Image Recognition with Neural Networks

based on www.neuralnetworksanddeeplearning.com

Recognizing handwritten digits

Problem: Make a neural network recognizing handwritten digits.

Use a large set of normalized pictures of handwritten digits:
27x27 pixels, greyscale, centered digit, upright

Figure: Sample of the MNIST data set.

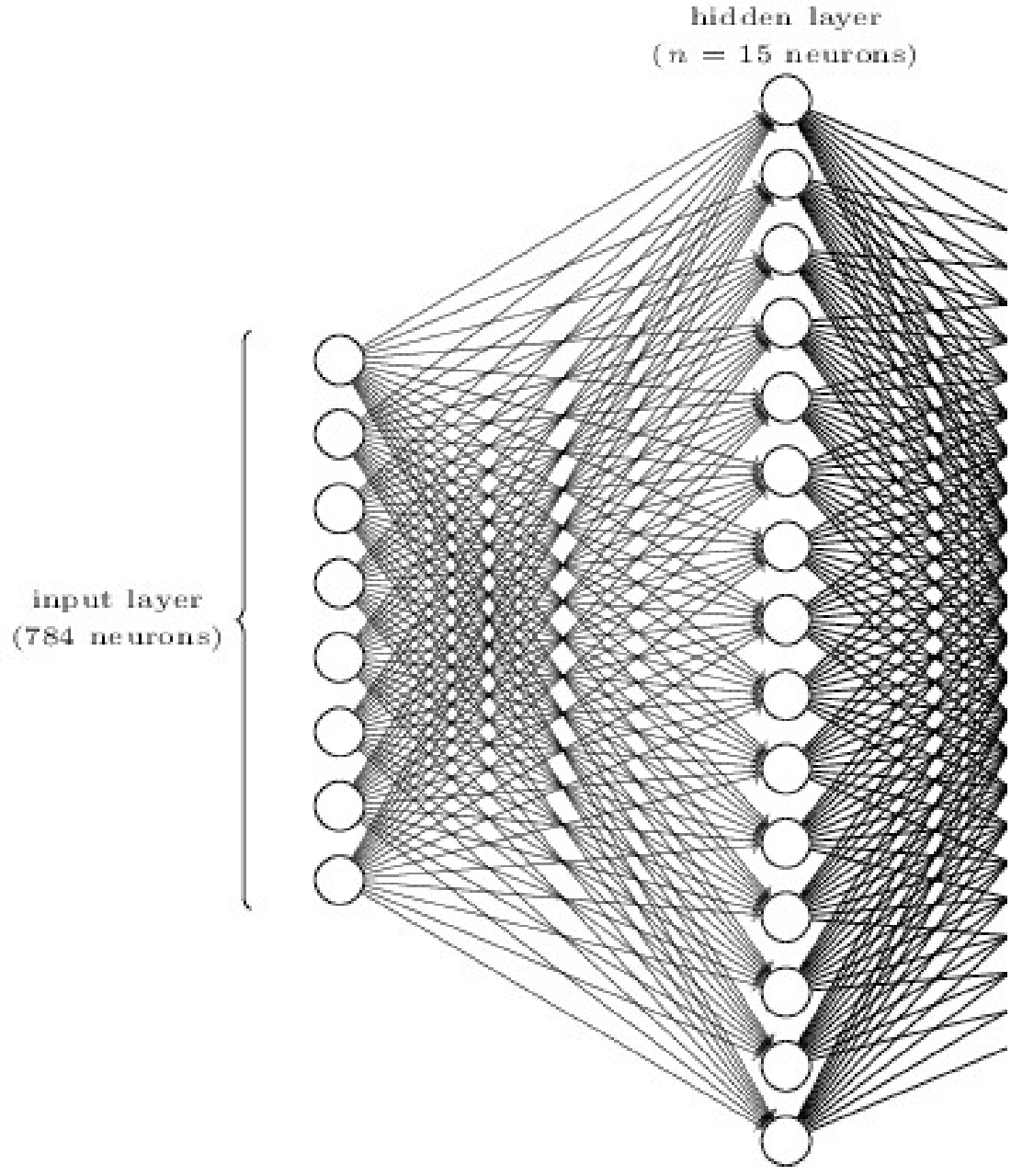
Input layer

How can we encode such a picture and feed it into a neural network?

Each pixel is associated with a number between 0 and 1.

This gives a 27x27 vector of such numbers
$\rightarrow$ 27x27 neurons for the input layer

Input layer



Output layer

How does the output layer look like?

