

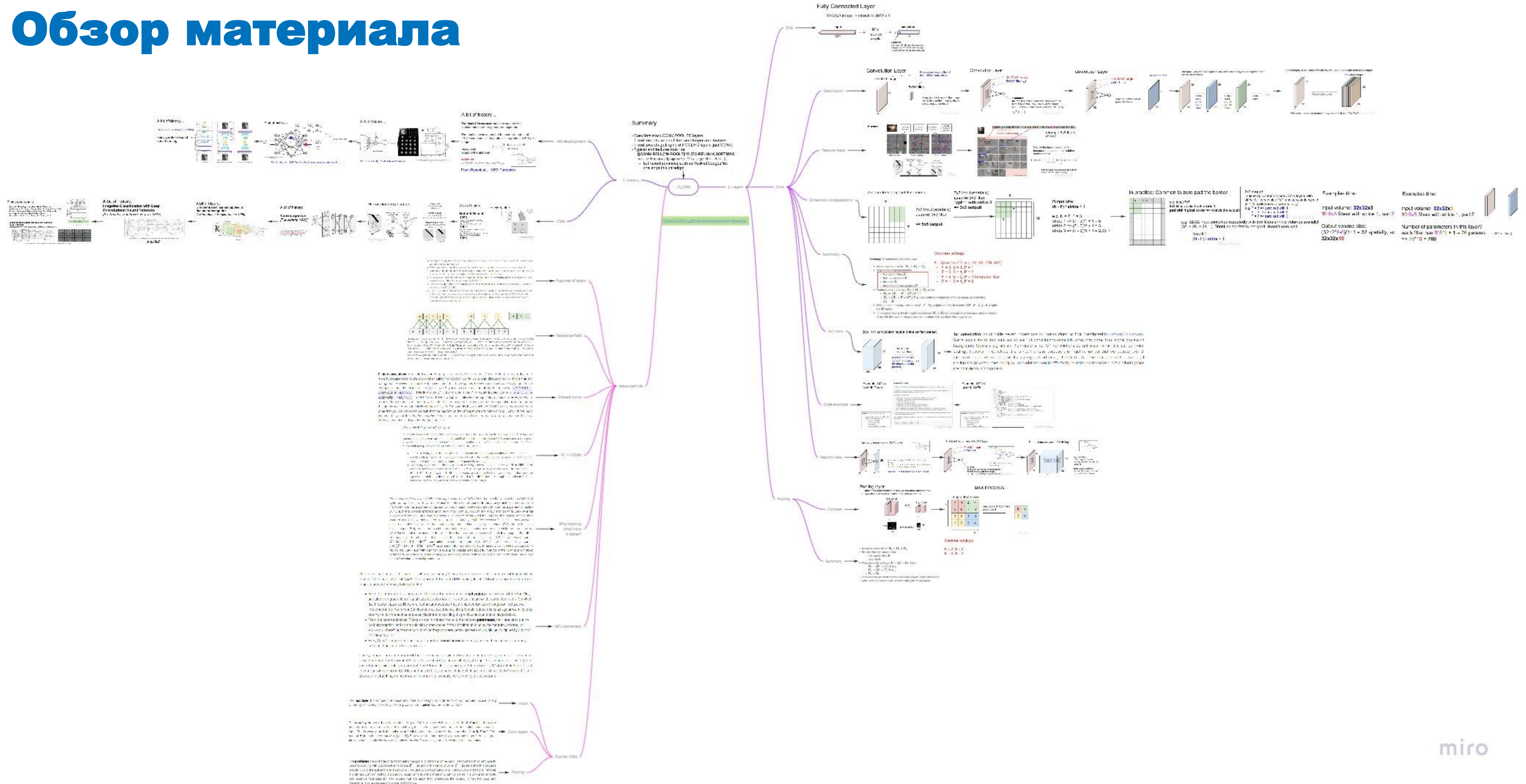
Лекция 9

Введение в глубокое обучение для компьютерного зрения

Курс «Компьютерное зрение»

