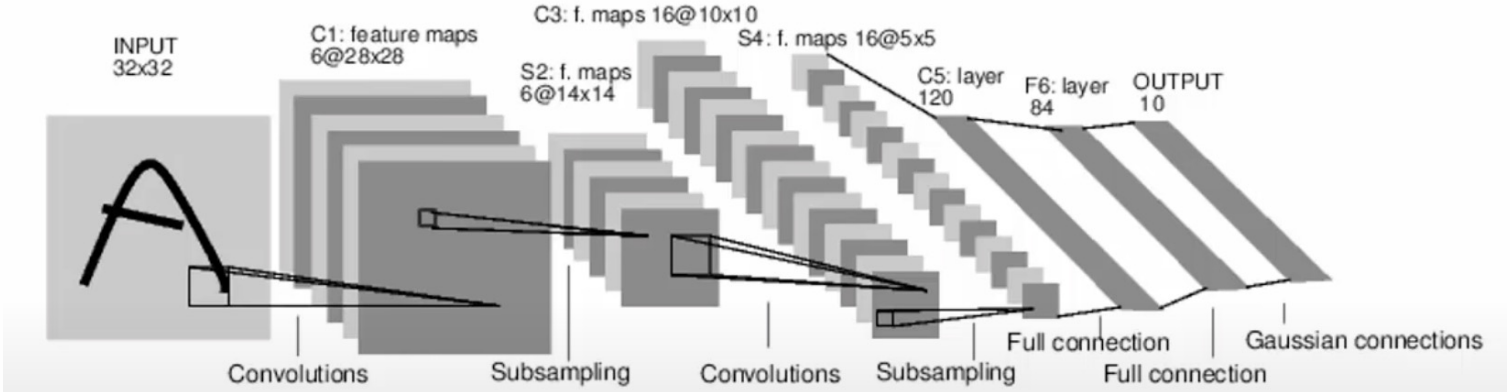
CNN Features Extraction

- The higher the layer, the larger and more complex image elements it can recognize.



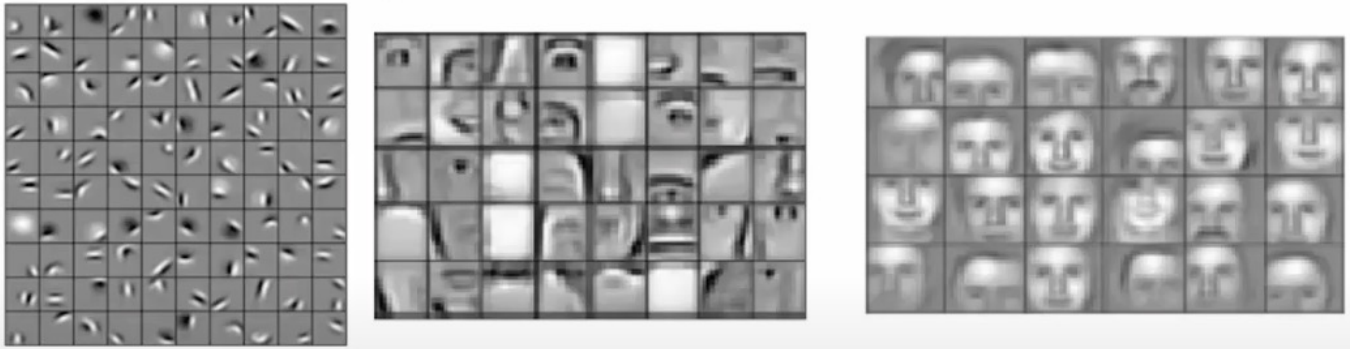
Krizhevsky A., Sutskever I., Hinton G. ImageNet Classification with Deep Convolutional Neural Networks. 2012.

CNN LeNet-5



Back-Propagation Applied to Handwritten Zip Code Recognition / *Y. LeCun, B. Boser, J. S. Denker et al.* 1989