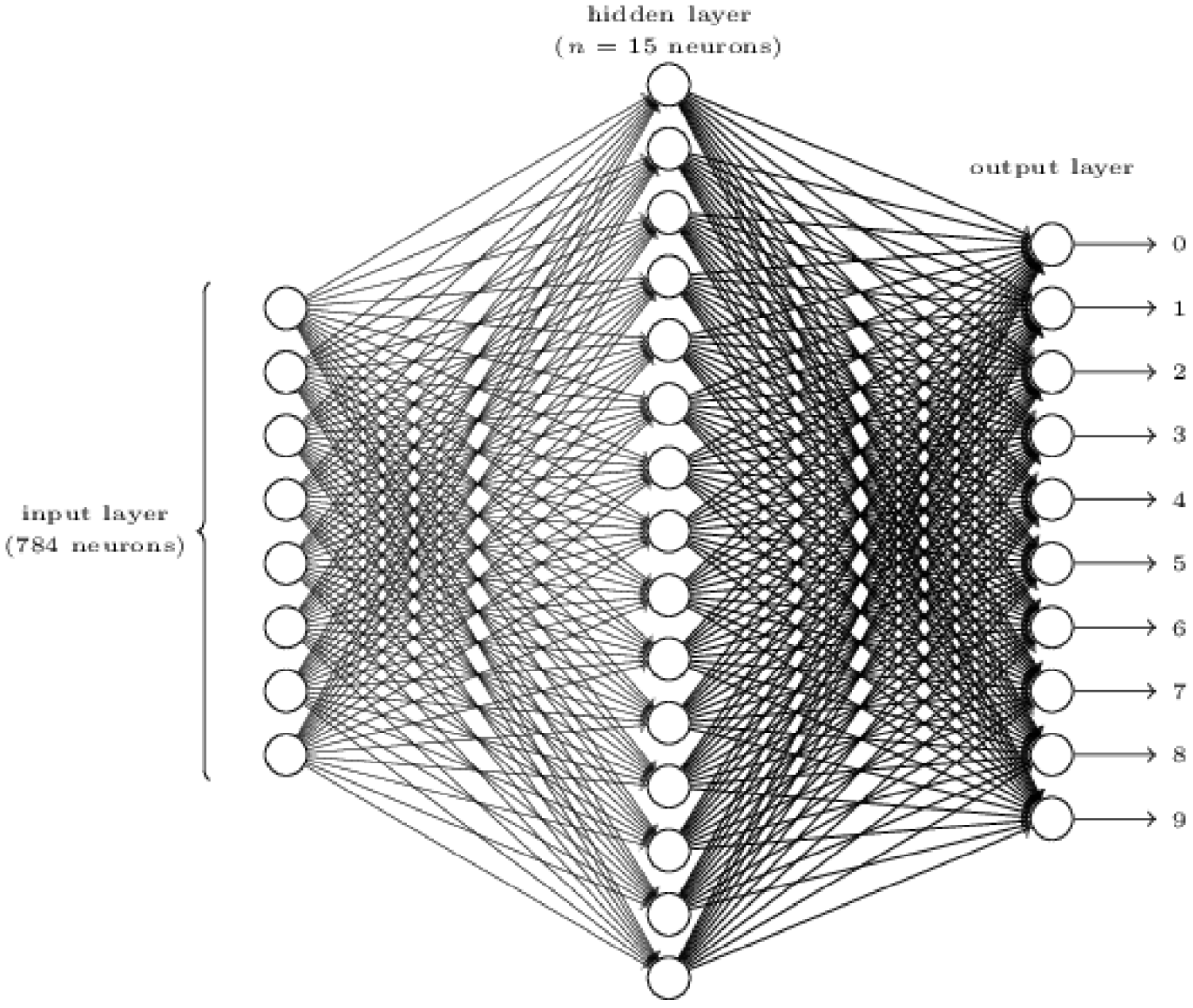
We need at least 10 different output possibilities to distinguish 10 different digits.

One possibility: 4 output neurons
$\rightarrow 2^4 = 16$ output possibilities

Alternative: 10 output neurons
$\rightarrow$ each neuron specializes on detection of one digit

It turns out empirically that the latter construction is more effective!

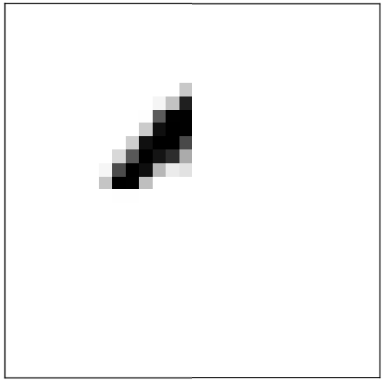
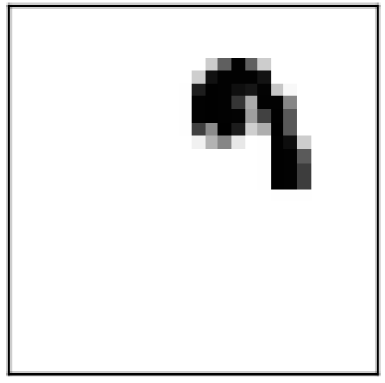
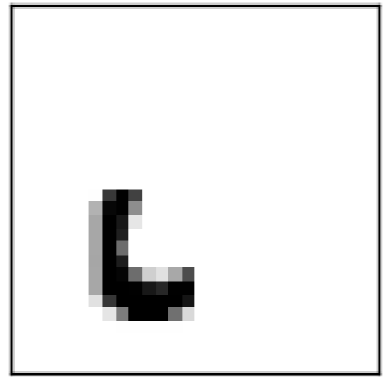
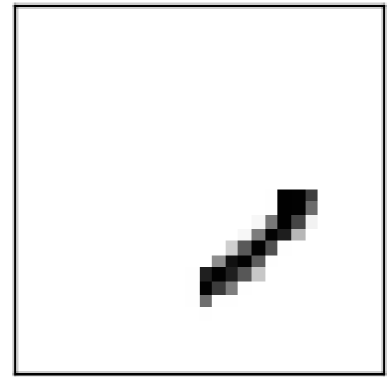
A fully connected neural network



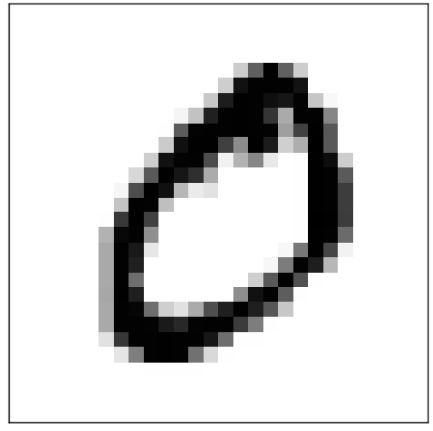
Can we understand this preference heuristically?

