Convolutional Neural Networks

Dr. Wojtek Kowalczyk

wojtek@liacs.nl

Why do we need many layers?

- In theory, one hidden layer is sufficient to model any function with arbitrary accuracy; however, the number of required nodes and weights grows exponentially fast
- The deeper the network the less nodes are required to model "complicated" functions
- Consecutive layers "learn" features of the training patterns; from simplest (lower layers) to more complicated (top layers)
- Visual cortex consists of about 10 layers

Why can't we just add more layers?

- In practice, "classical" multi-layer networks work fine only for a very small number of hidden layers (typically 1 or 2) - this is an empirical fact ...

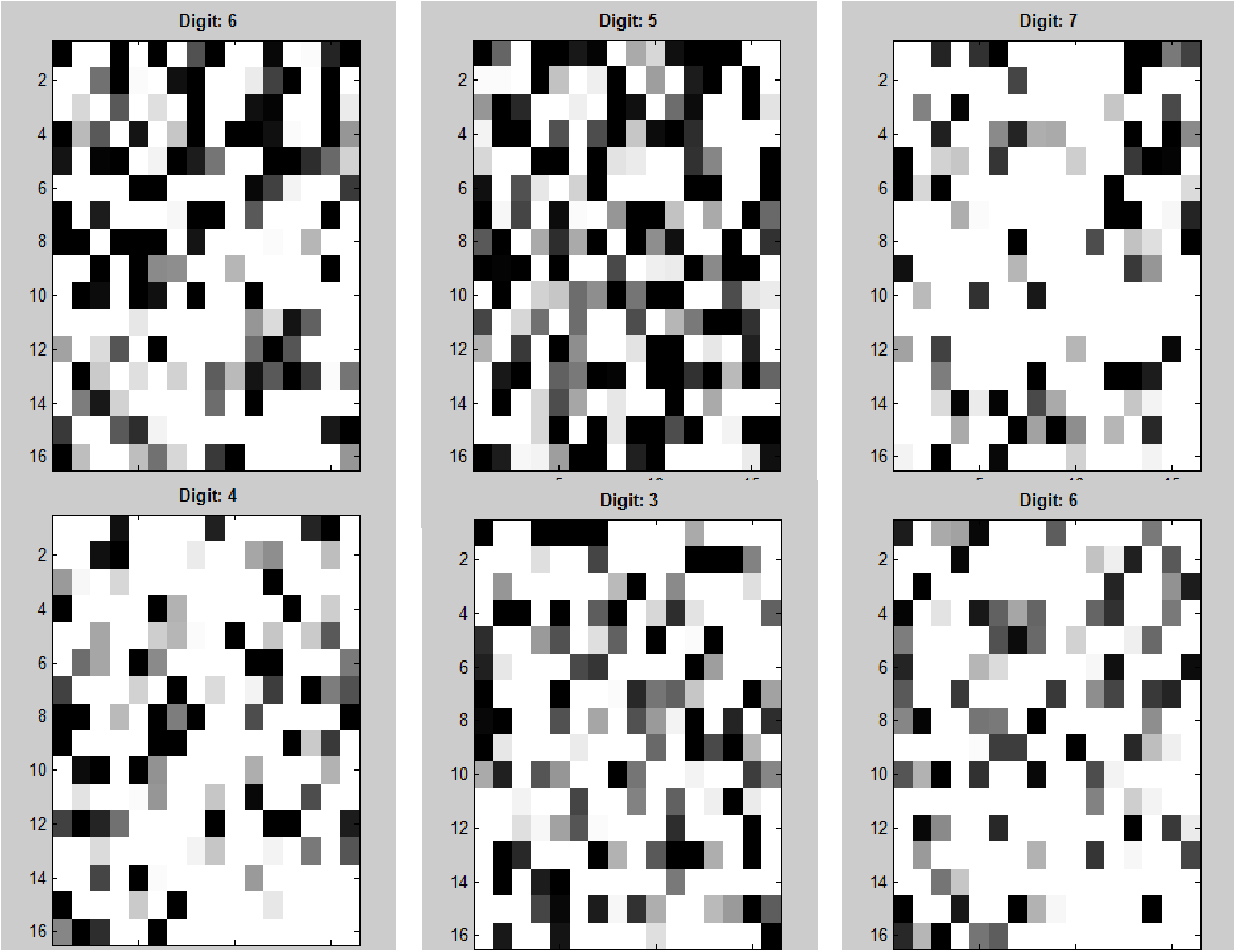
- Adding layers is harmful because:
  - the increased number of weights quickly leads to data overfitting (lack of generalization)
  - huge number of (bad) local minima trap the gradient descent algorithm
  - vanishing or exploding gradients (the update rule involves products of many numbers) cause additional problems