CNN for Face Recognition

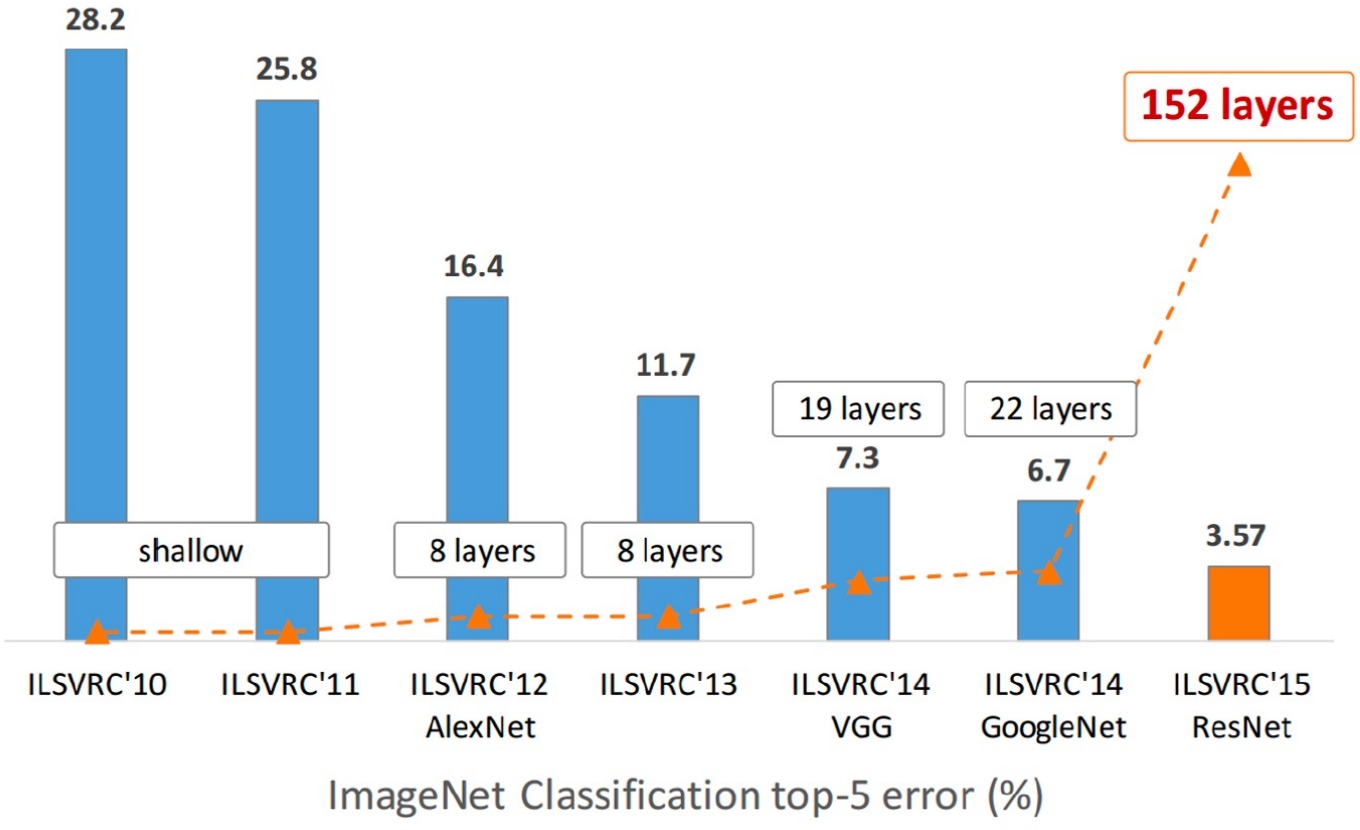


Honglak Lee, Roger Grosse, Rajesh Ranganath, and Andrew Y. Ng.
Unsupervised Learning of Hierarchical Representations with
Convolutional Deep Belief Networks (2011)

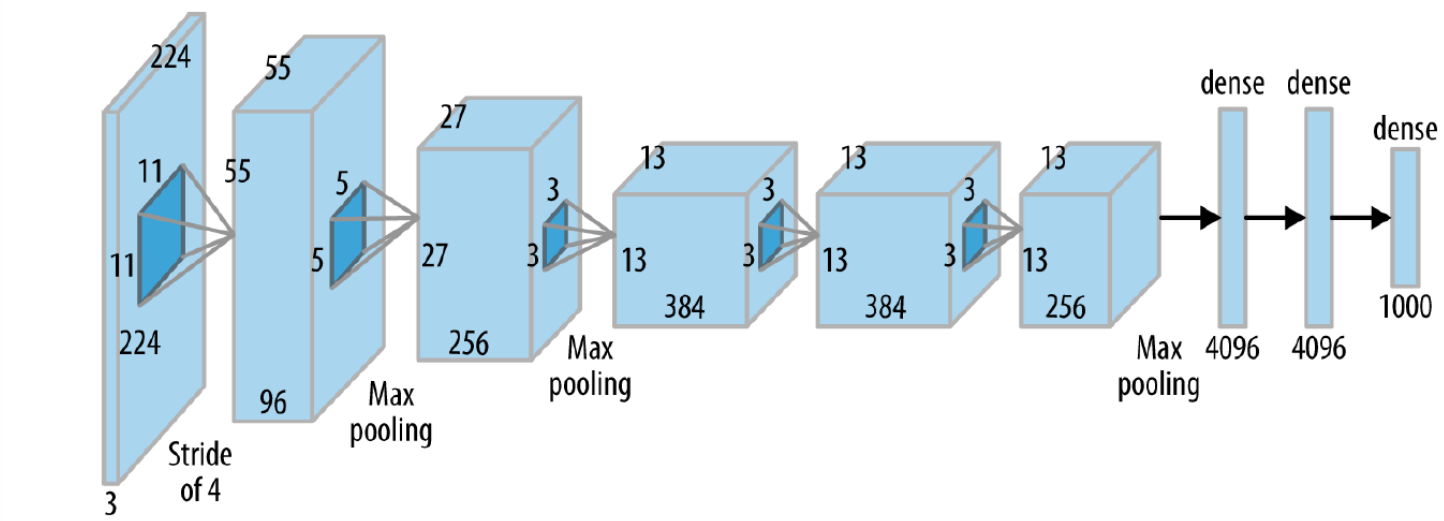
ImageNet – Large Set of Labeled Images

Li Fei-Fei et al. ImageNet: A large-scale hierarchical image database. 2009.
Li Fei-Fei et al. Construction and analysis of a large scale image ontology. 2009.