Suppose the network works in the following way:

Each hidden neuron checks for a certain geometric pattern in the picture.

This combines to:



If we had only 4 output neurons, each of them would have to tell from this kind of information the correct digit. This is because the combination of all outputs determines the result (for example (0,0,0,0)=0).

In the case of 10 output neurons each of them only has to decide whether it is the number it is specialized on or not.

Important: We do not really know whether this is the way the network functions. This is only heuristics!

Convolutional networks

So far we have seen a network that is fully connected; each neuron is connected to all neurons in the next layer.

Such networks can achieve accuracies of 98% but have no notion of spatial structure.

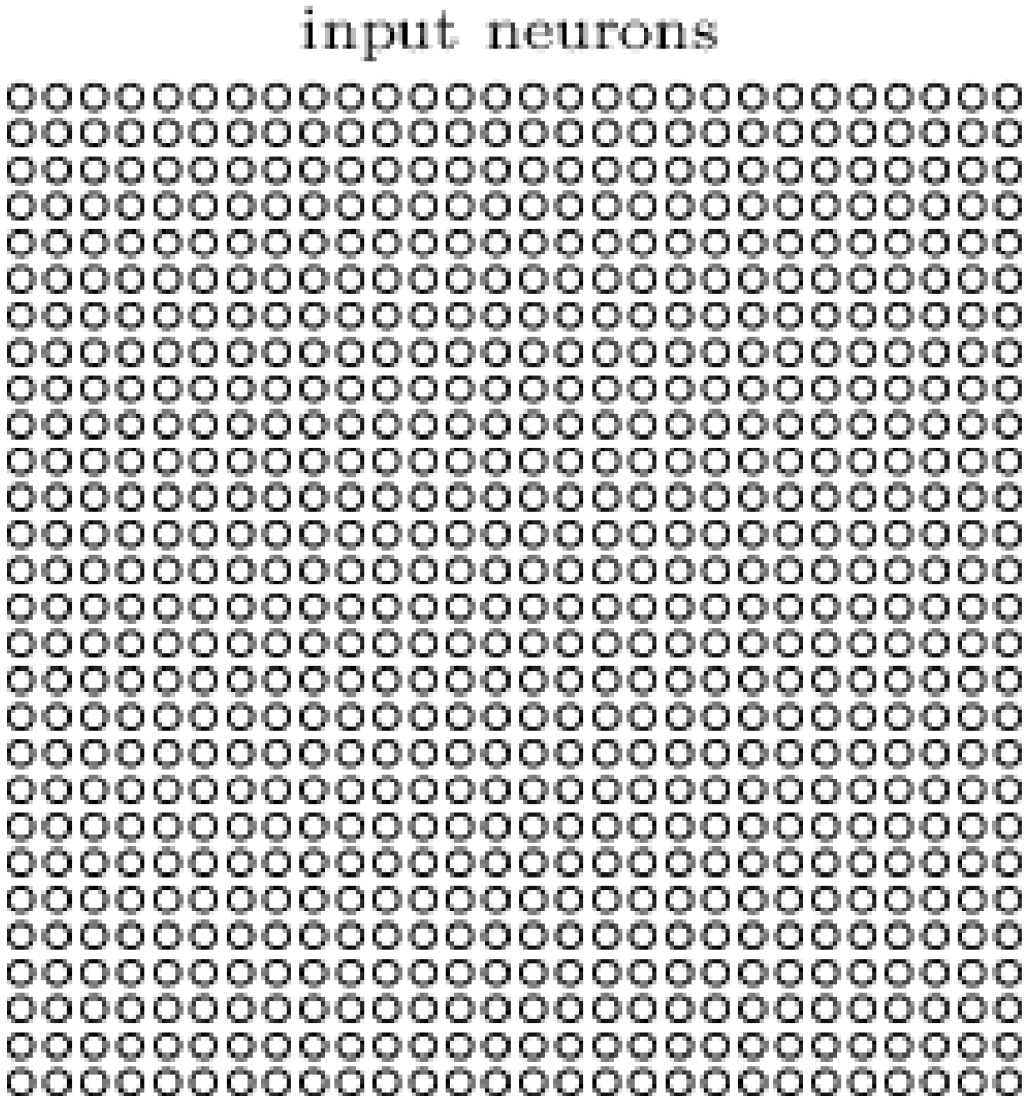
This problem can be solved by so-called convolutional networks.

