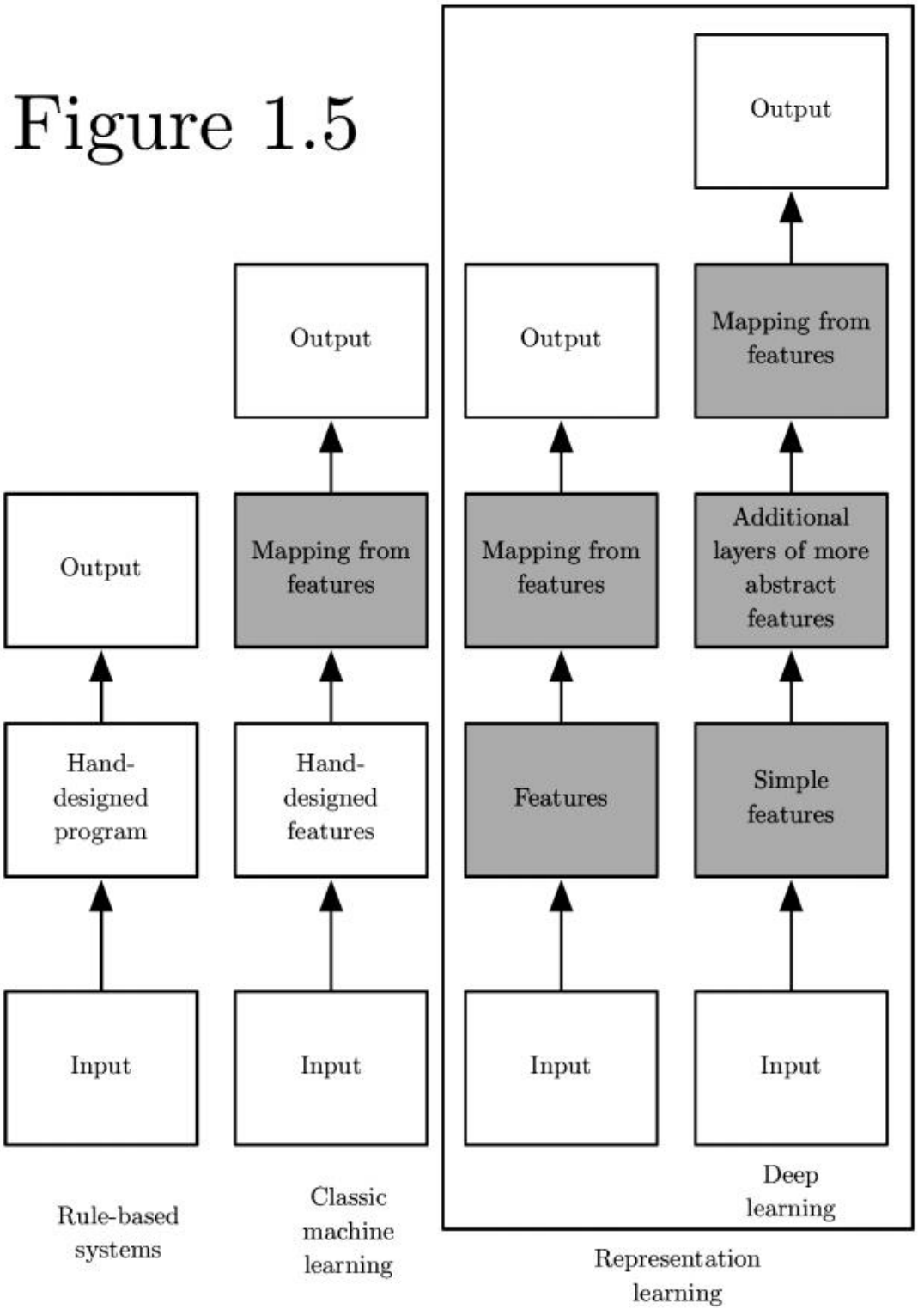
Learning Paradigms



Historical Waves

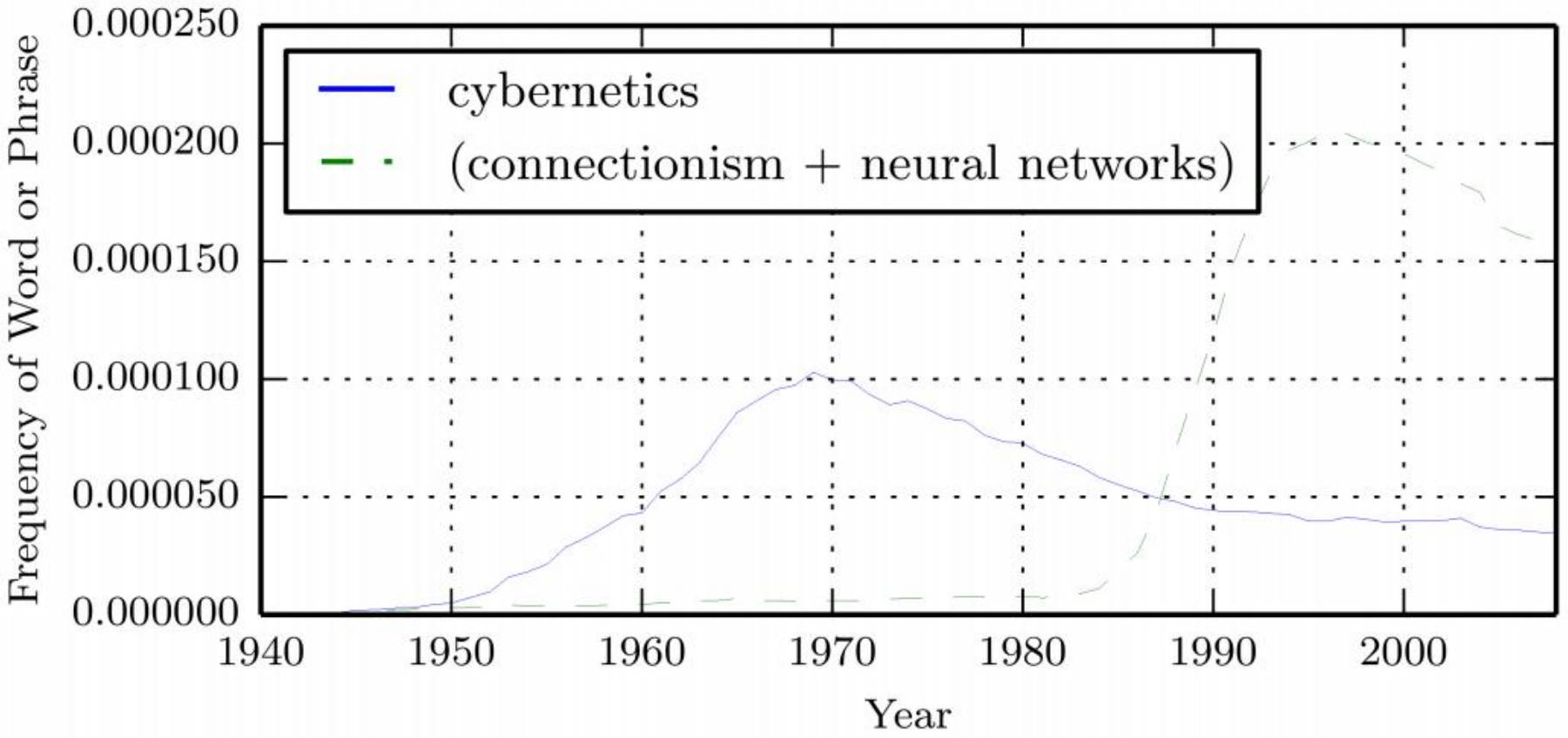