CNN Evolution



AlexNet: First Breakthrough on ImageNet

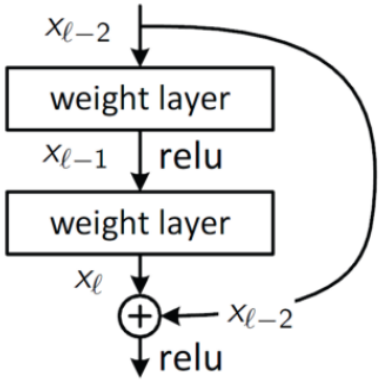


- ReLU + Dropout
- 60 000 000 parameters
- Filter and pooling sizing
- GPU

ResNet: Residual Neural Network

Skip connection of layer l with the previous layer $l - d$

$$x_\ell = \sigma(Wx_{\ell-1}) + x_{\ell-d}$$



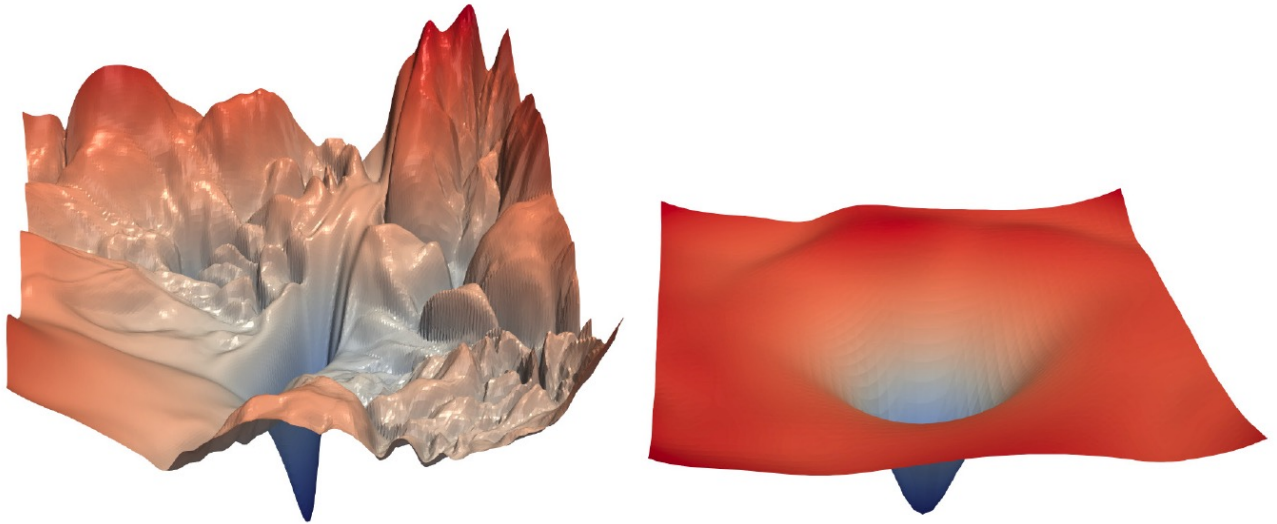
The layer does not learn a new vector representation, but its increment

- Increments are more stable – convergence is improved.
- It is possible to increase the number of layers.
- Generalization – Highway Networks:

$$x_\ell = \sigma(Wx_{\ell-1})\,\underbrace{\tau(W'x_{\ell-1})}_{\text{transform gate}} + x_{\ell-d}\,\underbrace{\big(1-\tau(W'x_{\ell-1})\big)}_{\text{carry gate}}$$

ResNet: Optimal Criterion Visualization

- Skip connections simplify the criterion to be optimized,
- Eliminating local extrema and saddle points.



