ELEN 4720: Machine Learning for Signals, Information and Data

Lecture 14, 3/25/2020

Prof. John Paisley

Department of Electrical Engineering
& Data Science Institute

Columbia University

Neural Networks

Many figures borrowed from: Martin Gorner, Andrew Ng, Ruslan Salakhutdinov, Louis Serrano

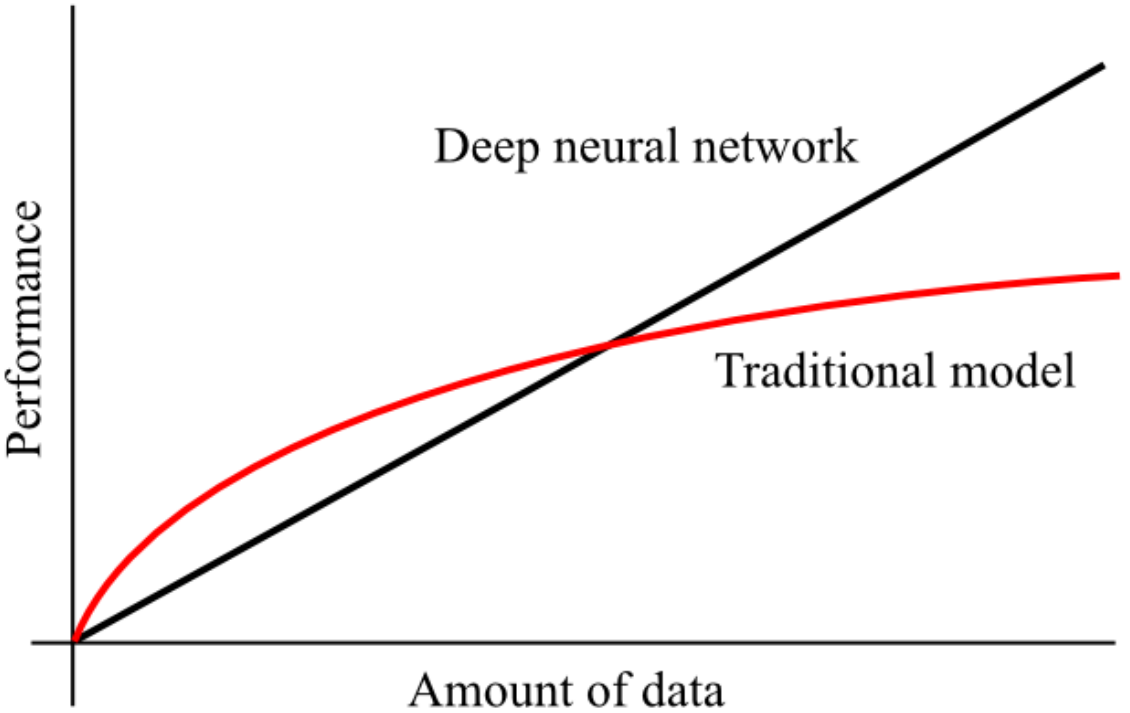
Deep learning: why now?

- Neural networks have been around since the 1980s.
- Scalability in two directions: computation and data
- Complex models learn tasks better, however...

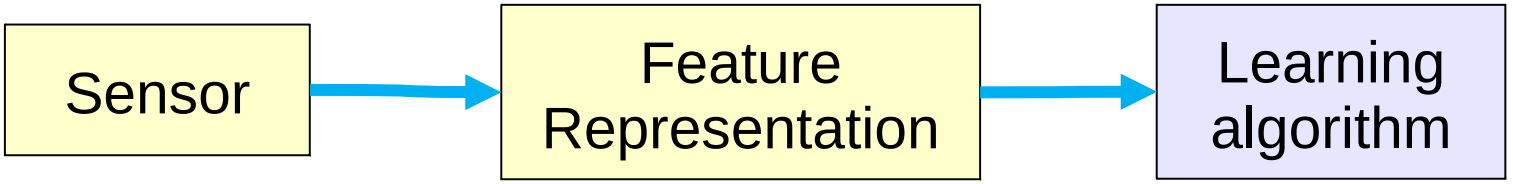
✓ Need more data

✓ Need faster computation

✓ Need expressive models and inference tricks



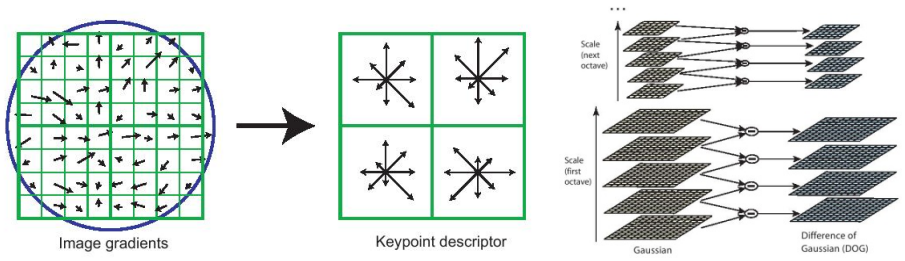
Traditional machine learning methodology



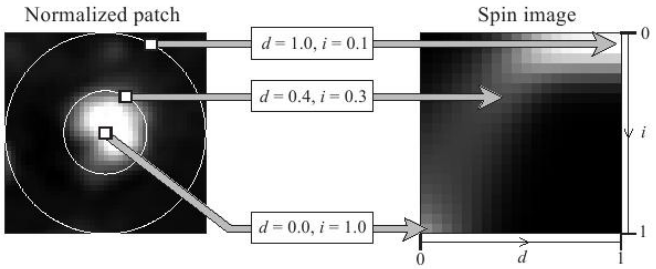
Problems of hand-tuned features

1) Needs expert knowledge
2) Time-consuming and expensive
3) Does not generalize to other domains

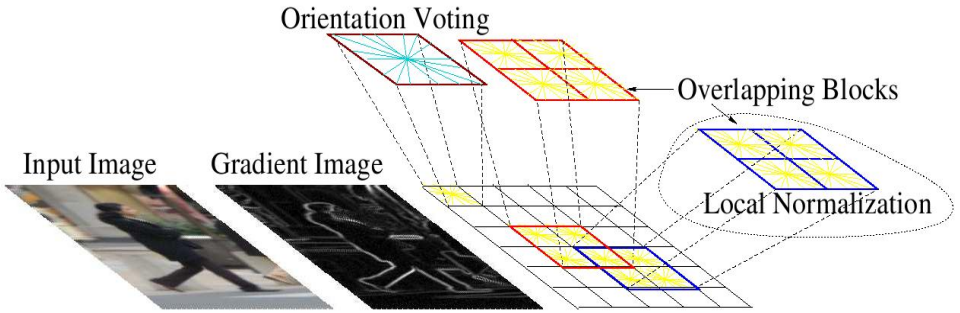
Some computer vision features



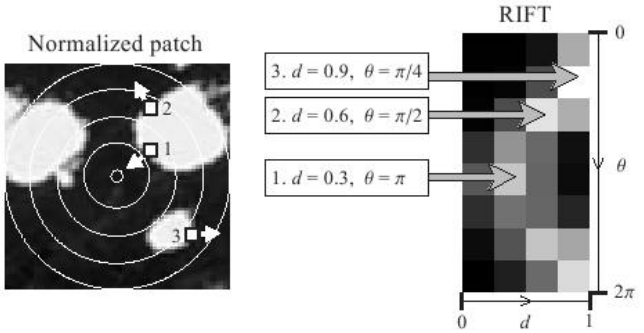
SIFT



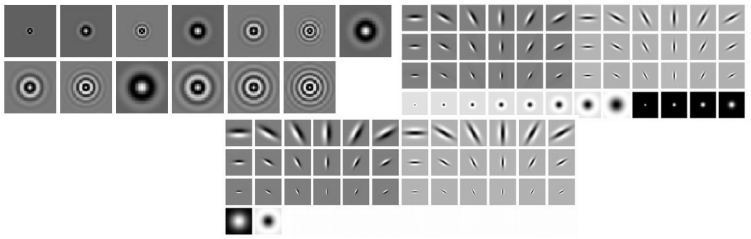
Spin image



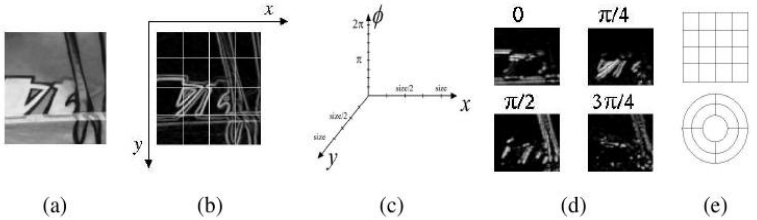
HoG



RIFT

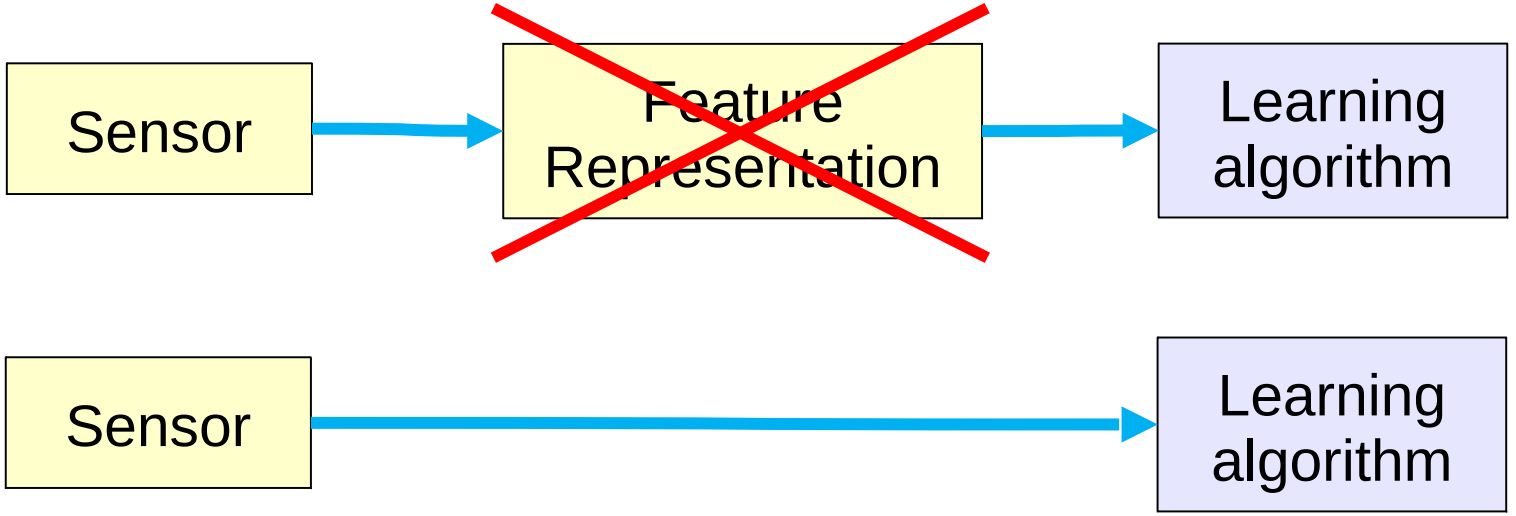


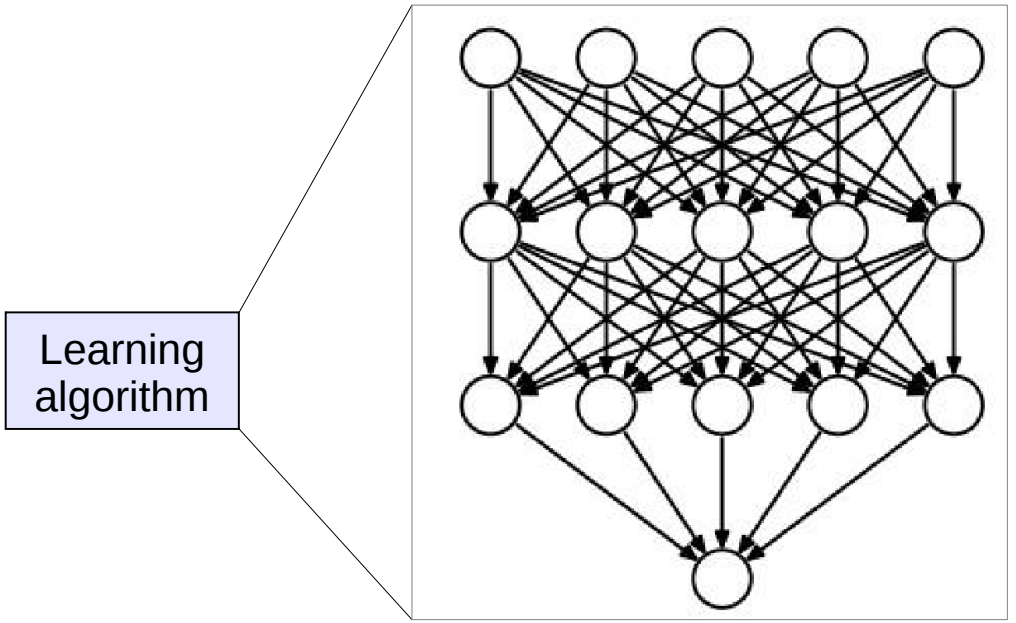
Textons

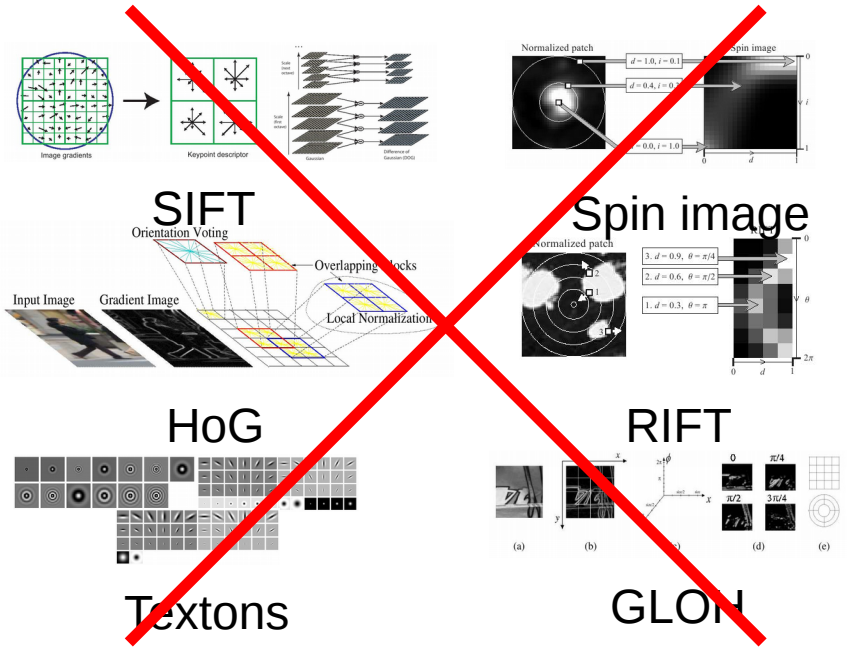


GLOH

Automatic representation learning







What can deep learning do?

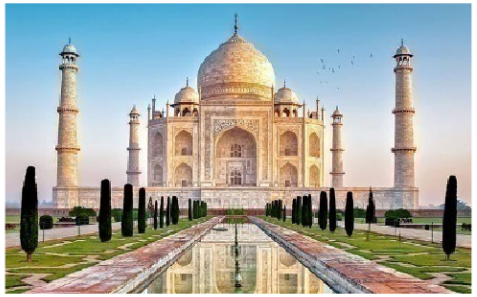
- **Short answer**: Nothing that other algorithms can't do already, but they often have much better performance.

- **Longer answer**: Almost any machine learning problem can use a neural network in some way.

  - Supervised: Predict output corresponding to input.
    - Deep learning excels here. I will focus on this problem.

  - Unsupervised: Learn structure from data.
    - A newer research thrust. Many people focusing on this recently because there's still much room for improvement.

Traditional ML problem: Tag an image



mosque, tower, building, cathedral, dome, castle

ski, skiing, skiers, skiiers, snowmobile

kitchen, stove, oven, refrigerator, microwave

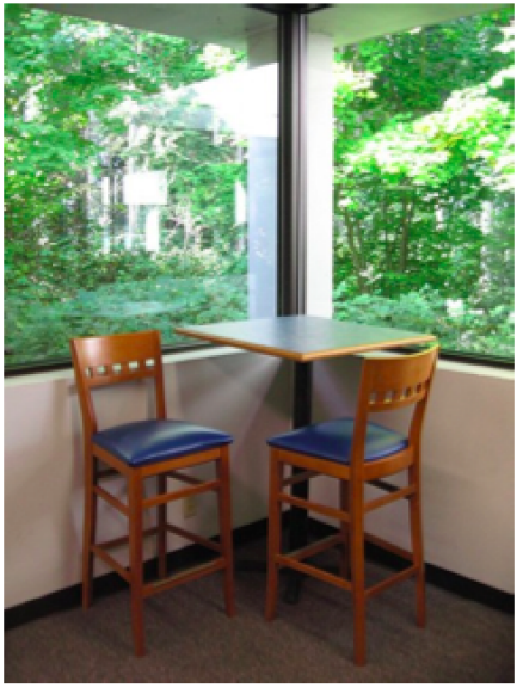
bowl, cup, soup, cups, coffee

beach

snow

Modern ML problem: Create a unique caption

a car is parked in
the middle of nowhere .



a wooden table and chairs
arranged in a room .