Figure 1.7

Historical Trends: Growing Datasets

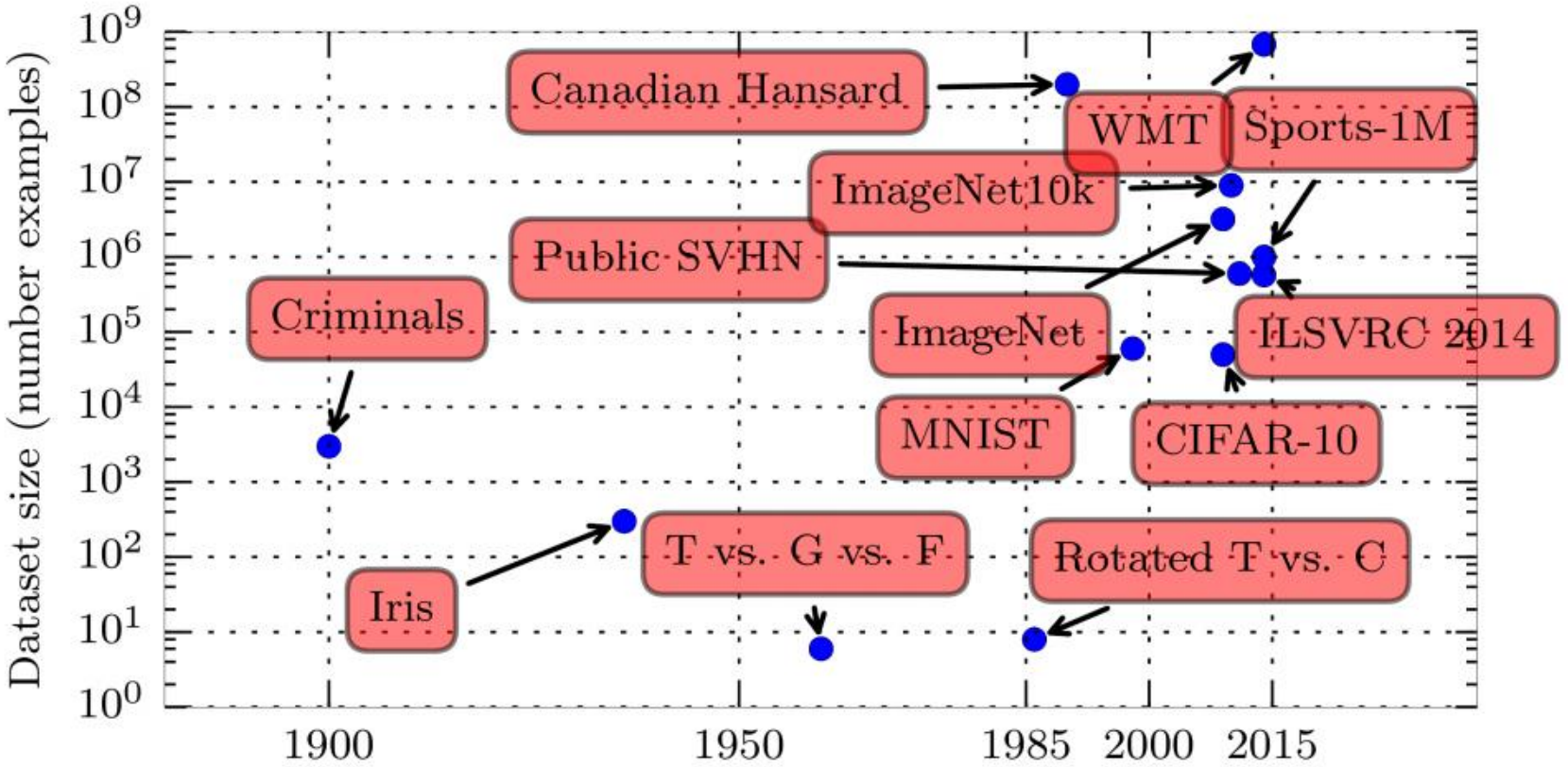


Figure 1.8

(Goodfellow 2016)