without skip connections

with skip connections

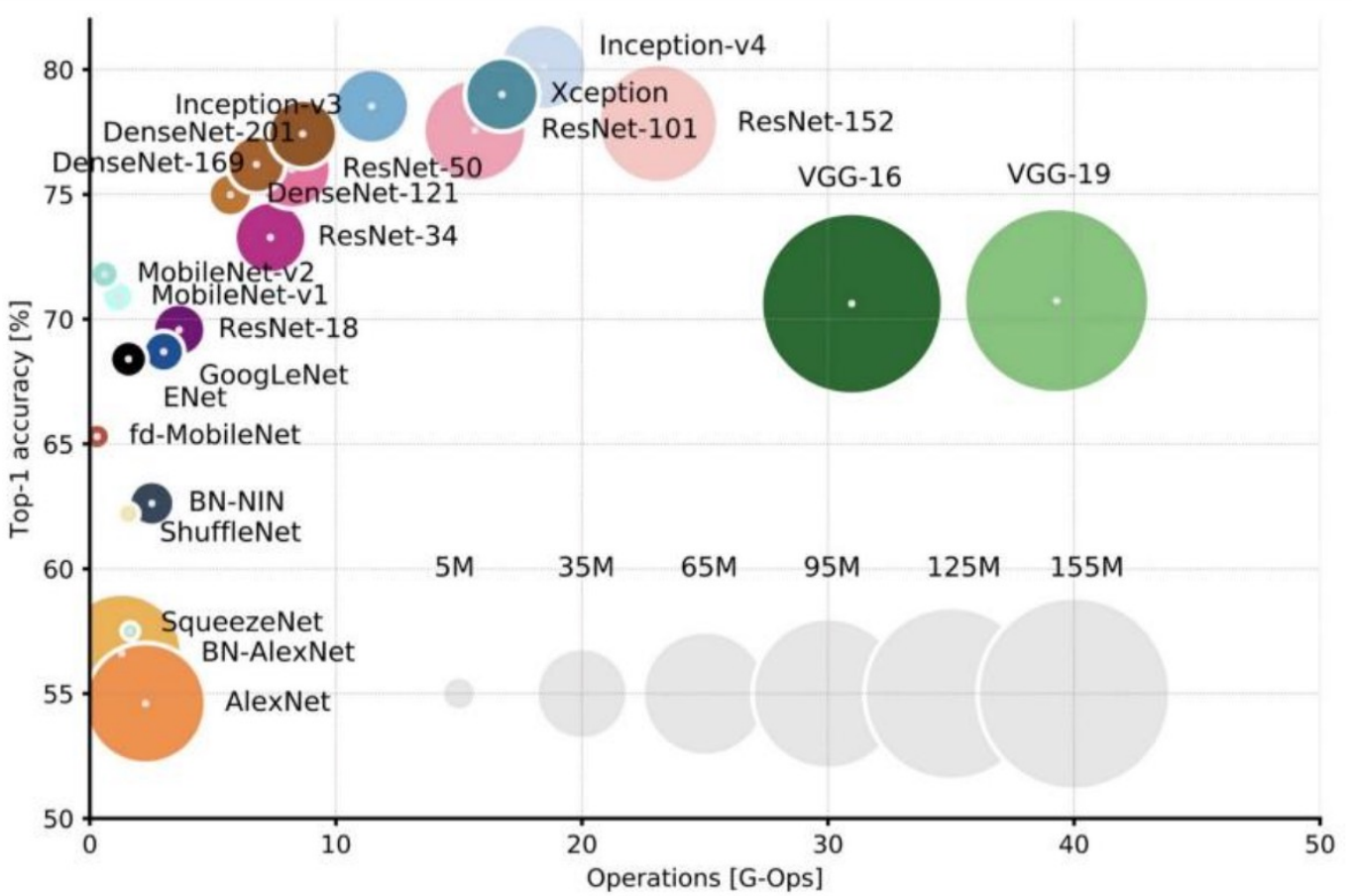
Hao Li et al. Visualizing the Loss Landscape of Neural Nets. 2018

Commonly Used Tricks in CNN

ITMO

- Activation functions without horizontal asymptotes, such as ReLU.
- Adaptive gradient methods.
- Dropout.
- Batch normalization.
- Residual NN.
- Selection of the number of layers and their sizes.
- Dataset augmentation.

CNN: Types of Architectures

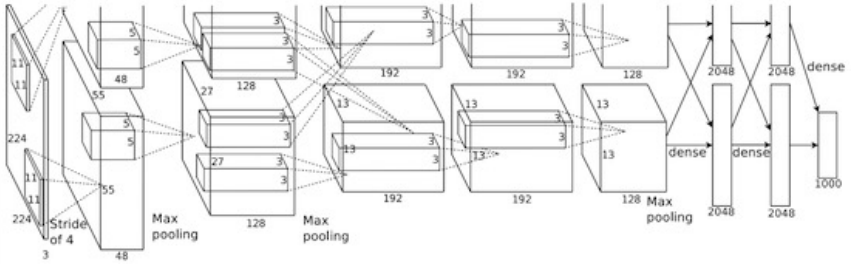


LeNet-5

- A convolutional neural network using a sequence of three layers: convolution layers, pooling layers and non-linearity layers –> since the publication of LeCun's work, this is perhaps one of the main features of deep learning in relation to images.
  - Uses convolution to extract spatial properties.
  - Subsampling using spatial averaging of maps.
  - Nonlinearity, specified as a hyperbolic tangent or sigmoid.
  - The final classifier in the form of a multilayer neural network.
  - The sparse connectivity matrix between the layers makes it possible to reduce the number of calculations.

Dan Ciresan Net, AlexNet

- **Dan Ciresan Net**
  - In 2010, Dan Claudiu Ciresan and Jurgen Schmidhuber published one of the first descriptions of the implementation of GPU neural networks. Their work contained a forward and backward implementation of a 9-layer neural network on the NVIDIA GTX 280.
- **AlexNet**
  - Use of linear rectification units (ReLU) as nonlinearities.
  - Using a discard technique to selectively ignore individual neurons during training, which avoids overfitting the model.
  - Overlapping max pooling to avoid the effects of average pooling.
  - Using NVIDIA GTX 580 to speed up learning.