Глубокое обучение в CV

Обзор материала



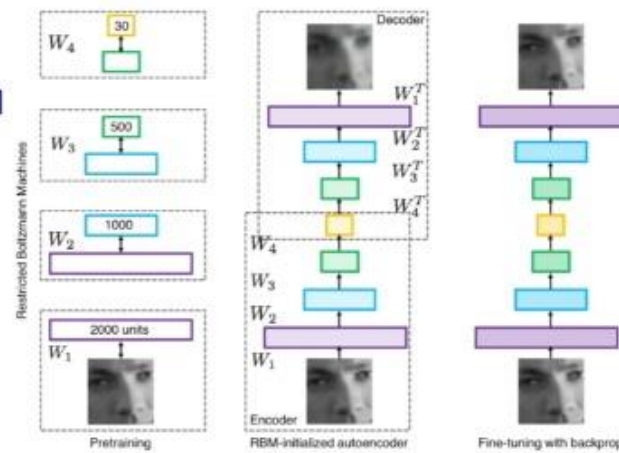
Исторические предпосылки

Развитие нейросетевых методов и сверточных сетей

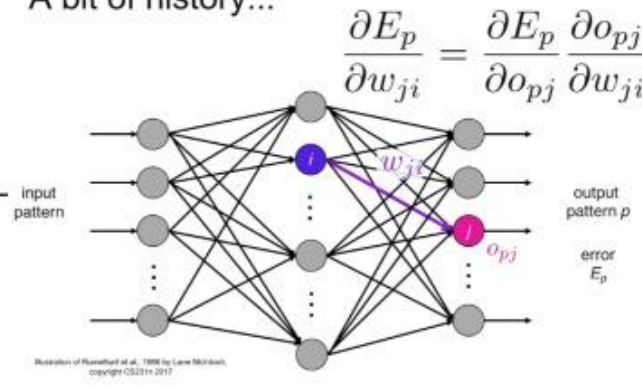
A bit of history...

[Hinton and Salakhutdinov 2006]

Reinvigorated research in Deep Learning

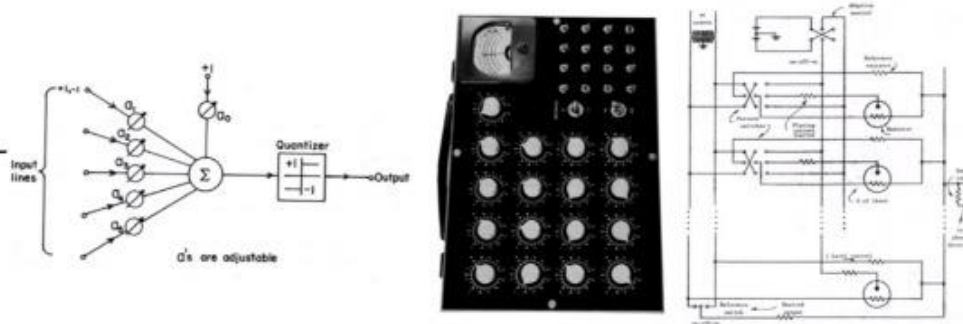


A bit of history...



Rumelhart et al., 1986: First time back-propagation became popular

A bit of history...



Widrow and Hoff, ~1960: Adaline/Madaline

A bit of history...

The **Mark I Perceptron** machine was the first implementation of the perceptron algorithm.

The machine was connected to a camera that used 20×20 cadmium sulfide photocells to produce a 400-pixel image.

recognized letters of the alphabet

update rule:
 $w_i(t+1) = w_i(t) + \alpha(d_j - y_j(t))x_{ji}$

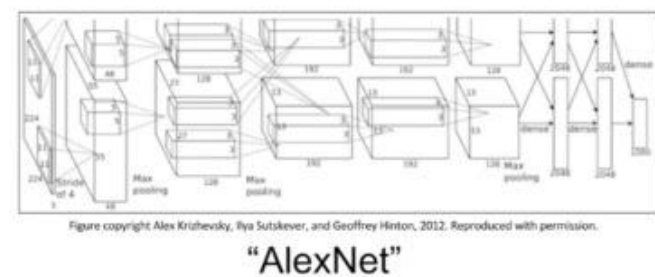
Frank Rosenblatt, ~1957: Perceptron

$$f(x) = \begin{cases} 1 & \text{if } w \cdot x + b > 0 \\ 0 & \text{otherwise} \end{cases}$$