Connections per Neuron

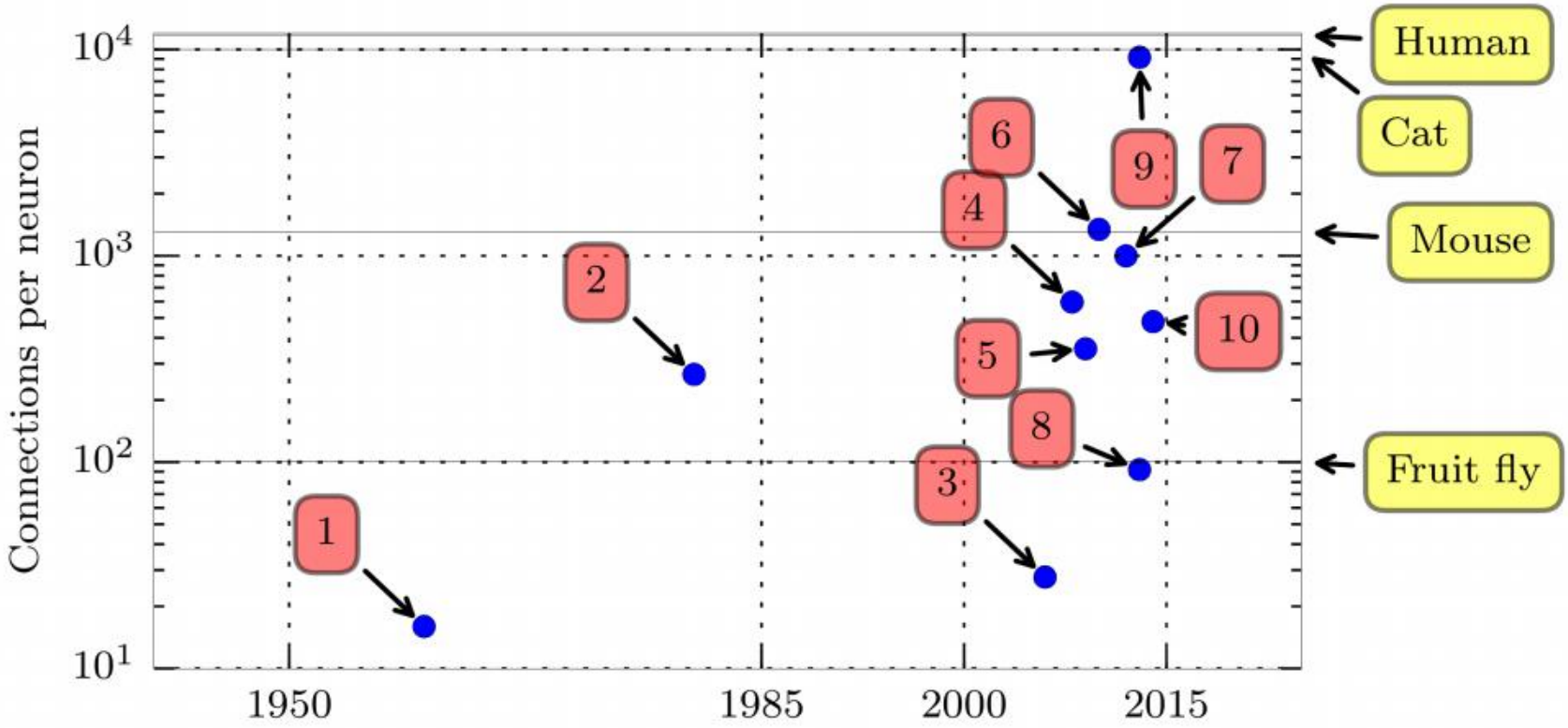


Figure 1.10

Number of Neurons

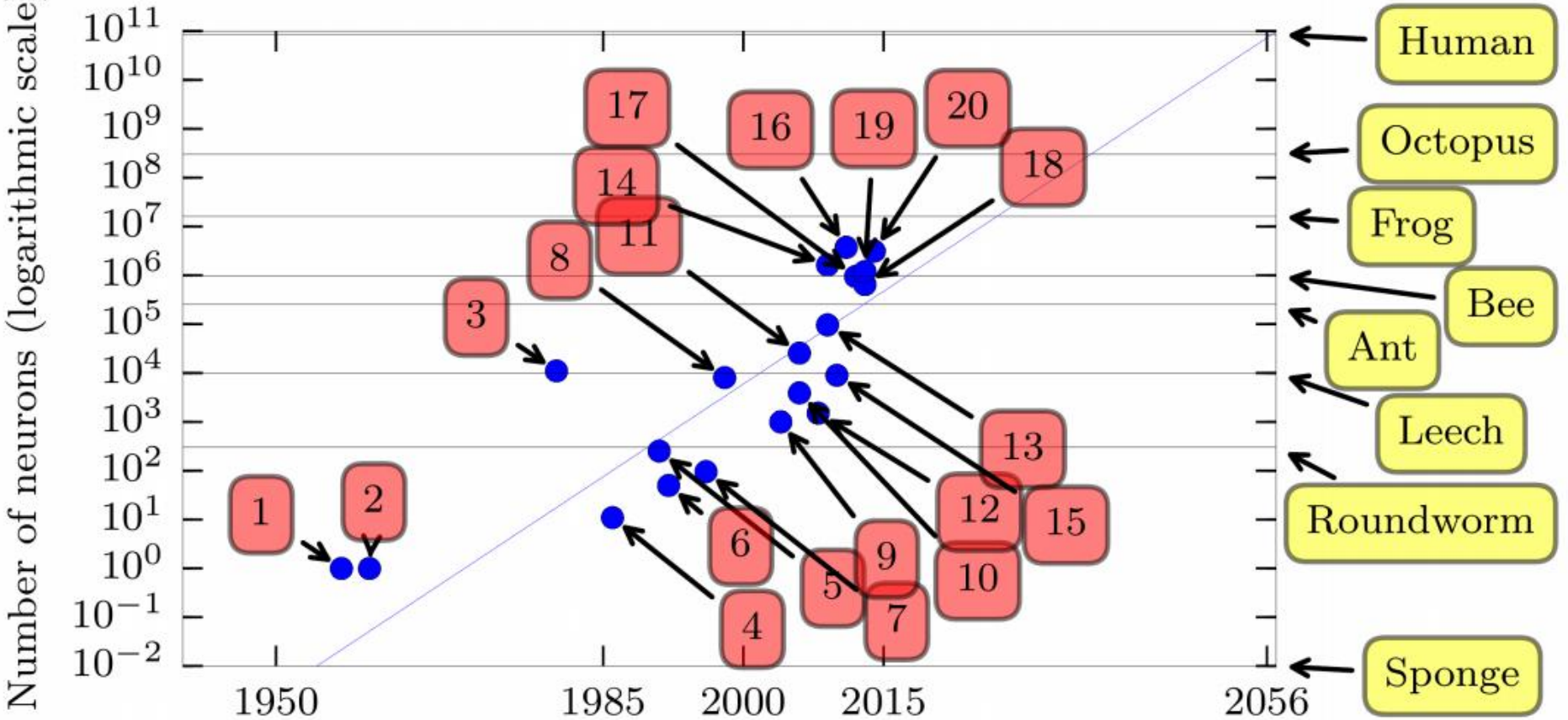


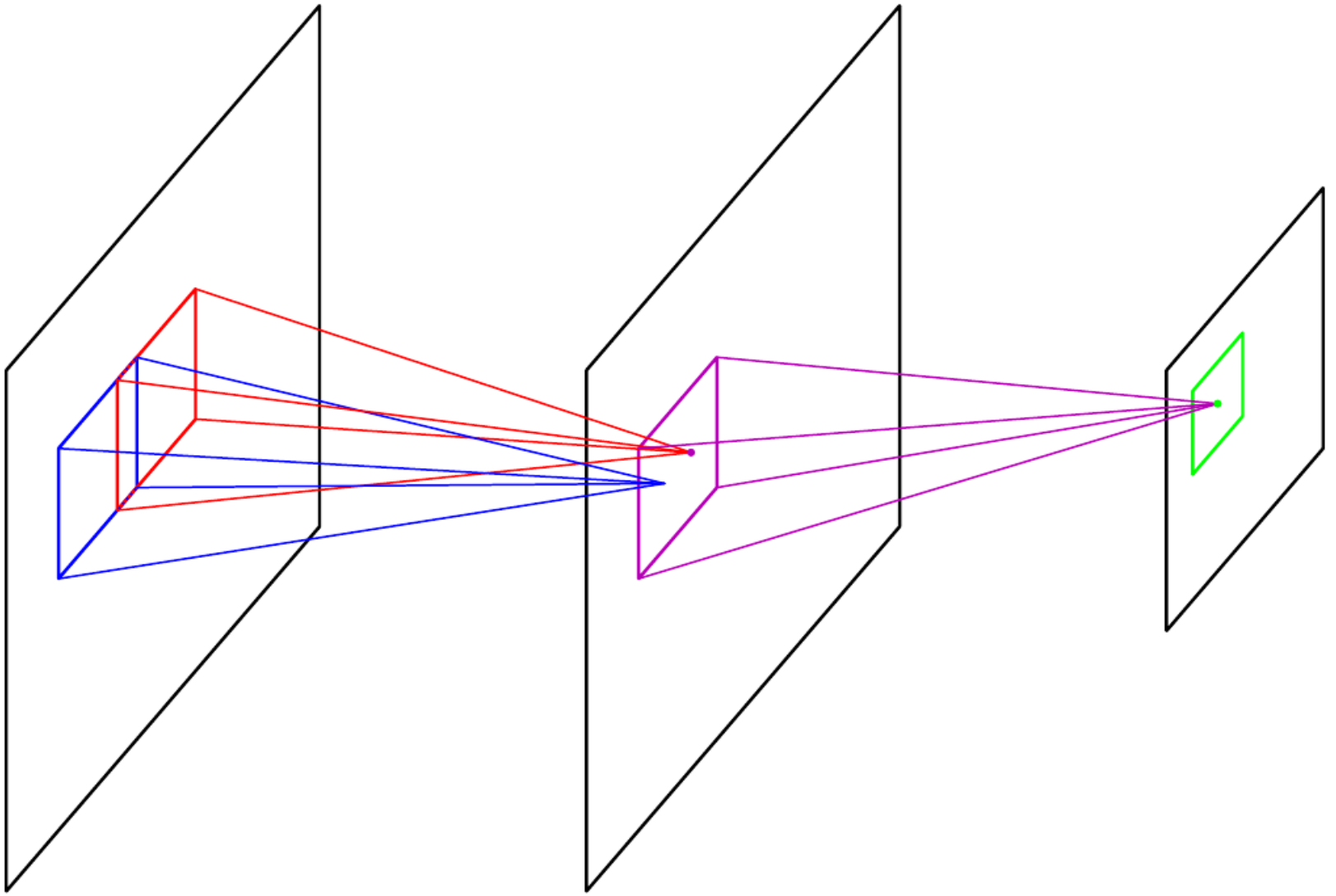
Figure 1.11

(Goodfellow 2016)

Convolutional Neural Network

1. Introduction

Convolutional Neural Layers

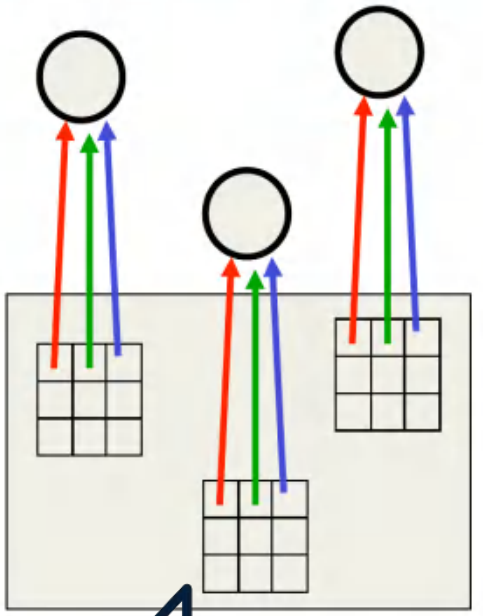


مزیت ویژگیهای تکرارشونده در تصاویر

The replicated feature approach
(currently the dominant approach for neural networks)

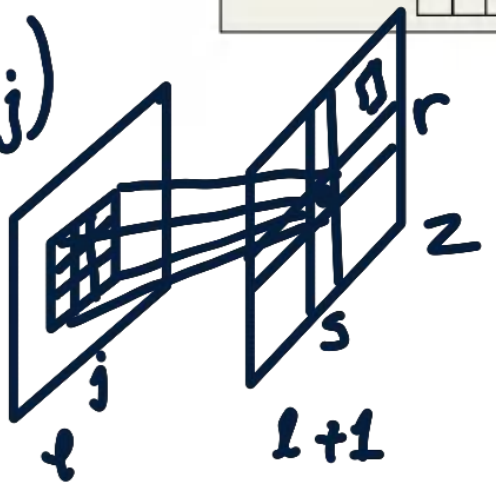
- Use many different copies of the same feature detector with different positions.