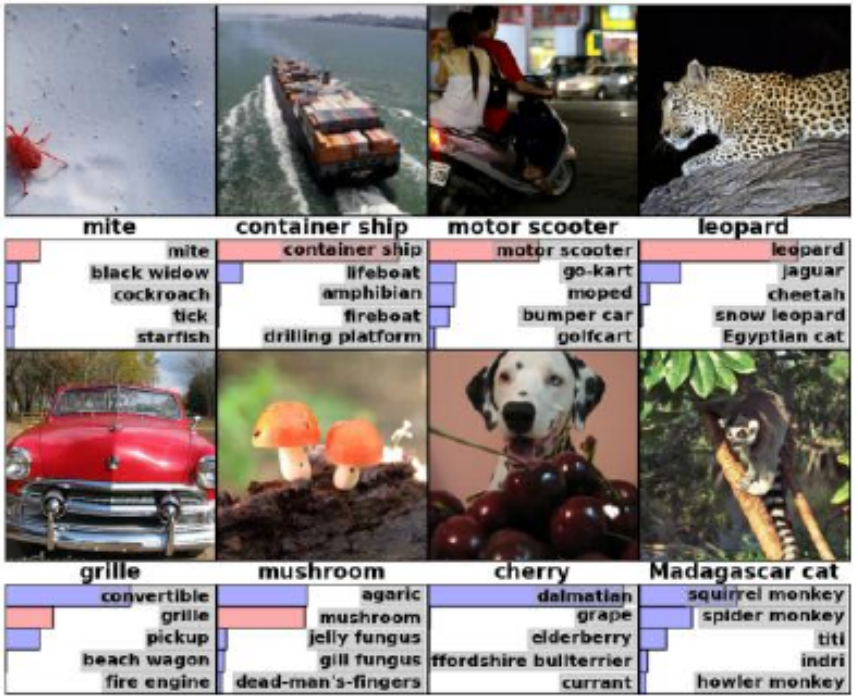
there is a cat sitting on a shelf .

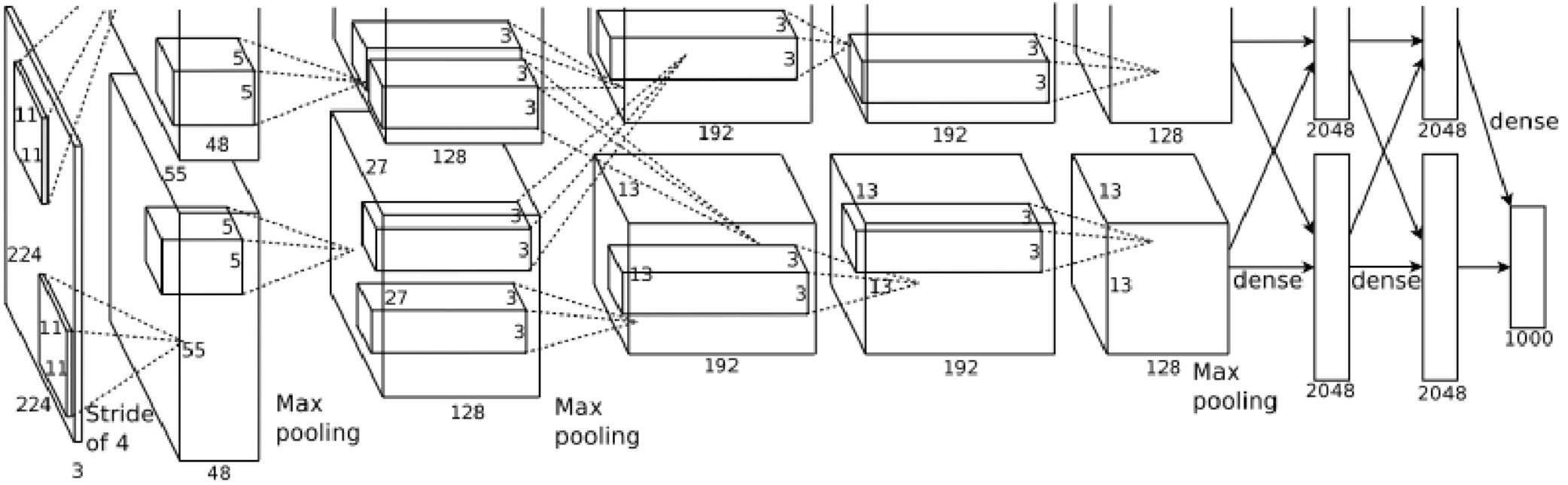
a ferry boat on a marina
with a group of people .

a little boy with a bunch
of friends on the street .

ImageNet

- A difficult classification challenge dataset
  - Training data: 1.2M images, 1K categories
  - Testing: label 150K new images
- "AlexNet" shocked with 15.3% error rate versus 26%, which was best at that time

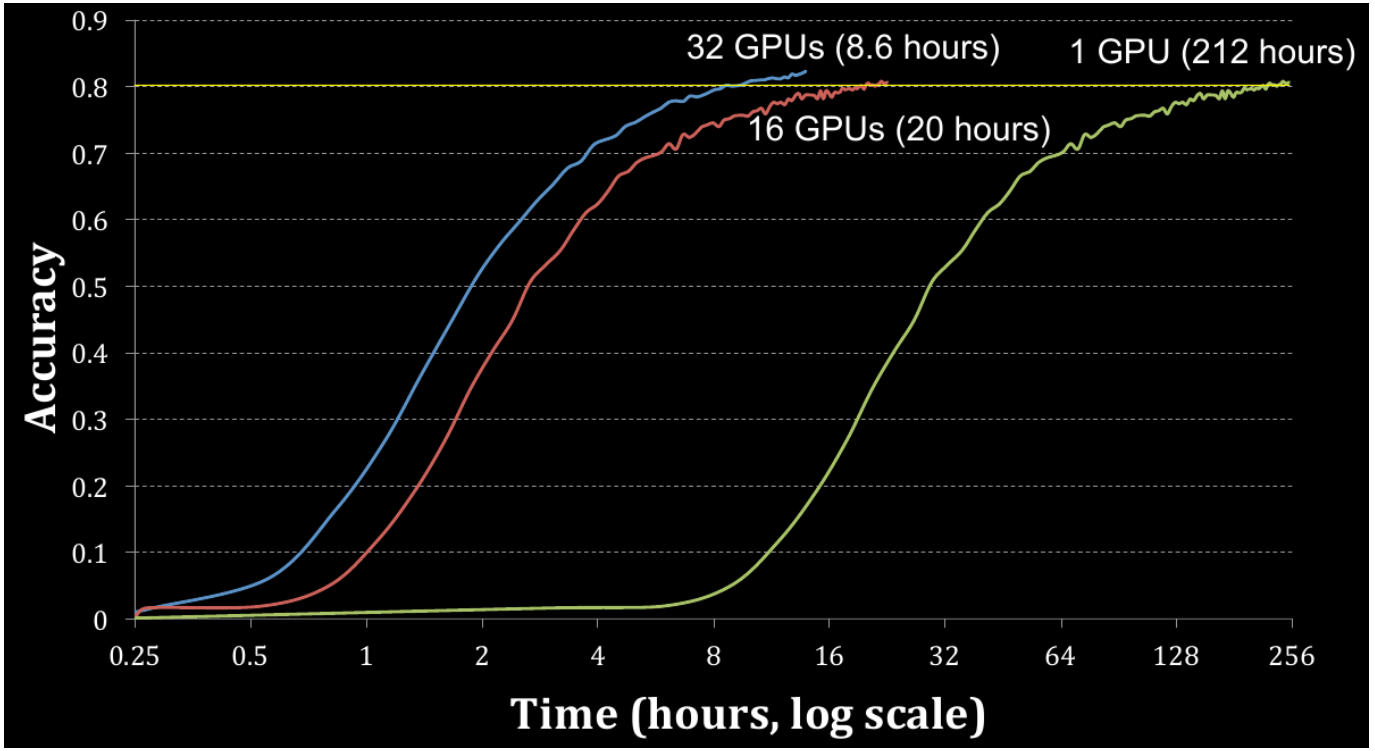




Alex Krizhevsky, Ilya Sutskever, Geoffrey Hinton, "ImageNet classification with deep convolutional neural networks," *Advances in Neural Information Processing Systems,* 2012.

Public software (a selection)

- TensorFlow (Google)
- Caffe (UC Berkeley)
- PyTorch (Facebook)
- Implementation: GPUs are important for scaling

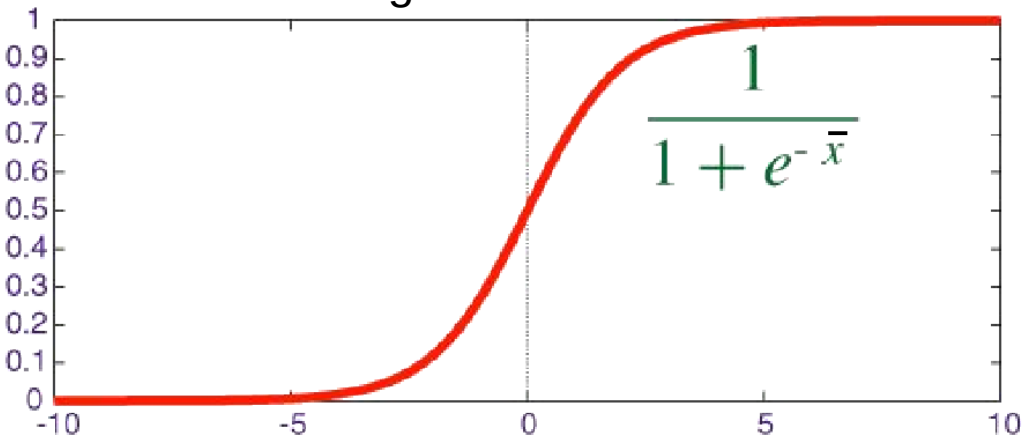


Building blocks of a deep learning model

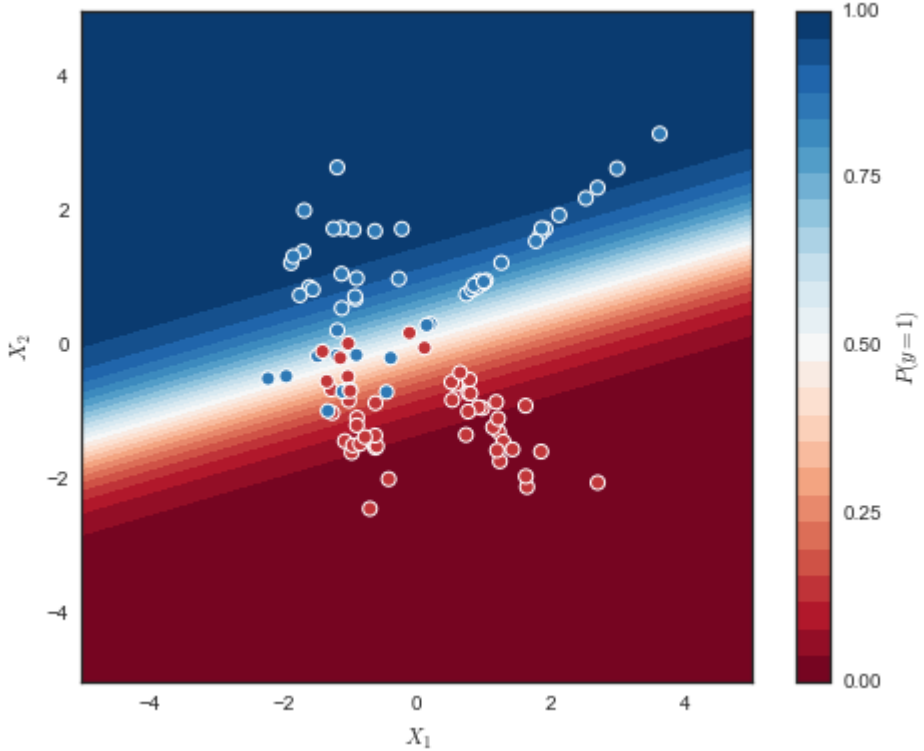
Part 1: Logistic regression (review)

- Logistic regression learns 2 classes
- The key is using the sigmoid function. This maps R → [0,1].
- A *dot-product* gives the value for sigmoid: $\bar{x} = w^T x + b$

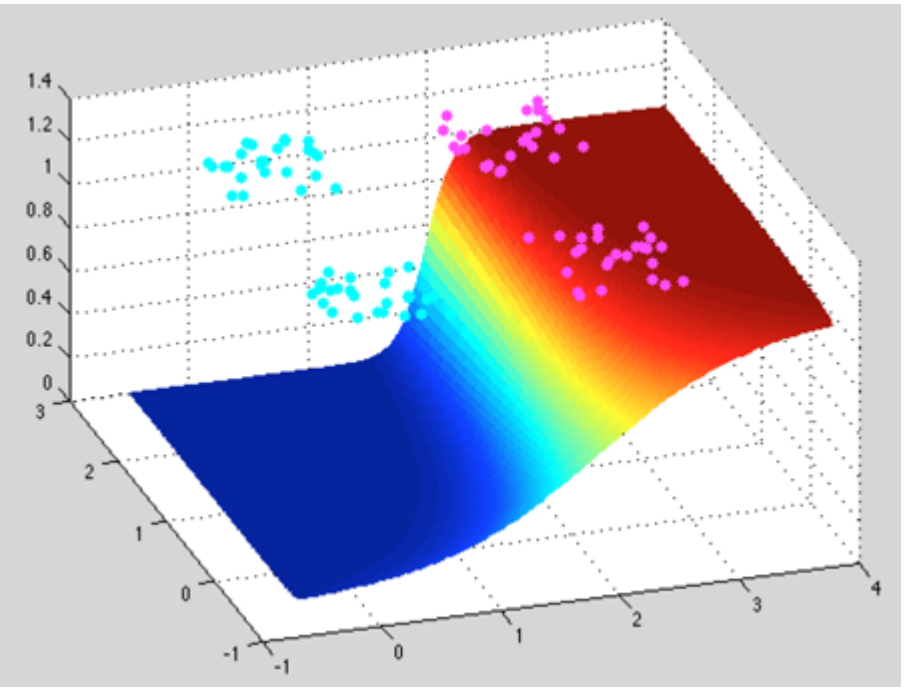
Sigmoid function



One view of decision boundary



A different view (and different data)



Building blocks of a deep learning model

Part 2: Multinomial logistic regression

- For multiple classes, we generalize this with the ***softmax*** function.
- Imagine our data is *p*-dimensional and we have *M* classes.

$$\mathrm{x} \in \mathbb{R}^p \quad \mathrm{y} \in \{1, \ldots, M\}$$

- We now have *M* vectors, organized in a *M* x *p* matrix, *W.* We add a class-specific bias vector *b* as well.

$$W\mathrm{x} + b$$

$$\Pr(\mathrm{y} = k | W, \mathrm{x}, b) = \sigma_k(W\mathrm{x} + b) = \frac{e^{(W\mathrm{x}+b)_k}}{\sum_{j=1}^{p} e^{(W\mathrm{x}+b)_j}}$$

- Our goal is to learn good values for matrix *W* and vector *b*. We're given *n* input-output pairs of (*y,x*) to do this.

- How do we measure “good”? Set up an ***objective function***. We will minimize the negative log-probability of the data.