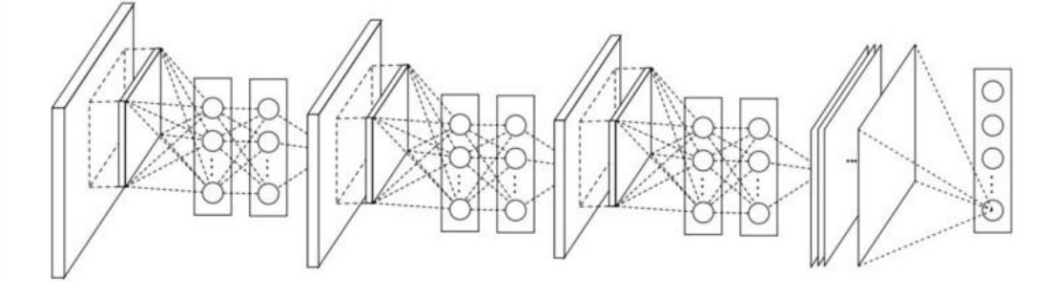
Overfeat, VGG

- **Overfeat**
  - In December 2013, Jan LeCun's NYU lab published a description of Overfeat, a variation of AlexNet. The article also described learnable bounding boxes, and subsequently many other works on this topic came out.
- **VGG**

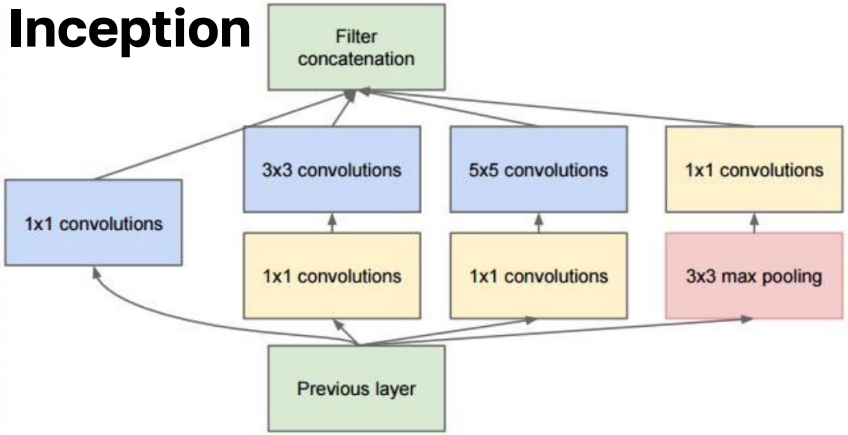
ConvNet Configuration					
A	A-LRN	B	C	D	E
11 weight layers	11 weight layers	13 weight layers	16 weight layers	16 weight layers	19 weight layers
input (224 × 224 RGB image)					
conv3-64	conv3-64 **LRN**	conv3-64 **conv3-64**	conv3-64 conv3-64	conv3-64 conv3-64	conv3-64 conv3-64
maxpool					
conv3-128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

