Deep Learning

7. Convolutional neural networks for classification.

Viacheslav Dudar

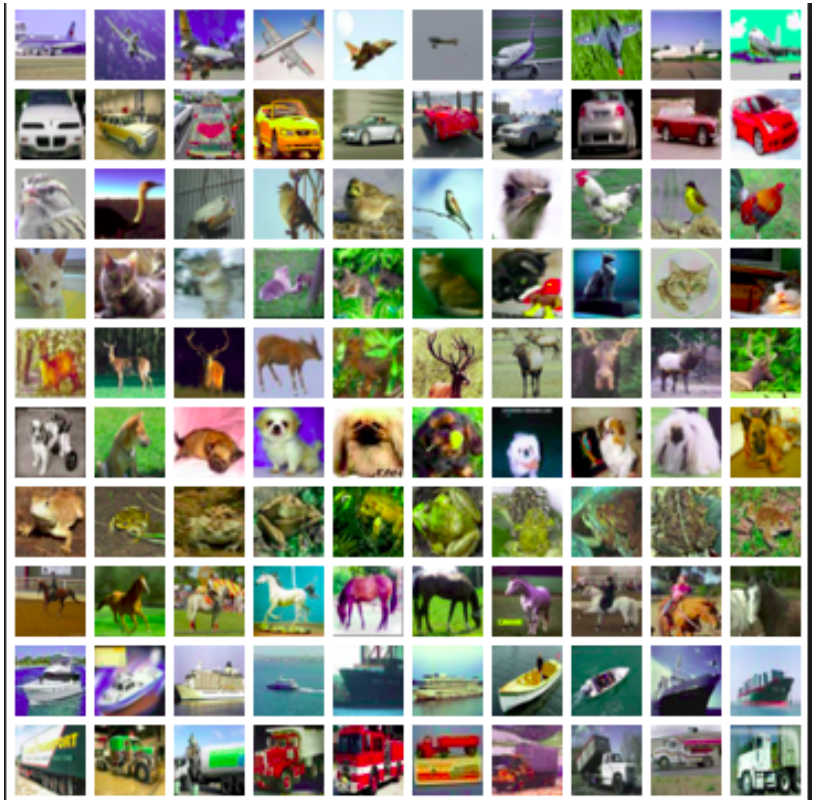
Taras Shevchenko National University of Kyiv

2018

Why fully connected networks are not enough

- We do not account for spatial shape of the image
- Number of coefficients even for the simple model could be giant
- We want to gain translation invariance
- Fully connected networks do not generalize well to the new data

Example: CIFAR-10



50000 training images, 10000 test images

10 classes

3*32*32 color images

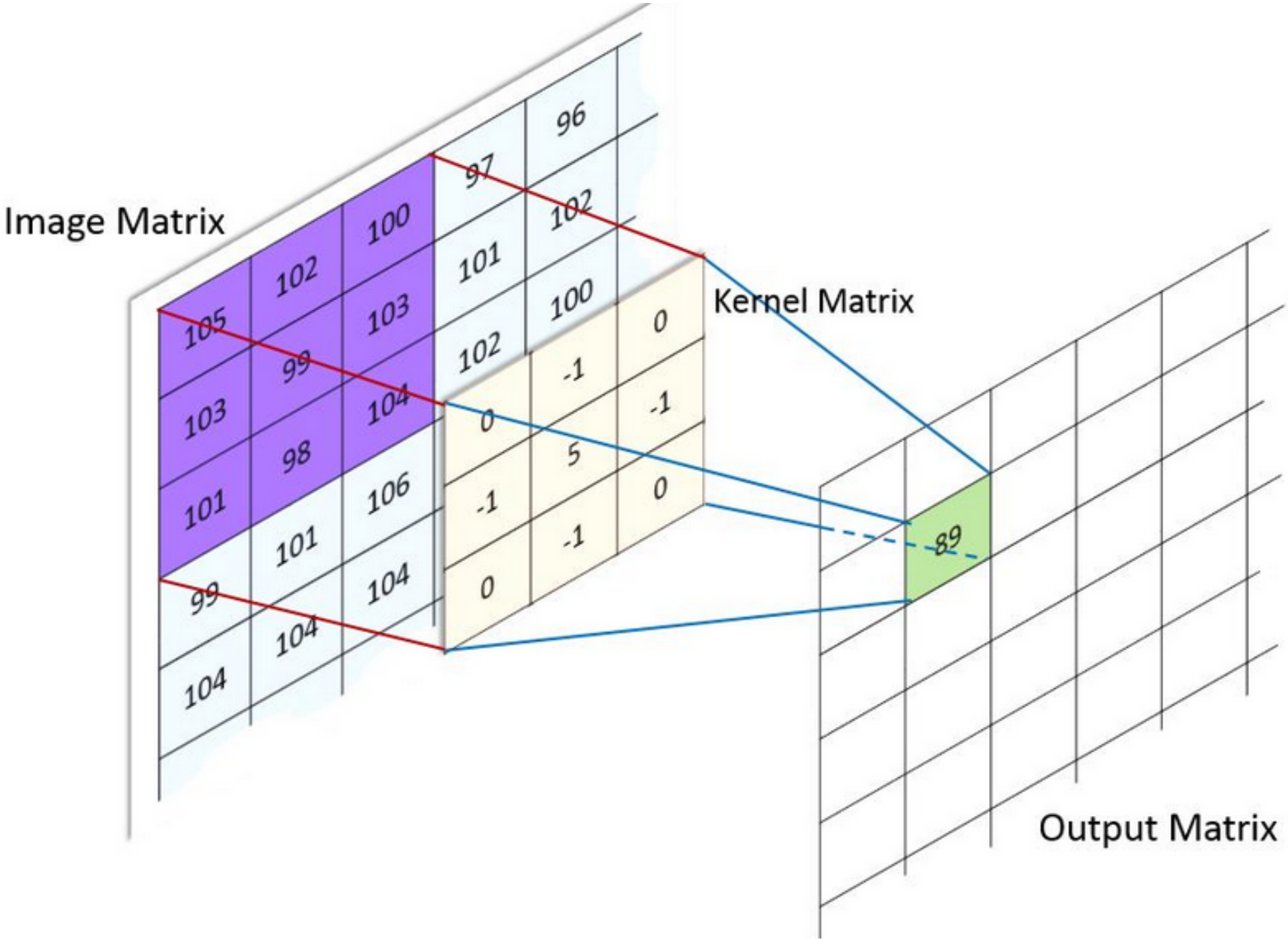
Fully connected nets: 100% train accuracy Only about 55% test accuracy

