

ALBANY STATE UNIVERSITY, GA

WEB DESIGN AND DEVELOPMENT USING AI

CHAPTER THREE -- BRIEF OF MACHINE LEARNING

BY

Wanjun Hu

Department Of Math, Cs & Physics

Albany State University

504 College Dr, Albany Ga

Wanjun.Hu@asurams.edu

<https://www.backupspirit.com/camp>

Table of Contents

Section 3.1 What is a linear regression	3
Section 3.2 Cost function	6
Section 3.3 Example	7
Section 3.4 Multivariant regression	8
Section 3.5 Artificial neural network to represent machine learning models	9
Section 3.6 Deep Learning neural networks	11
Section 3.7 Convolution neural network	13

CHAPTER THREE — BRIEF OF MACHINE LEARNING

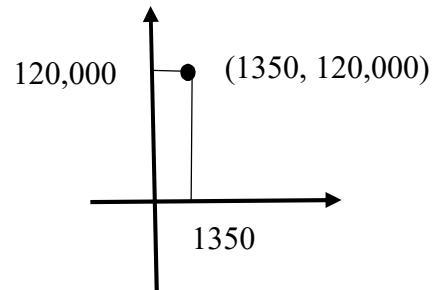
Machine learning and AI are based on the very basic operations. This is just like any science. For instance, mathematics is based on the very basic operations: plus, minus, multiplication and divisions. From those basic operations, we build Algebra 1, Algebra 2, Precalculus, Calculus, and a lot of other modern mathematics subjects. In a similar situation, modern computer is constructed based on the so-called gates. There are six basic gates: AND-gate, OR-gate, XOR-gate, NOT-gate, NAND-gate, NOR-gate. From there, we build adders, multiplexers, CPU, and others, and finally modern computers.

No surprise, Machine Learning and AI are based on the very basic concepts, i.e., the linear regression.

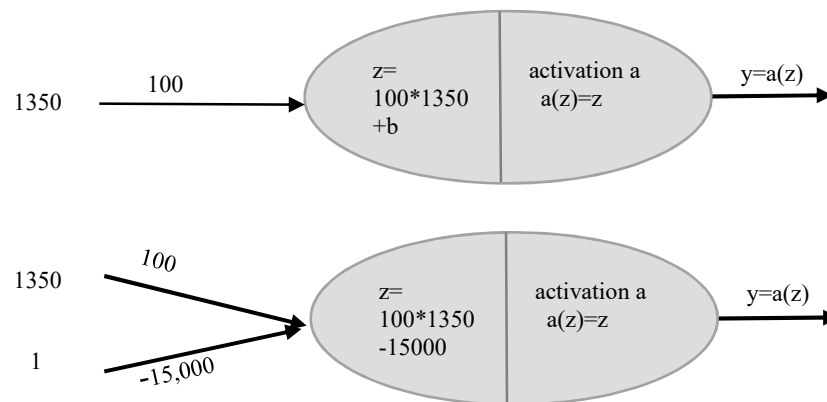
Section 3.1 What is a linear regression

Using a simple sentence, one can say that linear regression is the method that tries to use linear function to describe the world around us. For instance, you sell your house with 1,350 square feet for \$120,000. The square feet of your house are the input, written as x , and selling price is the output or y . You can describe the relationship in several ways

1. As a number pair (1,350, \$120,000)
2. As a function $\$120,000 = 1,350 * \$100 - \$15,000$, or
 $\$120,000 = 1,350 * \$90 - \$1,500$, or any other ways in a generic form
 $\$120,000 = 1,350 * w + b$, where w and b are real numbers
3. As a point in a coordinate system



4. As a network (in either one of the following graphs)



The activation in above network can be anything function. Here we just let $a(z)=z$. In machine learning, an activation can be a function such as Sigmoid, ReLu, Tanh(x), etc.

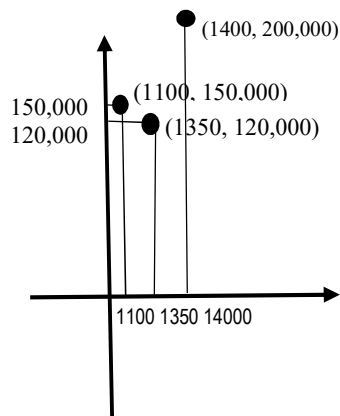
What if there are several houses sold recently? For instance,

Your house: 1350 sqft → \$120,000

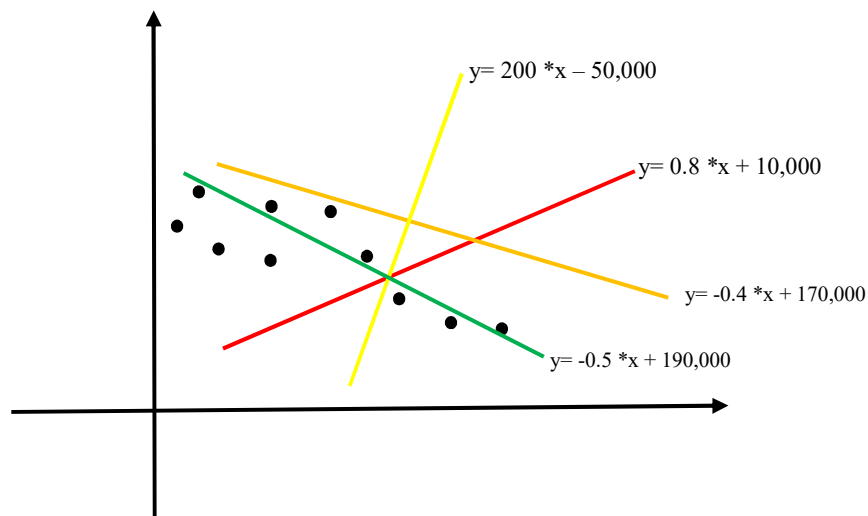
Another house: 1400 sqft → \$200,000

Another house: 1100sqft → \$150,000.

