

# Neural Network and TensorFlow

---

Dapeng Hu, Qinglong Tian, Haozhe Zhang, Min Zhang

April 23, 2017

STAT 580 Statistical Computing  
Department of Statistics  
Iowa State University

# Content

1. History of Neural Network
2. Neural Network
3. Convolutional Neural Network
4. TensorFlow

# History of Neural Network

---

# History of Neural Networks

- The 1940s: The Beginning of Neural Networks
- The 1950s and 1960s: The First Golden Age of Neural Networks
- The 1970s: The Quiet Years
- The 1980s: Renewed Enthusiasm

## The 1940s: The Beginning of Neural Networks

McCulloch and Pitts, 1943

- Employed logic and mathematical notion of computation to explain how neural mechanism might realize mental functions.
- Commonly regarded as the inception of artificial neural networks.

## The 1940s: The Beginning of Neural Networks

McCulloch and Pitts, 1943

- Employed logic and mathematical notion of computation to explain how neural mechanism might realize mental functions.
- Commonly regarded as the inception of artificial neural networks.

# The 1940s: The Beginning of Neural Networks

## Hebb's Learning Rule:

- Connectionism
- A method of determining how to alter the weights between model neurons.



In work with another Banting colleague, Leonard Andrewes, Hebb conditioned dogs and became less impressed with Pavlov's ideas. After a period of searching as to whether he should continue in psychology, he decided in 1934 to move to Canada, where he moved to Chicago to continue his doctoral research under Charles Sherrington.

The older scientist was to exert a profound influence on Hebb's approach, although he did not know it at the time. As a result of his work with Andrewes, he had salvaged slides of a frog brain from a discarded experiment. These slides, in the neural connections some due to memory, provided the first attempts to derive memory traces in the brain, inventing techniques for making them visible. By 1937 he had become convinced that memories could

not be stored in a single region of the brain but must be spread throughout. In 1934, when Hebb went to Chicago, he left behind his work on memory, concentrating on the study of vision.

After later Lauder was offered a professorship at Harvard University, he accepted. However, before taking up the post, he had to earn his research free scratch, and having only enough money to buy a typewriter, he conducted an experiment that could support a thesis on the visual system. Hebb, however, tried to adapt his interest in Lauder's nature-nurture question to Lauder's interests by investigating the effect of early experience on the development of the visual system.

Courtesy to the empiricist idea of his master's thesis, Hebb found that rats raised in a dark environment could

not distinguish the size and brightness of patterns as accurately as rats reared nor-

mal. This finding indicated that the organization of the visual system was innate and independent of environmental factors. Hebb's work was well received at the Gestalt school, to which Lauder was sympathetic (see *The Legacy of Gestalt Psychology*, by R. L. Solso and Stephen Palmer; SCIENTIFIC AMERICAN, April 1987).

After Lauder died in 1945, Hebb and Stephen Palmer, SCIENTIFIC AMERICAN, April 1987). Although the results were included in a paper he published at the time, he did not publish them again until much later than normal rate to learn to discriminate between different sizes of patterns. Only many years later, after he had again changed his ideas about the relationship between memory and learning mechanisms, did he appreciate the significance of this work.

Hebb obtained his Ph.D. from Harvard in the middle of the Depression, but he did not have the opportunity to practice psychology to his heart. He therefore started working as a teaching assistant while he was still a graduate student to finance his work with Lauder. In 1937 there was a post available at McGill University, but Hebb's luck held out. His sister was taking her Ph.D. in physiology at the University of Toronto, and she had a surgeon who had just established a clinic in Montreal. Hebb approached this surgeon, who was the main-

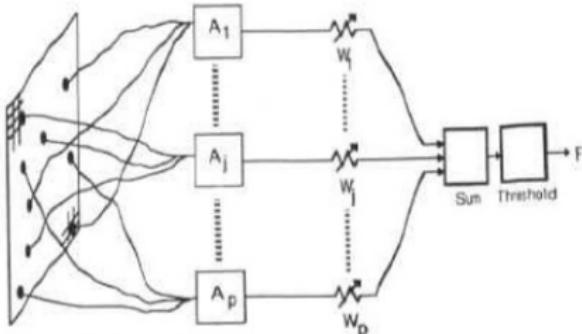
tenance of facial palsies by surgically resecting the nerve that controls the muscle. Hebb was looking for someone to study the consequences of brain surgery on the nervous system, and he obtained the information in his brother, and the opportunity to do so. His application was successful. He married again and returned to Montreal. The young man with a Ph.D. in psychology, far from his family destiny and become a member of a medical group pioneering the treatment of neurological disorders.

Hebb's first wife was the main-

# The 1950s and 1960s: First Golden Age of Neural Networks

Perceptron (Rosenblatt's perceptron), 1957

## Perceptron (1957)

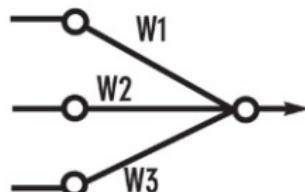


Original Perceptron

(From *Perceptrons* by M. L Minsky and S. Papert, 1969, Cambridge, MA: MIT Press. Copyright 1969 by MIT Press.



Frank Rosenblatt  
(1928-1971)



Simplified model:

# The 1970s: The Quiet Years

Marvin Minsky

- “Perceptron”, 1969.
- “Fatal flaw” of the perceptron; inability to solve non-linear problems.