Convolutional neural net (CNN) components

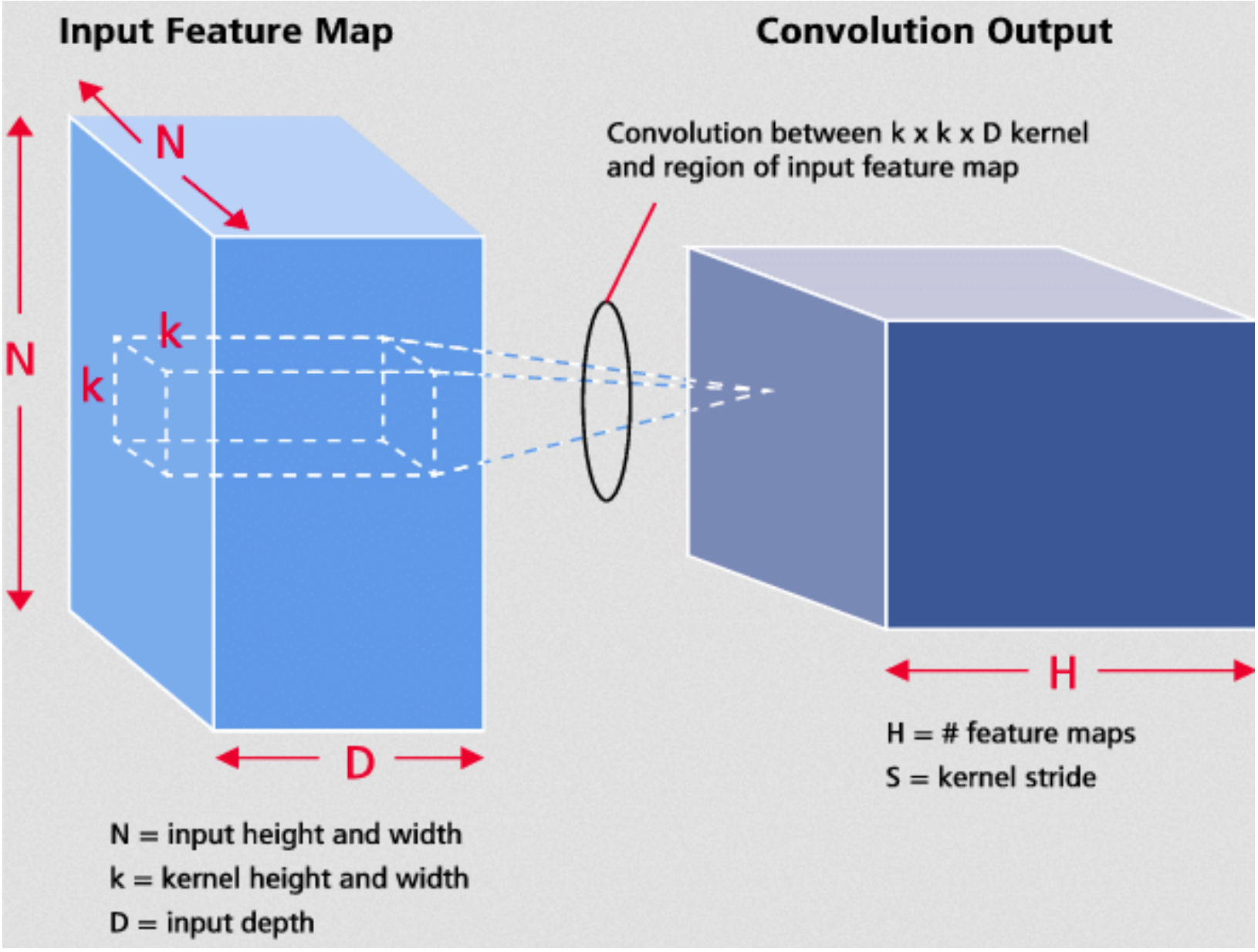
The simplest convolutional net consists of such blocks:

- Convolutional Layers
- Nonlinear function application
- Pooling layers
- Fully connected layers
- Softmax

Recall: convolution

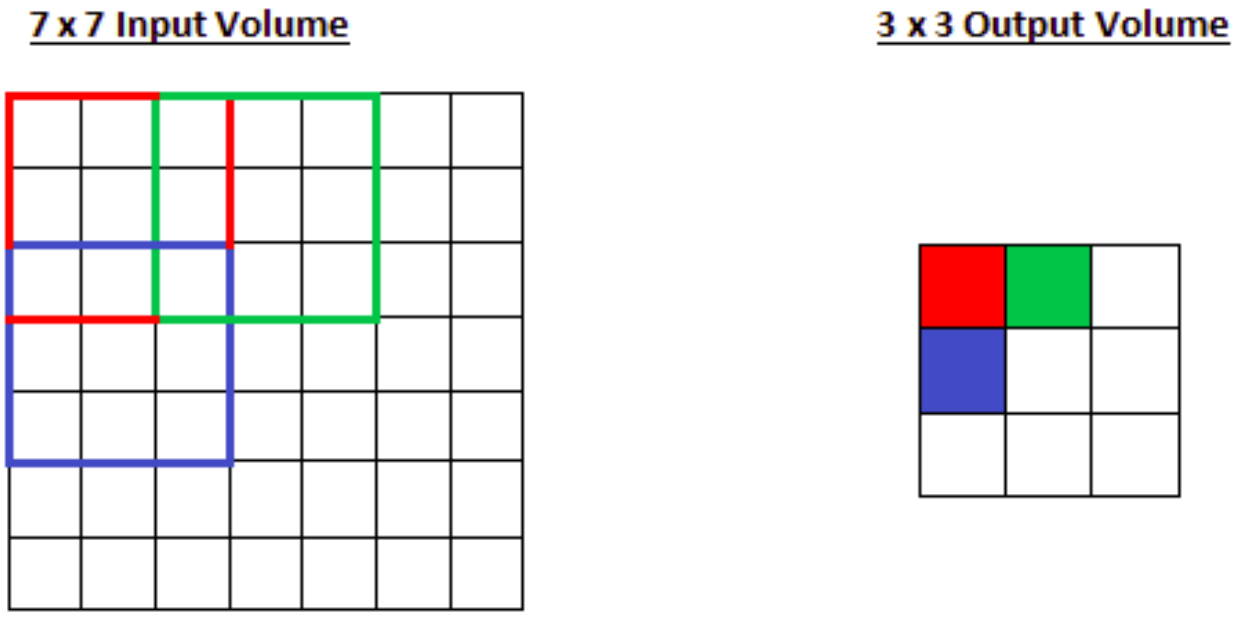


Convolution for tensors



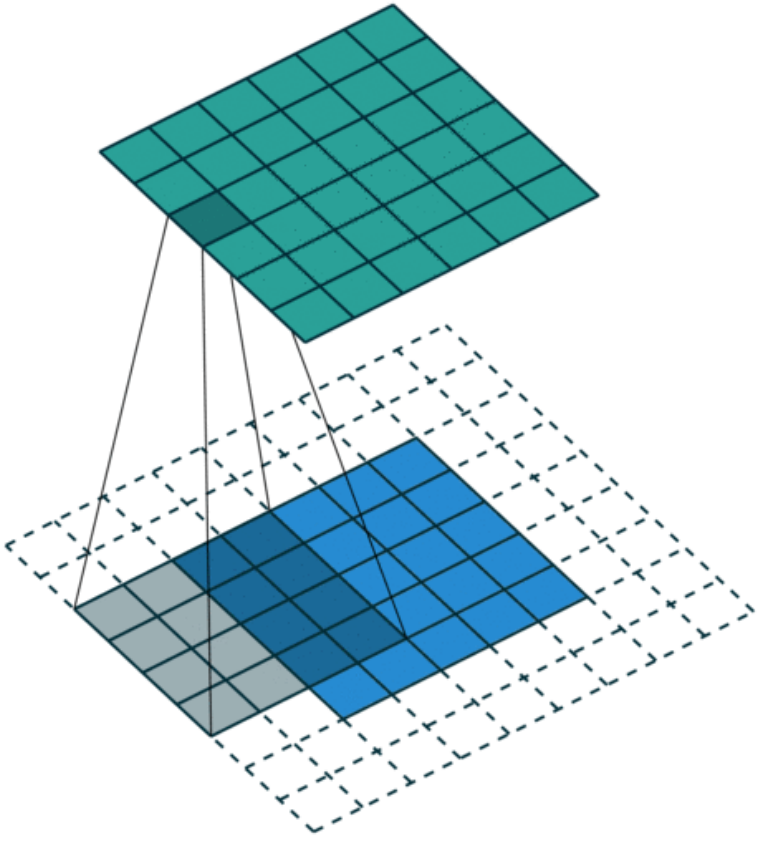
Stride

Stride: length of step of convolution
Example: stride = 2 (vertical and horizontal)