Network-in-Network, GoogLeNet, Inception

- **Network-in-Network**

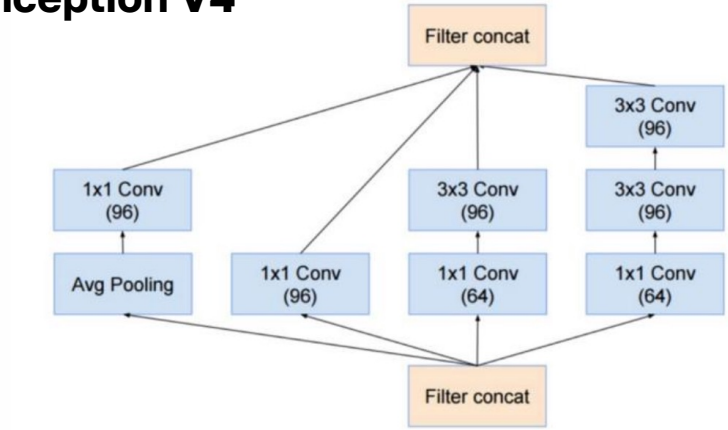


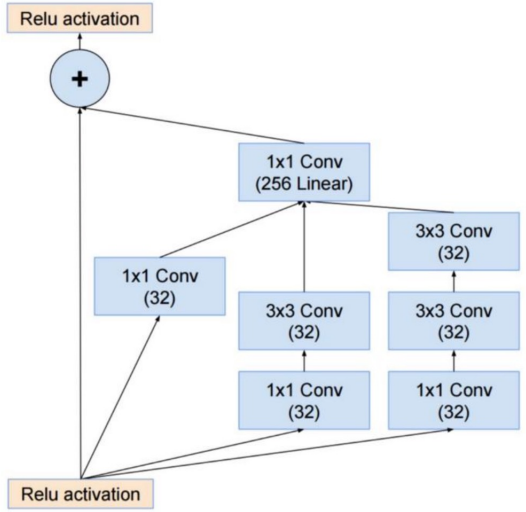
- **GoogLeNet and Inception**



Inception

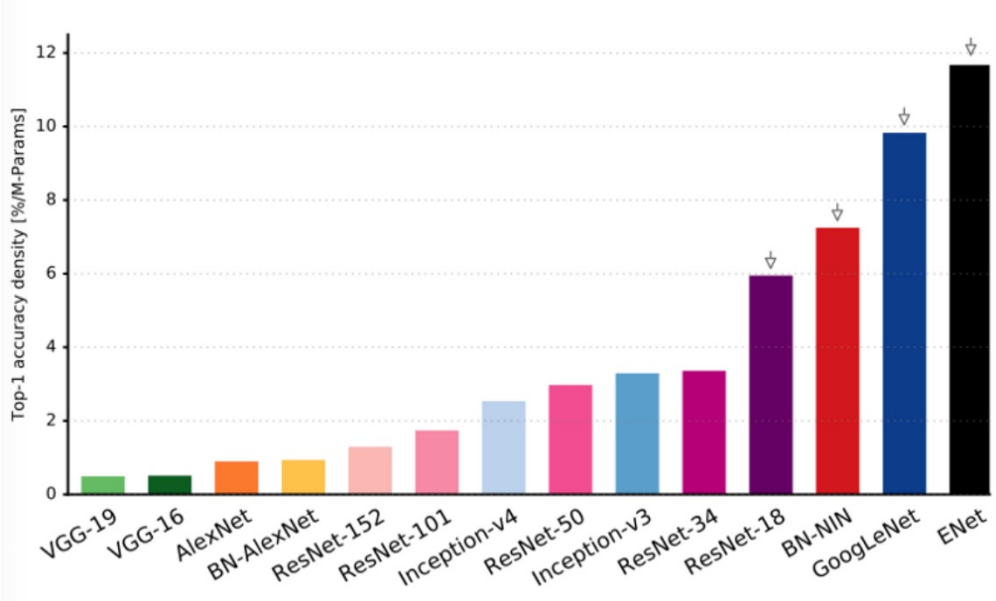
- **Inception V2 (V3)**
  - Maximizing the flow of information in the network by carefully balancing its depth and width. Before each pooling, property maps increase.
  - As the depth increases, the number of properties or the width of the layer also systematically increases.
  - The width of each layer increases to increase the combination of properties before the next layer.
  - Whenever possible, only 3x3 convolutions are used. Given that 5x5 and 7x7 filters can be decomposed with multiple 3x3.
- **Inception V4**





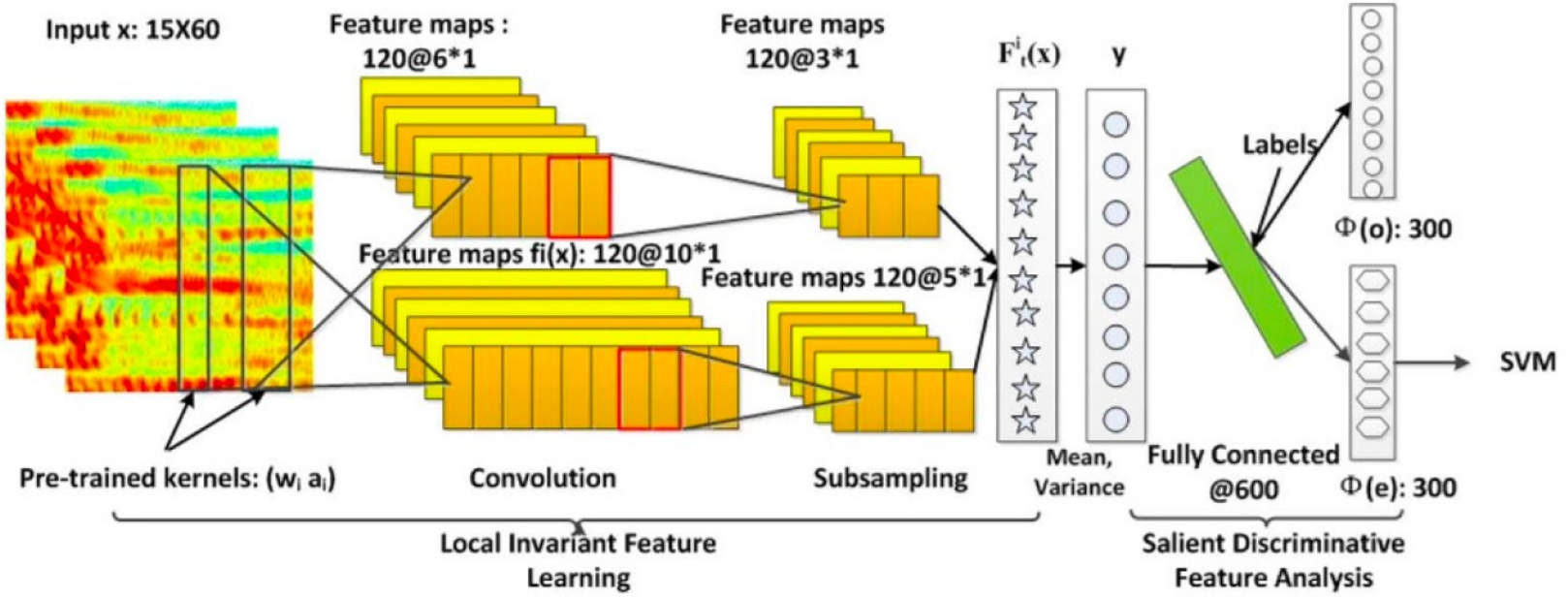
SqueezeNet

- Alteration in a new way of many concepts from ResNet and Inception. The authors have demonstrated that improving the architecture can reduce the size of networks and the number of parameters without complex compression algorithms.