  - Could also replicate across scale and orientation (tricky and expensive)
  - Replication greatly reduces the number of free parameters to be learned.
- Use several different feature types, each with its own map of replicated detectors.
  - Allows each patch of image to be represented in several ways.

The red connections all have the same weight.



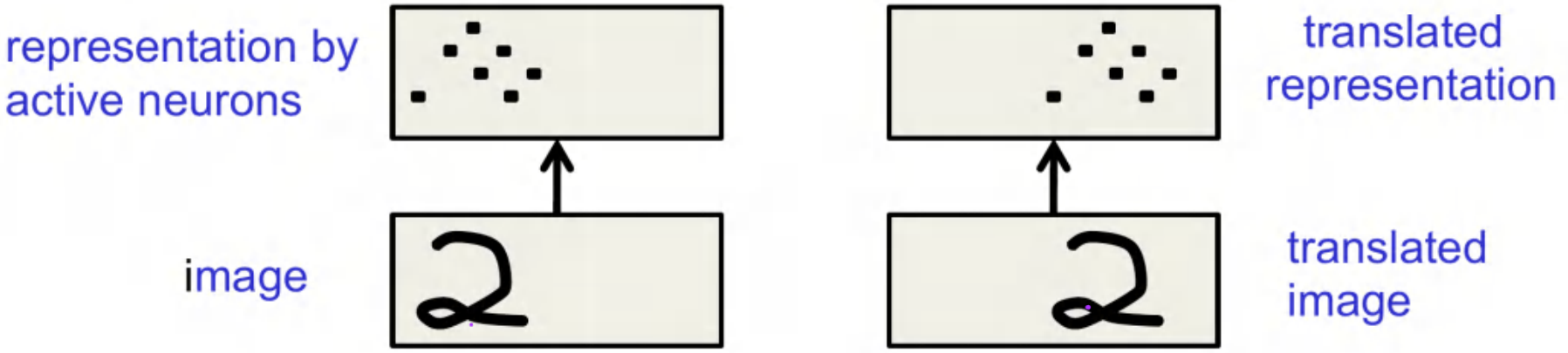
$$z_{r,s} = f\left(\sum_{\substack{r-k\le i\le r+k \\ s-k\le j\le s+k}} w_{i-r,j-s}\, x_{i,j}\right)$$

عرض کرنل $(2k+1)\times(2k+1)$



What does replicating the feature detectors achieve?