NN development

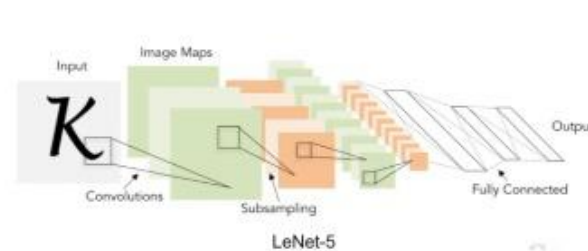
A bit of history:

ImageNet Classification with Deep Convolutional Neural Networks
[Krizhevsky, Sutskever, Hinton, 2012]



A bit of history:

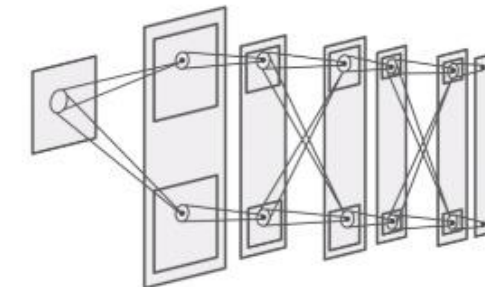
Gradient-based learning applied to document recognition
[LeCun, Bottou, Bengio, Haffner 1998]



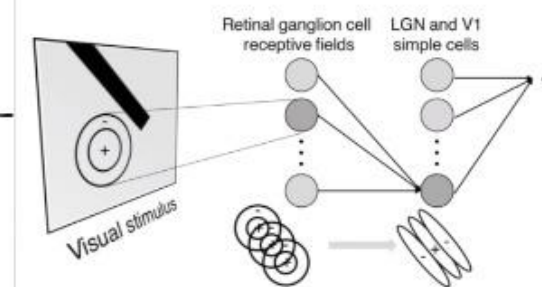
A bit of history:

Neurocognitron
[Fukushima 1980]

"sandwich" architecture (SCSCSC...)
simple cells: modifiable parameters
complex cells: perform pooling



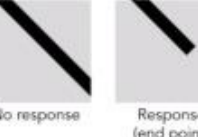
Hierarchical organization



Simple cells:
Response to light orientation

Complex cells:
Response to light orientation and movement

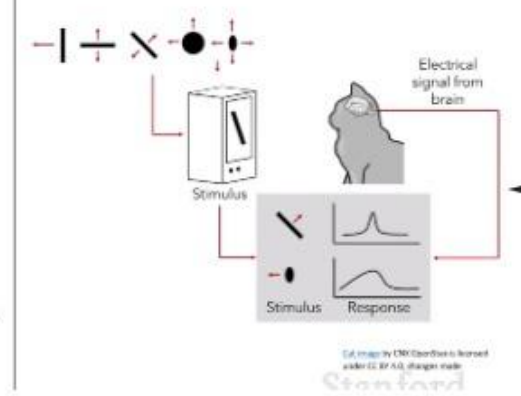
Hypercomplex cells:
response to movement with an end point



A bit of history:

Hubel & Wiesel, 1959
RECEPTIVE FIELDS OF SINGLE NEURONES IN THE CAT'S STRIATE CORTEX

1962
RECEPTIVE FIELDS, BINOCULAR INTERACTION AND FUNCTIONAL ARCHITECTURE IN THE CAT'S VISUAL CORTEX
1968...



CNN

miro

Сверточные нейронные сети

Сверточные слои

Description

Convolution Layer

32x32x3 image

5x5x3 filter

Filters always extend the full depth of the input volume

Convolve the filter with the image i.e. "slide over the image spatially, computing dot products"

1 number: the result of taking a dot product between the filter and a small 5x5x3 chunk of the image (i.e. 5*5*3 = 75-dimensional dot product + bias)

$w^T x + b$

convolve (slide) over all spatial locations

activation map

Preview: ConvNet is a sequence of Convolutional Layers, interspersed with activation functions

For example, if we had 6 5x5 filters, we'll get 6 separate activation maps:

We stack these up to get a "new image" of size 28x28x6!