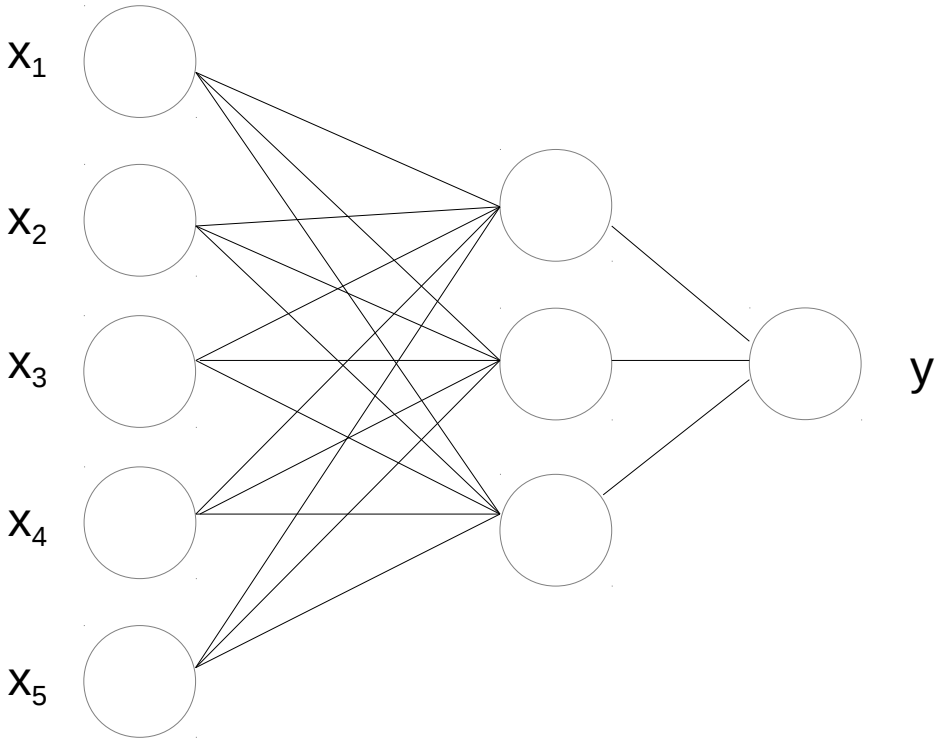
$$\mathcal{L} = -\sum_{i=1}^{n} \ln \Pr(\mathrm{y}_i | W, \mathrm{x}_i, b) = -\sum_{i=1}^{n} \sum_{k=1}^{M} \mathbb{1}(\mathrm{y}_i = k) \ln \sigma_k(W\mathrm{x}_i + b)$$

- Finally, this can be minimized using ***gradient methods***. Simply take derivatives and update parameters.

$$W \leftarrow W - \alpha \nabla_W \mathcal{L}, \qquad b \leftarrow b - \alpha \nabla_b \mathcal{L}$$

Multinomial logistic regression as a “neural network”

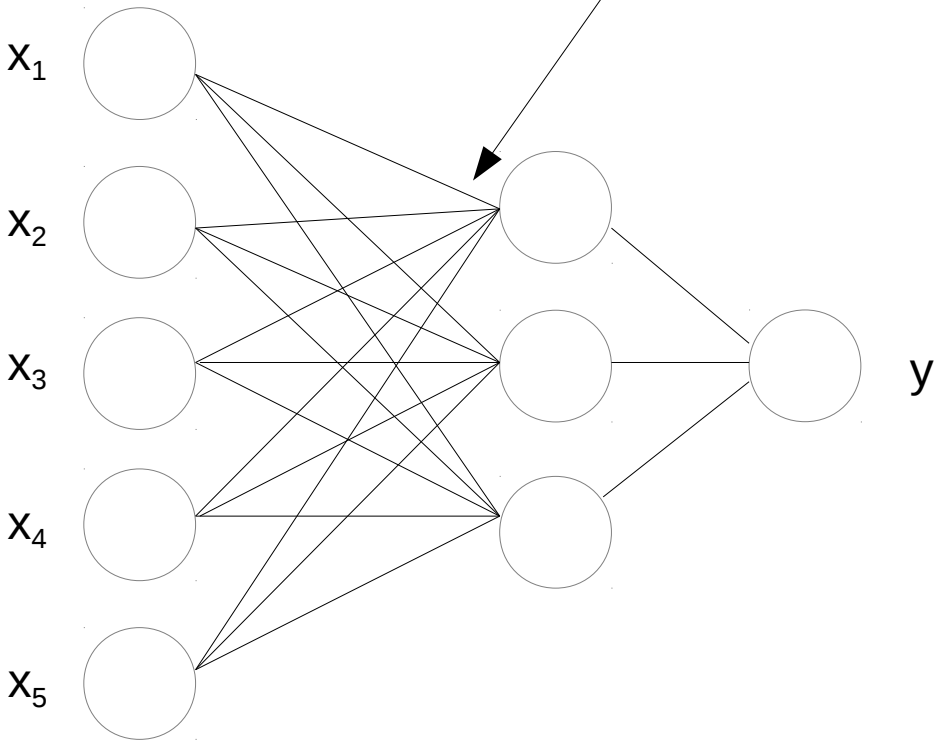
- Imagine our data is 5-dimensional and we have three classes.
- We can visualize this multinomial logistic regression problem graphically. This visualization is done to build NN intuitions.
- This picture is for ***<u>one</u>*** input-output pair. The network operates by trying to map all inputs in the data to their outputs.



Multinomial logistic regression as a “neural network”

$$\hat{y}_k = W_{k,1}x_1 + \ldots + W_{k,5}x_5 + b_k$$

(*k* = 1 here)

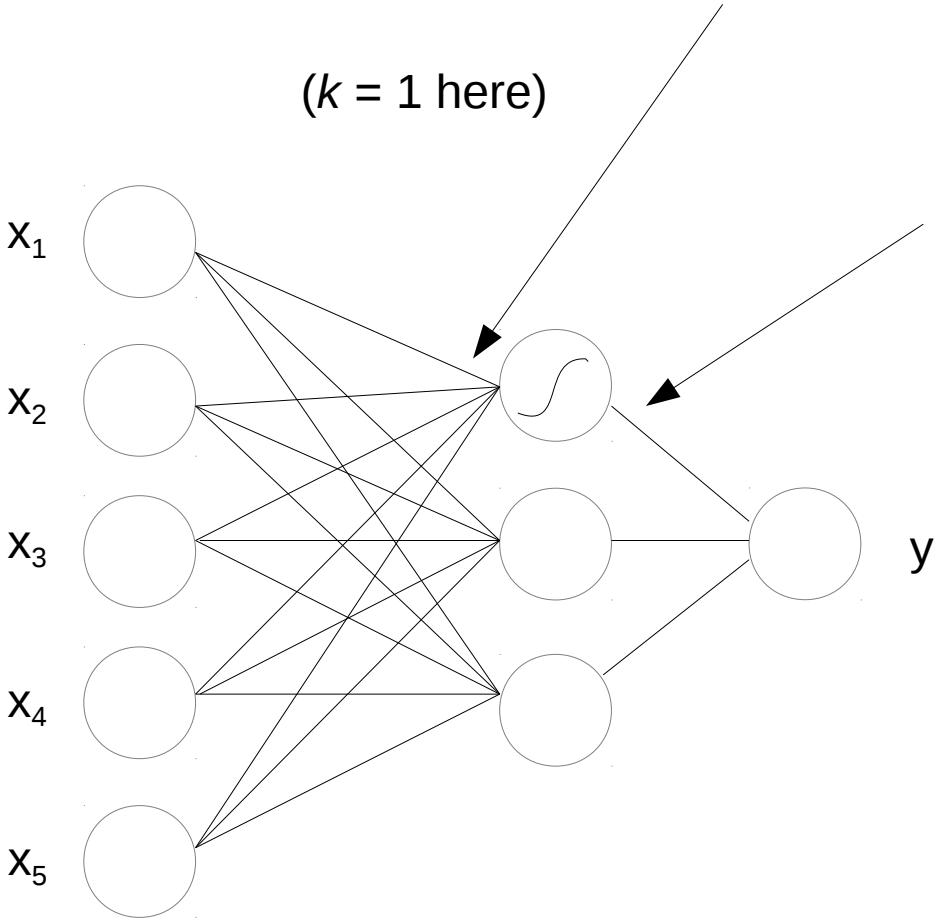


Multinomial logistic regression as a “neural network”

$$\hat{y}_k = W_{k,1}x_1 + \ldots + W_{k,5}x_5 + b_k$$

(*k* = 1 here)

$$\Pr(\mathrm{y} = k|\hat{y}_k) = \sigma_k(\hat{y}_k)$$

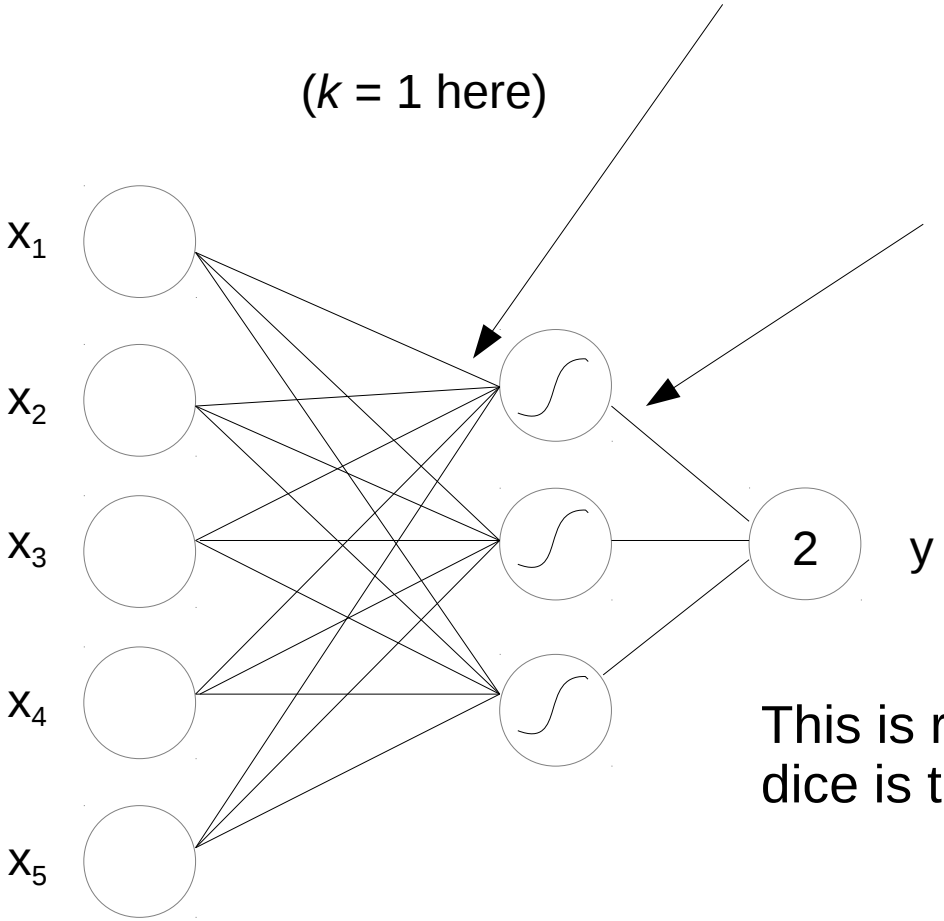


Multinomial logistic regression as a “neural network”

$$\hat{y}_k = W_{k,1}x_1 + \ldots + W_{k,5}x_5 + b_k$$

(*k* = 1 here)

$$\Pr(\mathrm{y} = k|\hat{y}_k) = \sigma_k(\hat{y}_k)$$

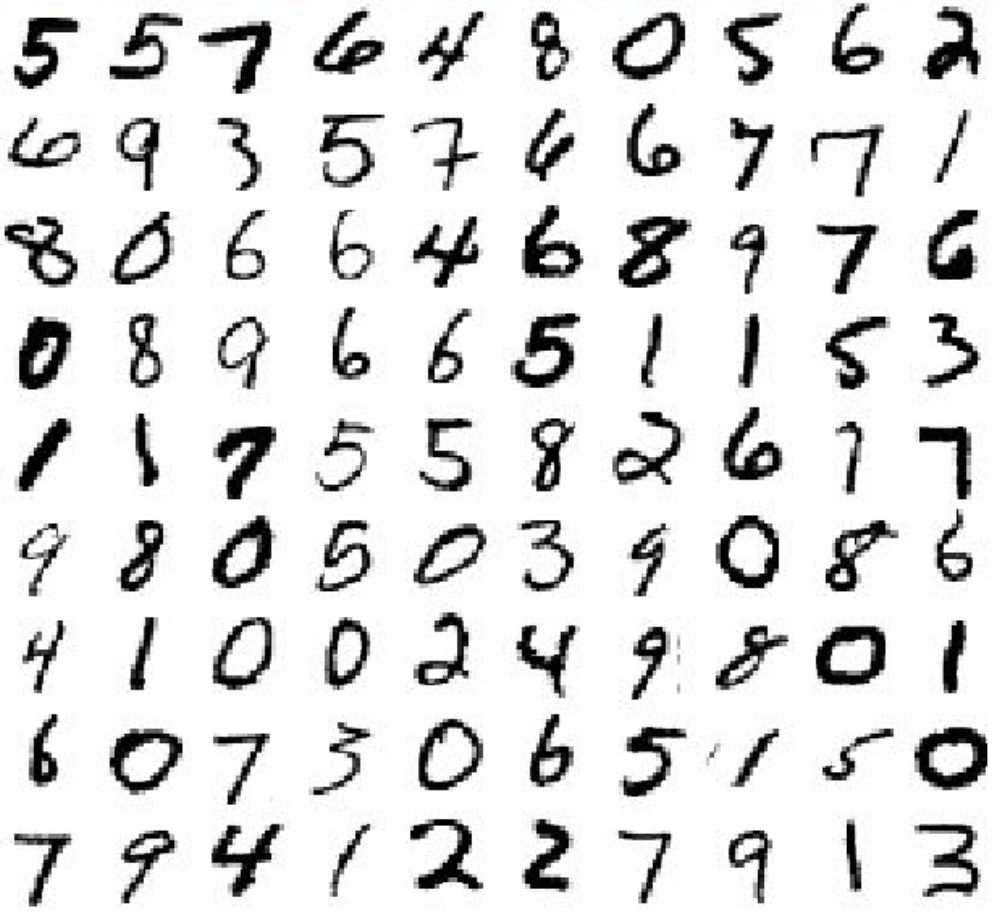


This is repeated for all classes. Then a 3-sided dice is thrown to decide what y equals.

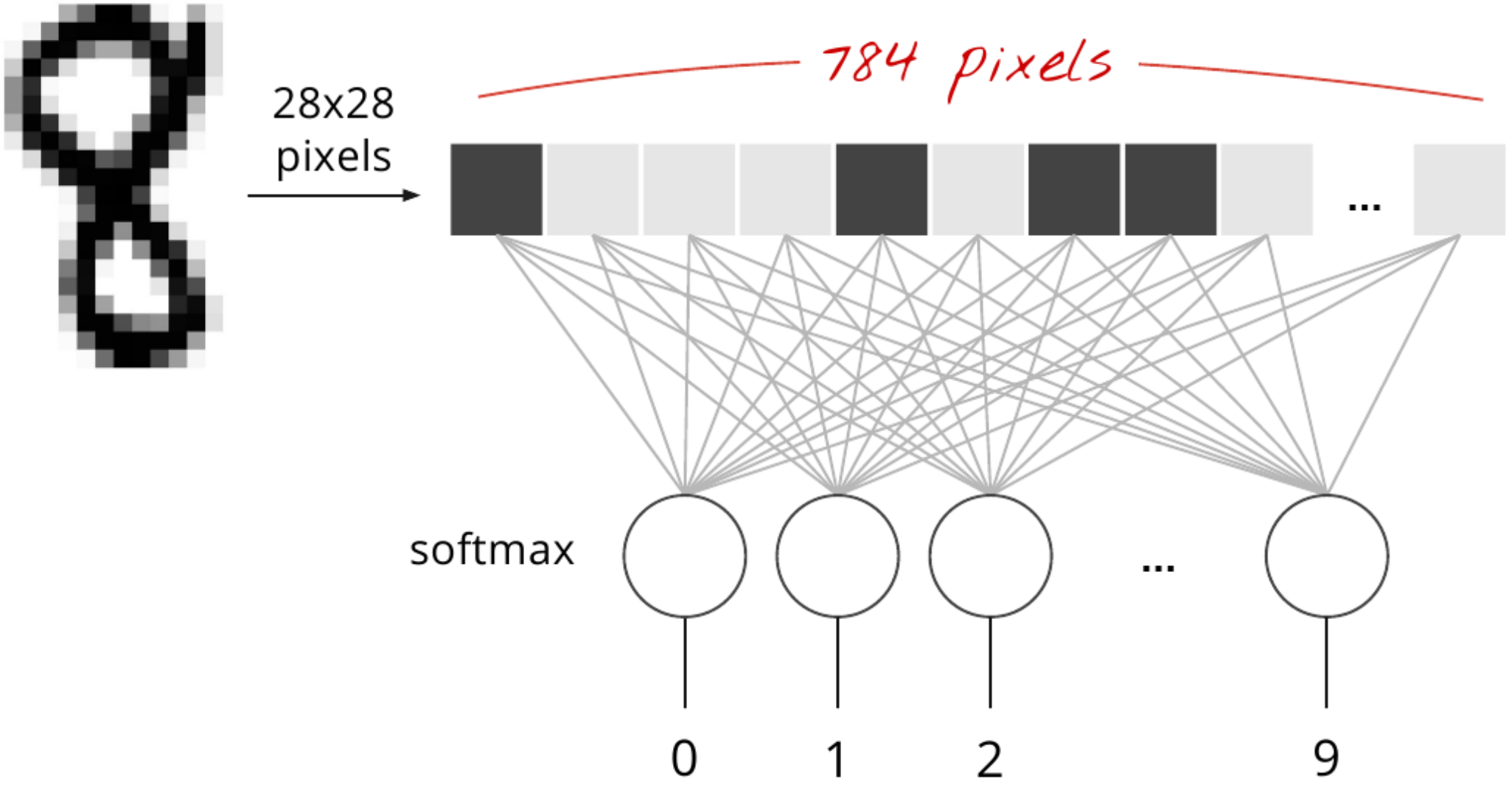
MNIST handwritten digits

A benchmark data set for many years. Good for illustrative purposes.