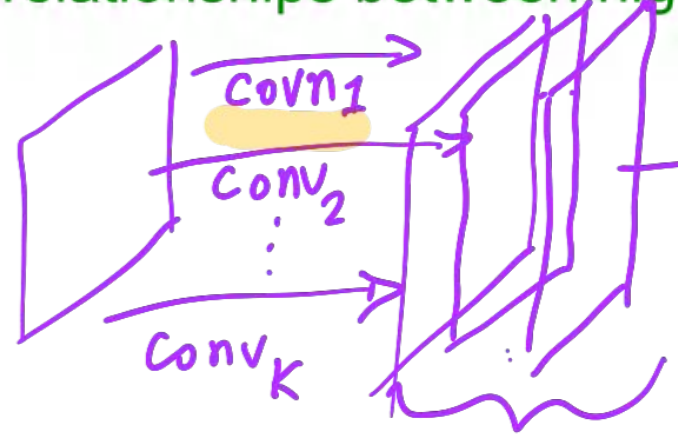
- Equivariant activities: Replicated features do not make the neural activities invariant to translation. The activities are equivariant.



- Invariant knowledge: If a feature is useful in some locations during training, detectors for that feature will be available in all locations during testing.

Pooling the outputs of replicated feature detectors

- Get a small amount of translational invariance at each level by averaging four neighboring replicated detectors to give a single output to the next level.
  - This reduces the number of inputs to the next layer of feature extraction, thus allowing us to have many more different feature maps.
  - Taking the maximum of the four works slightly better.
- Problem: After several levels of pooling, we have lost information about the precise positions of things.
  - This makes it impossible to use the precise spatial relationships between high-level parts for recognition.



The MNIST Dataset

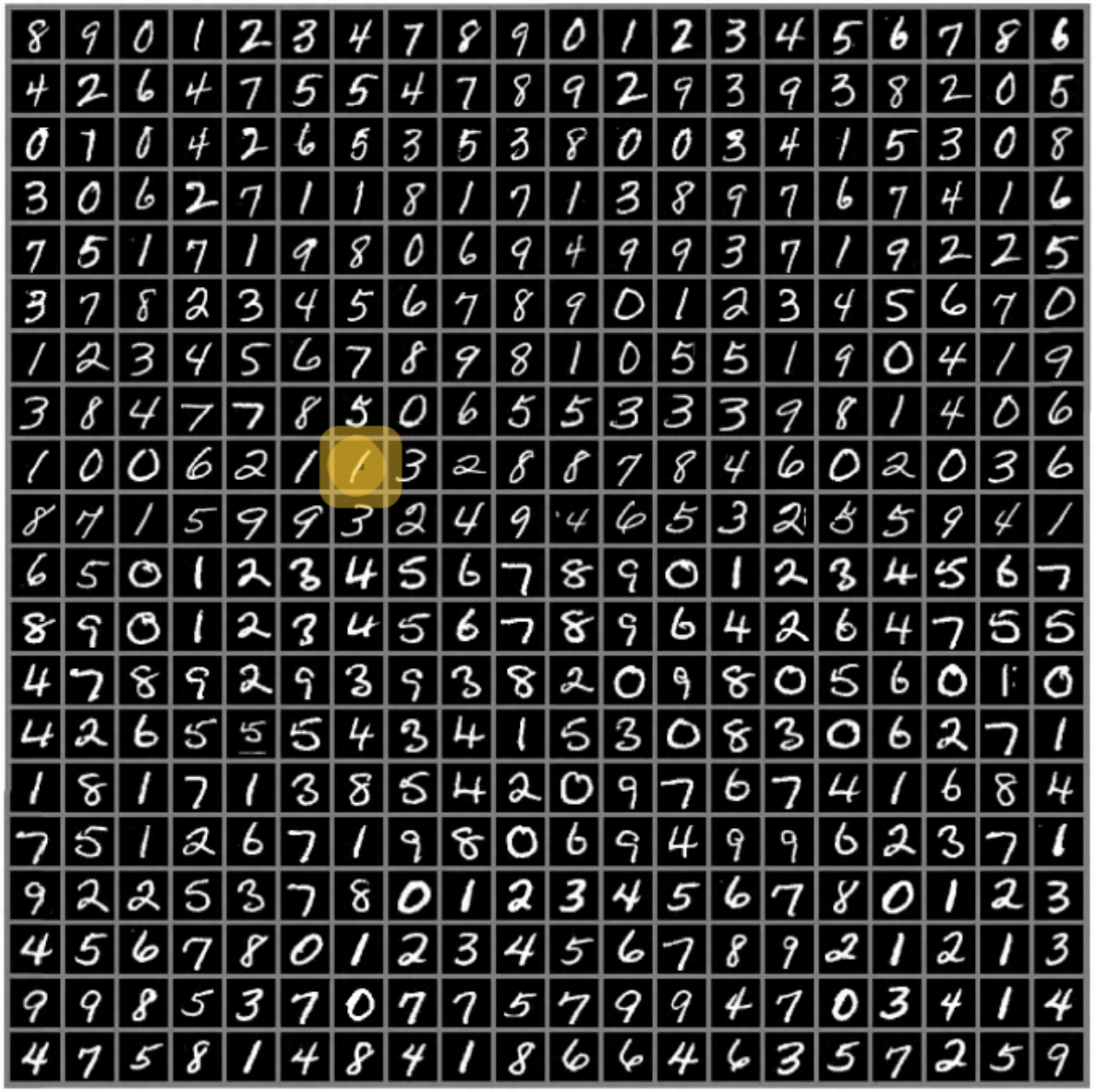
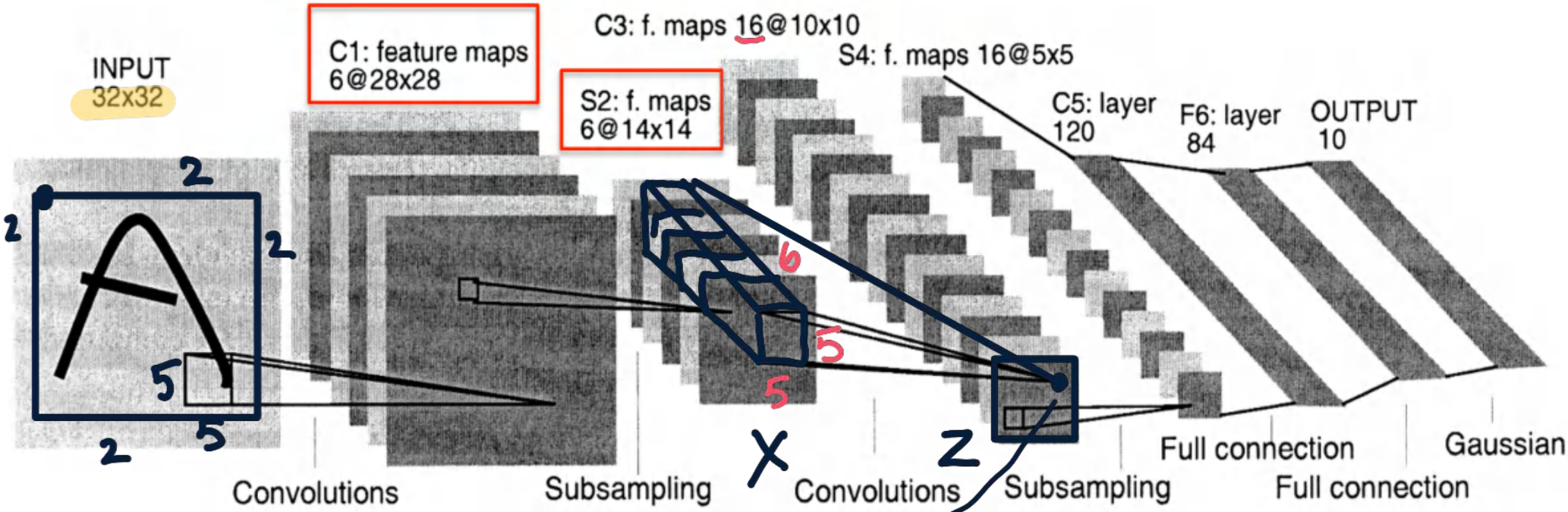