A disturbing observation

- Consider the *digit recognition problem* (16x16)
- Let us modify the images by *randomly permuting all pixels*: take a random permutation *p* and change every image x[0:16x16] into x[p(0:16x16)]
- What accuracy can be achieved by a single layer perceptron on such a “randomly permuted” data?

THE SAME AS ON THE ORIGINAL DATA!

(the same holds for a multi-layer perceptron)



Digit: 6
Digit: 5
Digit: 7
Digit: 4
Digit: 3
Digit: 6
2
4
6
8
10
12
14
16

MNIST data set

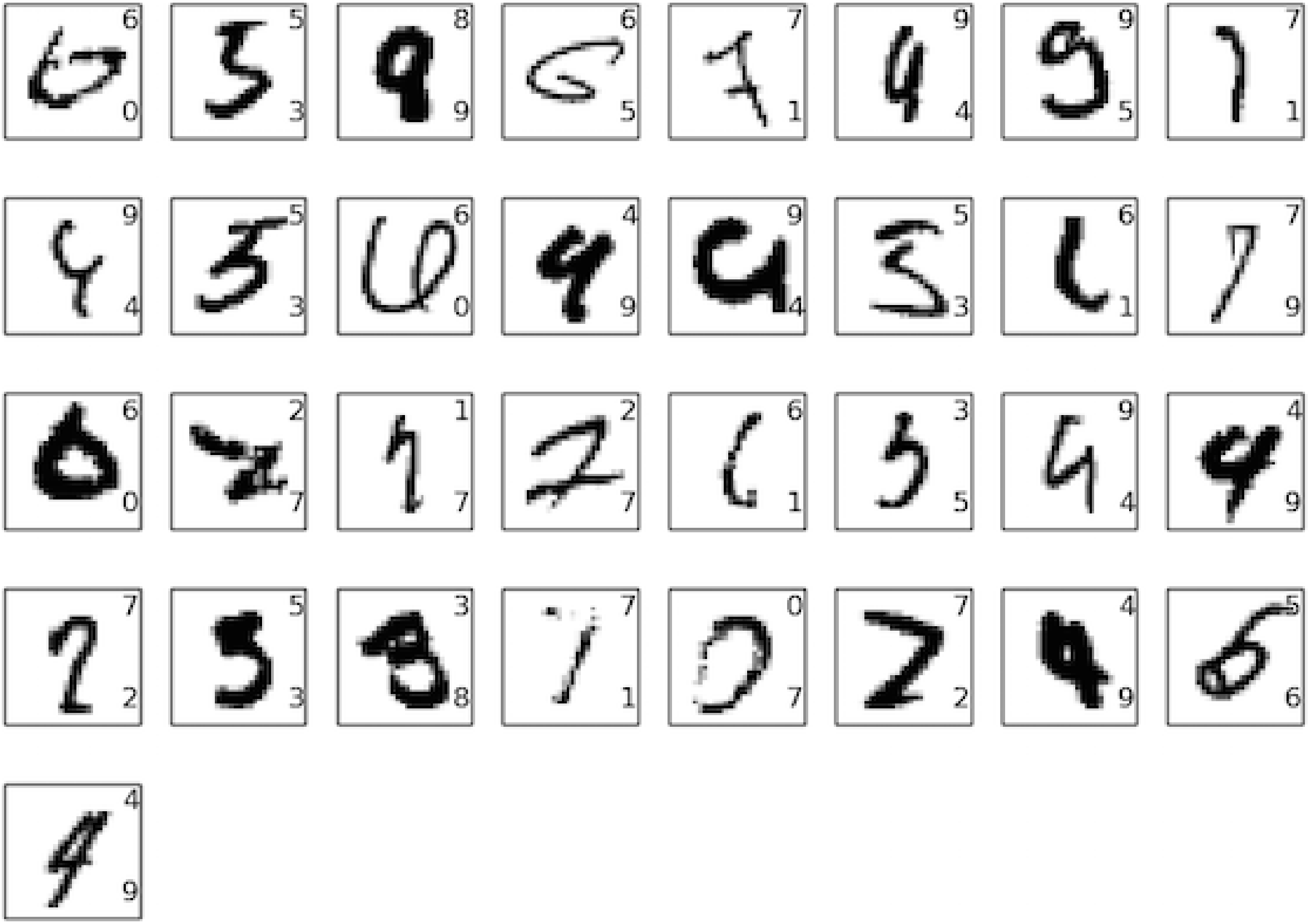
training:
60.000 images

testing:
10.000 images

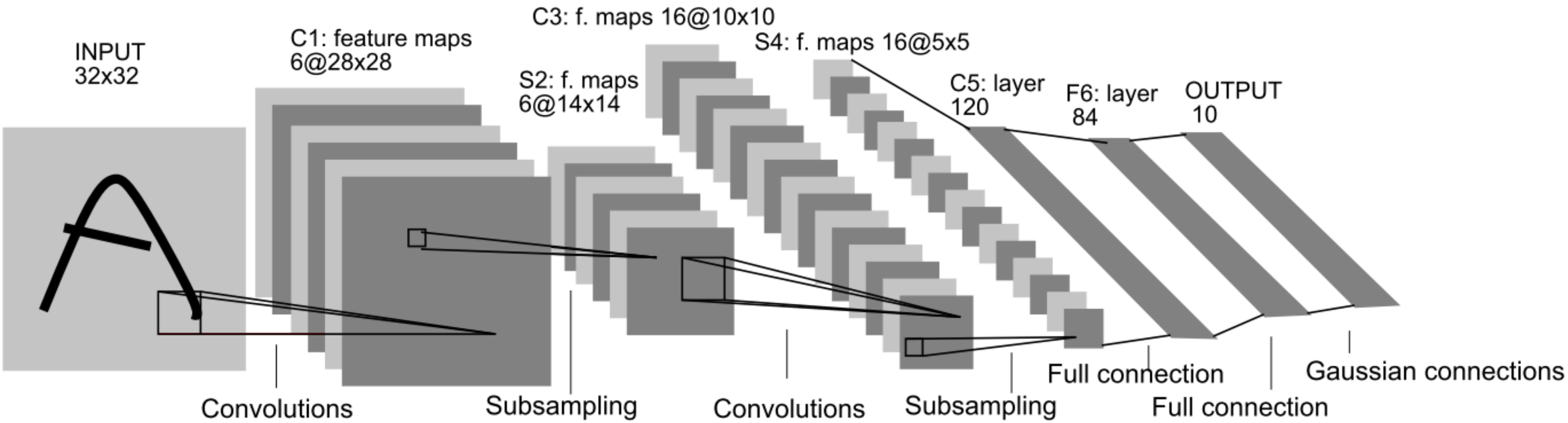
each image:
32x32 pixels

accuracy: 99.7%
(on the test set)