It contains 28 x 28 images, 60K to train the model, 10K to test.



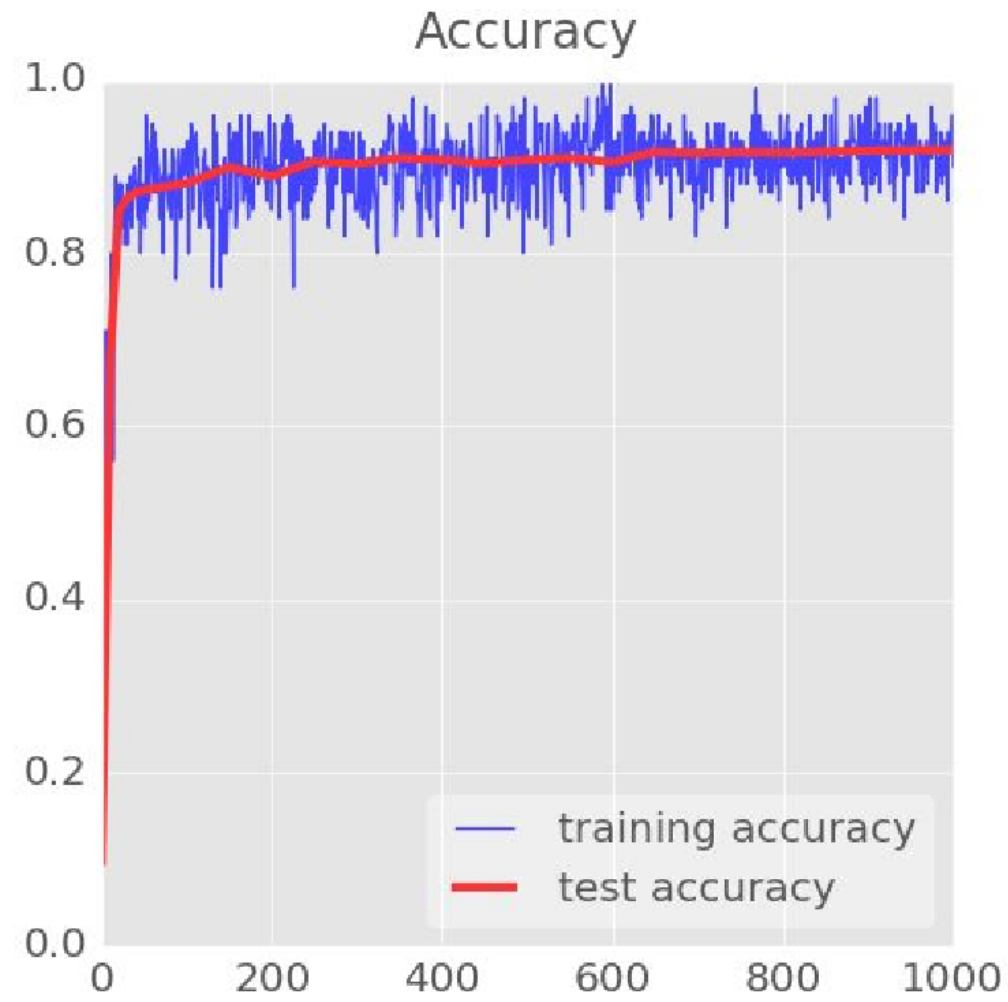
MNIST: a classification problem

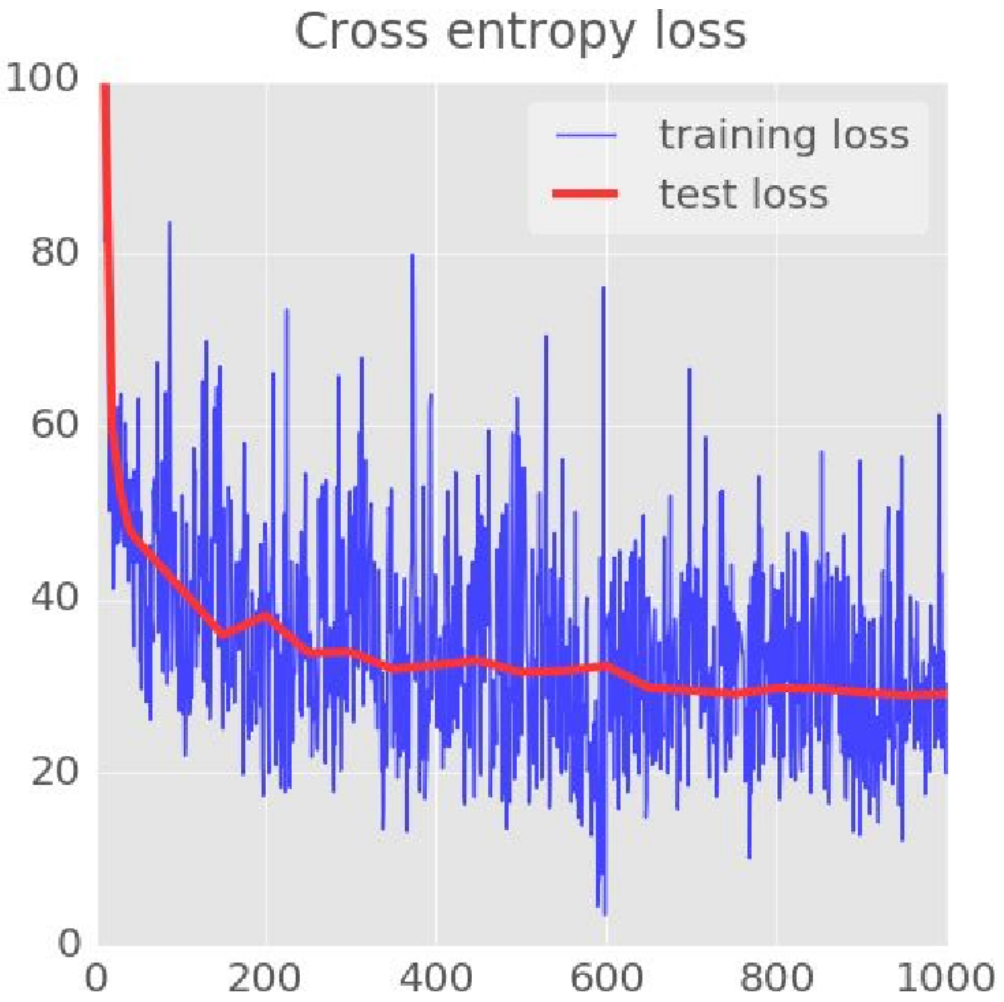


Learn 10 vectors (784 dims) and 10 biases to classify digits.

Left: The accuracy on new data (about 92%)

Right: The objective function we're minimizing



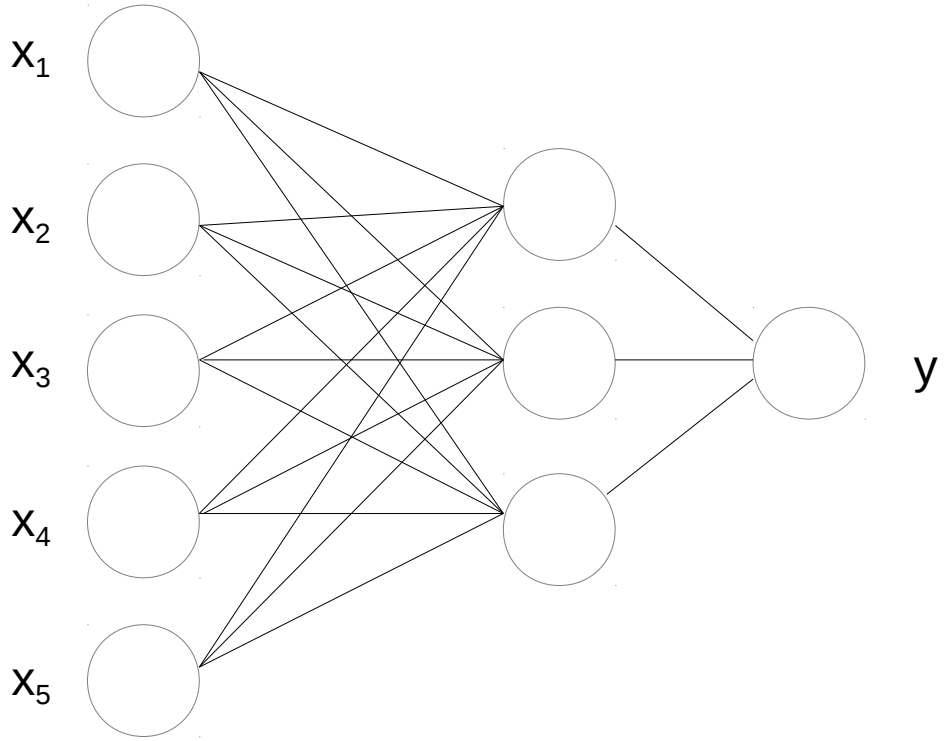


Building blocks of a deep learning model

Part 3: Multilayer perceptron

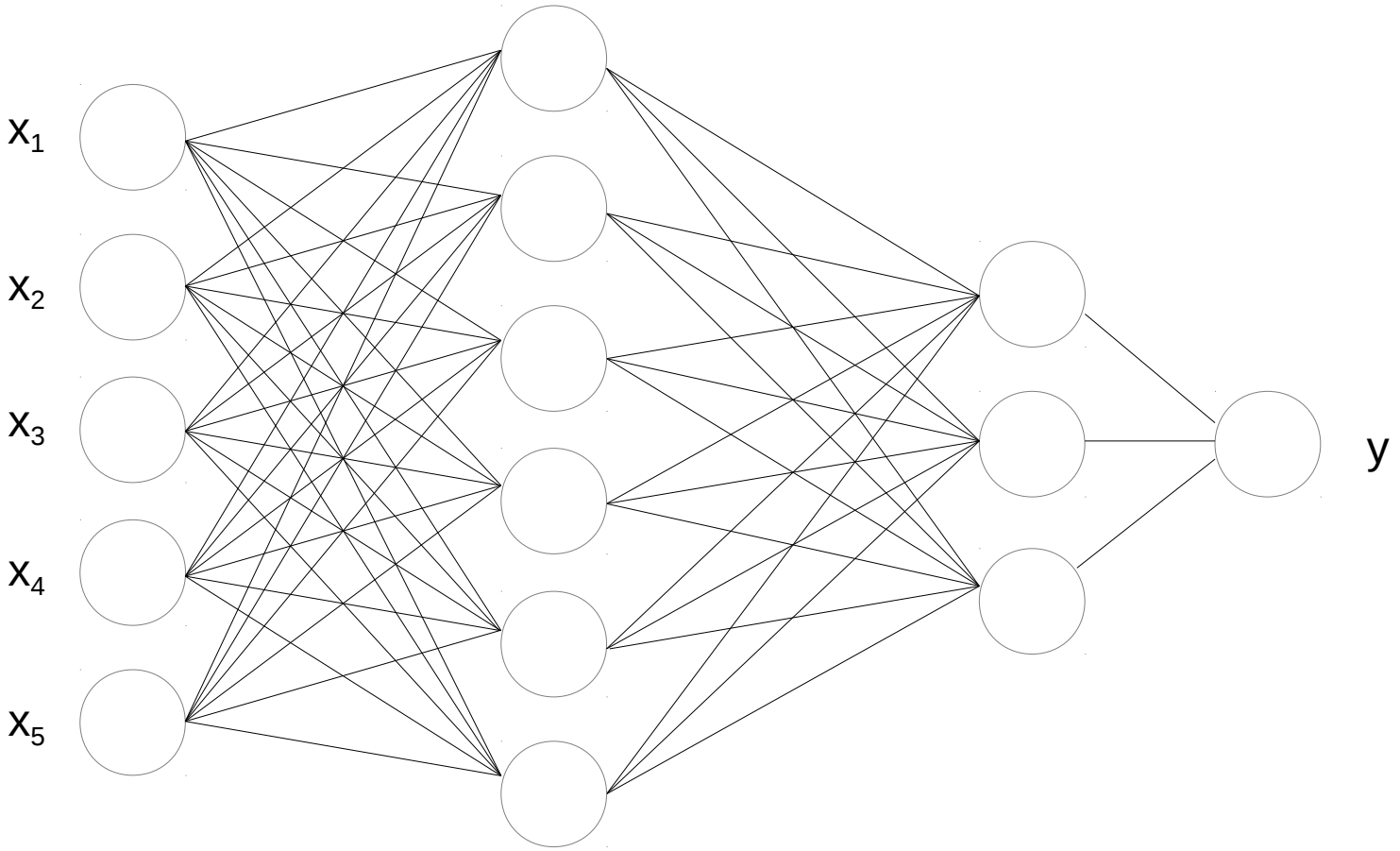
Making multinomial logistic regression a neural network

- The basic idea behind neural networks is to expand the multinomial logistic regression to include "**hidden layers**".

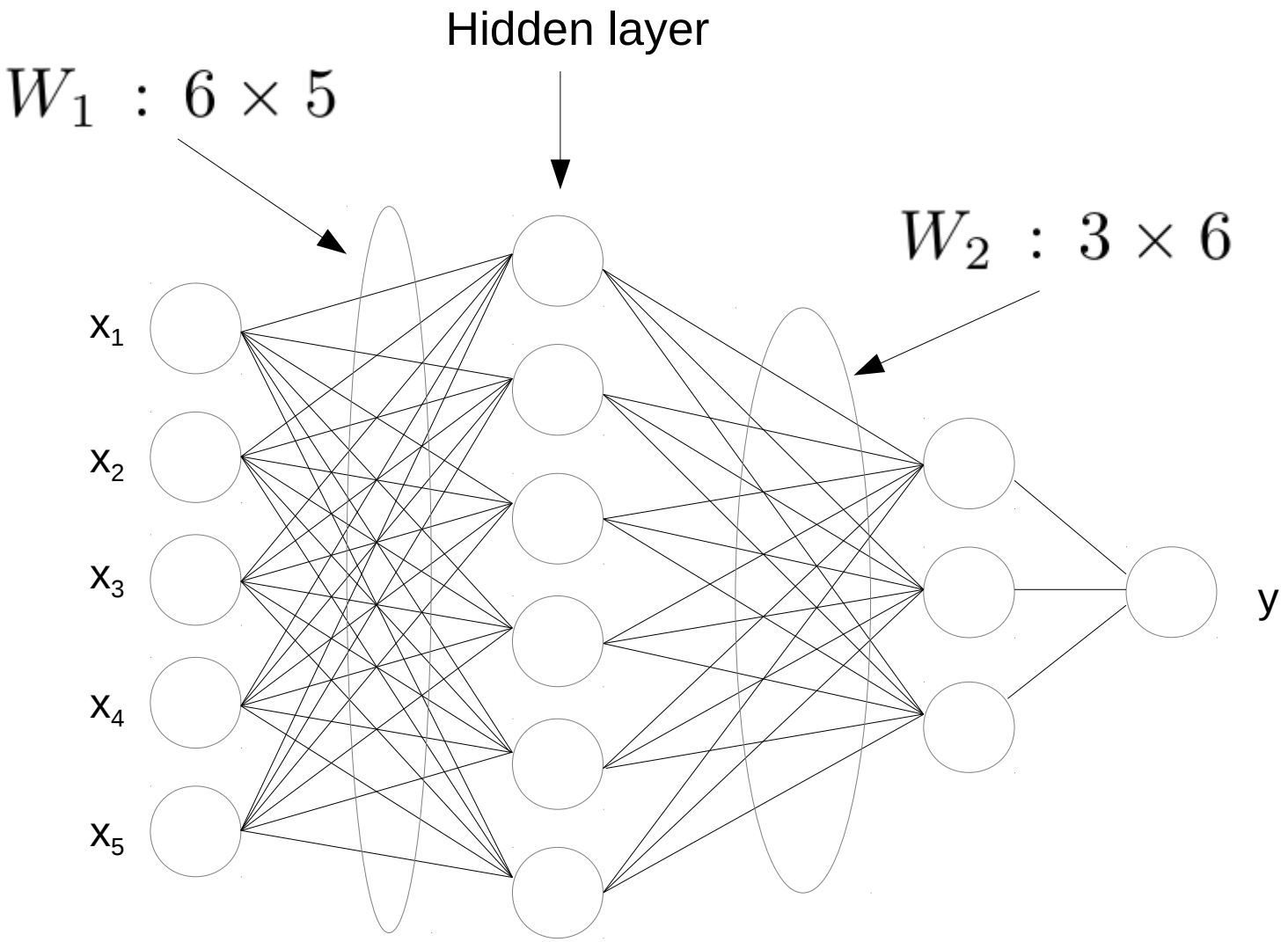


Making multinomial logistic regression a neural network

- The basic idea behind neural networks is to expand the multinomial logistic regression to include "**hidden layers**".