Figure 1.9

Le Net

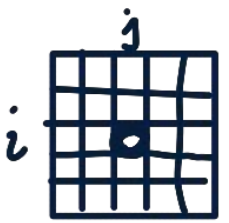
- Yann LeCun and his collaborators developed a really good recognizer for handwritten digits by using backpropagation in a feedforward net with:
  - Many hidden layers
  - Many maps of replicated units in each layer.
  - Pooling of the outputs of nearby replicated units.
  - A wide net that can cope with several characters at once even if they overlap.
  - A clever way of training a complete system, not just a recognizer.
- This net was used for reading ~10% of the checks in North America.
- Look the impressive demos of LENET at http://yann.lecun.com

The architecture of LeNet5



for each $c_{out} = 1 \cdots 16$ do $Z_{ij}^{c_{out}} = f\left(\sum_{k,l,c_{in}} W_{k,l}^{c_{in},c_{out}} X_{i+k-3,j+l-3}^{c_{in}} \right)$

k,l,c_{in}
$k=1,\cdots,5$
$l=1,\cdots,5$
$c_{in}=1,\cdots,6$



The 82 errors made by LeNet5

Notice that most of the errors are cases that people find quite easy.

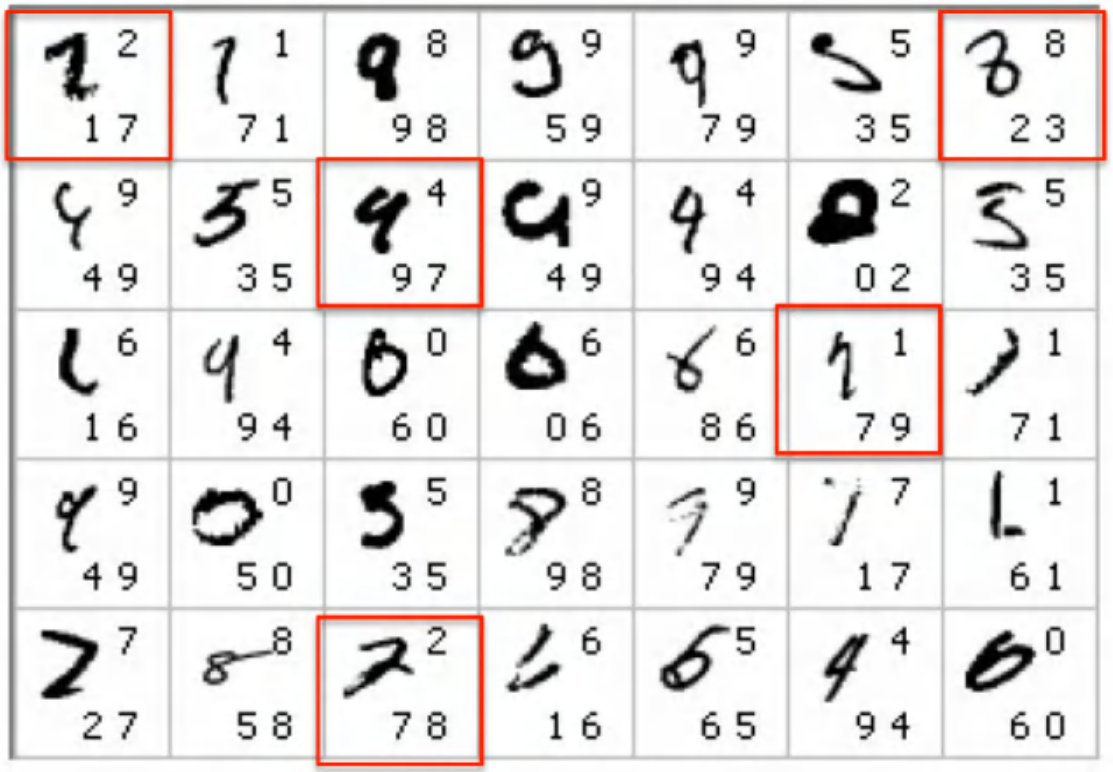
The human error rate is probably 20 to 30 errors but nobody has had the patience to measure it.

The brute force approach

- LeNet uses knowledge about the invariances to design:
  - the local connectivity
  - the weight-sharing
  - the pooling.
- This achieves about 80 errors.
  - This can be reduced to about 40 errors by using many different transformations of the input and other tricks (Ranzato 2008)

- Ciresan *et. al.* (2010) inject knowledge of invariances by creating a huge amount of carefully designed extra training data:
  - For each training image, they produce many new training examples by applying many different transformations.
  - They can then train a large, deep, dumb net on a GPU without much overfitting.
- They achieve about 35 errors.

The errors made by the Ciresan *et. al.* net



The top printed digit is the right answer. The bottom two printed digits are the network's best two guesses.

The right answer is almost always in the top 2 guesses.

With model averaging they can now get about 25 errors.