There are three important concepts:

- Local receptive fields
- Shared weights and biases
- Pooling layers

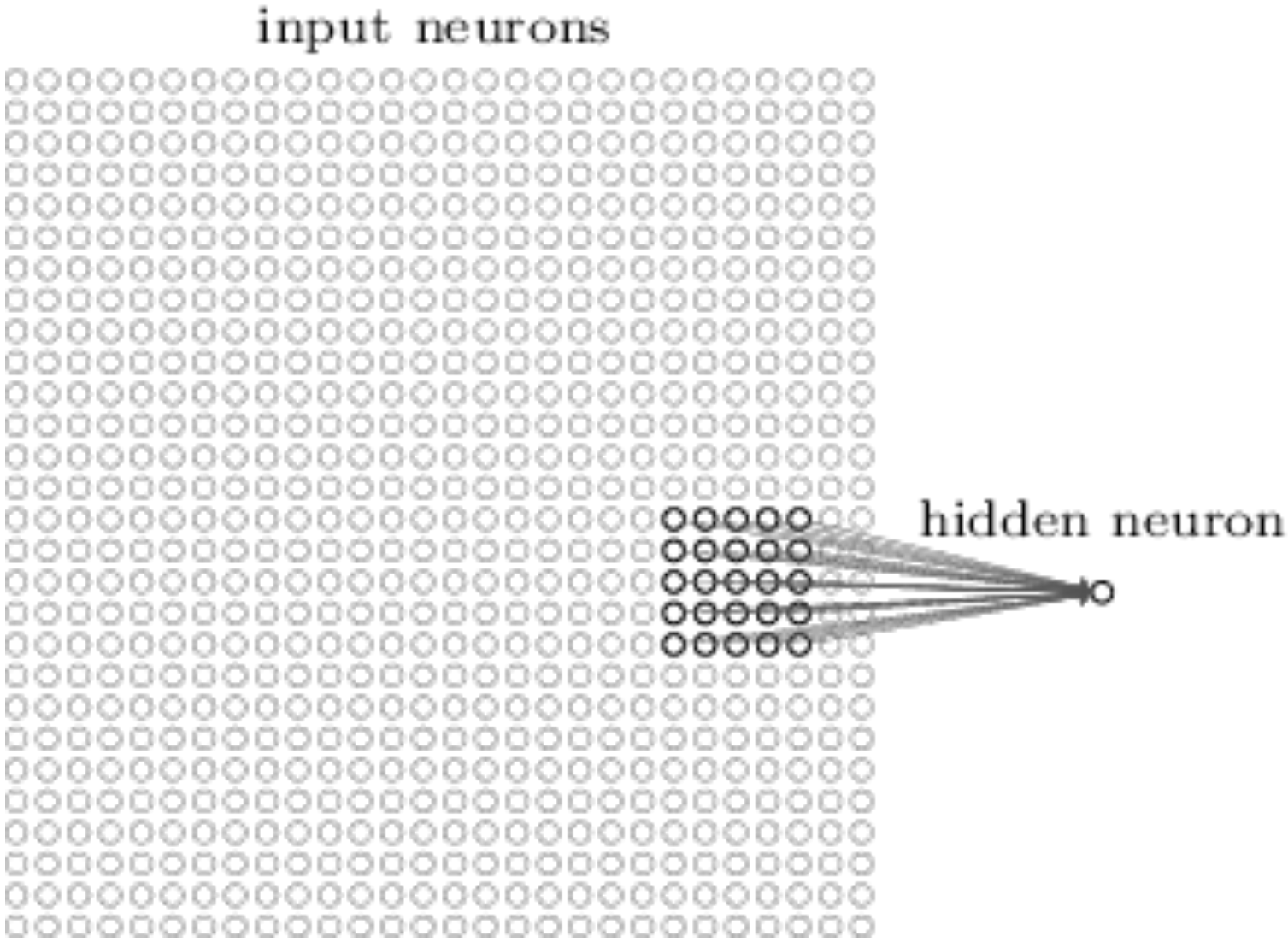
Local receptive fields

Arrange input layer in a matrix corresponding to the image pixels.