Padding

Example padding $= 2$ (horizontal and vertical)

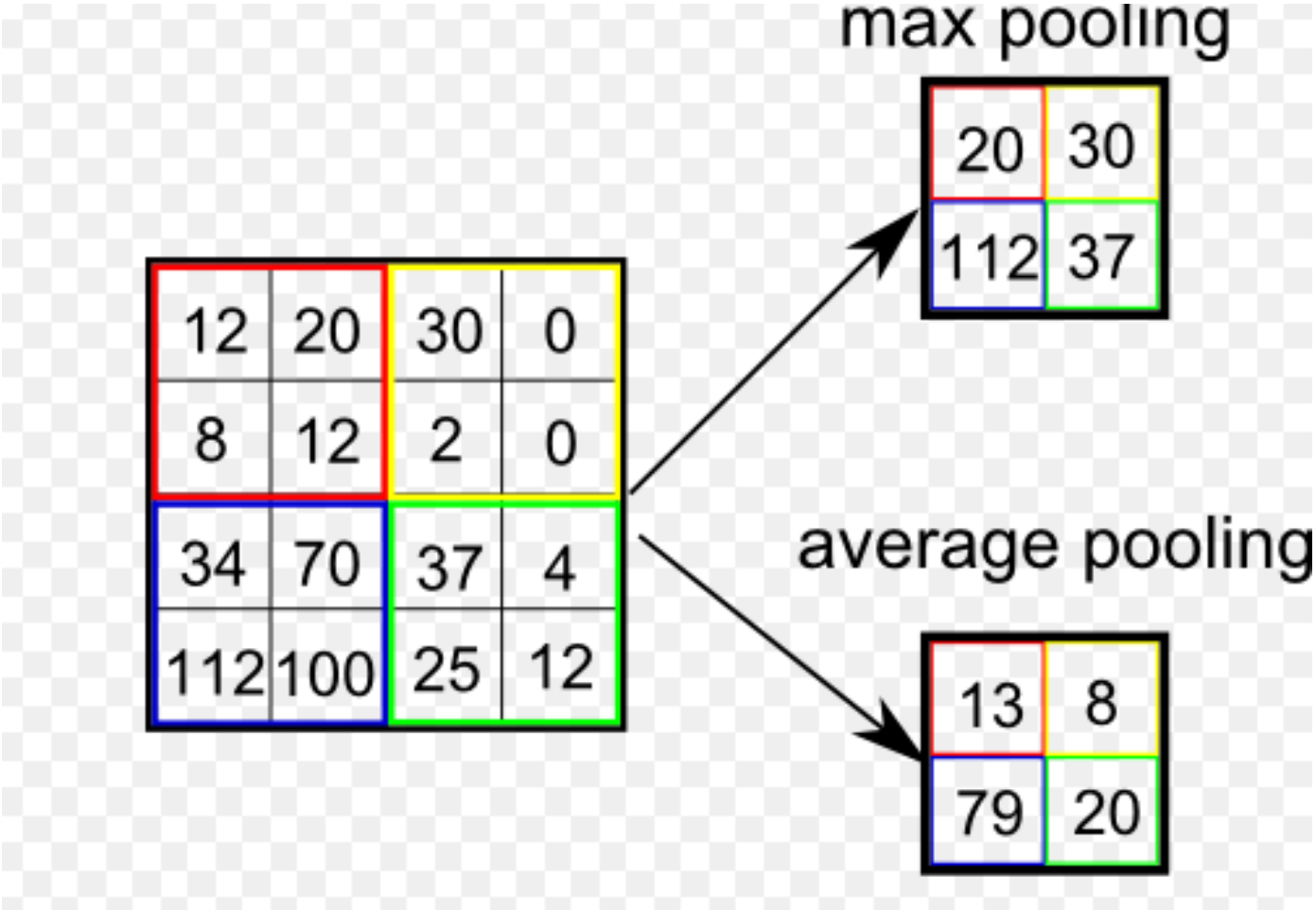


Summary for convolutional layer

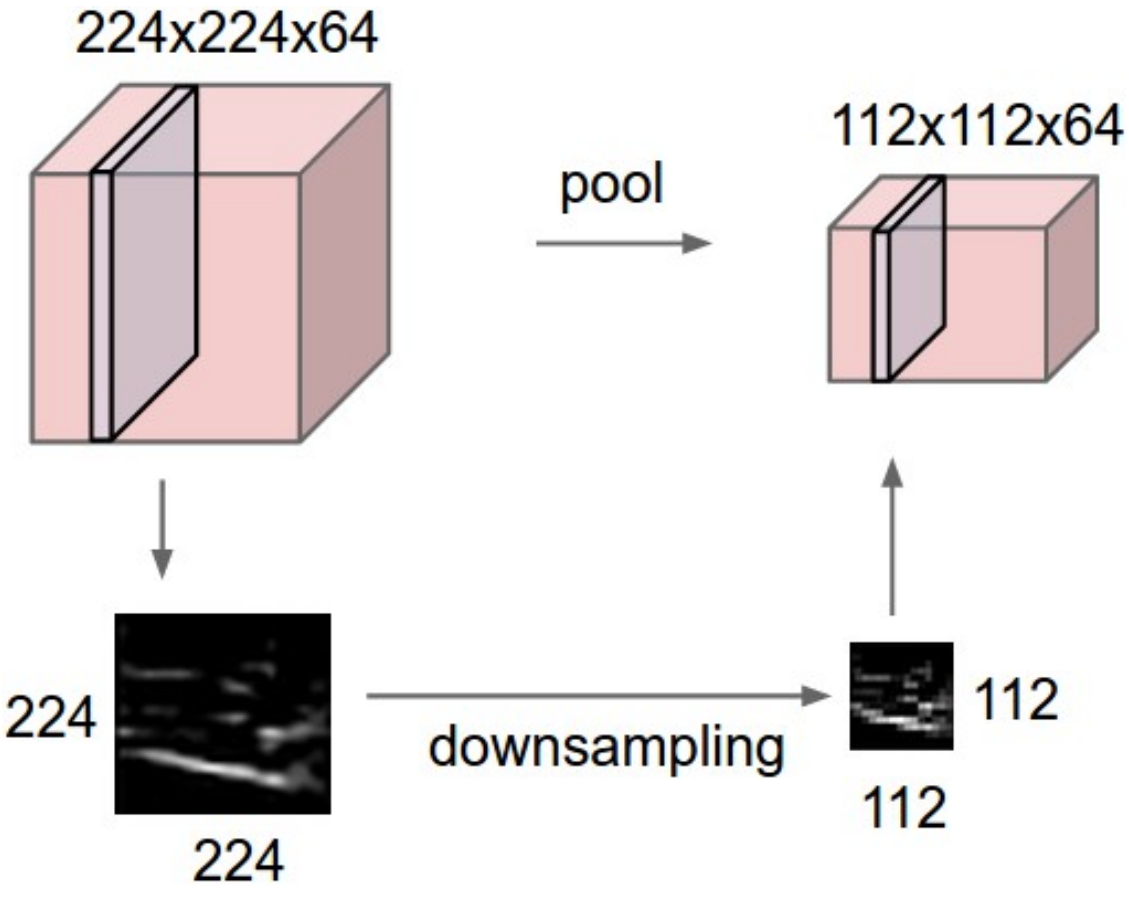
- Takes tensor of depth D as input
- Uses H convolutional kernels of size $k \times k \times D$
- Performs convolution with stride S and padding P