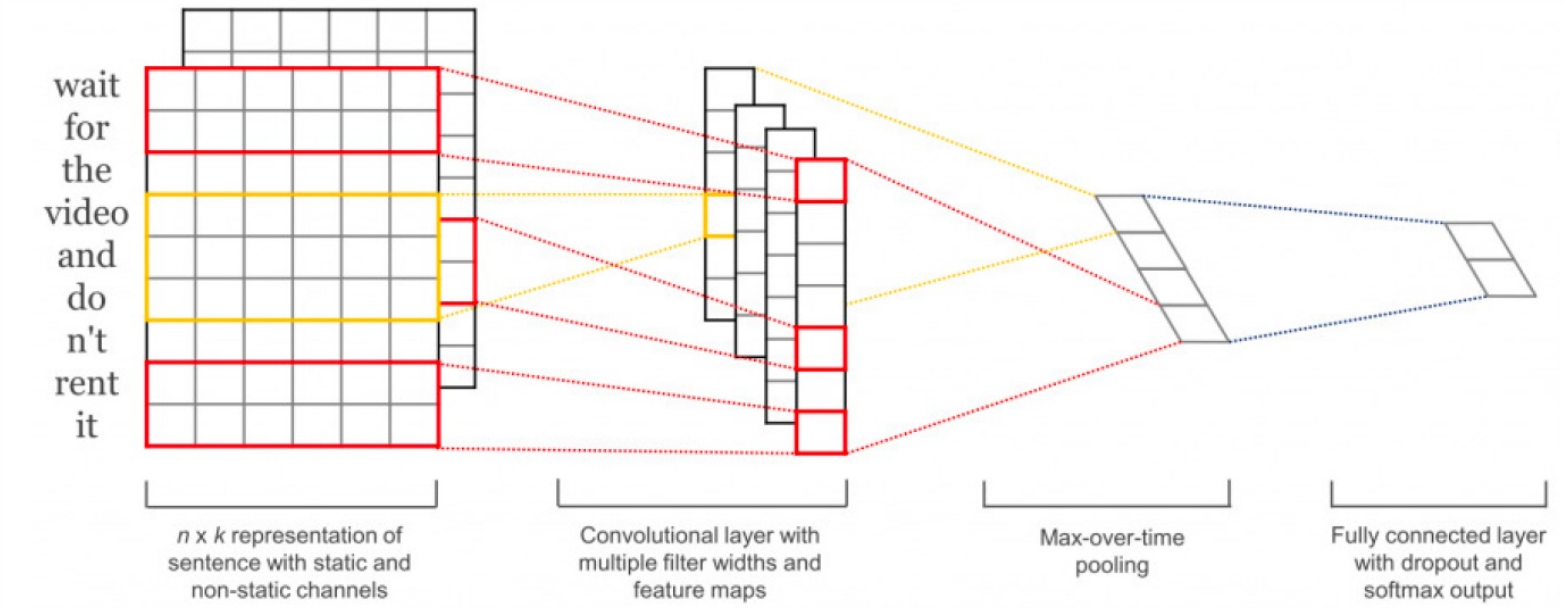
CNN: Speech Recognition

- Successive signal fragments are represented by spectral decomposition vectors.

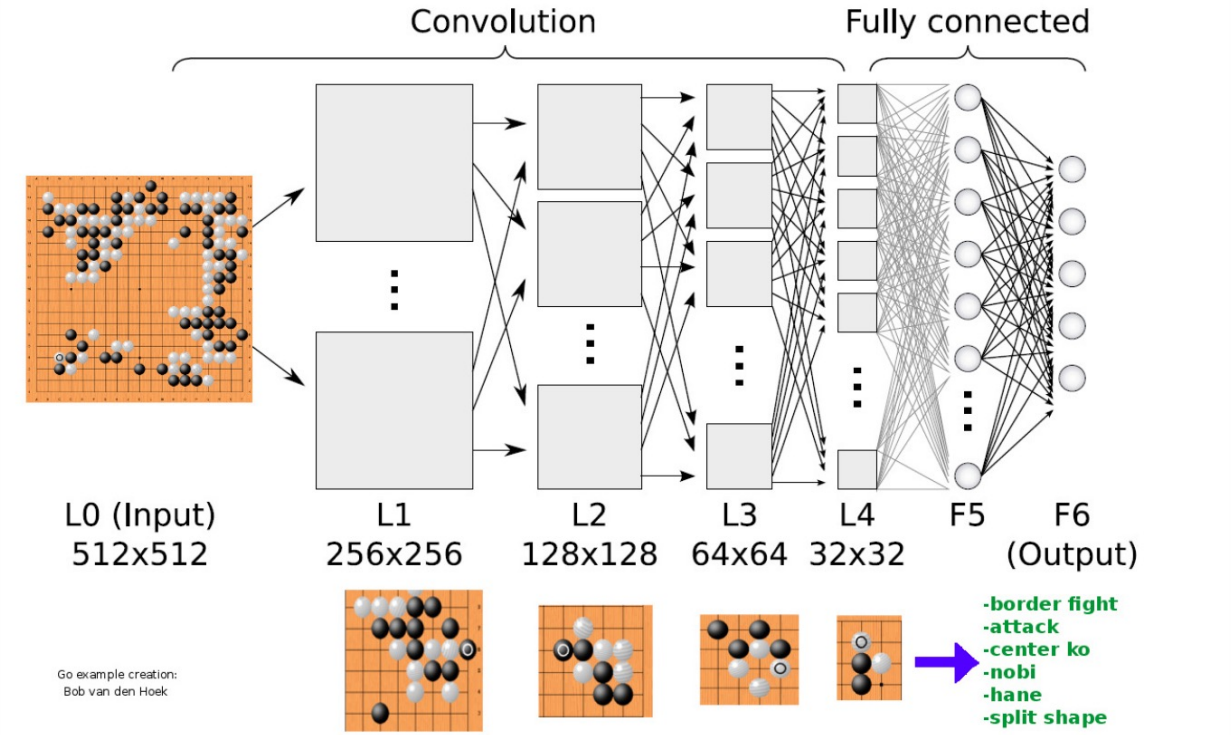


CNN: Classification of Sentences in Text

- Sequential words in the text are represented by frames using vector representations (word2vec, etc.)