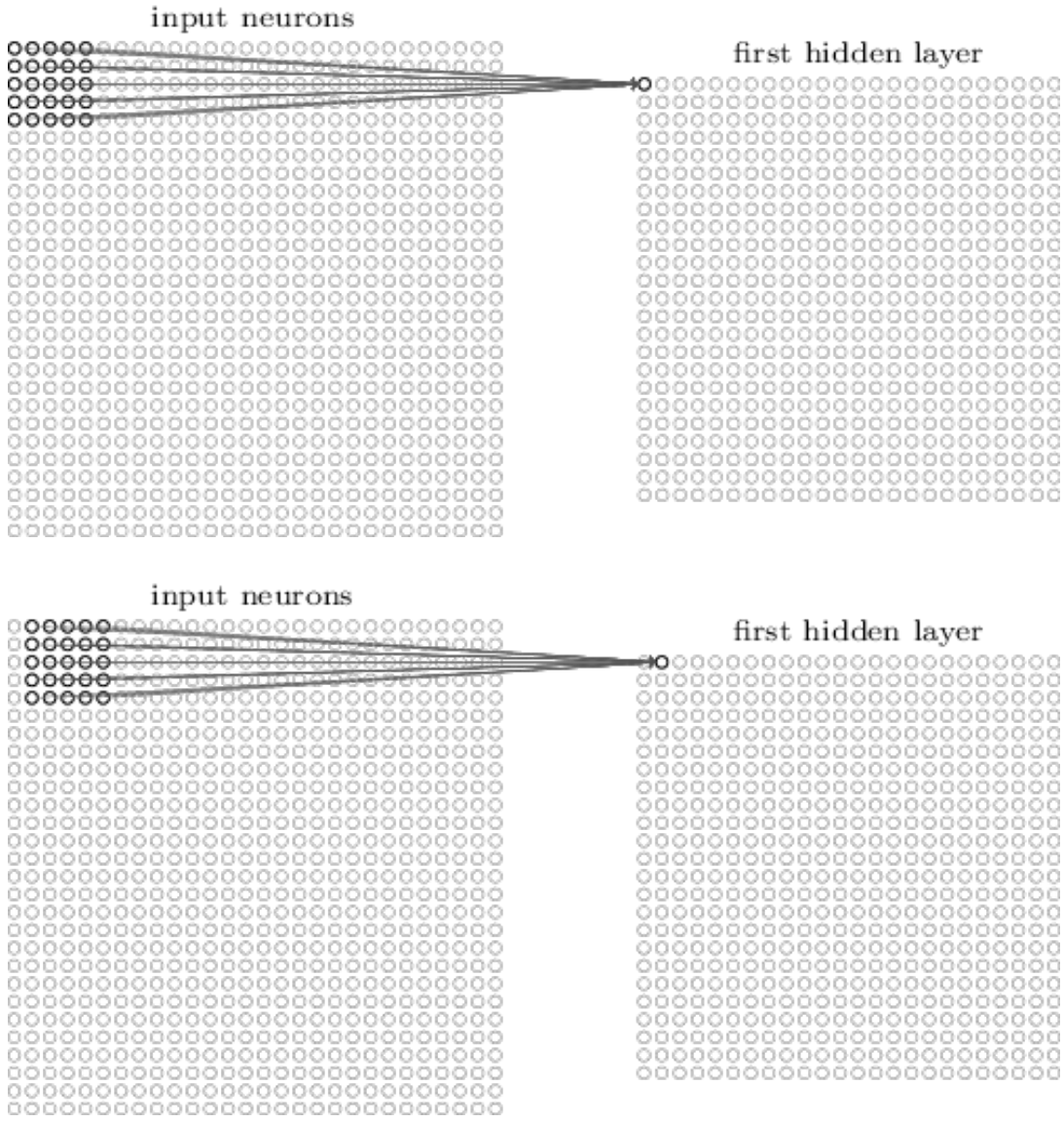
Local receptive fields

We fully connect a 5x5 submatrix (local receptive field) to a neuron in the next layer.



Local receptive fields

We stride this submatrix over the complete input matrix (here: stride length=1).



Shared weights and biases

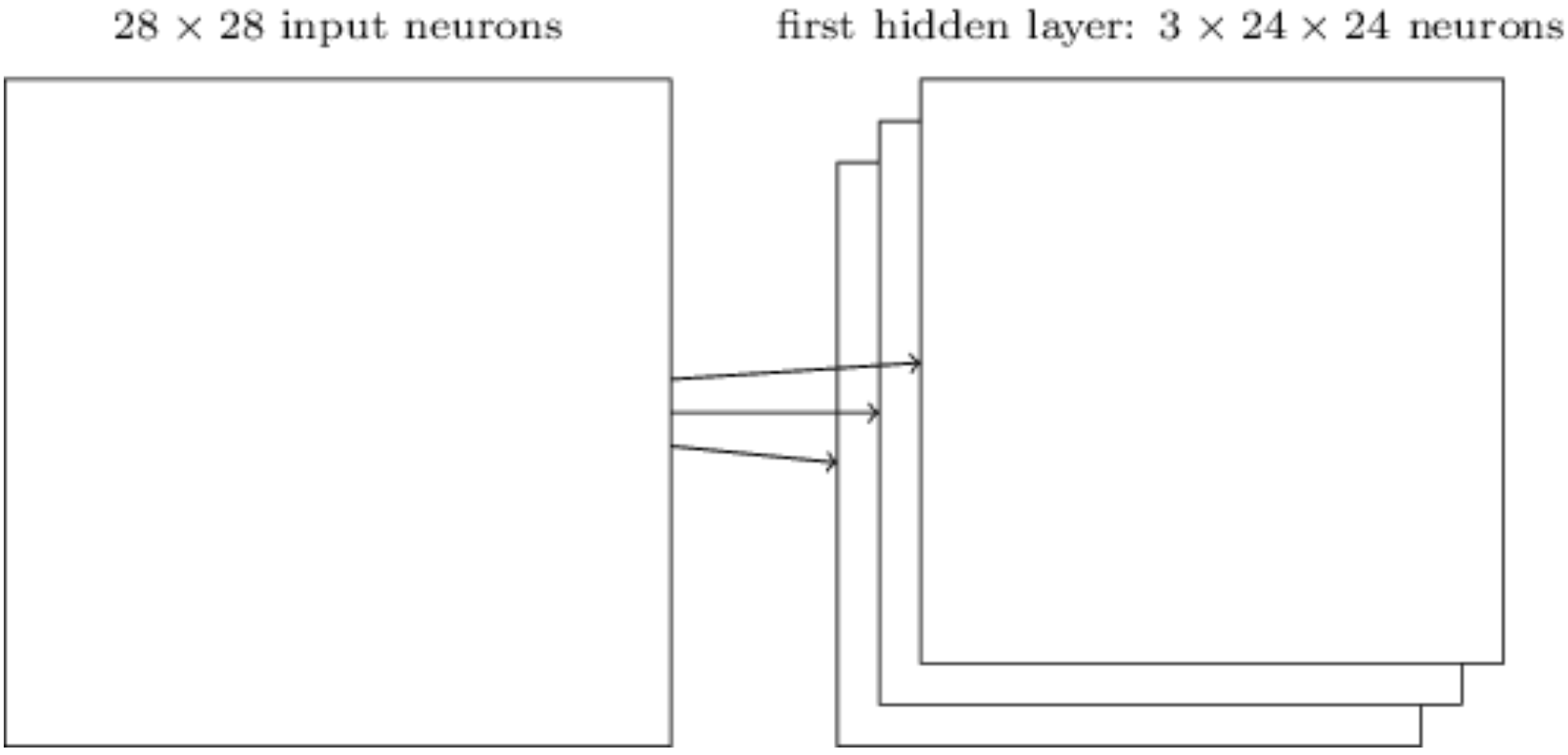
Use the same weights and biases for each hidden neuron.