Feature maps

Preview

Low-level features

Mid-level features

High-level features

Linearly separable classifier

example 5x5 filters (32 total)

We call the layer convolutional because it is related to convolution of two signals:

$$f(x,y) \otimes g(x,y) = \sum_{x_1} \sum_{y_1} f(x_1, y_1) g(x-x_1, y-y_1)$$

elementwise multiplication and sum of a filter and the signal (image)

Dimension computations

A closer look at spatial dimensions:

7x7 input (spatially) assume 3x3 filter

=> 5x5 output

7x7 input (spatially) assume 3x3 filter applied with stride 2 => 3x3 output!

Output size: $(N - F) / \text{stride} + 1$

e.g. N = 7, F = 3:

stride 1 => $(7 - 3) / 1 + 1 = 5$

stride 2 => $(7 - 3) / 2 + 1 = 3$

stride 3 => $(7 - 3) / 3 + 1 = 2.33 \dots$

In practice: Common to zero pad the border

e.g. input 7x7 3x3 filter, applied with stride 1 pad with 1 pixel border => what is the output?

7x7 output! in general, common to see CONV layers with stride 1, filters of size FxF, and zero-padding with (F-1)/2. (will preserve size spatially)

e.g. F = 3 => zero pad with 1

F = 5 => zero pad with 2

F = 7 => zero pad with 3

E.g. 32x32 input convolved repeatedly with 5x5 filters shrinks volumes spatially! (32 -> 28 -> 24 ...). Shrinking too fast is not good, doesn't work well.

(recall:)

$(N - F) / \text{stride} + 1$

Examples time:

Input volume: 32x32x3

10 5x5 filters with stride 1, pad 2

Output volume size:

$(32 + 2 \cdot 2 - 5) / 1 + 1 = 32$ spatially, so 32x32x10

Examples time:

Input volume: 32x32x3

10 5x5 filters with stride 1, pad 2

Number of parameters in this layer?

each filter has $5 \cdot 5 \cdot 3 + 1 = 76$ params (+1 for bias)

=> $76 \cdot 10 = 760$

Summary

Summary To summarize, the Conv Layer:

- Accepts a volume of size $W_1 \times H_1 \times D_1$
- Requires four hyperparameters:
 - Number of filters K ,
 - their spatial extent F ,
 - the stride S ,
 - the amount of zero padding P
- Produces a volume of size $W_2 \times H_2 \times D_2$ where:
 - $W_2 = (W_1 - F + 2P) / S + 1$
 - $H_2 = (H_1 - F + 2P) / S + 1$ (i.e. width and height are computed equally by symmetry)
 - $D_2 = K$
- With parameter sharing, it introduces $F \cdot F \cdot D_1$ weights per filter, for a total of $(F \cdot F \cdot D_1) \cdot K$ weights and K biases.
- In the output volume, the d -th depth slice (of size $W_2 \times H_2$) is the result of performing a valid convolution of the d -th filter over the input volume with a stride of S , and then offset by d -th bias.