# The 1970s: The Quiet Years

Kunihiro Fukushima

- Neocognitron

Tuomo Kohonen

- Self-Organizing Map (SOM) algorithm

# The 1980s: Renewed Enthusiasm

John Hopfield

- Hopfield Neural Network, 1982

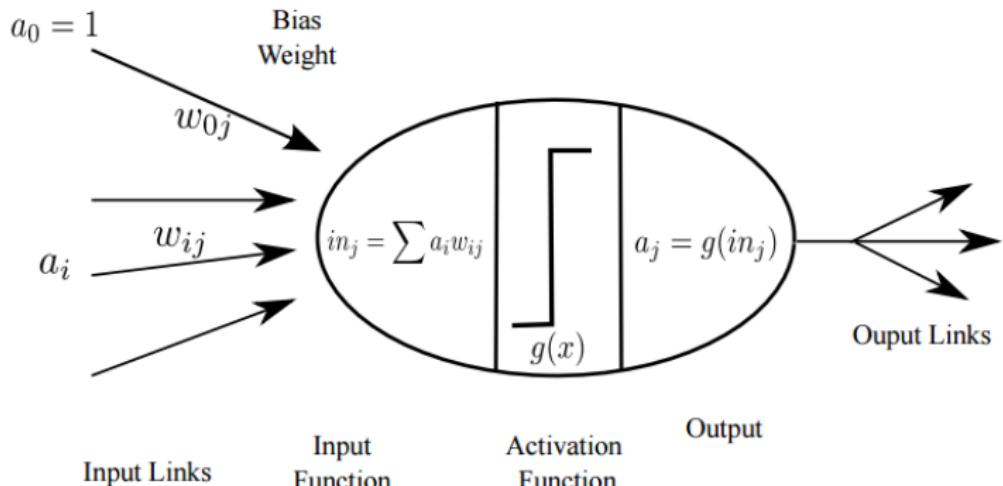
Geoffrey Hinton, etc.