Hence, all hidden neurons detect the same kind of features in the input picture.

This is therefore called a feature map.

Shared weights and biases

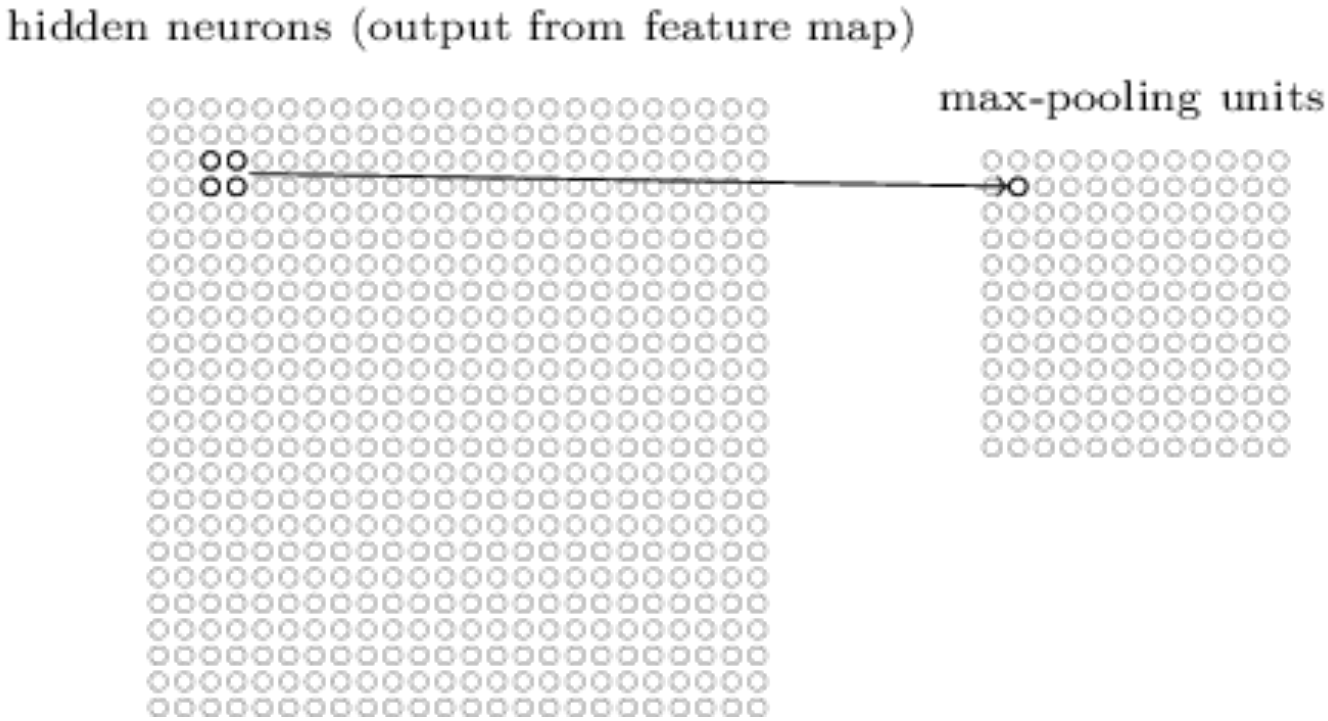
Since a picture has many different features it is useful to have several such feature maps.



This hidden layer is called convolutional layer.

Pooling layers

Pooling layers condense the output of a convolutional layer.