Basic neural networks (most subscripts now indicate layers)



Basic neural networks (most subscripts now indicate layers)

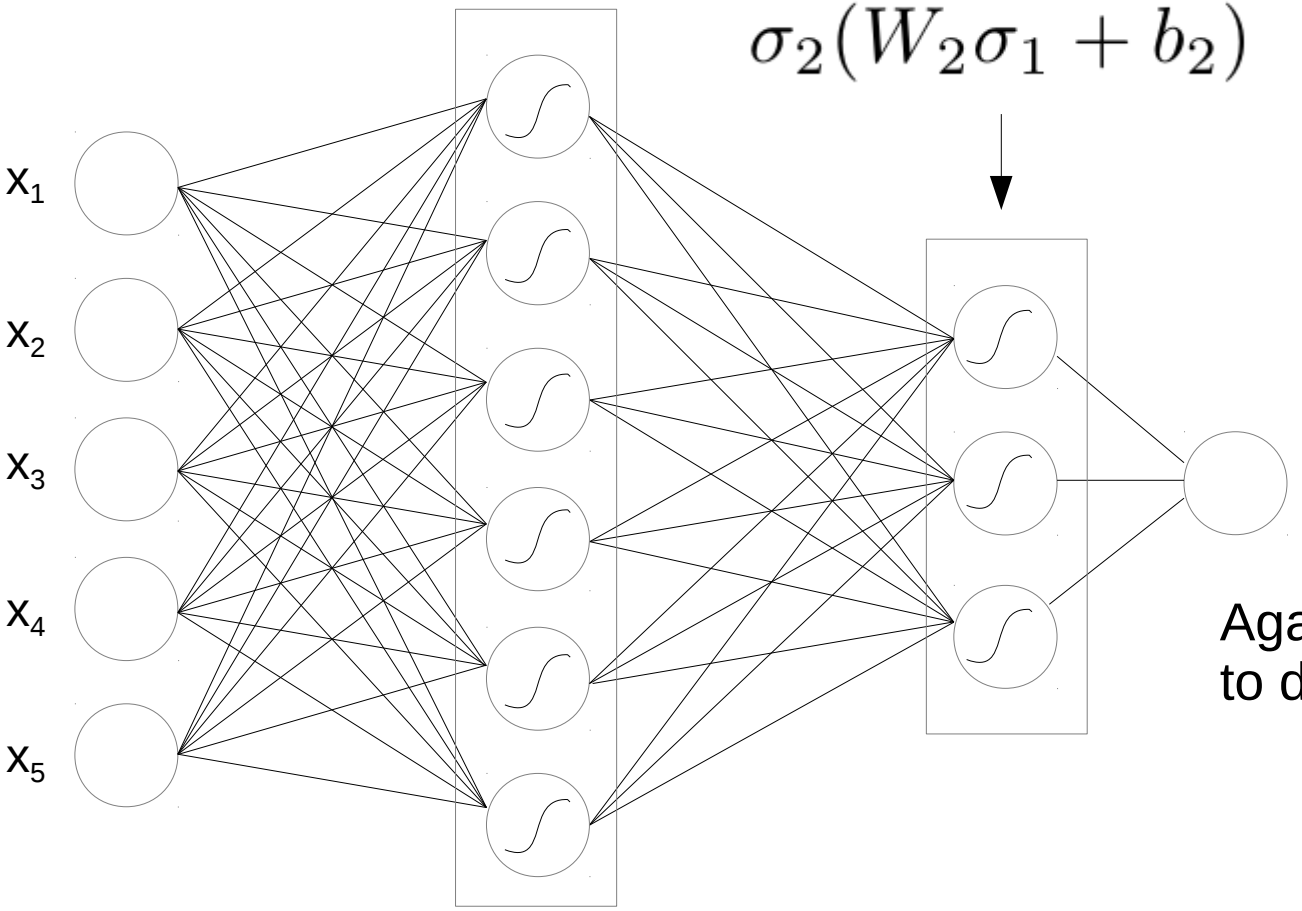
Alert: Notation overload!

Only the last layer is softmax.

All others are: $\sigma(a) = \frac{e^a}{1+e^a}$

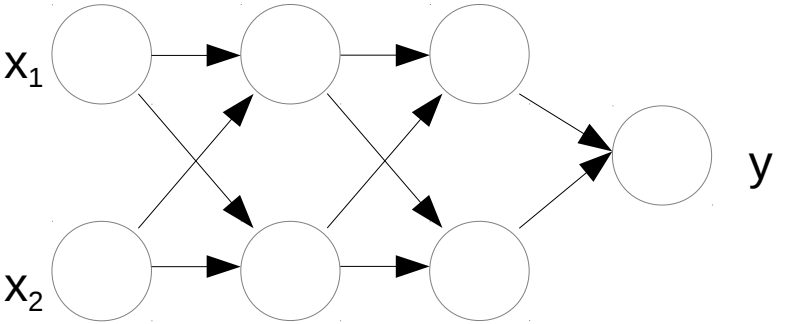
$$\sigma_1(W_1 \mathrm{x} + b_1)$$

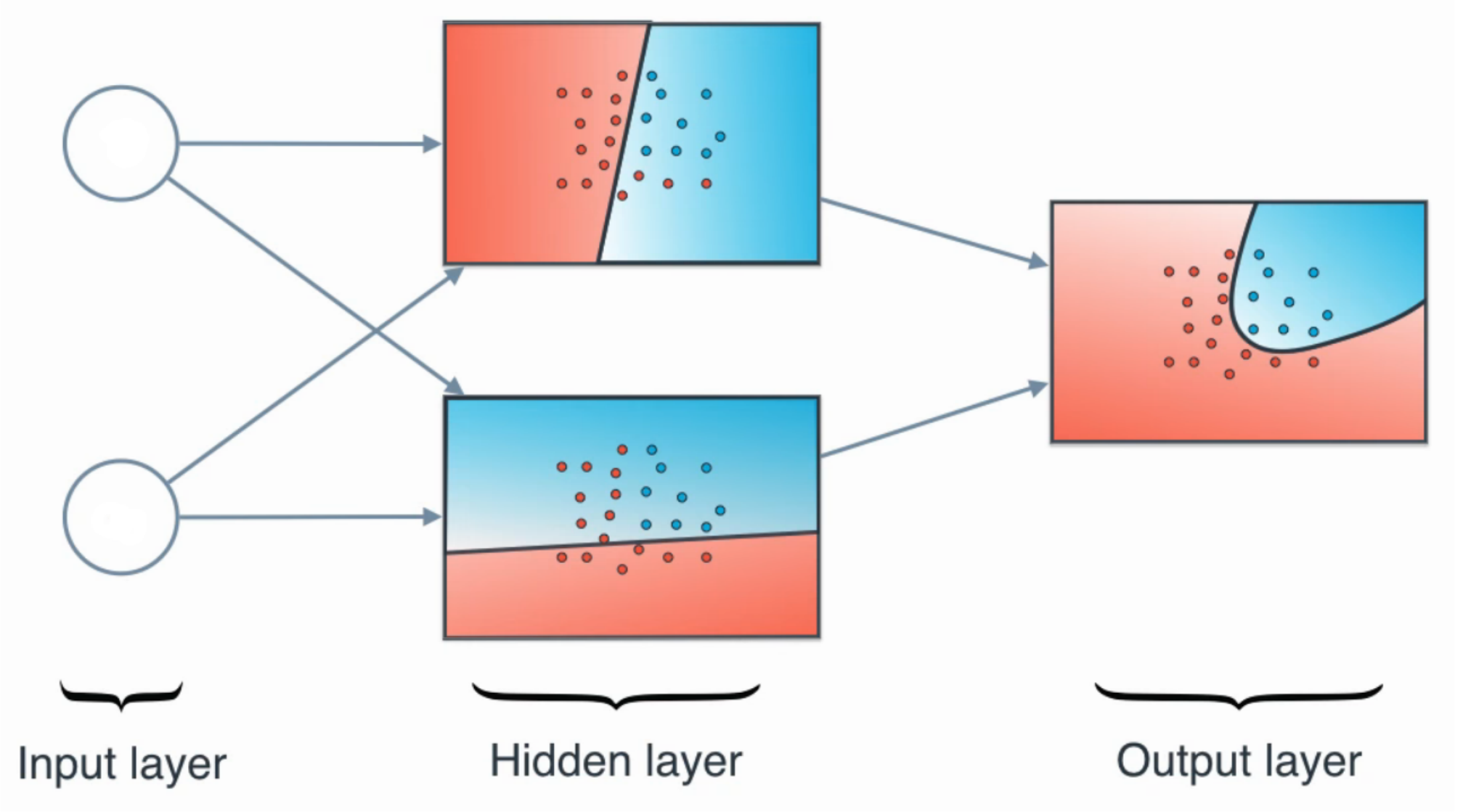
$$\sigma_2(W_2 \sigma_1 + b_2)$$



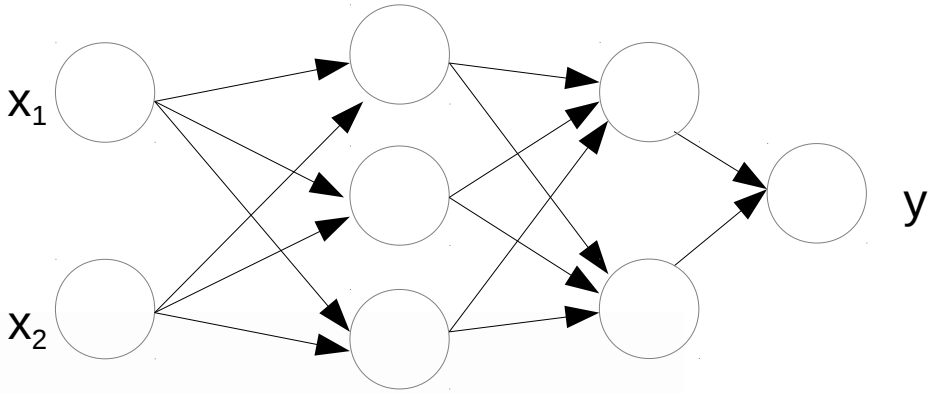
Again throw a 3-sided dice to decide what y equals.

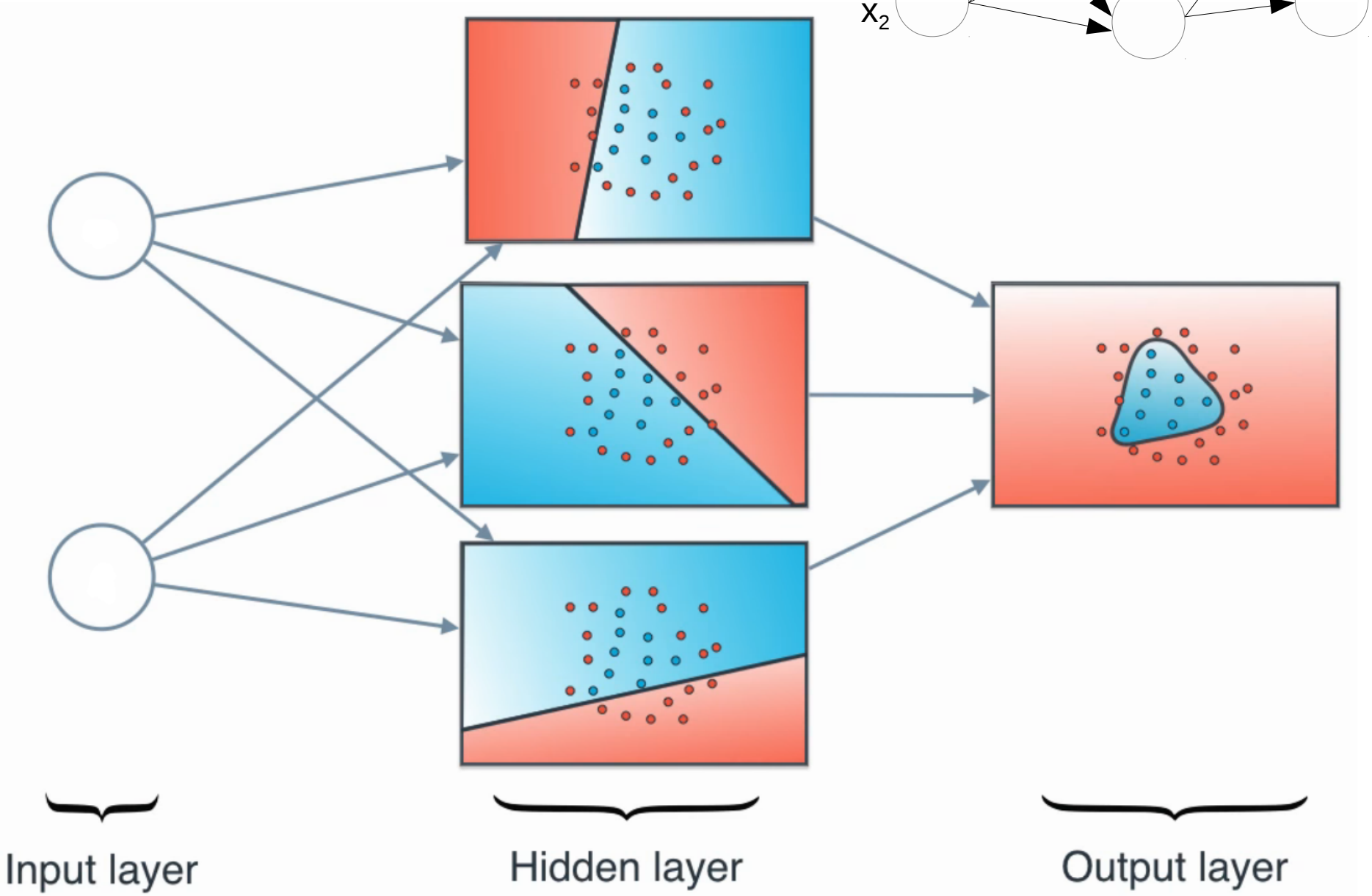
Why is this a good idea?





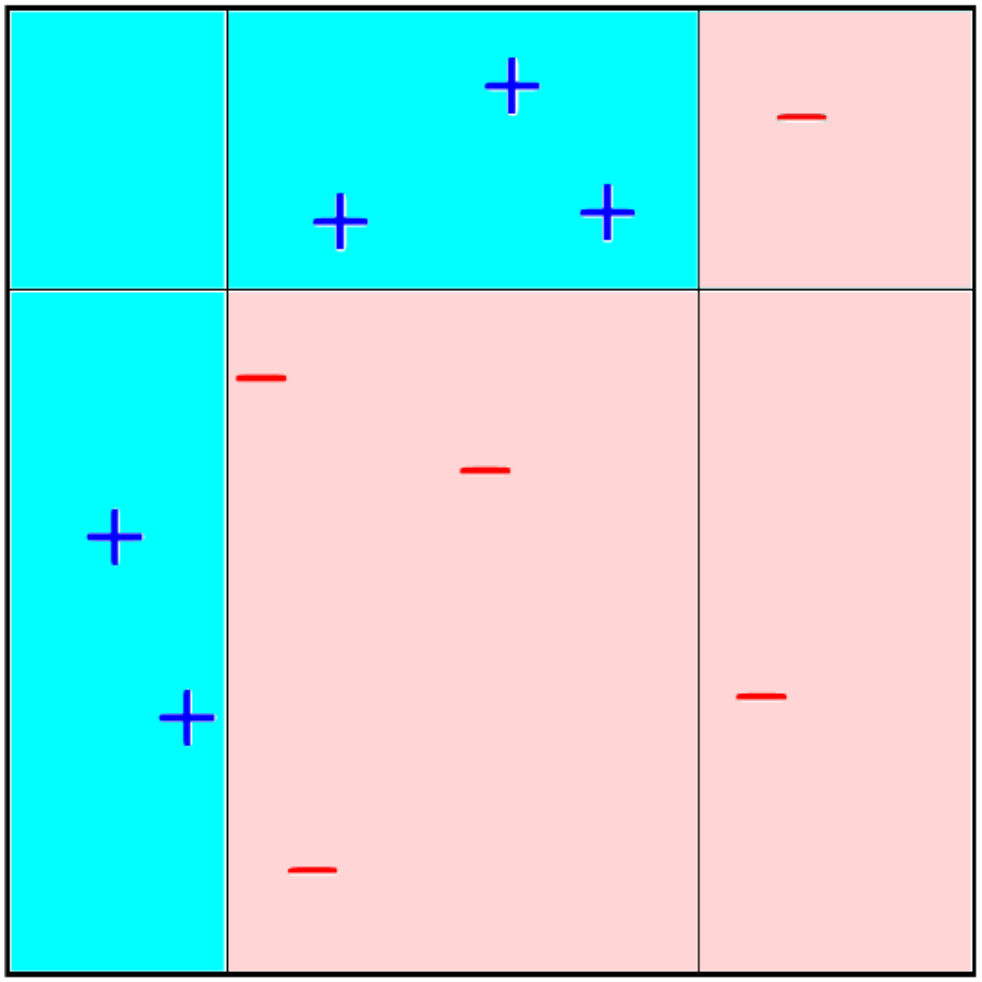
Why is this a good idea?