- Boltzmann Machine
- Back-propagation (BP) algorithm

# **Neural Network**

---

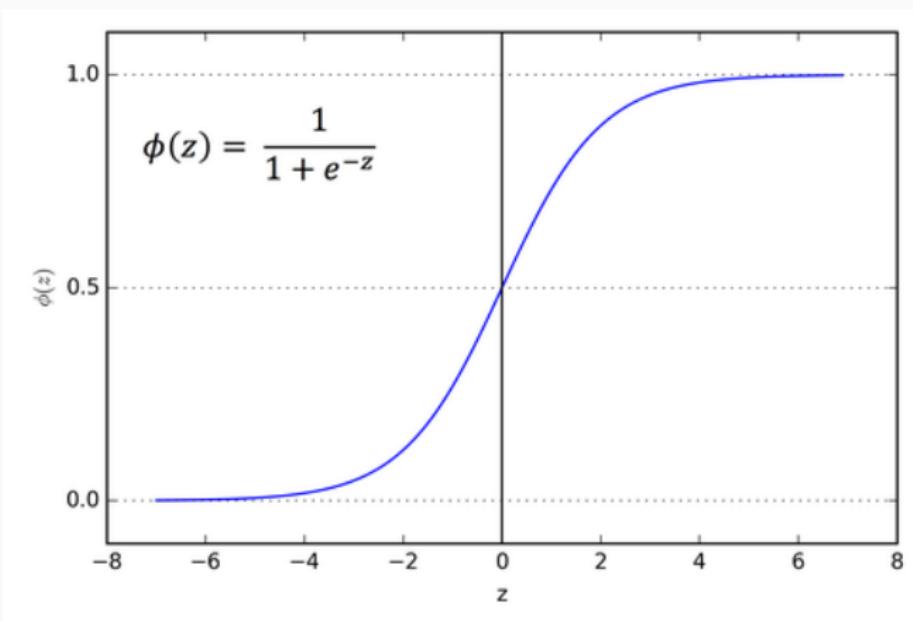
# A Neuron Model



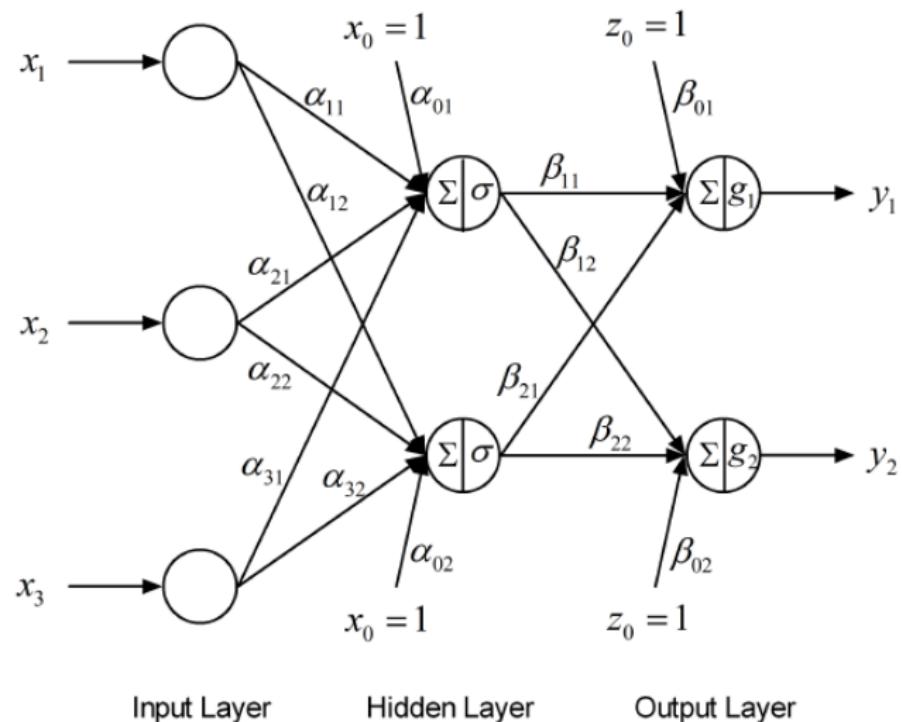
## A Neuron Model

- The weights of the neuron are the parameters of the model.
- The input is computed as a weighted sum of the inputs.
- The activation function determines the output and can be different functions.
- The output is obtained by applying the activation function to the input.
- How does the neuron work?

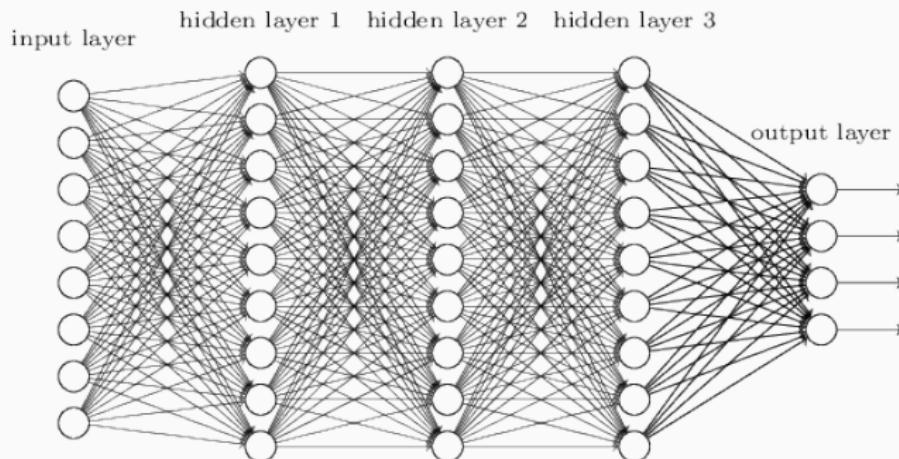
# A Neuron Model



# Single-layer Neural Network



# Multiple-layer Neural Network



## Delta Learning Rule

- Based on gradient descent algorithm.
- Only applicable for continuous activation function.
- The learning signal  $r$  is called delta and defined as:

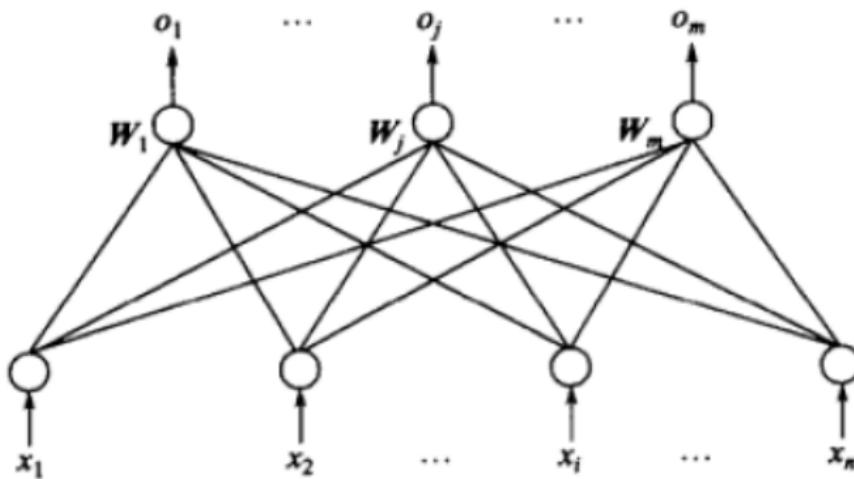
$$r = \{d_j - f(\mathbf{W}_j^T \mathbf{X})\}f'(\mathbf{W}_j^T \mathbf{X}) \approx (d_j - o_j)f'(\text{net}_j)$$

- The weights will be adjusted as:

$$\Delta \mathbf{W}_j = \eta(d_j - o_j)f'(\text{net}_j) \mathbf{X}$$

## Backpropagation Algorithm

- BP algorithm is a process of training a neural network
- Based on delta learning rule



## Backpropagation Algorithm

- For the output layer

$$\Delta\omega_{jk}^{h+1} = \eta\delta_k^{h+1}y_j^h = \eta(d_k - o_k)o_k(1 - o_k)y_j^h,$$

where  $j = 0, \dots, m_h$ ,  $k = 1, \dots, \ell$ .

- For the  $h$ th hidden layer

$$\Delta\omega_{ij}^h = \eta\delta_j^h y_i^{h-1} = \eta \left( \sum_{k=1}^l \delta_k^o \omega_{jk}^{h+1} \right) y_j^h (1 - y_j^h) y_i^{h-1}.$$

where  $i = 0, \dots, m_{h-1}$ ,  $j = 1, \dots, m_h$ .