How to detect a significant drop in the error rate

- Is 30 errors in 10,000 test cases significantly better than 40 errors?
  - It all depends on the particular errors!
  - The McNemar test uses the particular errors and can be much more powerful than a test that just uses the number of errors.

	model 1 wrong	model 1 right
model 2 wrong	29	1
model 2 right	11	9959

	model 1 wrong	model 1 right
model 2 wrong	15	15
model 2 right	25	9945

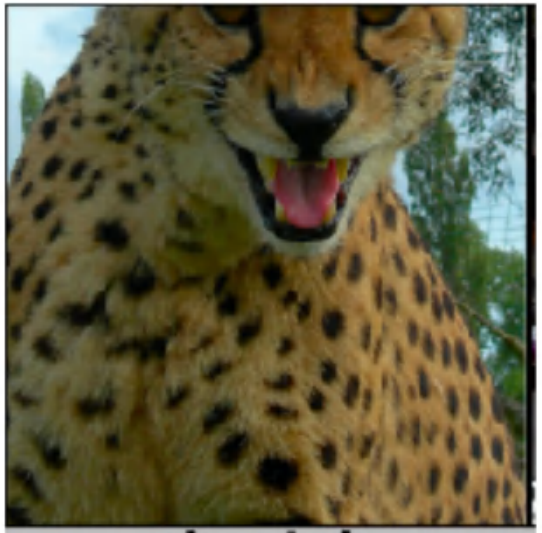
From hand-written digits to 3-D objects

- Recognizing real objects in color photographs downloaded from the web is much more complicated than recognizing hand-written digits:
  - Hundred times as many classes (1000 vs 10)
  - Hundred times as many pixels (256 x 256 color vs 28 x 28 gray)
  - Two dimensional image of three-dimensional scene.
  - Cluttered scenes requiring segmentation
  - Multiple objects in each image.
- Will the same type of convolutional neural network work?

The ILSVRC-2012 competition on ImageNet

- The dataset has 1.2 million high-resolution training images.
- The classification task:
  - Get the "correct" class in your top 5 bets. There are 1000 classes.
- The localization task:
  - For each bet, put a box around the object. Your box must have at least 50% overlap with the correct box.
- Some of the best existing computer vision methods were tried on this dataset by leading computer vision groups from Oxford, INRIA, XRCE, …
  - Computer vision systems use complicated multi-stage systems.
  - The early stages are typically hand-tuned by optimizing a few parameters.

Examples from the test set (with the network's guesses)



cheetah

cheetah
leopard
snow leopard
Egyptian cat

bullet train

bullet train
passenger car
subway train
electric locomotive

hand glass

scissors
hand glass
frying pan
stethoscope

- University of Toronto (Alex Krizhevsky) — 16.4% — 34.1%

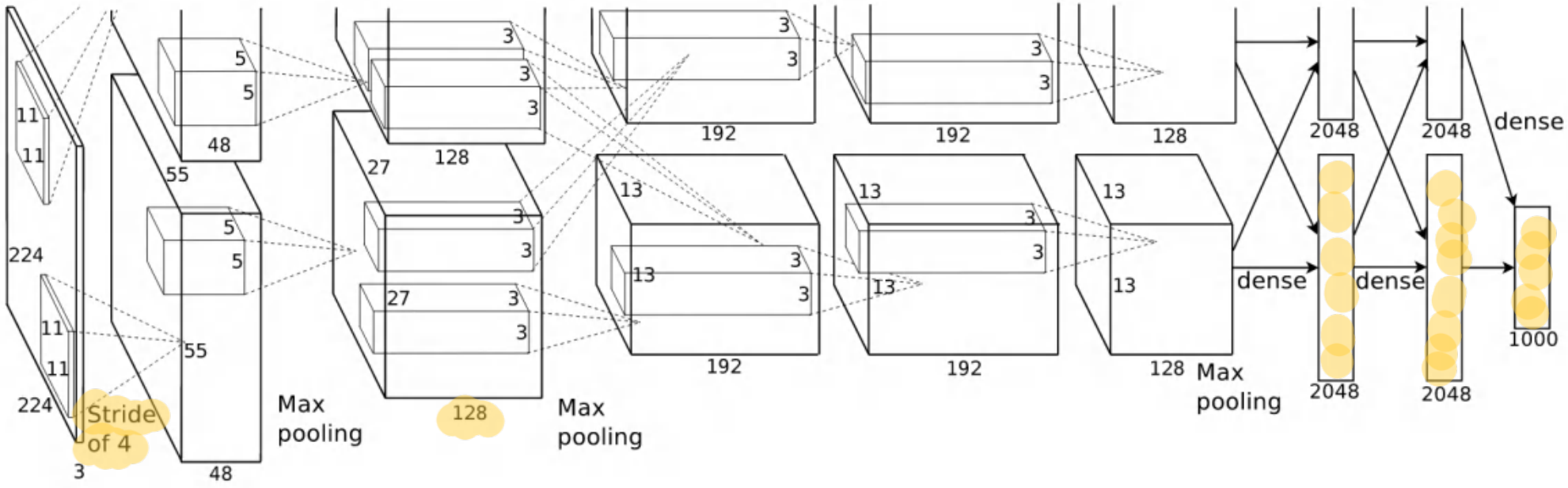
Error rates on the ILSVRC-2012 competition

	classification	classification &localization
• University of Tokyo	• 26.1%	53.6%
• Oxford University Computer Vision Group	• 26.9%	50.0%
• INRIA (French national research institute in CS) + XRCE (Xerox Research Center Europe)	• 27.0%	
• University of Amsterdam	• 29.5%	

A neural network for ImageNet

- Alex Krizhevsky (NIPS 2012) developed a very deep convolutional neural net of the type pioneered by Yann Le Cun. Its architecture was:
  - 7 hidden layers not counting some max pooling layers.
  - The early layers were convolutional.
  - The last two layers were globally connected.
- The activation functions were:
  - Rectified linear units in every hidden layer. These train much faster and are more expressive than logistic units.
  - Competitive normalization to suppress hidden activities when nearby units have stronger activities. This helps with variations in intensity.

AlexNet Structure



Tricks that significantly improve generalization

- Train on random 224x224 patches from the 256x256 images to get more data. Also use left-right reflections of the images.
  - At test time, combine the opinions from ten different patches: The four 224x224 corner patches plus the central 224x224 patch plus the reflections of those five patches.
- Use "dropout" to regularize the weights in the globally connected layers (which contain most of the parameters).
  - Dropout means that half of the hidden units in a layer are randomly removed for each training example.
  - This stops hidden units from relying too much on other hidden units.