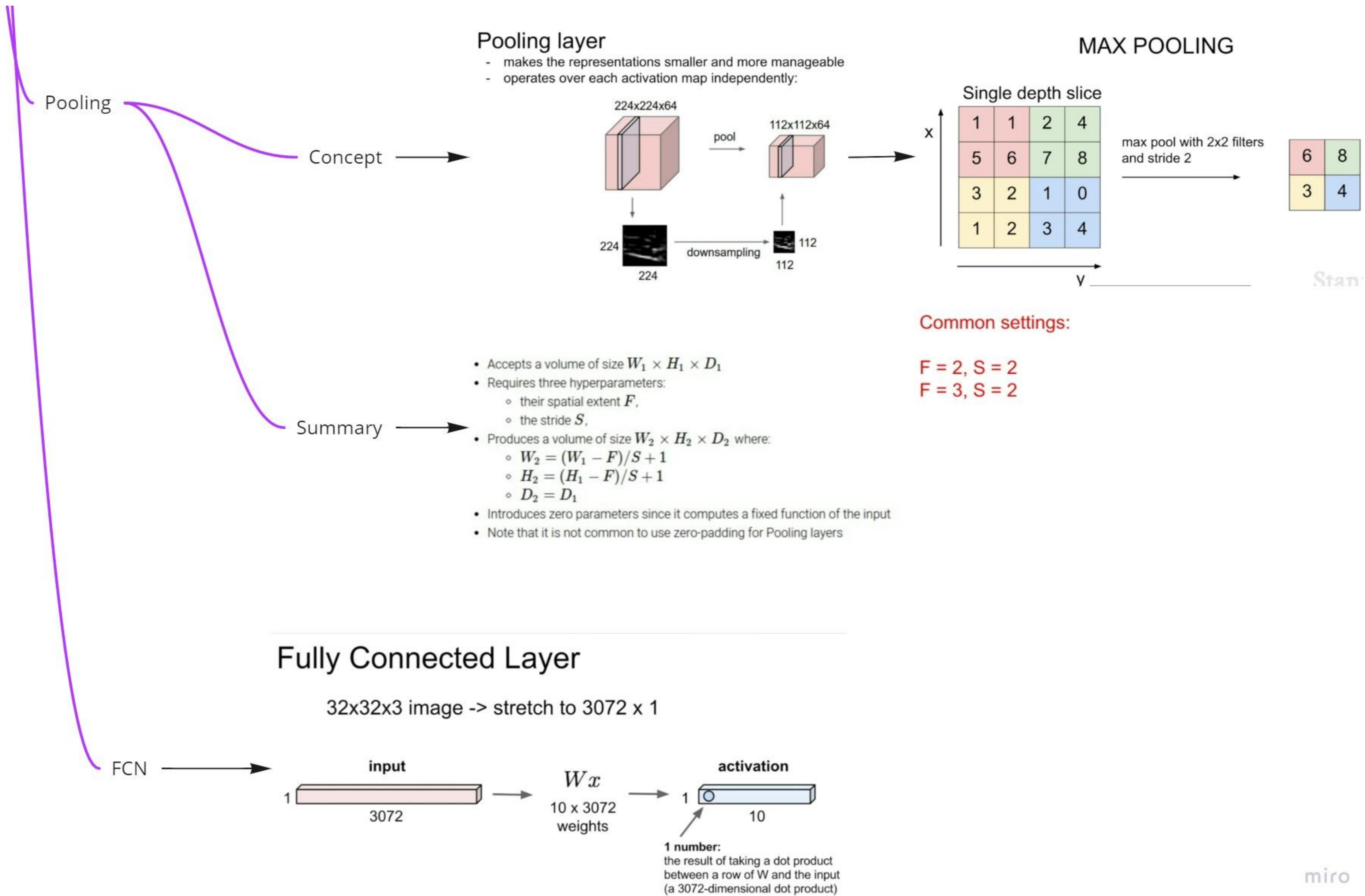
Common settings:

$K = (\text{powers of 2, e.g. 32, 64, 128, 512})$

- $F = 3, S = 1, P = 1$
- $F = 5, S = 1, P = 2$
- $F = 5, S = 2, P = ?$ (whatever fits)
- $F = 1, S = 1, P = 0$

Сверточные нейронные сети

Pooling и полносвязные слои



Сверточные нейронные сети

Назначение слоев

- INPUT [32x32x3] will hold the raw pixel values of the image, in this case an image of width 32, height 32, and with three color channels R,G,B.
- CONV layer will compute the output of neurons that are connected to local regions in the input, each computing a dot product between their weights and a small region they are connected to in the input volume. This may result in volume such as [32x32x12] if we decided to use 12 filters.
- RELU layer will apply an elementwise activation function, such as the $\max(0, x)$ thresholding at zero. This leaves the size of the volume unchanged ([32x32x12]).
- POOL layer will perform a downsampling operation along the spatial dimensions (width, height), resulting in volume such as [16x16x12].
- FC (i.e. fully-connected) layer will compute the class scores, resulting in volume of size [1x1x10], where each of the 10 numbers correspond to a class score, such as among the 10 categories of CIFAR-10. As with ordinary Neural Networks and as the name implies, each neuron in this layer will be connected to all the numbers in the previous volume.