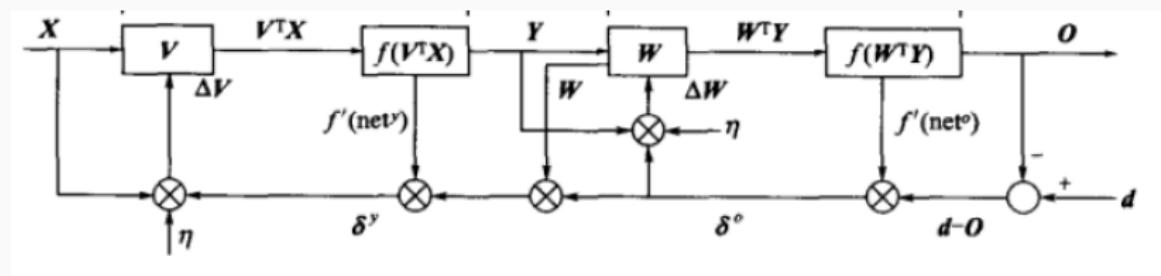
- The first hidden layer

$$\Delta\omega_{pq}^1 = \eta\delta_q^1 x_p = \eta \left( \sum_{r=1}^{m_2} \delta_r^2 \omega_{qr}^2 \right) y_q^1 (1 - y_q^1) x_p.$$

where  $p = 0, \dots, n$ ,  $k = 1, \dots, m_1$ .

# Backpropagation Algorithm

The flow of signal in BP algorithm:



## More Hidden Layers?

- The signal is getting weaker
- Non-convex problem
- Slow Convergence
- Weakness of Gradient Descent: Local Optima

Hinton, G. E., & Salakhutdinov, R. R. (2006). Reducing the dimensionality of data with neural networks. *Science*, 313(5786), 504-507.

## Reducing the Dimensionality of Data with Neural Networks

G. E. Hinton\* and R. R. Salakhutdinov

High-dimensional data can be converted to low-dimensional codes by training a multilayer neural network with a small central layer to reconstruct high-dimensional input vectors. Gradient descent can be used for fine-tuning the weights in such “autoencoder” networks, but this works well only if the initial weights are close to a good solution. We describe an effective way of initializing the weights that allows deep autoencoder networks to learn low-dimensional codes that work much better than principal components analysis as a tool to reduce the dimensionality of data.

Dimensionality reduction facilitates the classification, visualization, communication, and storage of high-dimensional data. A simple and widely used method is principal components analysis (PCA), which

finds the directions of greatest variance in the data set and represents each data point by its coordinates along each of these directions. We describe a nonlinear generalization of PCA that uses an adaptive, multilayer “encoder” network

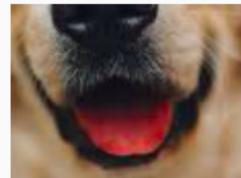
# **Convolutional Neural Network**

---



# Why CNN?

- Fully connected neural network can bring us too many parameters.
- Some patterns are much smaller than the whole image.



# Why CNN?

The same pattern appears in different regions.

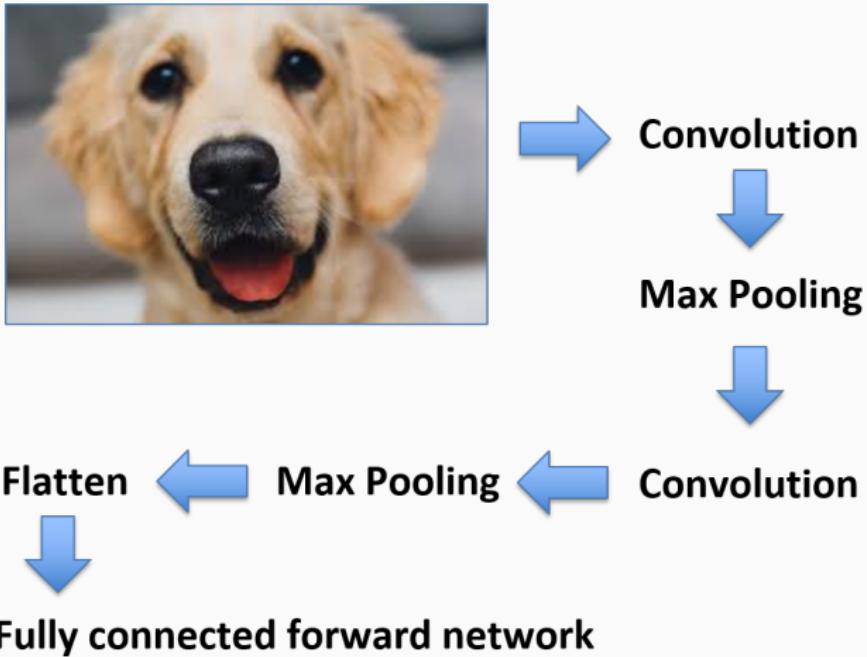


# Why CNN?

Subsampling the pixels will not change the object.