Depth: Repeated Composition

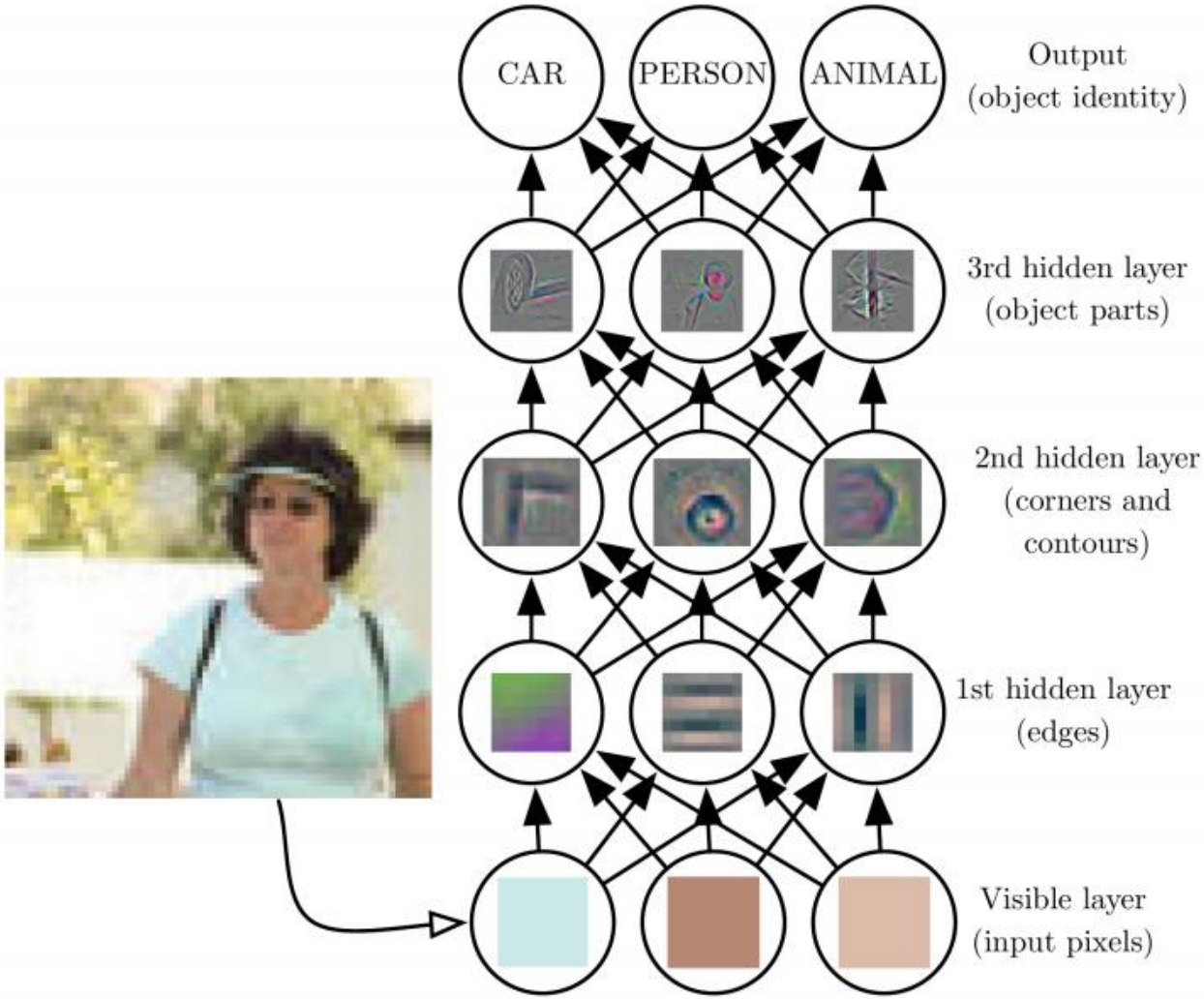


Figure 1.2

Better Generalization with Greater Depth

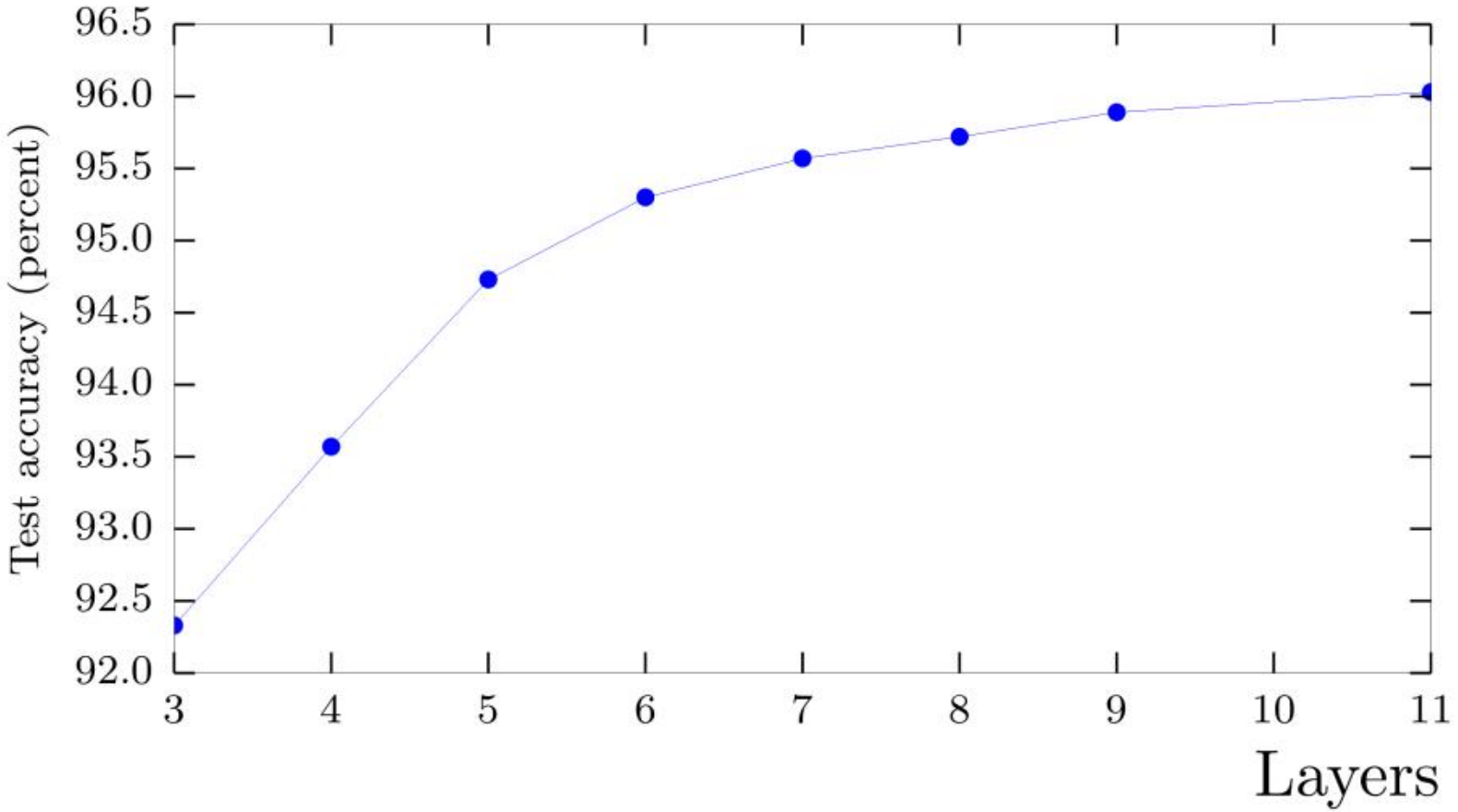


Figure 6.6