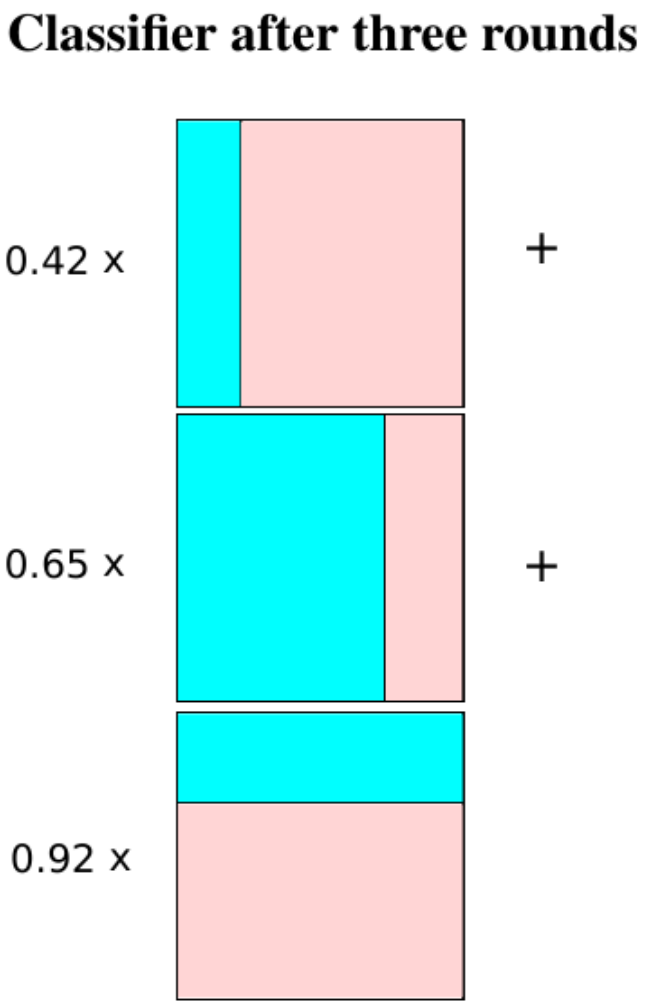




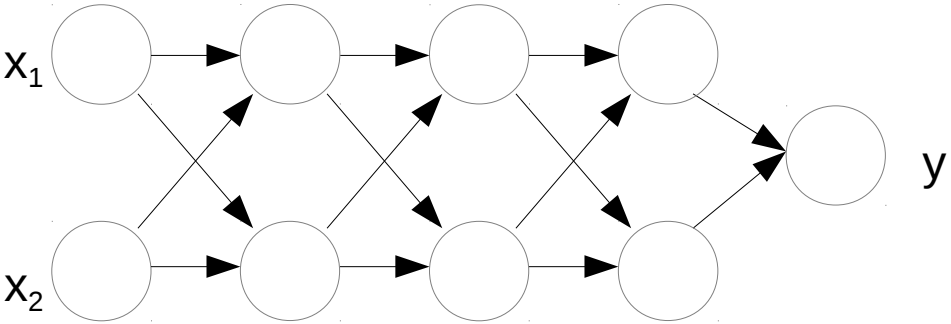
We've seen this idea before... $f_{boost}(x_0) = \text{sign}\left(\sum_{t=1}^{T} \alpha_t f_t(x_0)\right)$

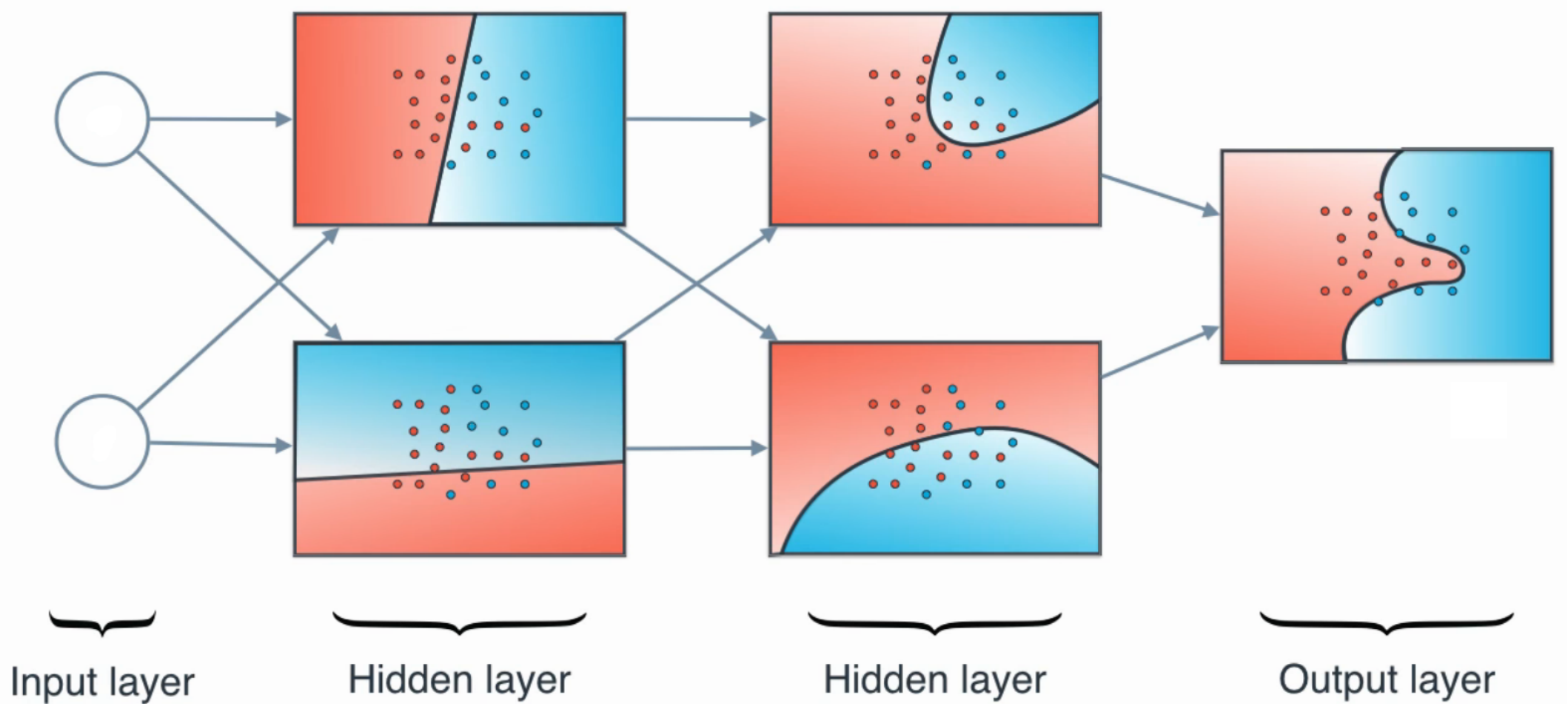
BOOSTING A DECISION STUMP (EXAMPLE 1)



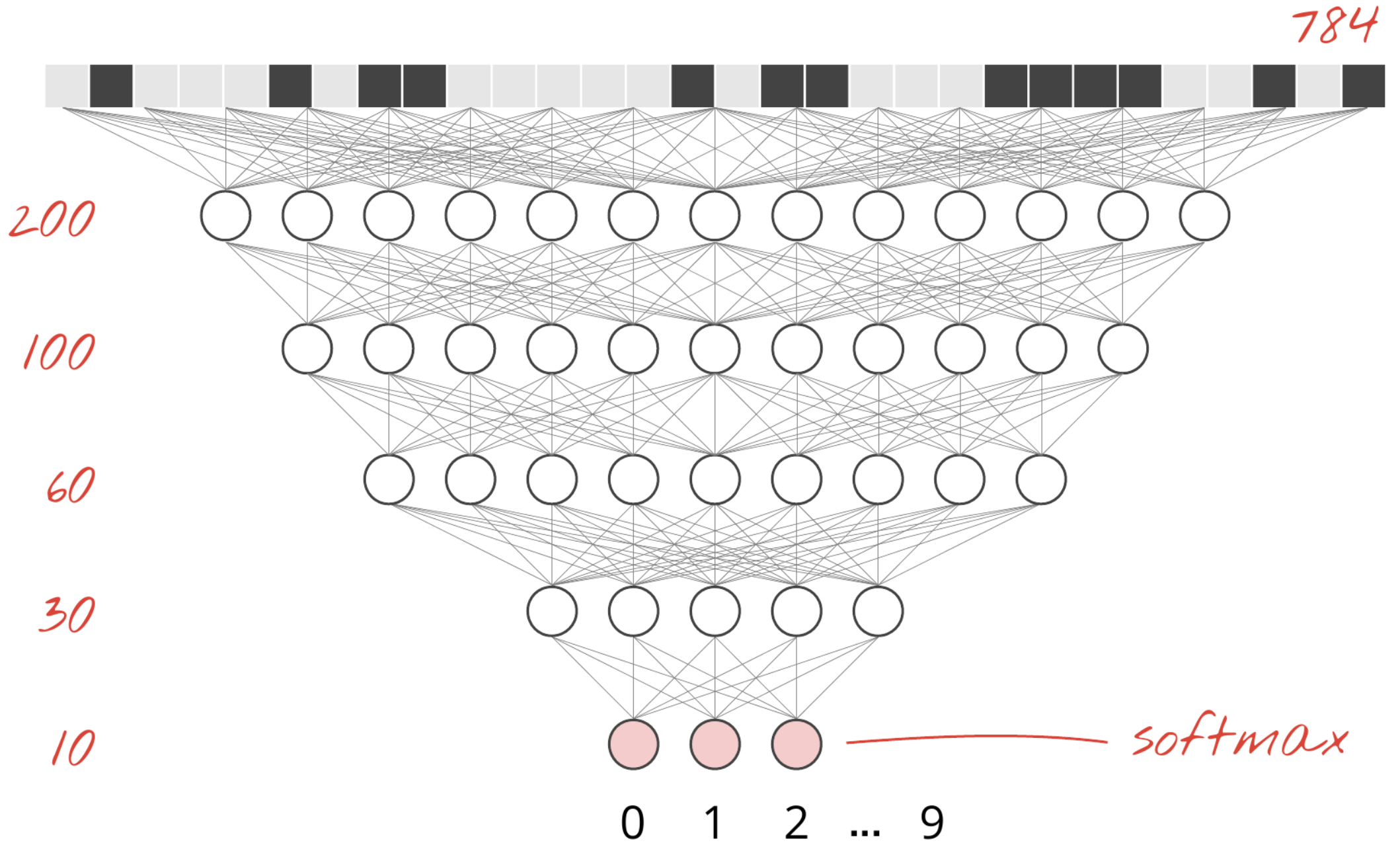


Why is this a good idea?





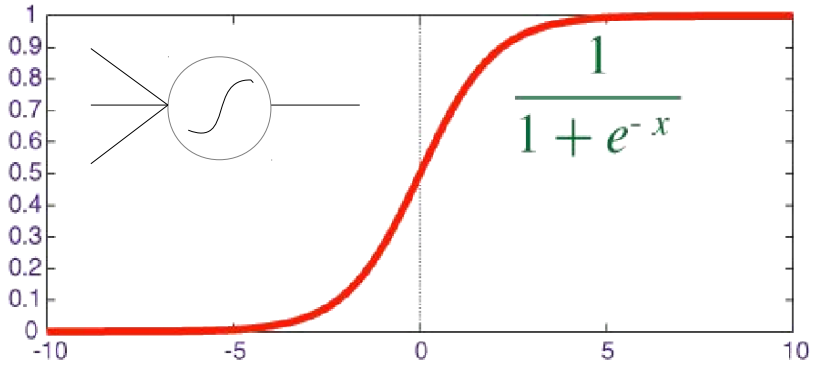
MNIST revisited



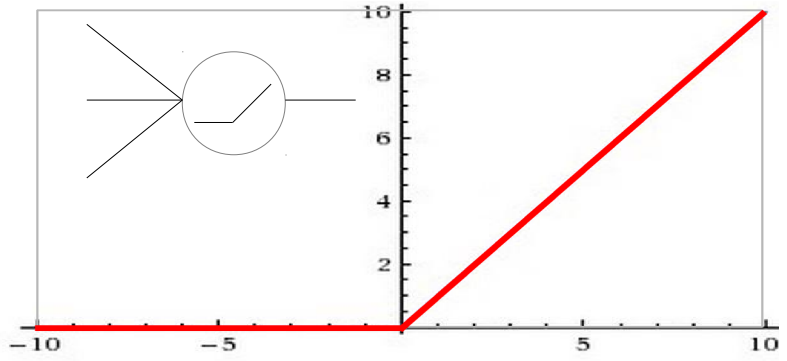
Technical problems and some tricks/hacks to fix them

- Scalability: use **stochastic optimization** with a decreasing **step size**
- Gradient vanishing: change activation function

Sigmoid function



ReLU: Rectified Linear Unit

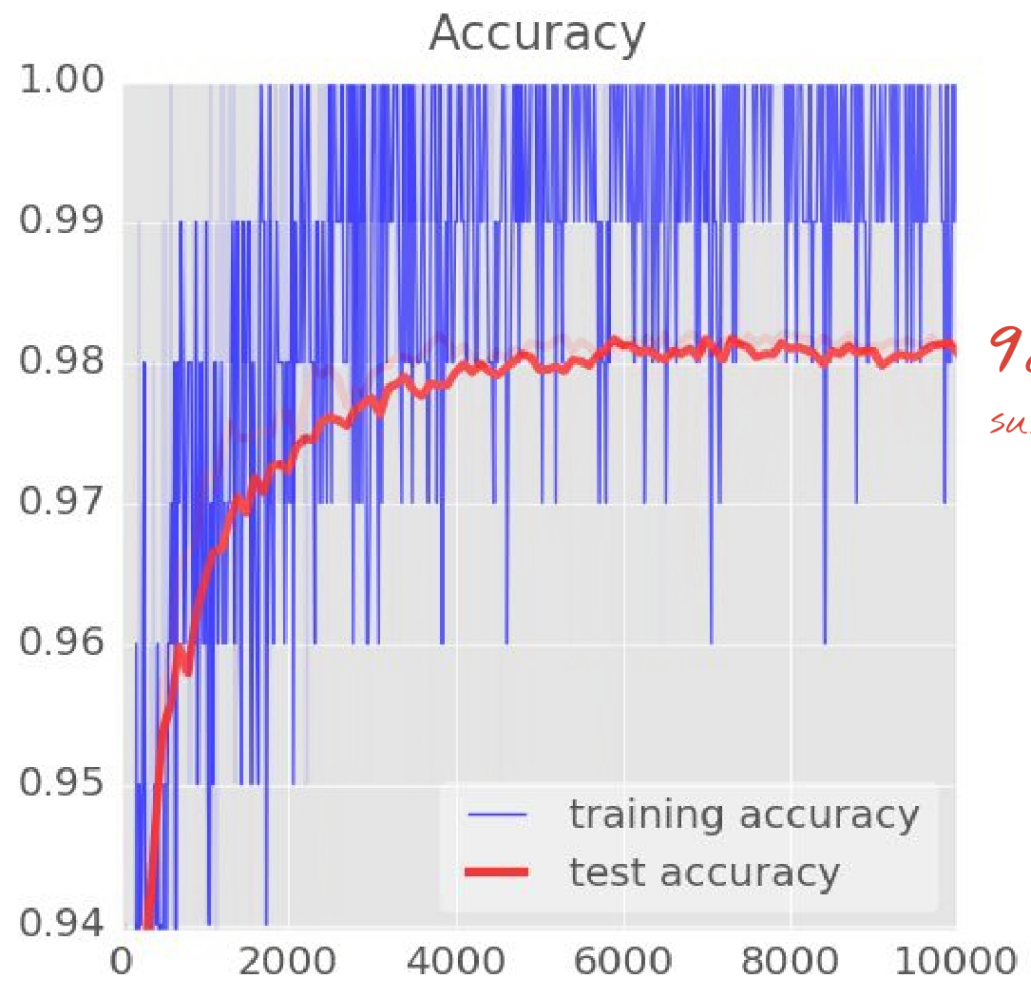


- Other keywords: backpropagation (gradient calculation)
  dropout
  early stopping
  batch normalization
  etc., etc., etc., etc., etc., etc., etc., etc., etc., etc.