## Whole Structure

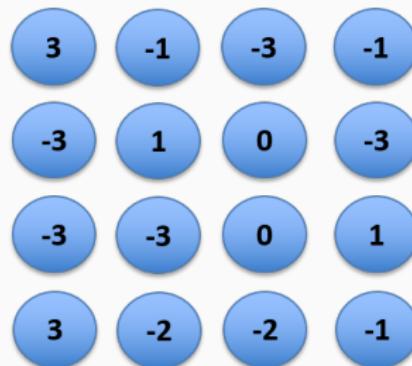


# How convolution works?

Stride = 1

1	0	0	0	0	1
0	1	0	0	1	0
0	0	1	1	0	0
1	0	0	0	1	0
0	1	0	0	1	0
0	0	1	0	1	0

1	-1	-1
-1	1	-1
-1	-1	1



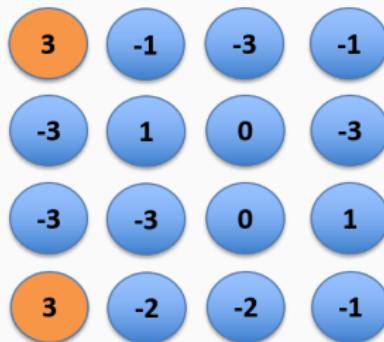
# How convolution works?

Stride = 1

1	0	0	0	0	1
0	1	0	0	1	0
0	0	1	1	0	0
1	0	0	0	1	0
0	1	0	0	1	0
0	0	1	0	1	0

1	-1	-1
-1	1	-1
-1	-1	1

Filter



# How convolution works?

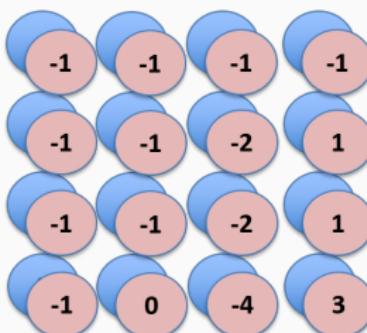
Stride = 1

1	0	0	0	0	1
0	1	0	0	1	0
0	0	1	1	0	0
1	0	0	0	1	0
0	1	0	0	1	0
0	0	1	0	1	0

1	-1	-1
-1	1	-1
-1	-1	1

-1	1	-1
-1	1	-1
-1	1	-1

2nd  
filter

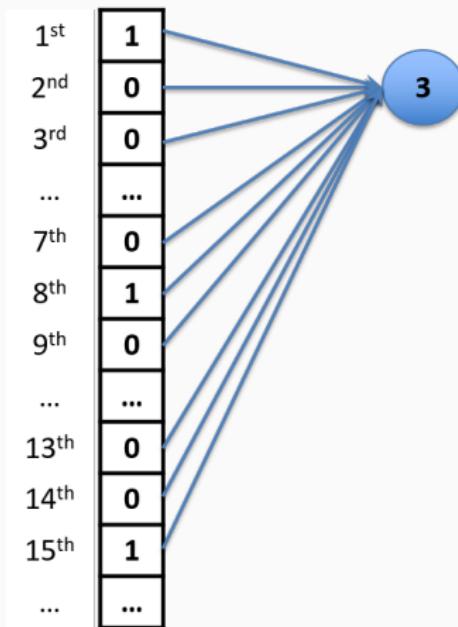


Feature map

# Connection between CNN and Neural Network

1	-1	-1
-1	1	-1
-1	-1	1

1	0	0	0	0	1
0	1	0	0	1	0
0	0	1	1	0	0
1	0	0	0	1	0
0	1	0	0	1	0
0	0	1	0	1	0

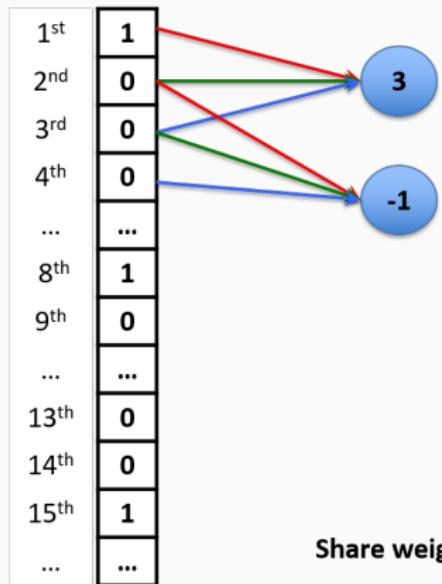


# Connection between CNN and Neural Network

1	-1	-1
-1	1	-1
-1	-1	1

1	0	0	0	0	1
0	1	0	0	1	0
0	0	1	1	0	0
1	0	0	0	1	0
0	1	0	0	1	0
0	0	1	0	1	0



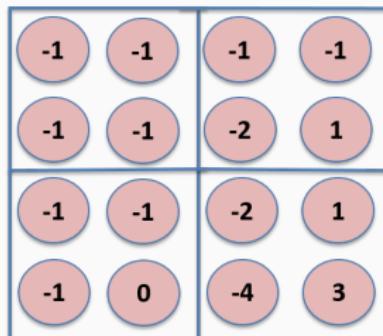
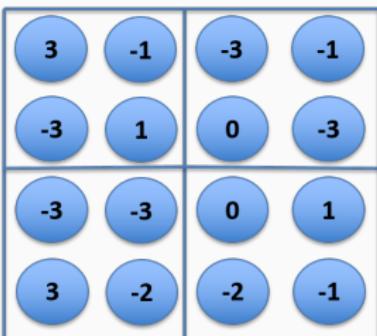
# Max pooling