All 33 misclassified digits



LeNet5



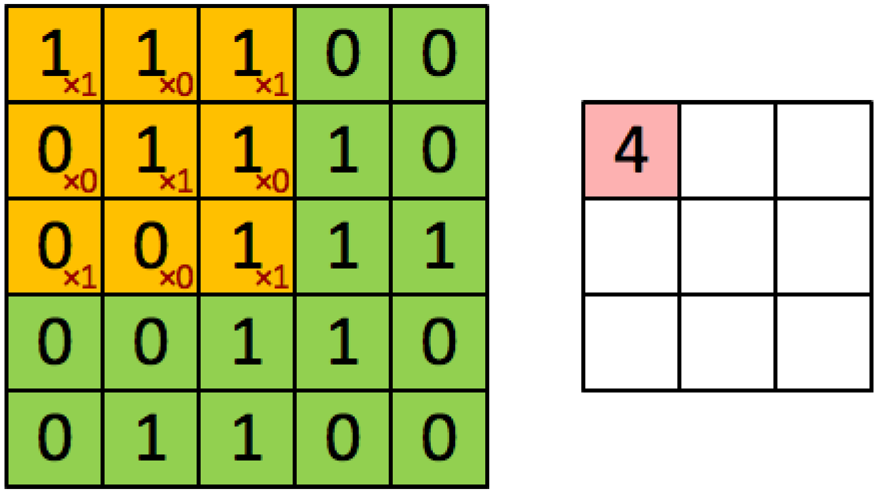
- Input: 32x32 pixel image
- Cx: Convolutional layer
- Sx: Subsample layer
  (reduces image size by averaging 2x2 patches)
- Fx: Fully connected layer

A Convolutional Filter

Let us suppose that in an input image we want to find locations that look like a 3x3 cross. Define a matrix **F** (called a **kernel, receptive field** or **convolutional filter**) and "convolve it" with all possible locations in the image. We will get another (smaller) matrix with "degrees of overlap":

$$F = \begin{matrix} 1 & 0 & 1 \\ 0 & 1 & 0 \\ 1 & 0 & 1 \end{matrix}$$

"multiply and add"