Сверточные нейронные сети

Receptive field

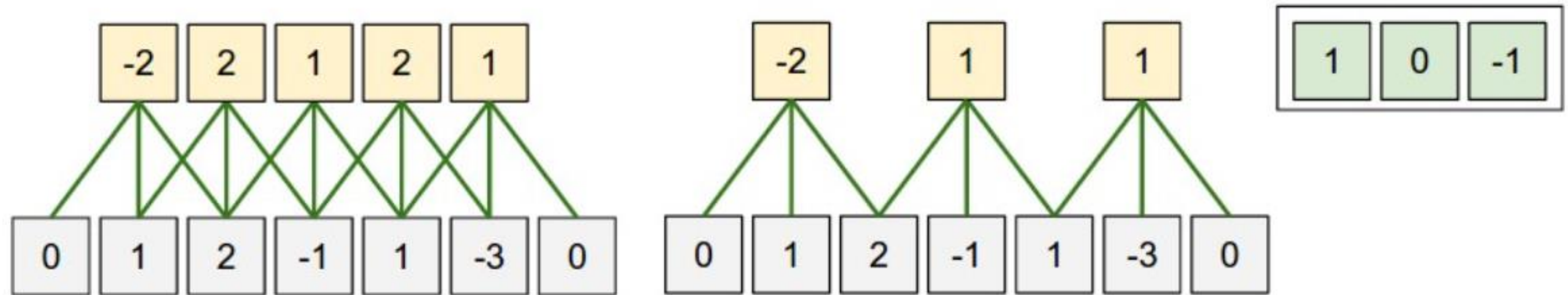


Illustration of spatial arrangement. In this example there is only one spatial dimension (x-axis), one neuron with a receptive field size of $F = 3$, the input size is $W = 5$, and there is zero padding of $P = 1$. **Left:** The neuron strided across the input in a stride of $S = 1$, giving an output of size $(5 - 3 + 2)/1 + 1 = 5$. **Right:** The neuron uses a stride of $S = 2$, giving an output of size $(5 - 3 + 2)/2 + 1 = 3$. Notice that stride $S = 3$ could not be used since it wouldn't fit neatly across the volume. In terms of the equation, this can be determined since $(5 - 3 + 2) = 4$ is not divisible by 3.

The neuron weights are in this example $[1, 0, -1]$ (shown on very right), and its bias is zero. These weights are shared across all yellow neurons (see parameter sharing below).