1	-1	-1
-1	1	-1
-1	-1	1

Filter 1

-1	1	-1
-1	1	-1
-1	1	-1

Filter 2



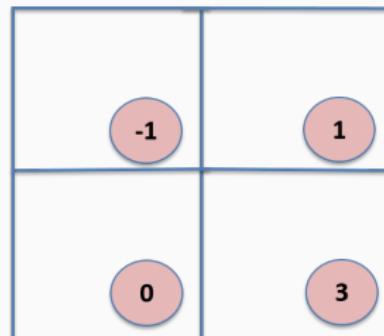
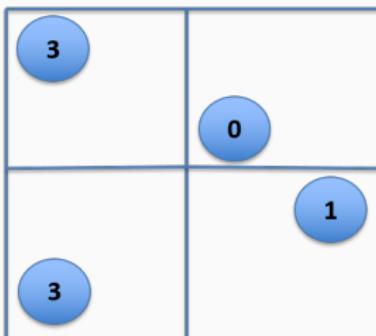
# Max pooling

1	-1	-1
-1	1	-1
-1	-1	1

Filter 1

-1	1	-1
-1	1	-1
-1	1	-1

Filter 2



1	0	0	0	0	1
0	1	0	0	1	0
0	0	1	1	0	0
1	0	0	0	1	0
0	1	0	0	1	0
0	0	1	0	1	0

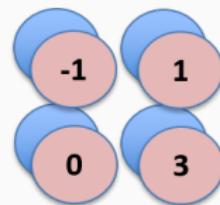


**Convolution**



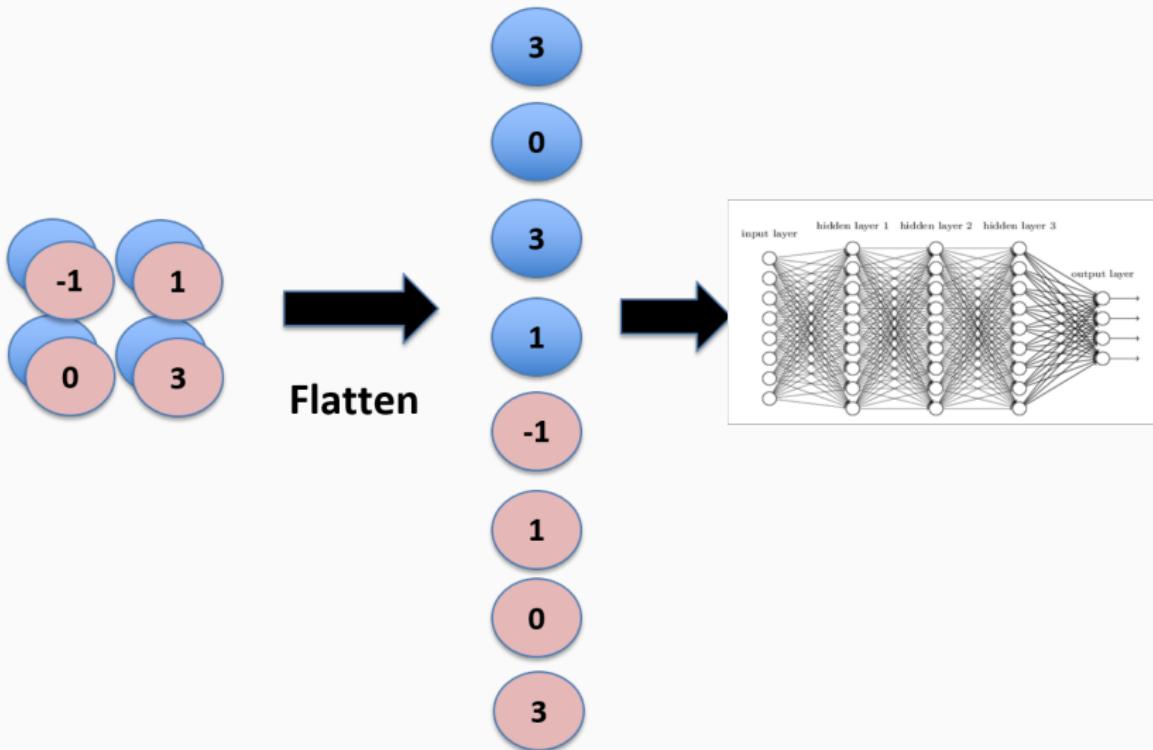
**Max Pooling**

Smaller image 2x2



Each filter is a channel

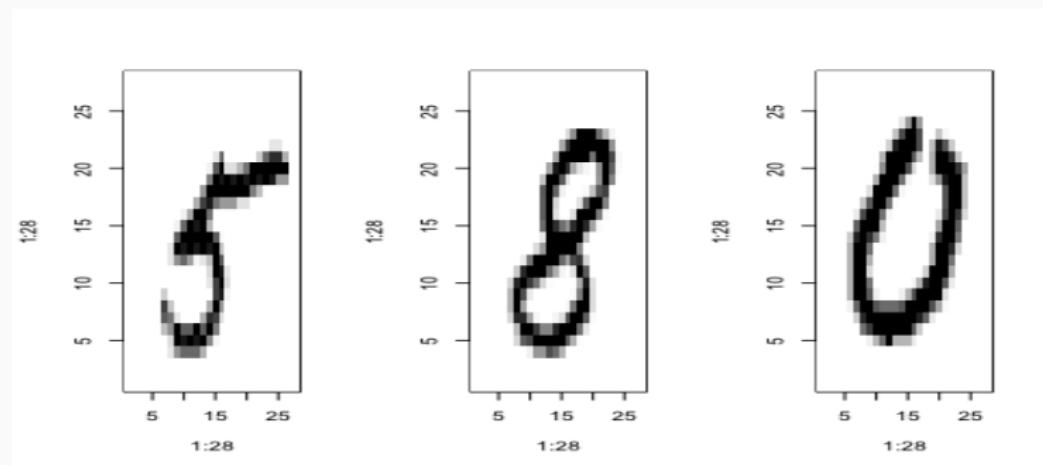
# Flatten



## Example of CNN: MNIST

The MNIST dataset contains images of handwritten digits like below. Each image is 28 pixels by 28 pixels. It also includes labels for each image, telling us which digit it is. Next, we are going to train a model to look at images and predict which digits they are.