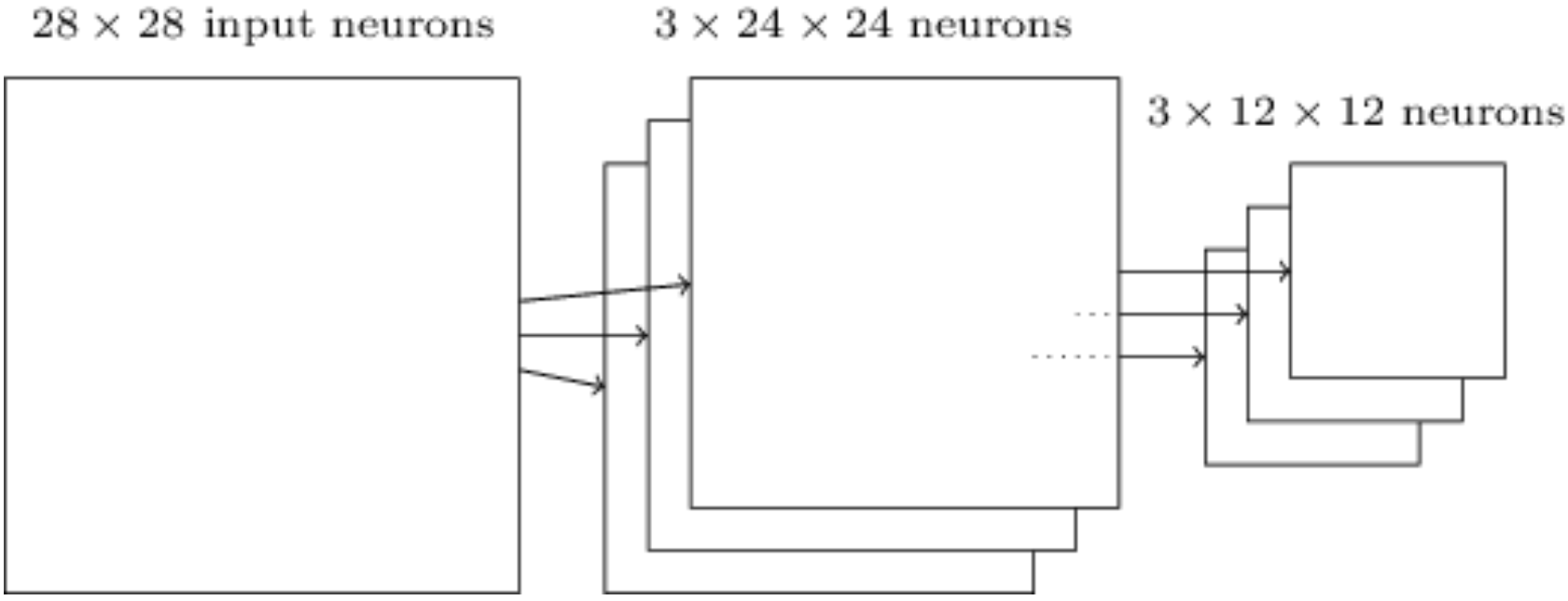


This can be done, for example, by taking only the largest number of the 4 output values (max-pooling).

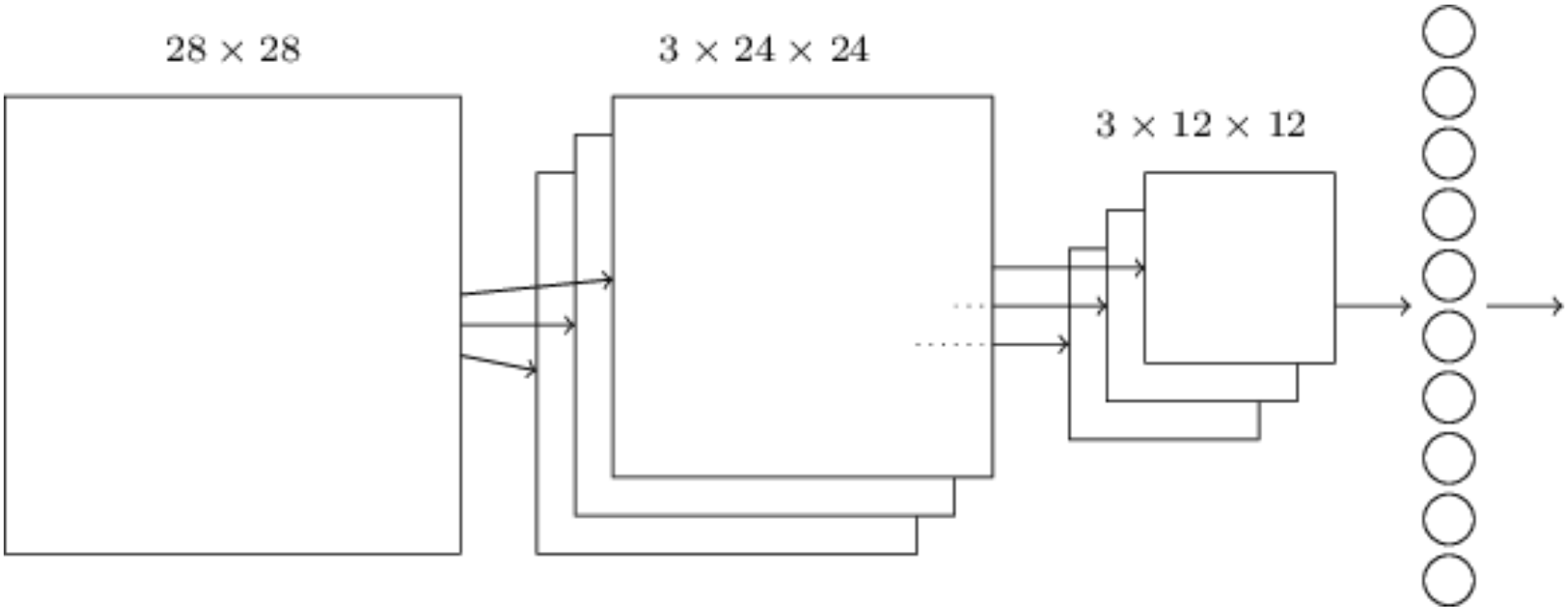
Note that there is (usually) no overlap between the 2x2 matrices in contrast to the convolutional layer.