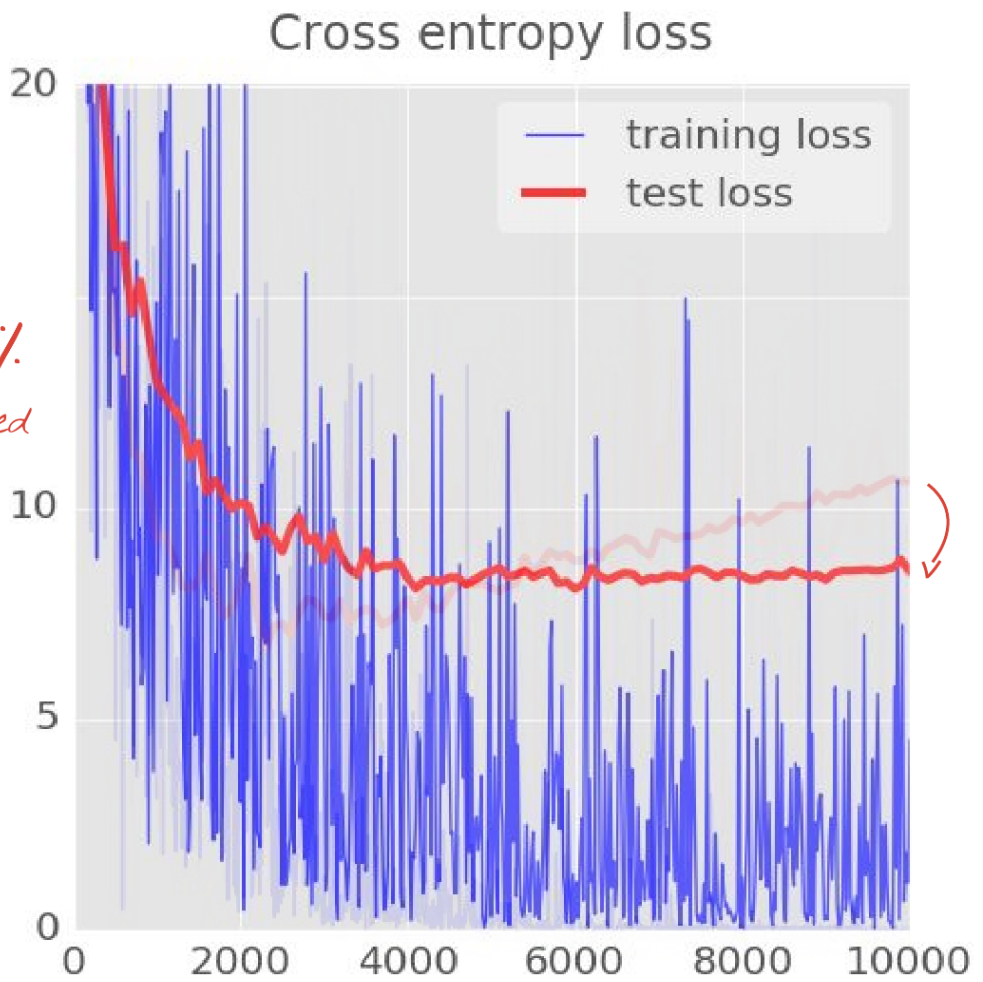
Specs: 4 hidden layers (200,100,60,30). Roughly 185K parameters

Left: The accuracy on new data (> 6% improvement)

Right: The objective function we're minimizing



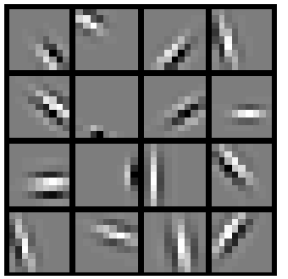
98.2% sustained



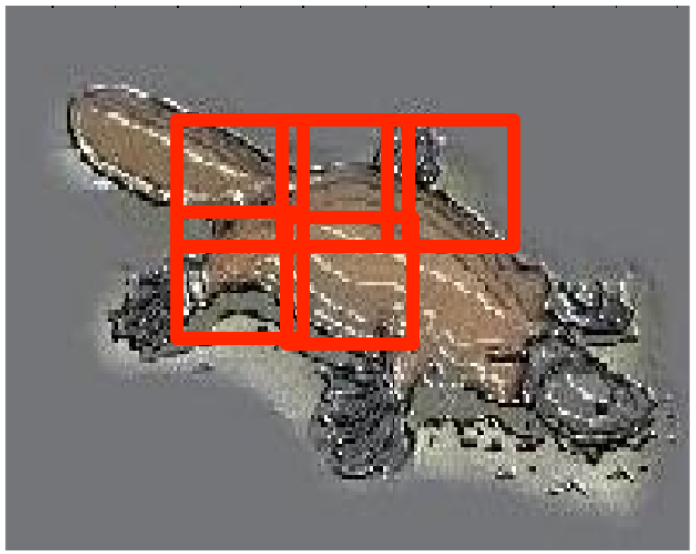
Building blocks of a deep learning model

Part 4: Convolutional neural networks (CNN)

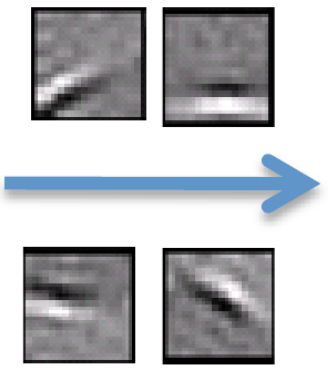
CNN: Deep learning with images



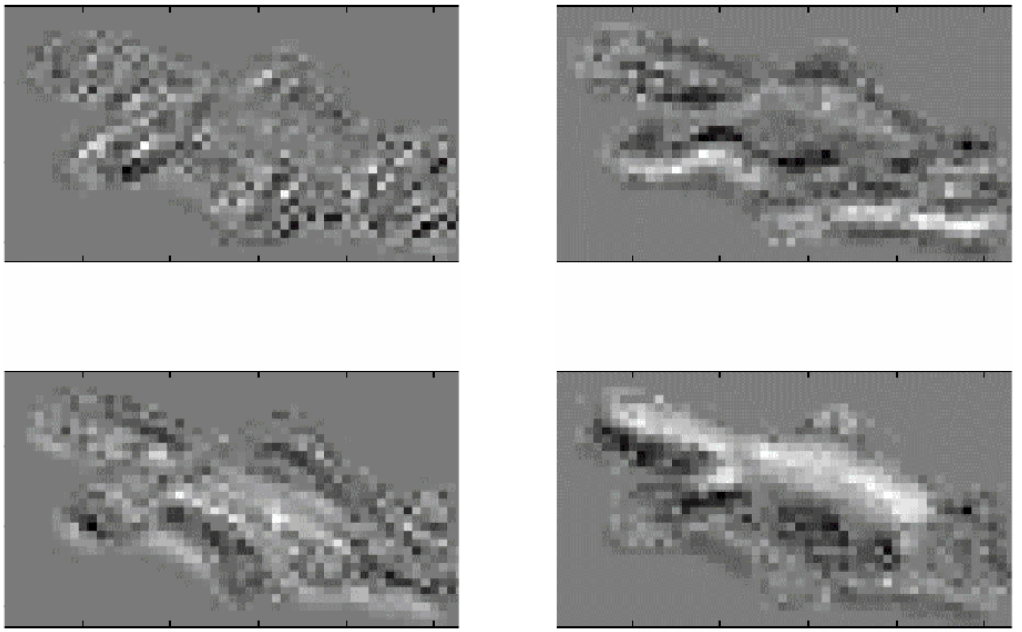
- A clear drawback of this approach with MNIST is that we removed structural information from the image (by vectorizing).
- **Convolutional neural networks** modify the previous framework by defining some layers to be convolutions using learned filters.



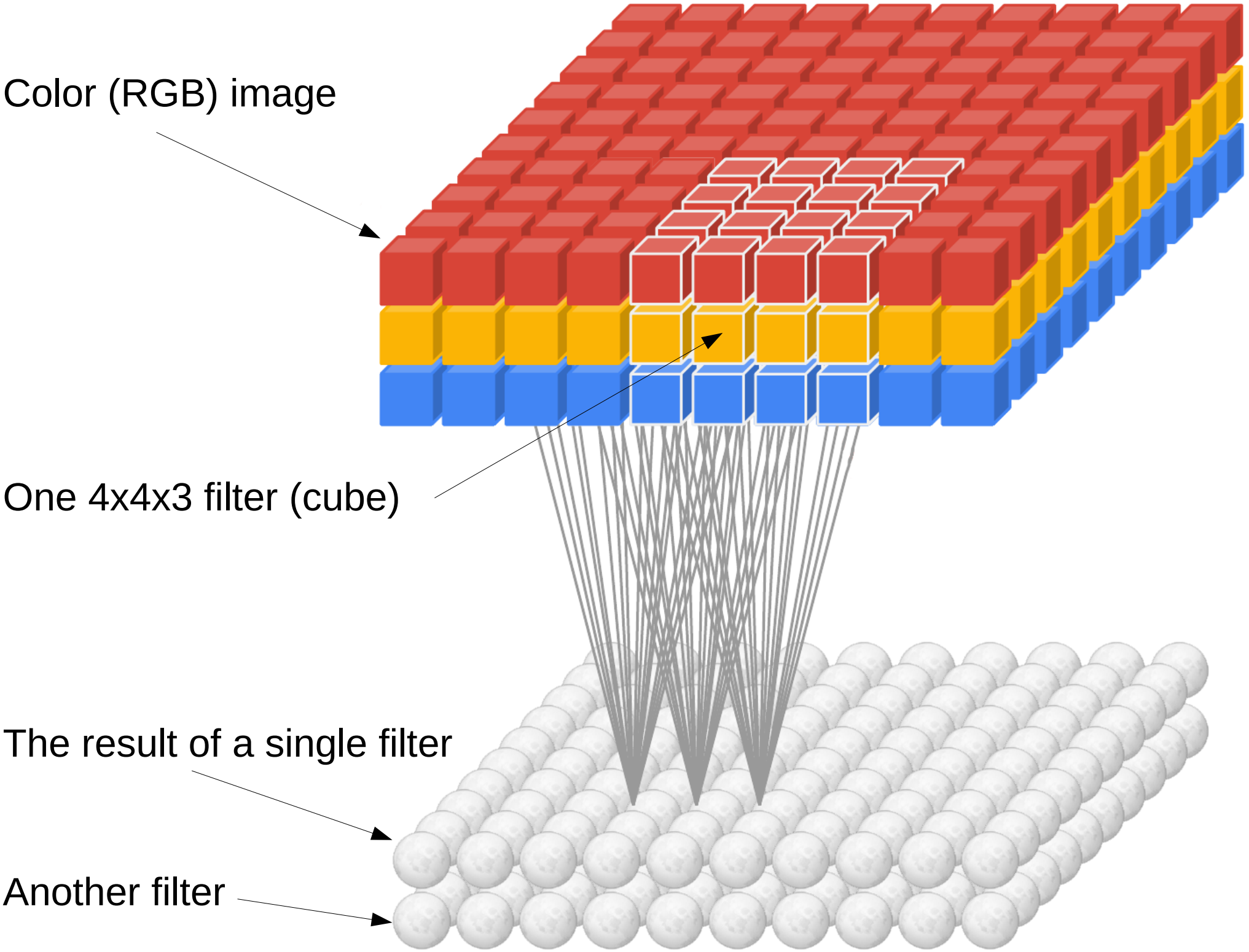
Input Image

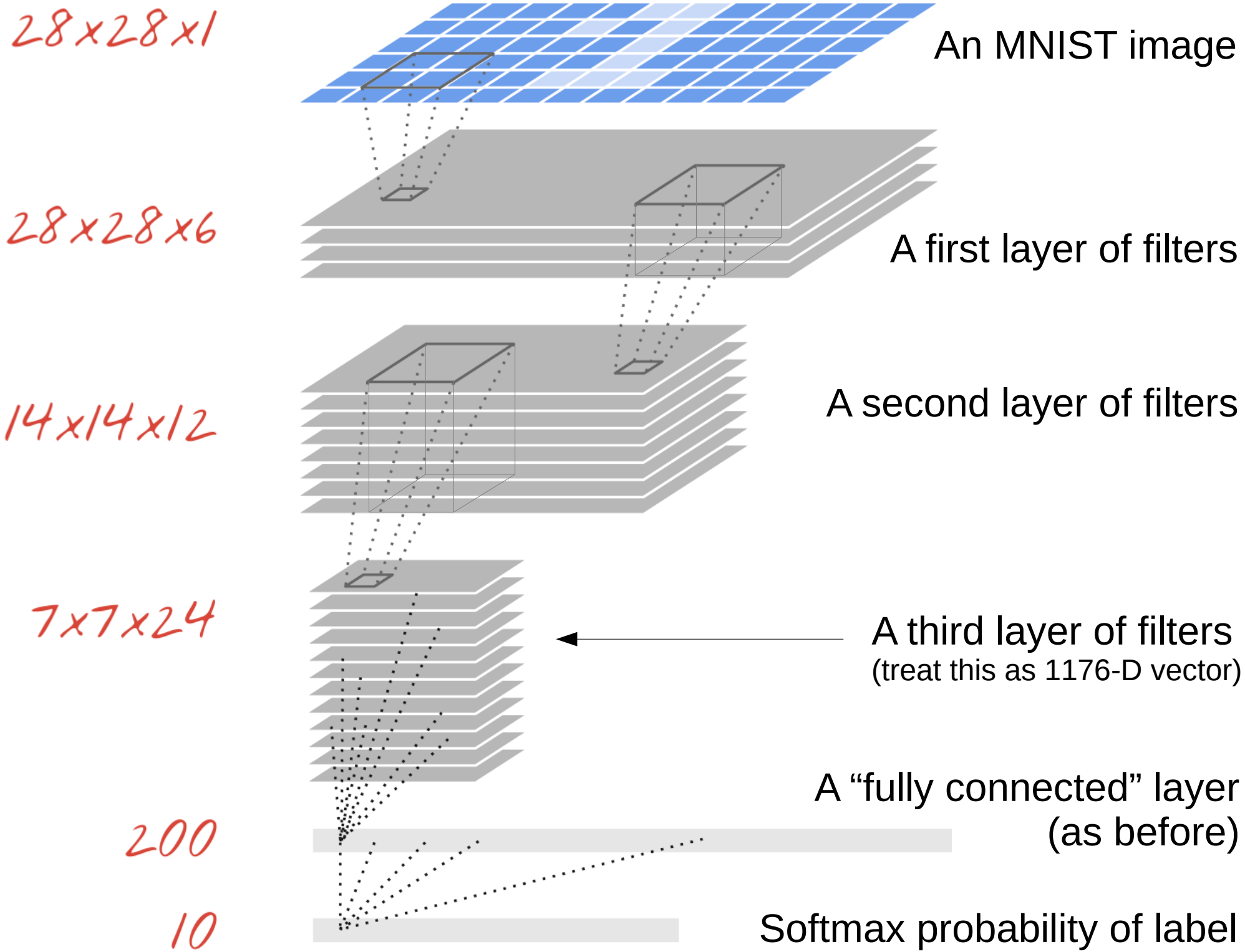


Learned bases



Features (coefficients)


Color (RGB) image
One 4x4x3 filter (cube)
The result of a single filter
Another filter

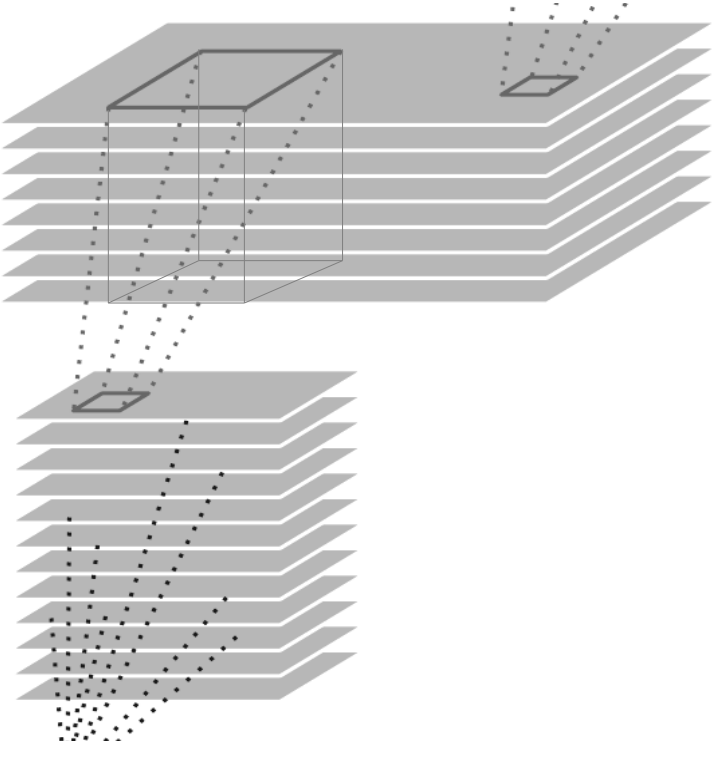

28x28x1
An MNIST image
28x28x6
A first layer of filters
14x14x12
A second layer of filters
7x7x24
A third layer of filters
(treat this as 1176-D vector)
A “fully connected” layer
(as before)
200
10
Softmax probability of label

How did we go from 14x14x8 to 7x7x12?