*First Convolutional Network Demo from 1993*



# TensorFlow

---



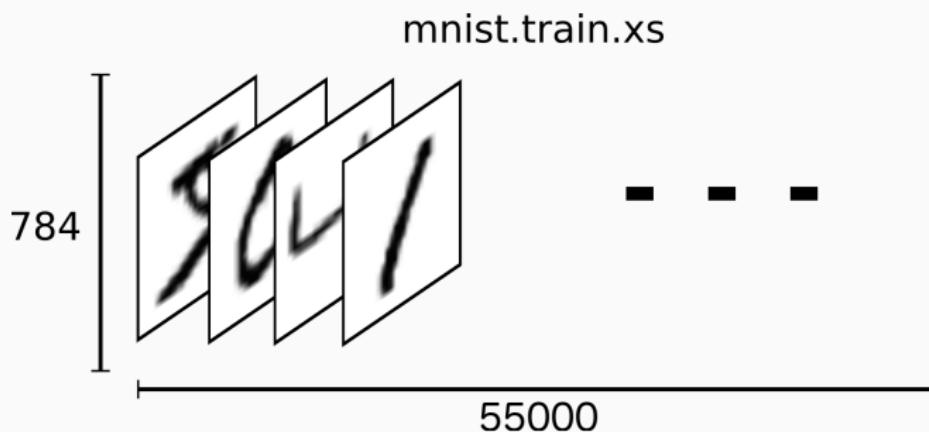
TensorFlow

## About TensorFlow:

- Tensorflow is an open source software library for numerical computation using data flow graphs. Nodes in the graph represent mathematical operations, which the graph edges represent the multidimensional data arrays (tensors) communicated between them.
- It allows you to deploy computation to one or more CPUs or GPUs in a desktop, server, or mobile device with a single API. The Tensorflow package provides access to the complete Tensorflow API from within R.

# MNIST

`mnist$train$image` is a tensor (an n-dimensional array) with shape (55000L, 784L).

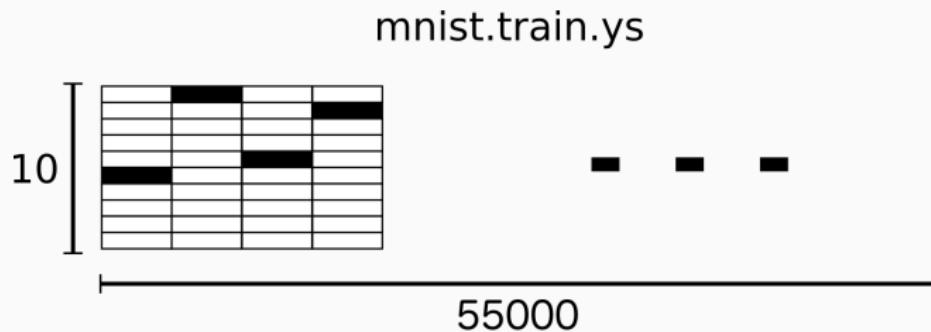


# MNIST

The label is “one-hot vectors”.

For example, 3 would be  $[0, 0, 0, 1, 0, 0, 0, 0, 0, 0]$ .

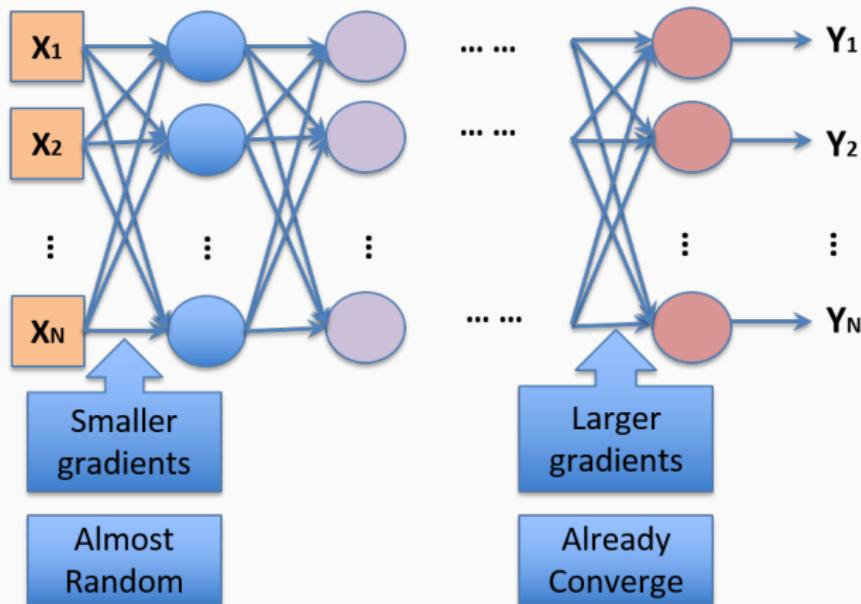
`mnist$train$labels` is a tensor with shape `(55000L, 10L)`.