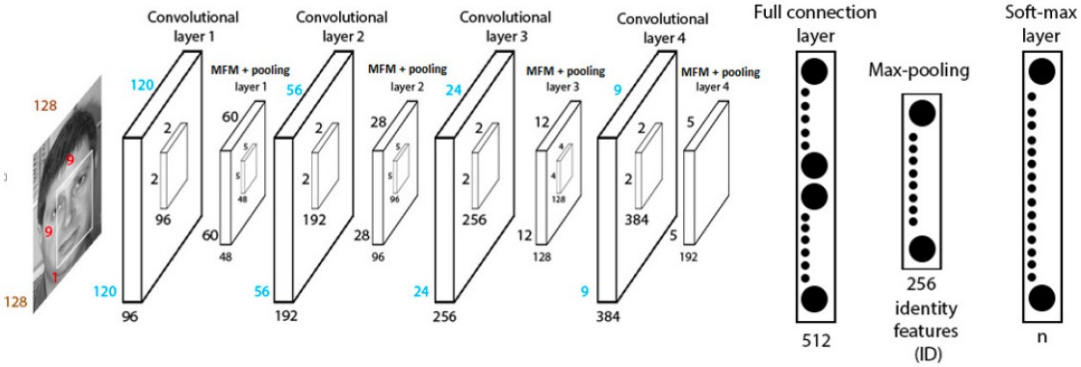


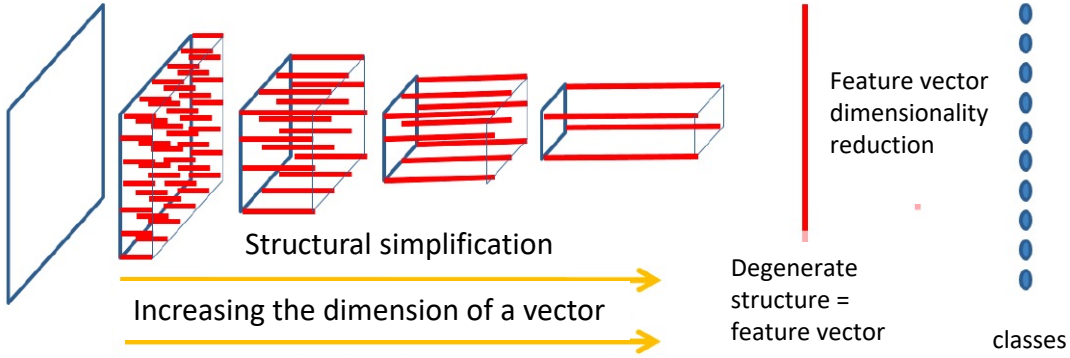
CNN: Decision Making in Logic Games



David Silver et al. (DeepMind) Mastering the game of Go without human knowledge. 2017.

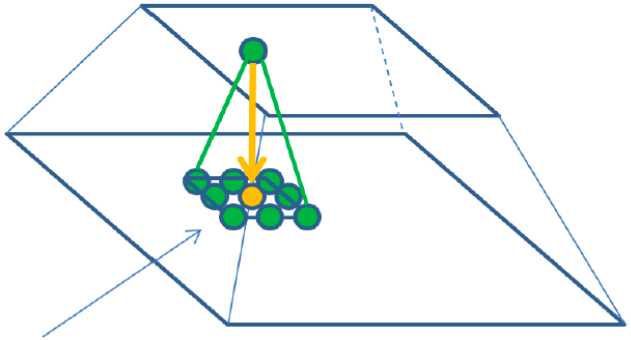
CNN: Images Vectorization



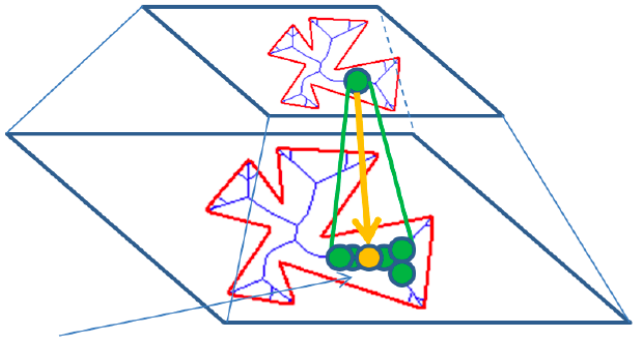


CNN: Idea of Generalizing for Any Structured Data

- Let's assume that each object has a structure given by a graph.
- The convolution is determined by the local neighborhood of the vertexing aggregates the vectors of the vertices of the local neighborhood.
- CNN is trained to find and classify subgraphs.



Rectangular window of given size centered on given point + window convolution operation



Local neighborhood defined for any graph vertex + neighborhood convolution operation

Test on Lectures 15-16

CS3 Test

AT3 Test

THANK YOU FOR YOUR TIME!

ITMO more than a UNIVERSITY

s.shavetov@itmo.ru