- Result is tensor of depth H
- Most popular setting is 3*3 convolution with stride 1 and padding 1

Pooling layer

Pooling layer types: max pooling and average pooling



Pooling layer

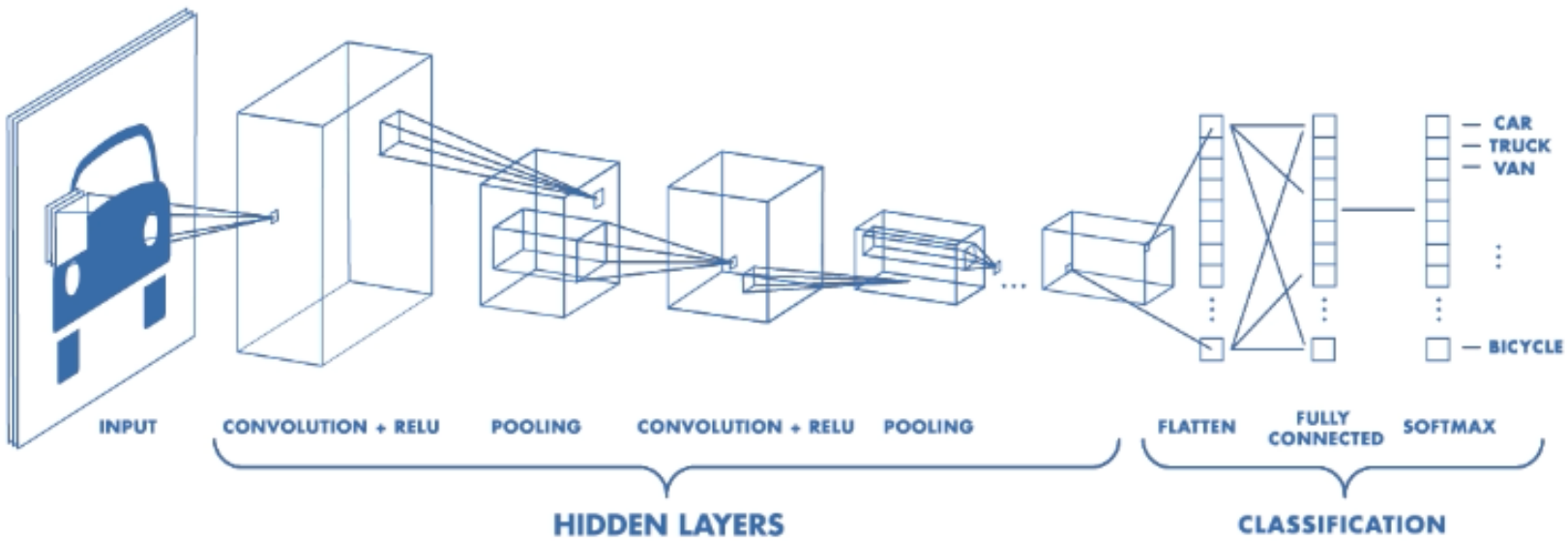


Pooling layers

Properties of pooling layers:

- Perform separatly for each tensor layer
- Typical window size is 2*2 (with stride 2)
- Depth of the tensor remains the same
- Descrease spatial dimension for simple further processing
- Makes representation invariant instead of equivariant

Typical CNN

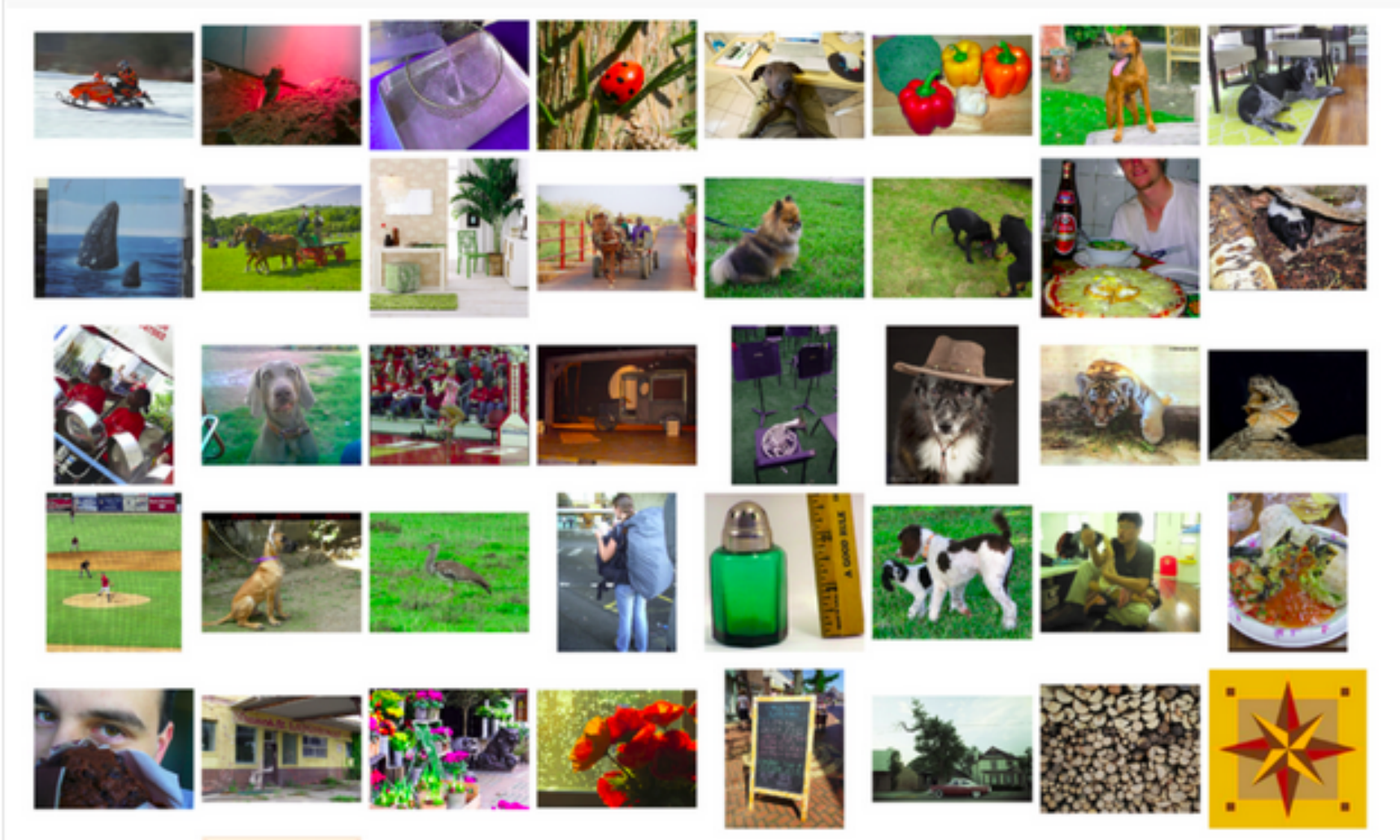

CAR
TRUCK
VAN
BICYCLE
INPUT
CONVOLUTION + RELU
POOLING
CONVOLUTION + RELU
POOLING
FLATTEN
FULLY CONNECTED
SOFTMAX
HIDDEN LAYERS
CLASSIFICATION