# MNIST

Why do we have 2 layers, not more?

Vanishing Gradient Problem  $\rightarrow$  Deeper does not usually imply better.

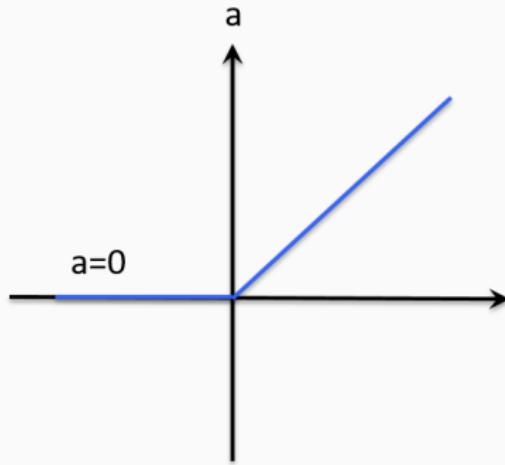


# MNIST

Activation function: Rectified Linear Unit (ReLU)

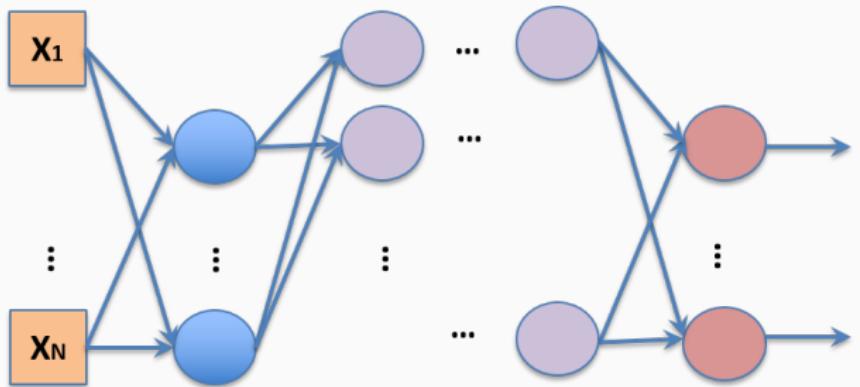
Reason:

- Infinite sigmoid with different biases
- Able to handle vanishing gradient problem
- Fast to compute
- Biological reason



# MNIST

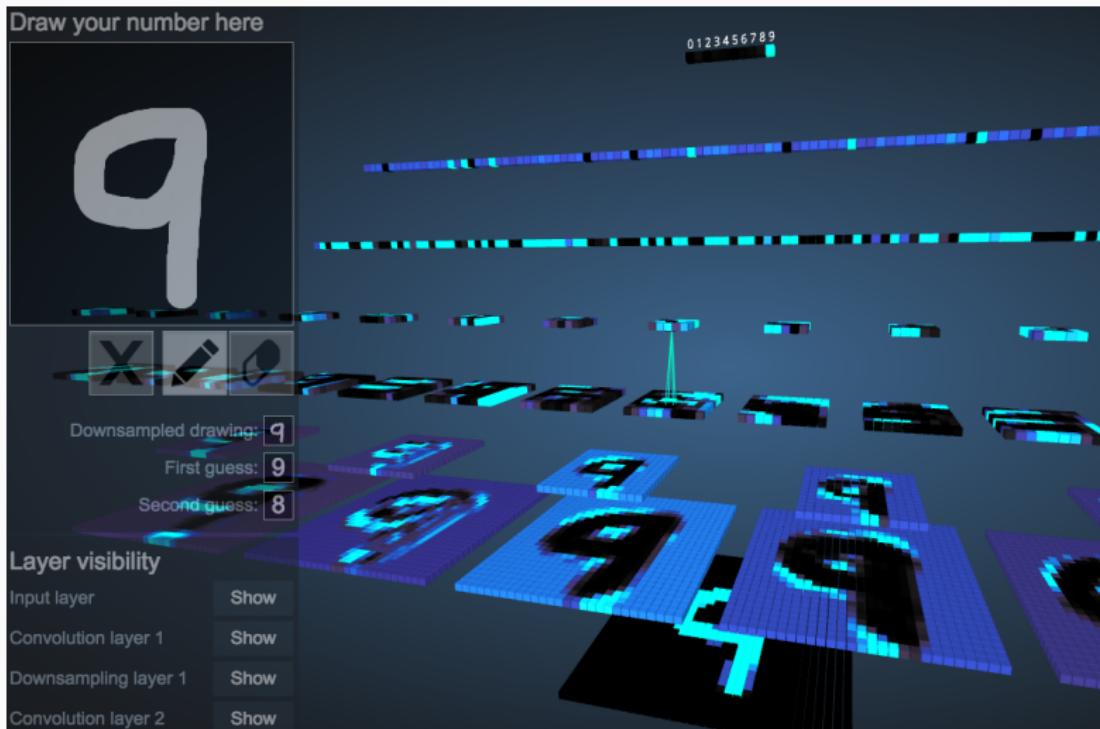
Dropout: when you do training, each time before updating the parameters, each neuron has p% to dropout.



Thinner  
structure

# CNN Visualization: Example of MNIST

*3D convolutional network visualization*



# Style Transfer

Style Transfer, A.K.A. Deep Style

Raymond Wong



Content

Snedecor Hall



Content → CNN → Style

# Style Transfer

Style Transfer, A.K.A. Deep Style