Image

Convolved Feature

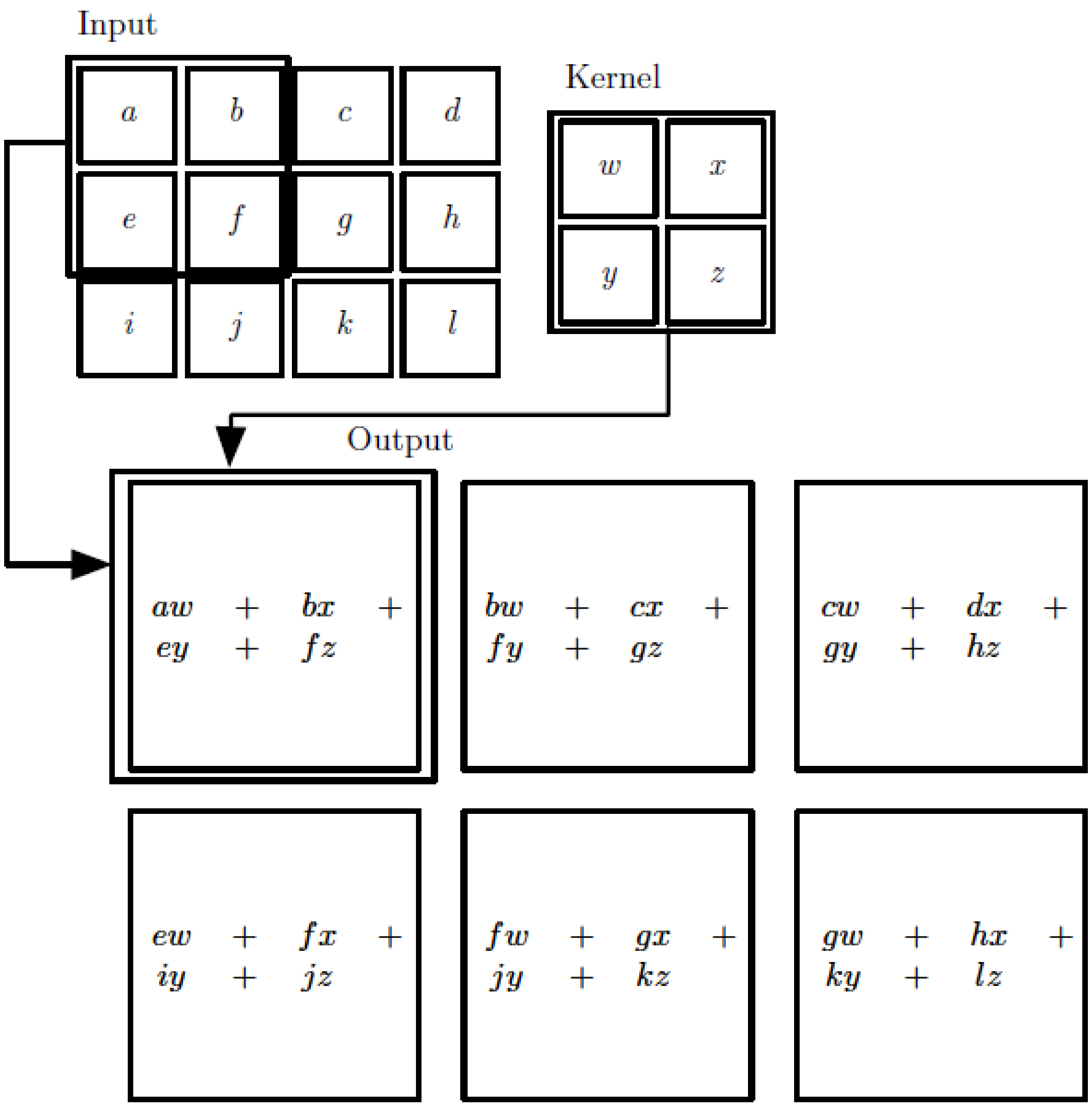


Figure 9.1 (the “Deep Learning” textbook)

Motivation

A filter: a “feature detector” – returns high values when the corresponding patch is similar to the filter matrix

Think about all pixels being **-1** (black) or **+1** (white) and filter parameters also restricted to **-1** and **1**

Example: what features are “detected” by:

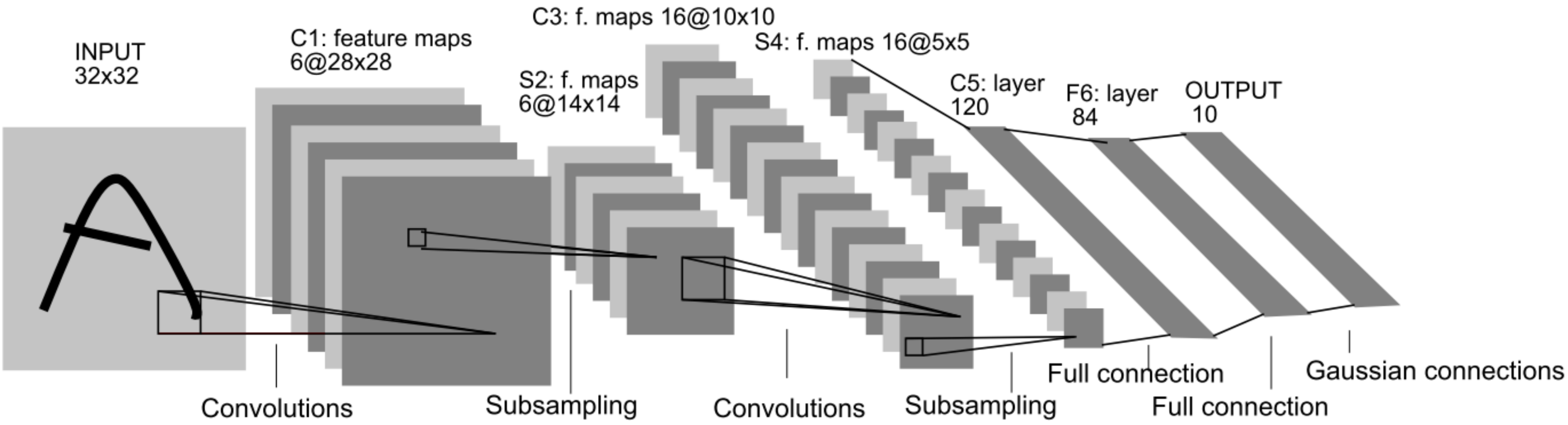
+	-	-
-	+	-
-	-	+

+	-	+
-	+	-
+	-	+

-	+	-
-	+	-
-	+	-

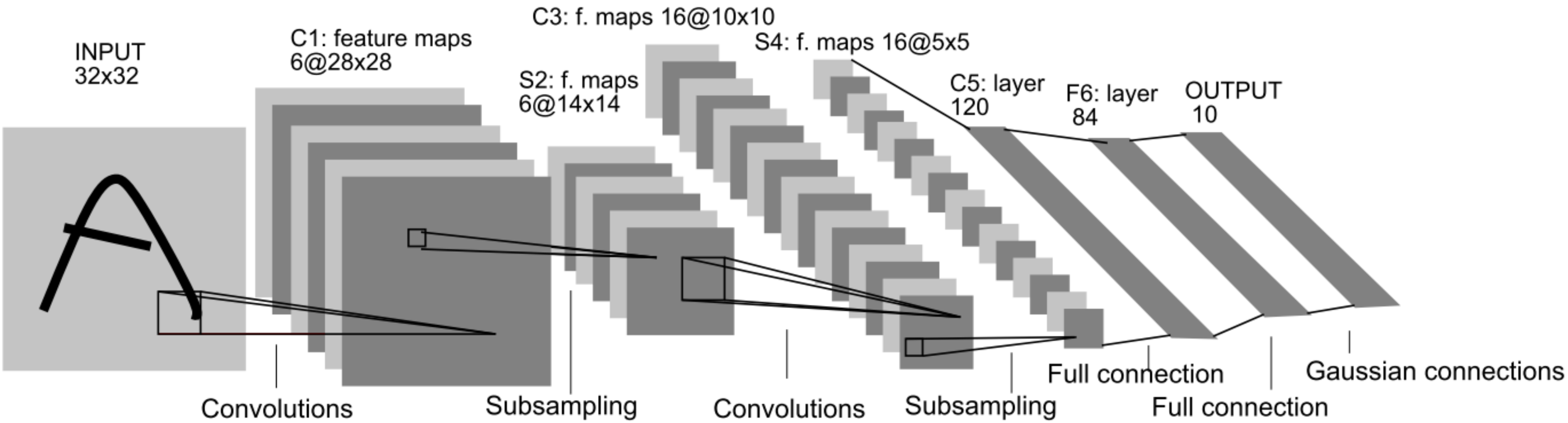
-	-	-
-	+	-
-	-	-

LeNet5