CNN properties

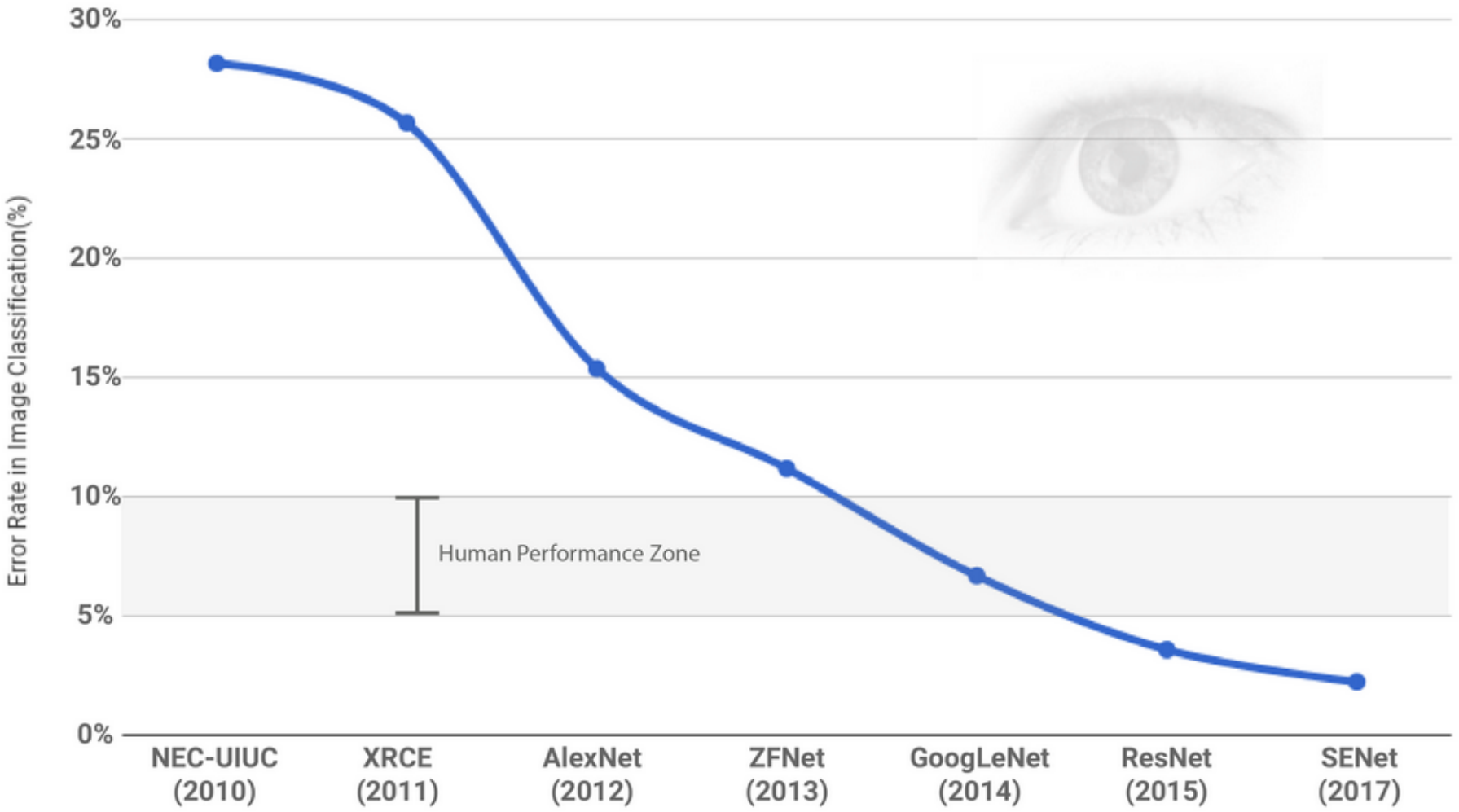
- Approximate translation invariance
- Total number of parameters is smaller because of weight sharing