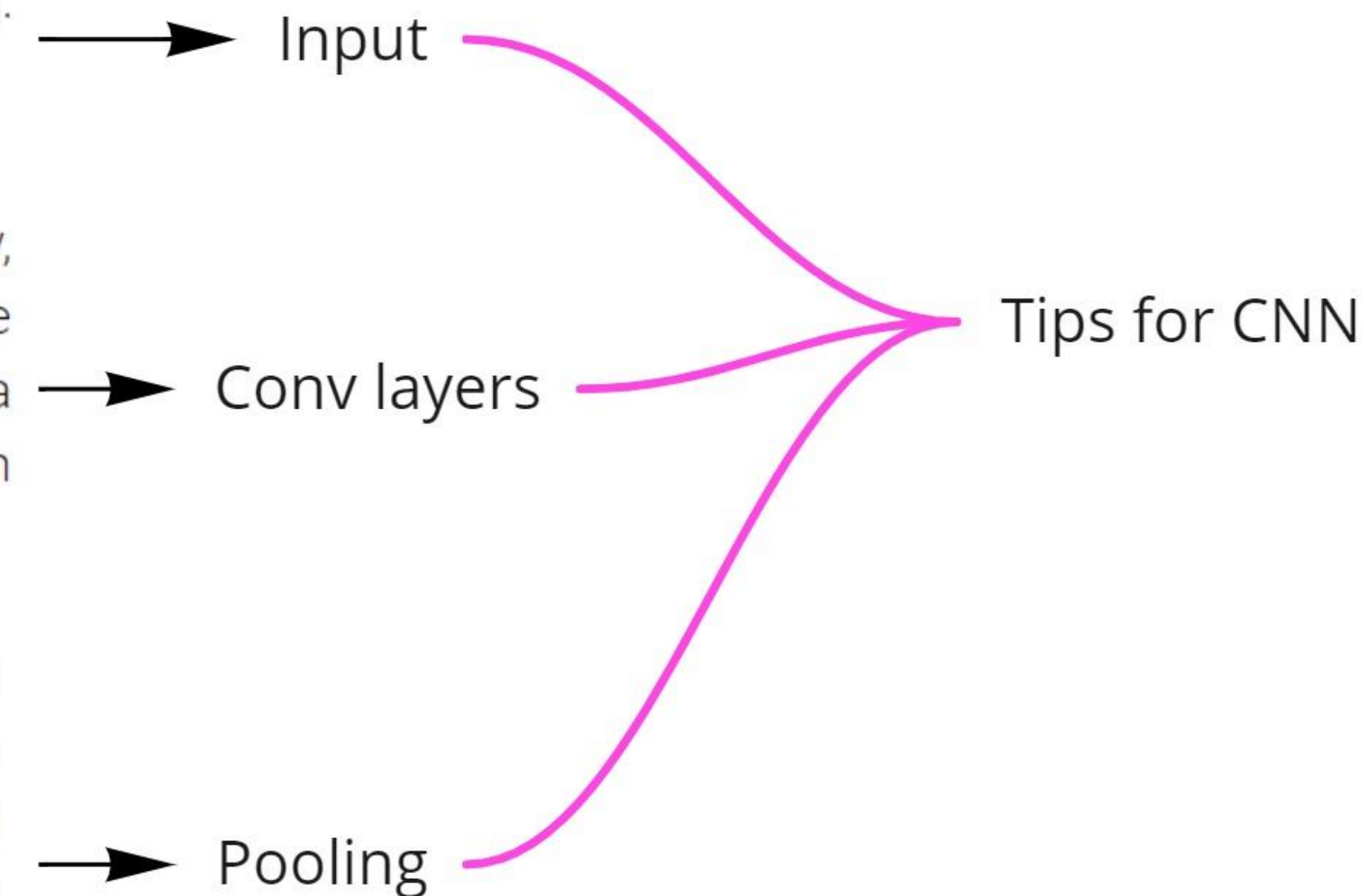
Сверточные нейронные сети

Практические советы

The **input layer** (that contains the image) should be divisible by 2 many times. Common numbers include 32 (e.g. CIFAR-10), 64, 96 (e.g. STL-10), or 224 (e.g. common ImageNet ConvNets), 384, and 512.

The **conv layers** should be using small filters (e.g. 3x3 or at most 5x5), using a stride of $S = 1$, and crucially, padding the input volume with zeros in such way that the conv layer does not alter the spatial dimensions of the input. That is, when $F = 3$, then using $P = 1$ will retain the original size of the input. When $F = 5$, $P = 2$. For a general F , it can be seen that $P = (F - 1)/2$ preserves the input size. If you must use bigger filter sizes (such as 7x7 or so), it is only common to see this on the very first conv layer that is looking at the input image.

The **pool layers** are in charge of downsampling the spatial dimensions of the input. The most common setting is to use max-pooling with 2x2 receptive fields (i.e. $F = 2$), and with a stride of 2 (i.e. $S = 2$). Note that this discards exactly 75% of the activations in an input volume (due to downsampling by 2 in both width and height). Another slightly less common setting is to use 3x3 receptive fields with a stride of 2, but this makes. It is very uncommon to see receptive field sizes for max pooling that are larger than 3 because the pooling is then too lossy and aggressive. This usually leads to worse performance.



- Исторические предпосылки нейросетевых методов
- Описание архитектуры сверточных нейронных сетей
- Особенности сверточных слоев
- Описание полносвязных и pooling слоев
- Практические советы при построении сверточных нейронных сетей

Спасибо за внимание!

**Колокольников
Георгий Андреевич**

Telegram: @Georg_Bell

E-mail: geokolok5@gmail.com

Сайт: <https://github.com/GeorgBell>

Использованные материалы:

- Гонсалес Р., Вудс Р. Цифровая обработка изображений. – М.: Техносфера, 2012. – 1104 с. – ISBN 978-5-94836-331-8.2.
- Курс лекций cs231n «Convolutional Neural Networks for Visual Recognition» (<http://cs231n.stanford.edu>).
- Курс лекций HSE «Deep Learning in Computer Vision» (<https://www.coursera.org/learn/deep-learning-in-computer-vision>)