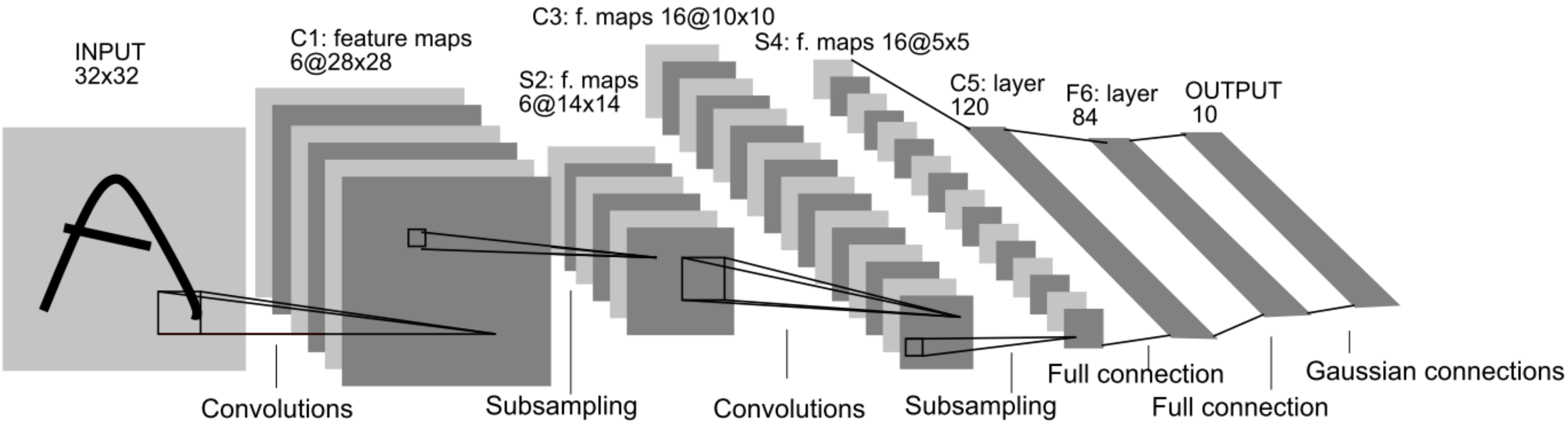
- Input: 32x32 pixel image
- Cx: Convolutional layer
- Sx: Subsample layer
  (reduces image size by averaging 2x2 patches)
- Fx: Fully connected layer