ImageNet dataset

- $>$ 14 m images
- $>$ 20 k categories



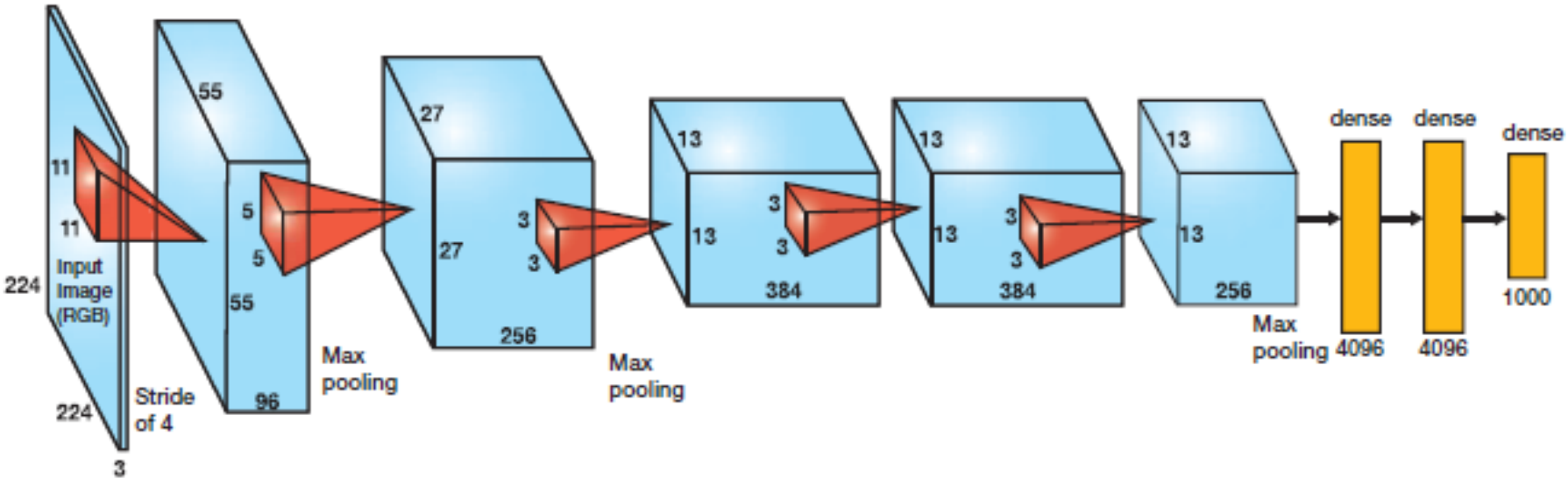
ImageNet Competition



AlexNet

[227x227x3] INPUT
[55x55x96] CONV1: 96 11x11 filters at stride 4, pad 0
[27x27x96] MAX POOL1: 3x3 filters at stride 2
[27x27x96] NORM1: Normalization layer
[27x27x256] CONV2: 256 5x5 filters at stride 1, pad 2
[13x13x256] MAX POOL2: 3x3 filters at stride 2
[13x13x256] NORM2: Normalization layer
[13x13x384] CONV3: 384 3x3 filters at stride 1, pad 1
[13x13x384] CONV4: 384 3x3 filters at stride 1, pad 1
[13x13x256] CONV5: 256 3x3 filters at stride 1, pad 1
[6x6x256] MAX POOL3: 3x3 filters at stride 2
[4096] FC6: 4096 neurons
[4096] FC7: 4096 neurons
[1000] FC8: 1000 neurons (class scores)

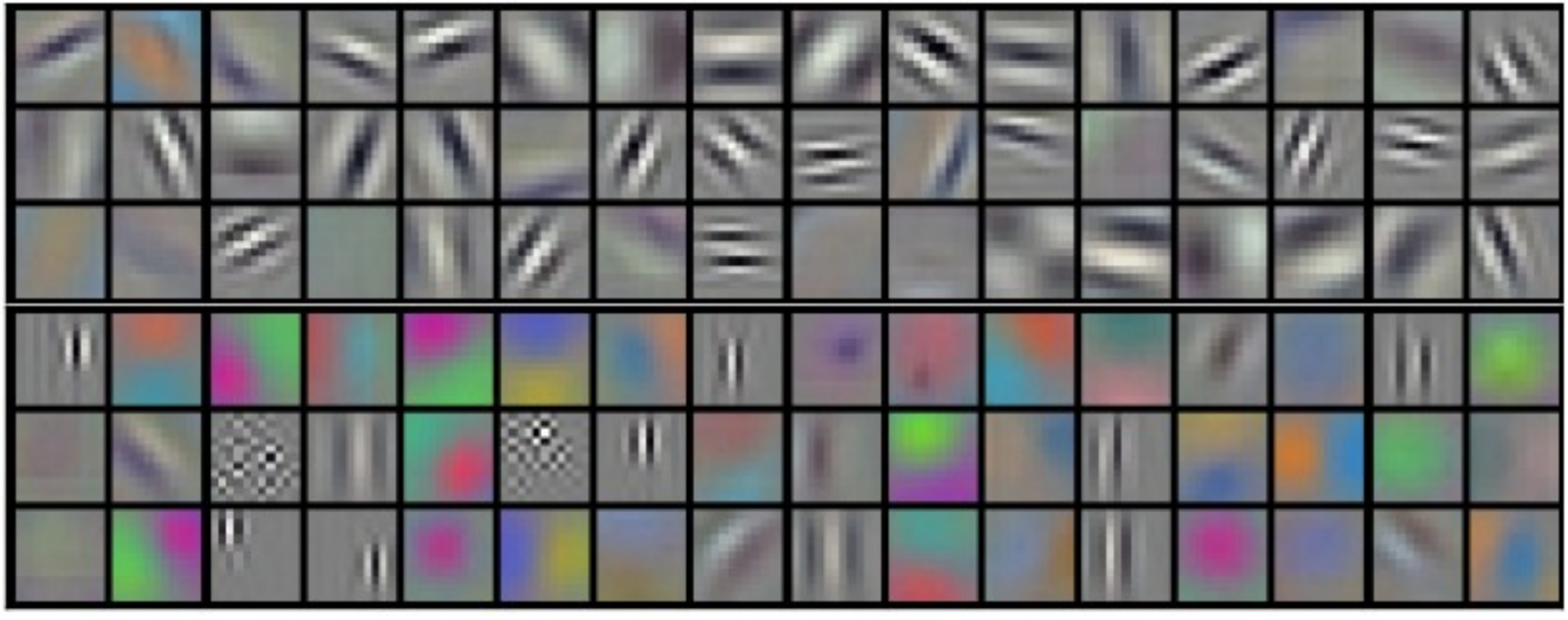
AlexNet



- Dropout 0.5
- Heavy data augmentation
- Random cropping to 224*224 for training and averaging 5 nets as test time
- Trained with momentum method on 2 GPUs for a week
- 60 m parameters

Learned kernels

First convolutional layer kernels of Alex-Net:



General properties of CNNs

- Need a lot of data for training
- Need a lot of computational power (GPUs)
- Still suffers from overfitting
- Needs regularization and data augmentation