- We used 12 different filters of size 4x4x8. But with correct padding, this should give 14x14x12 on the bottom row.
- Two common techniques are:
  - **Max-pooling:** Here, pick max of 2x2 windows
  - Change "**stride**": shift filters by 2

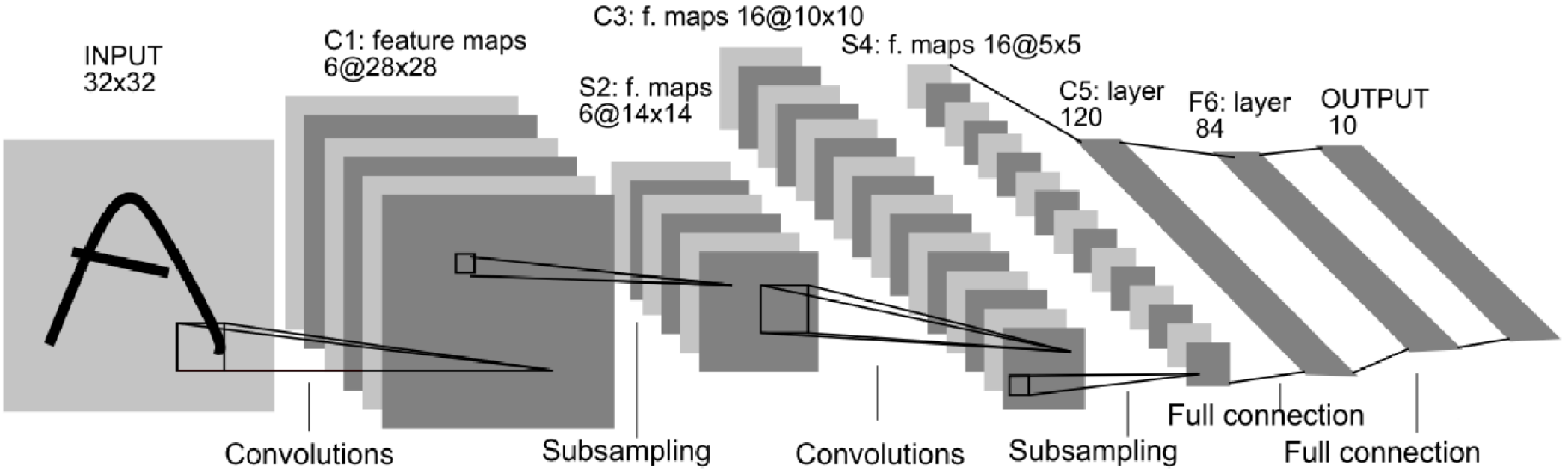
14x14x8

7x7x12



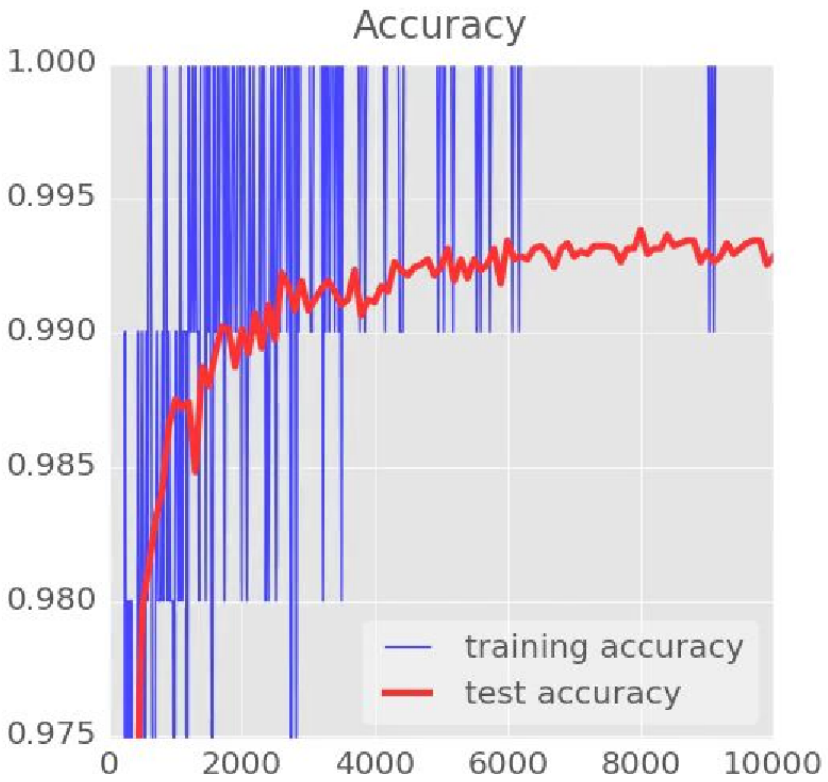
LeNet (Yann LeCun, 1998)

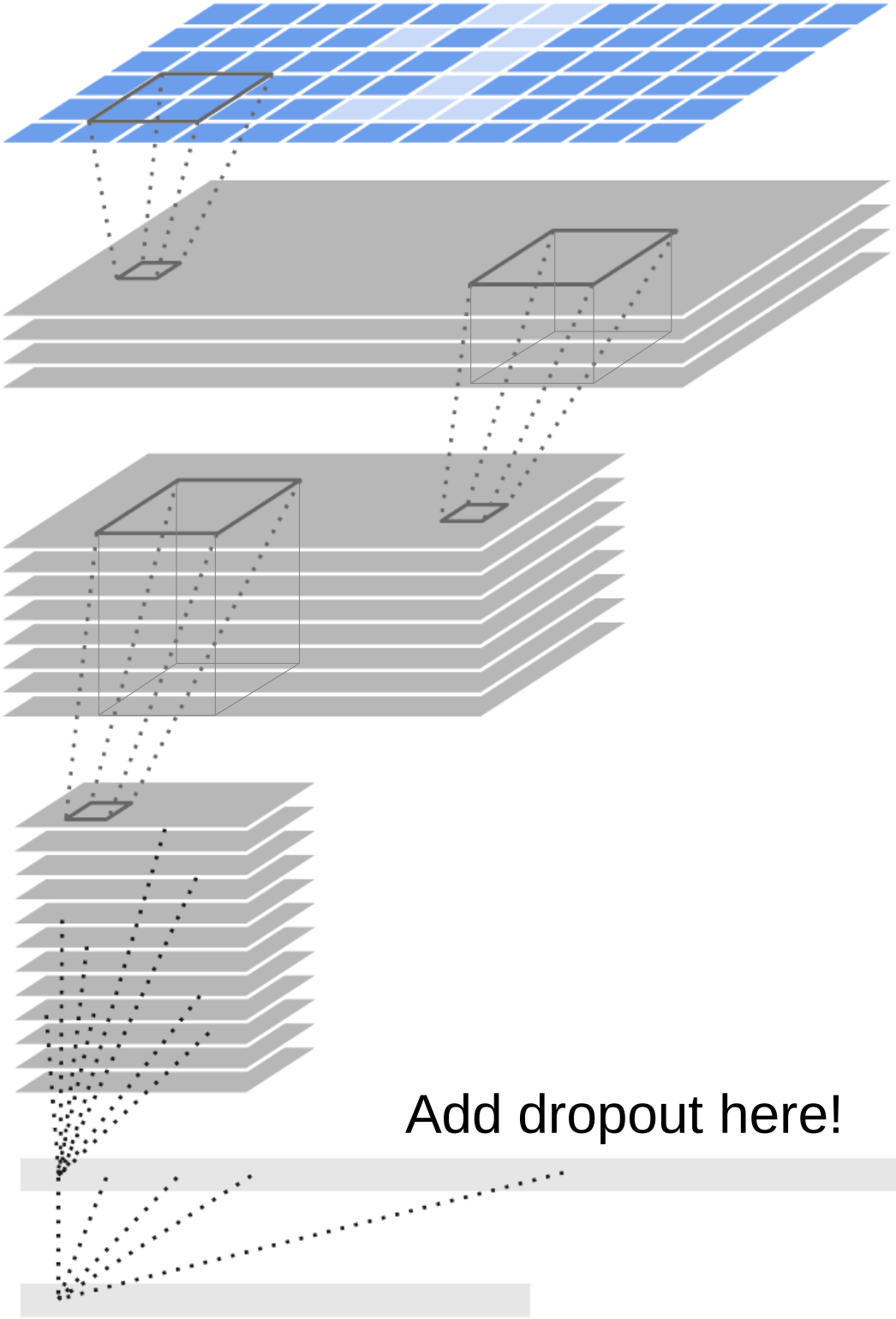
- An early CNN.
- Obtained roughly 98.9% accuracy on MNIST.
  Better than 98.2% with “fully connected” neural net of previous slide.



Using the whole bag of tricks

Accuracy 99.3%





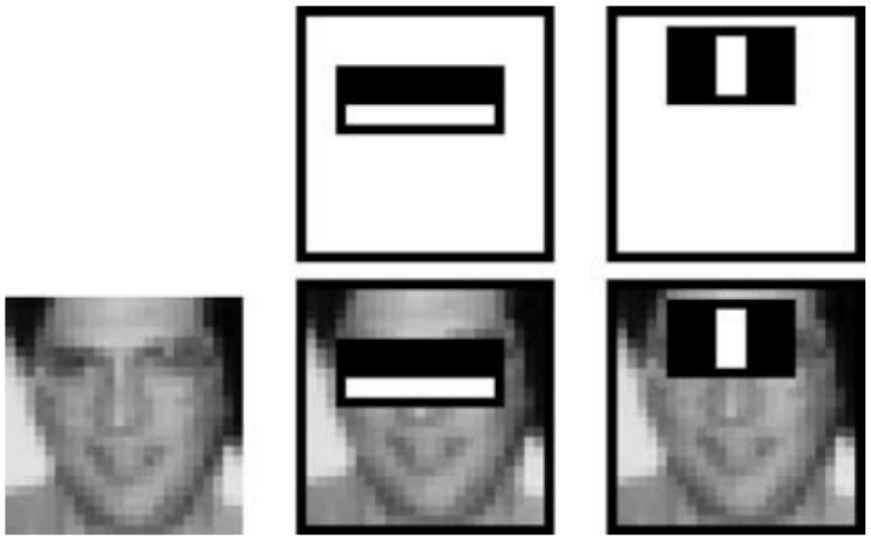
We've seen this idea before...

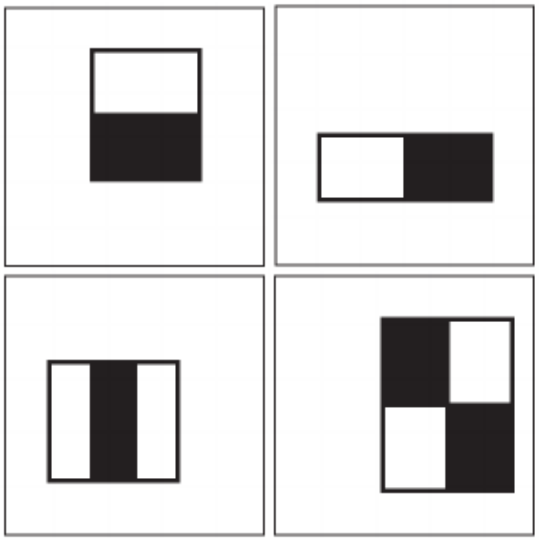
FACE DETECTION (VIOLA & JONES, 2001)

Problem: Locate the faces in an image or video.

Processing: Divide image into patches of different scales, e.g., 24×24, 48×48, etc. Extract *features* from each patch.

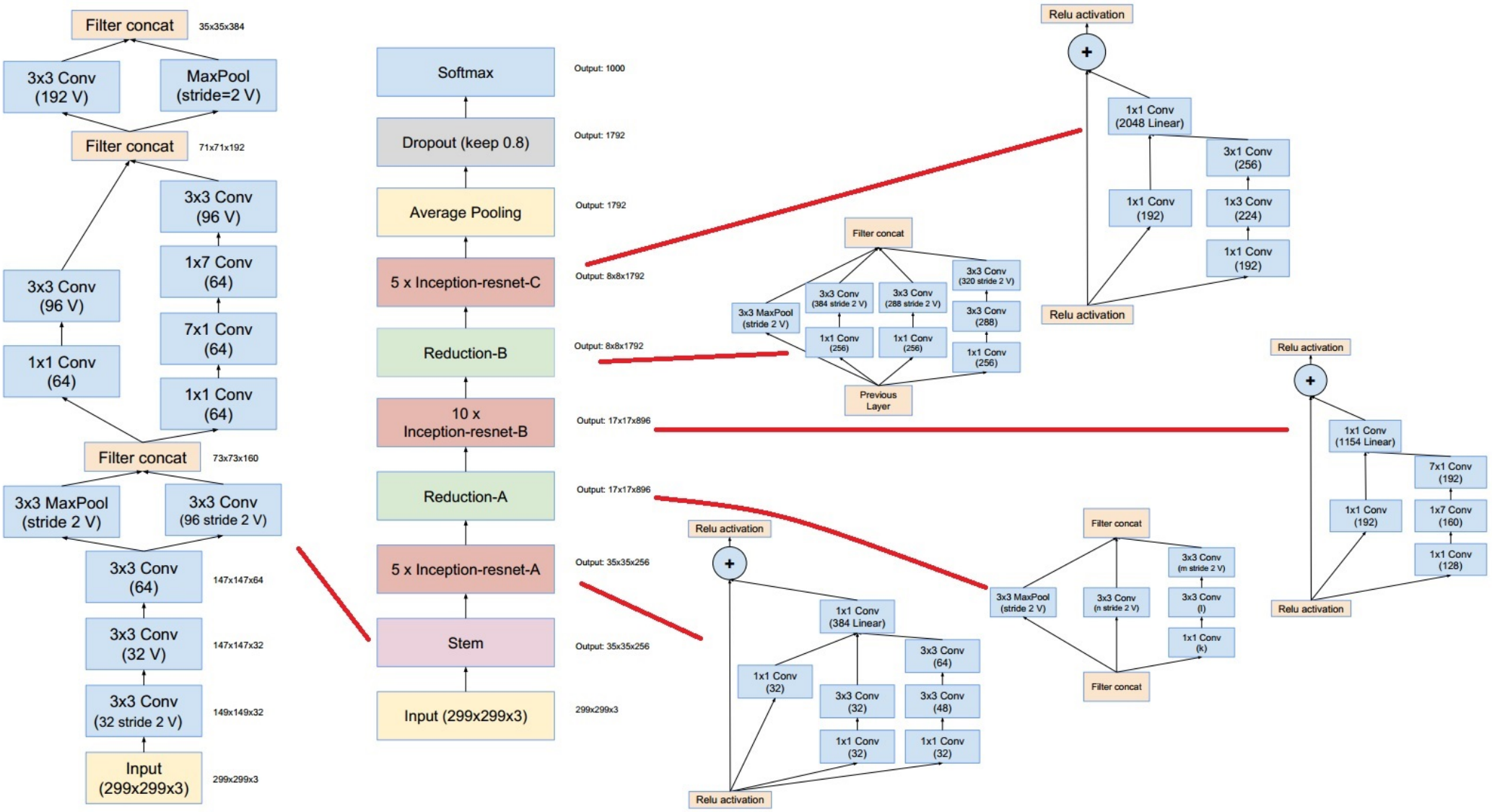
Classify each patch as face or no face using a *boosted decision stump*. This can be done in real-time, for example by your digital camera (at 15 fps).





- One patch from a larger image. Mask it with many "feature extractors."
- Each pattern gives one number, which is the sum of all pixels in black region minus sum of pixels in white region (total of 45,000+ features).

Deep convolutional neural networks have become very complicated. But they are modular and build upon the components we've been discussing.



Other deep learning frameworks

- Autoencoders (AE)
  - unsupervised learning, representation learning
- Generative adversarial networks (GAN)
  - try to “fool” a computer into thinking generated data is real
- Recurrent neural networks (RNN)
  - sequential data, especially language modeling

the two birds are trying to be seen in the water .

a giraffe is standing next to a fence in a field .

a parked car while driving down the road .