LeNet 5: Layer C1



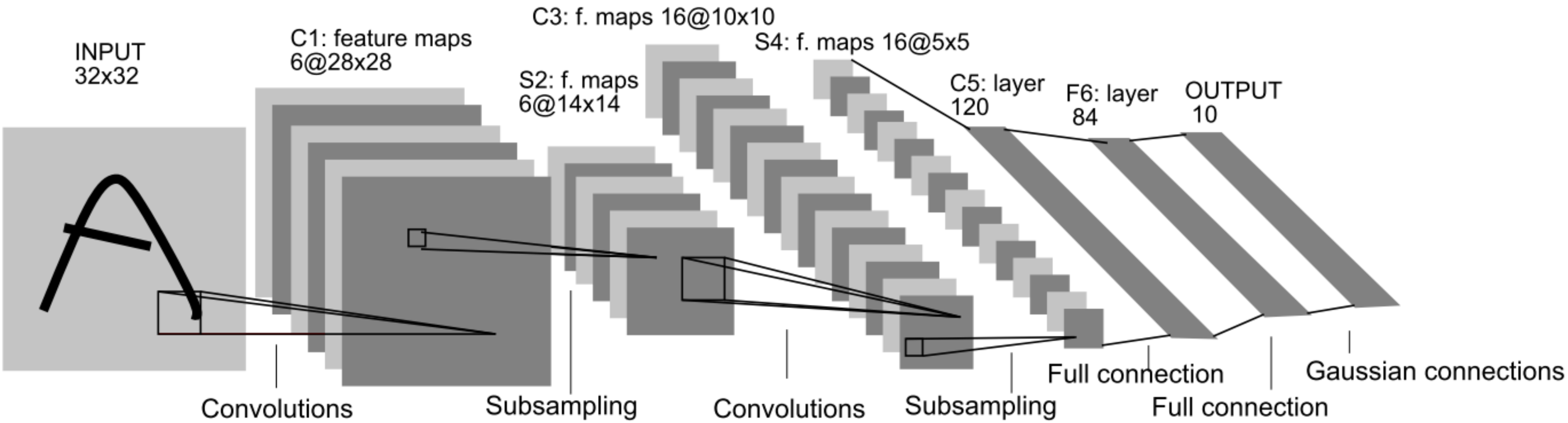
- C1: Convolutional layer with 6 feature maps of size 28x28.
- Each unit of C1 has a 5x5 receptive field in the input layer.
- Shared weights (5*5+1)*6=**156 parameters to learn**
  Connections: 28*28*(5*5+1)*6=122304
- If it was fully connected we had:
  (32*32+1)*(28*28)*6 = **4.821.600 parameters**

LeNet 5: Layer S2



- S2: Subsampling layer with 6 feature maps of size 14x14 2x2 nonoverlapping receptive fields in C1
- Layer S2: 6*2=**12 trainable parameters**.
- Connections: 14*14*(2*2+1)*6=5880