Pooling layers

There is a pooling layer for each feature map.

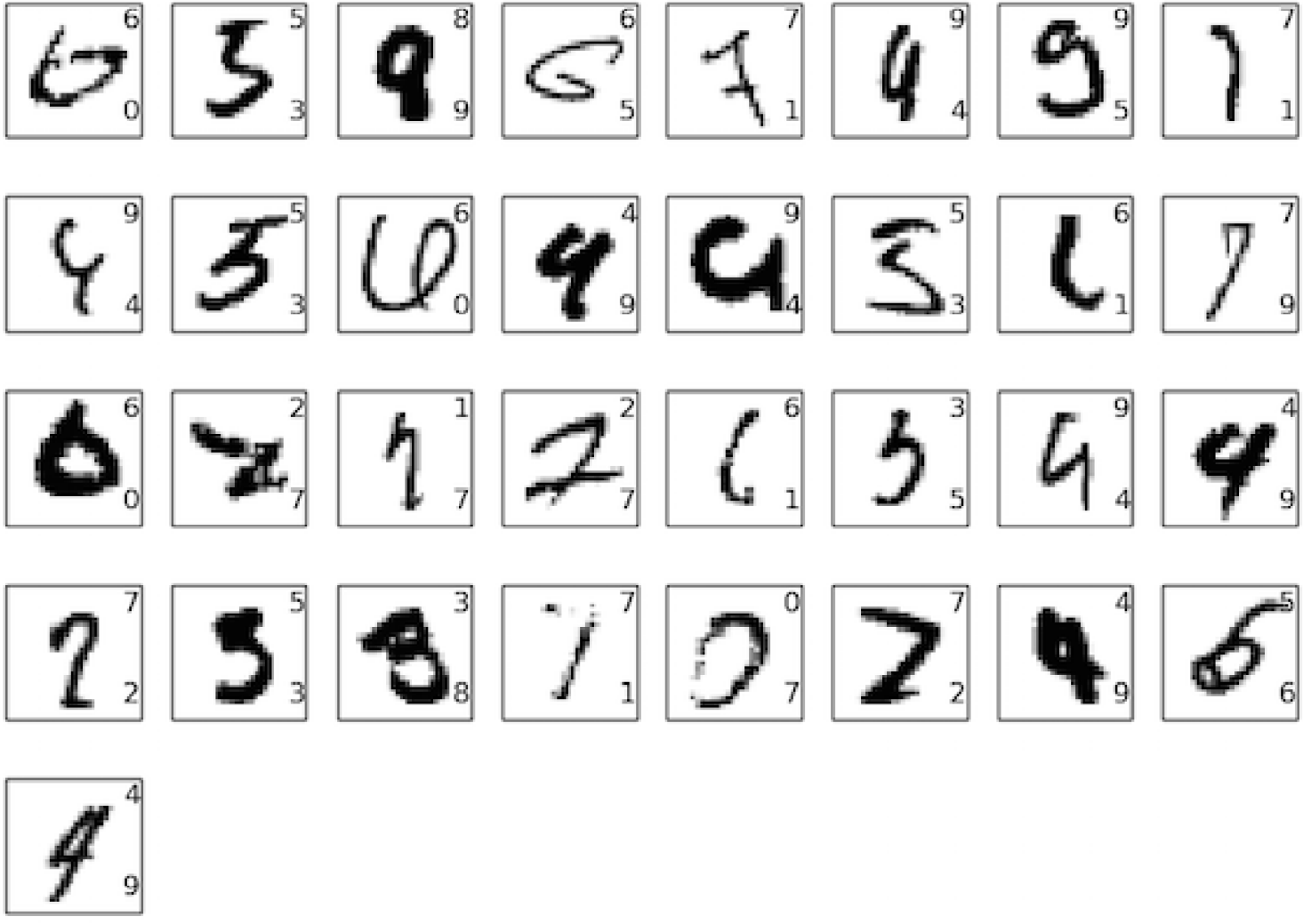


A complete convolutional neural network



Wrongly classified digits

This are 33 out of 10,000 digits a convolutional neural network has wrongly classified (99.67% correct).



Recent developments

Today, the MNIST data set is no longer a real challenge for modern neural networks.

Instead, a set of 1.2 million full color training images in 1000 categories, taken from the image database ImageNet, are used. The test set consists of 150,000 images.

In 2014, the network GoogLeNet achieved a 93.33 % accuracy in having the correct classification among the top 5 predictions.

A team at Google transcribed all Street View images of street numbers in France in less than one hour with an accuracy comparable to that of a human.

Recent developments

Andrej Karpathy and colleagues tried to compare this with human performance:

"...In the end I realized that to get anywhere competitively close to GoogLeNet, it was most efficient if I sat down and went through the painfully long training process and the subsequent careful annotation process myself... The labeling happened at a rate of about 1 per minute, but this decreased over time... Some images are easily recognized, while some images (such as those of fine-grained breeds of dogs, birds, or monkeys) can require multiple minutes of concentrated effort. I became very good at identifying breeds of dogs... Based on the sample of images I worked on, the GoogLeNet classification error turned out to be 6.8%... My own error in the end turned out to be 5.1%, approximately 1.7% better."

Recent developments

In 2013 a group slightly distorted correctly classified images such that they were then wrongly classified.