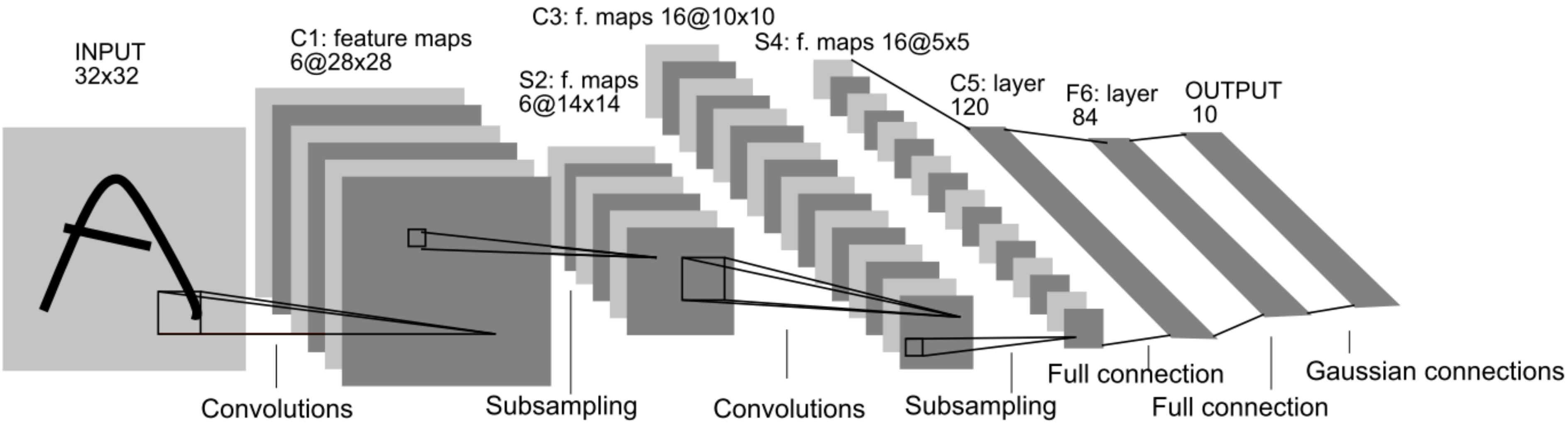
... and so on ...



Study slides 11-27 of DeepLearning.pdf

Read the original paper lecun-01a.pdf

LeNet 5: totals



- The whole network has:
  - 1256 nodes
  - 64.660 connections
  - **9.760 trainable parameters (and not millions!)**
  - **trained with the Backpropagation algorithm!**

Misclassified cases

From LeNet5 to ImageNet (2010/2012)

ImageNet

- 15M images
- 22K categories
- Images collected from Web
- RGB Images
- Variable-resolution
- Human labelers (Amazon's Mechanical Turk crowd-sourcing)

ImageNet Large Scale Visual Recognition Challenge (ILSVRC-2010)

- 1K categories
- 1.2M training images (~1000 per category)
- 50,000 validation images
- 150,000 testing images

ImageNet (study slides 28-40)

- ILSVRC-2010 test set

Model	Top-1	Top-5
Sparse coding [2]	*47.1%*	*28.2%*
SIFT + FVs [24]	*45.7%*	*25.7%*
CNN	**37.5%**	**17.0%**

- ILSVRC-2012 test set

Model	Top-1 (val)	Top-5 (val)	Top-5 (test)
SIFT + FVs [7]	—	—	26.2%
1 CNN	40.7%	18.2%	—
5 CNNs	38.1%	16.4%	**16.4%**
1 CNN*	39.0%	16.6%	—
7 CNNs*	36.7%	15.4%	**15.3%**

Key Points

- convolutions, feature maps, kernels, ...
- subsampling/pooling
- weights sharing
- ReLU (Rectified Linear Unit)
- Data Augmentation
- Dropout

Homework

- Study slides 11-40 of DeepLearning.pdf
- Study Chapter 6 of Nielsen's book:

  http://neuralnetworksanddeeplearning.com/chap6.html
- Read (but don't get intimidated!) Chapter 9 of the "Deep Learning" textbook
- How a random permutation of input pixels (in the training and testing sets) would affect the accuracy of CNNs?