You can see that one linear function can not describe above three sales. If using the coordinate system, there are three points that are not on the same line. A linear function is a line and vice versa.



In general, if you have a collection of number pairs of square feet and sale price, you can not use one linear function to describe accurately all of the sales. If you view each sale as a point (sqft, sale price), you will see the following picture



Each point represents one sale (sqft, sale price). There are 10 points, which means there are 10 sales. Each line represents one way to describe the relationship between the input “sqft” and

the output “price”. Looks like, we cannot decide which line or linear function to use. Is it the red line? Is it the orange line, or the green line? Which one?

Section 3.2 Cost function

Which linear function in above picture shall we choose? The gold, or the red, or the orange or the green?

How to decide?

To answer that question, one of the greatest mathematicians, Carl Gauss, invented a method, called **least square of errors**. Roughly, Gauss calculates the so-called square of all errors (losses), He then found the one that assumes the smallest sum.

In general, a linear function can be written as $y = w * x + b$, where w , and b are two coefficients to be decided. Different values of w and b decide different linear functions. To choose the “best” one, we shall invent the meaning of being best. That is done by designing a **cost function**.

Loss (error): For a house sale (1350, 120,000), we can calculate the expected output $y_{1345} = w * 1350 + b$. However, the actual sale price is \$120,000. There will have a difference between y_{1350} and 120,000, which is $y_{1350} - 120,000$. That is called the loss (error).

Cost function: For a collection of house sales, there will have a collection of losses. A cost function is calculated based on all losses. There are some of the following cost functions for linear regressions.

1. Mean squared error (MSE, a modified Gauss least square error method)
2. Distance-based error
3. Root Mean squared error (RMSE)
4. Cross-entropy function (using log functions)

5. Mean absolute error (MAE)
6. Kullback-Liebler (KL) Divergence
7. Hinge Loss

Section 3.3 Example

We will use the three sales discussed above to show what a cost function looks like. The three house sales are (1100, \$150,000) , (1350, \$120,000) and (1400, \$200,000).

Our linear function to be decided is

$$y = m * x + b$$

House sale	Expected price	Loss
(1100, 150,000)	$y_{1100} = m * 1100 + b$	$Loss_{1100} = y_{1100} - 150,000 = m * 1100 + b - 150,000$
(1350, 120,000)	$y_{1350} = m * 1350 + b$	$Loss_{1350} = y_{1350} - 120,000 = m * 1350 + b - 120,000$
(1400, 200,000)	$y_{1400} = m * 1400 + b$	$Loss_{1400} = y_{1400} - 200,000 = m * 1400 + b - 200,000$

The cost function using the mean square error (MSE) method will be

$$\begin{aligned} cost(m, b) &= \frac{(loss_{1100})^2 + (loss_{1350})^2 + (loss_{1400})^2}{3} \\ &= \frac{(m * 1100 + b - 150,000)^2 + (m * 1350 + b - 120,000)^2 + (m * 1400 + b - 200,000)^2}{3} \end{aligned}$$

Gradient Descent: After calculating the cost function, Machine Learning will use the so-called gradient descent method (invented by the great Newton) to find the value of m and b, so that $cost(m, b)$ reaches a minimum. The following is a picture shows how it works. The image is from the internet using Bing search. We will not go into details about that.



Section 3.4 Multivariate regression

When selling a house, people do not just look at the square feet. If the house is on the water front, the price goes up. If the house is on a golf course, the price goes up. If the house is close to a prestigious university, the price goes up. If the house is close to White House, the price goes up.

In other words, there are other factors that affect the sale price. Now, let's look at the three sales in more details.

House sale	Sqft	Water front	Golf course	Near White house	Sale price
1	1100	No	Yes	No	150,000
2	1350	No	No	No	120,000
3	1400	Yes	Yes	No	200,000

We will use

x_1 for sqft

x_2 for water front, $x_2 = 1$ means yes; $x_2 = 0$ means no

x_3 for golf course, $x_3 = 1$ means yes; $x_3 = 0$ means no

x_4 for nearwhitehouse, $x_4 = 1$ means yes; $x_4 = 0$ means no.

So, above three sales can be described by number tuples

(1100, 0, 1, 0, 150,000)

(1350, 0, 0, 0, 120,000)

(1400, 1, 1, 0, 200, 000)

We can write the sale price as a linear function of the four inputs x_1, x_2, x_3, x_4 , which is a

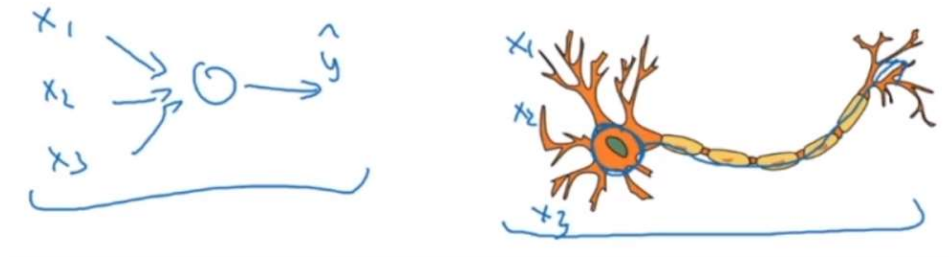
multivariate linear function.

$$y = m_1 * x_1 + m_2 * x_2 + m_3 * x_3 + m_4 * x_4 + b$$

Here, m_1, m_2, m_3, m_4, b are the coefficients of above function.

Section 3.5 Artificial Neural Networks to Represent Machine Learning

Models



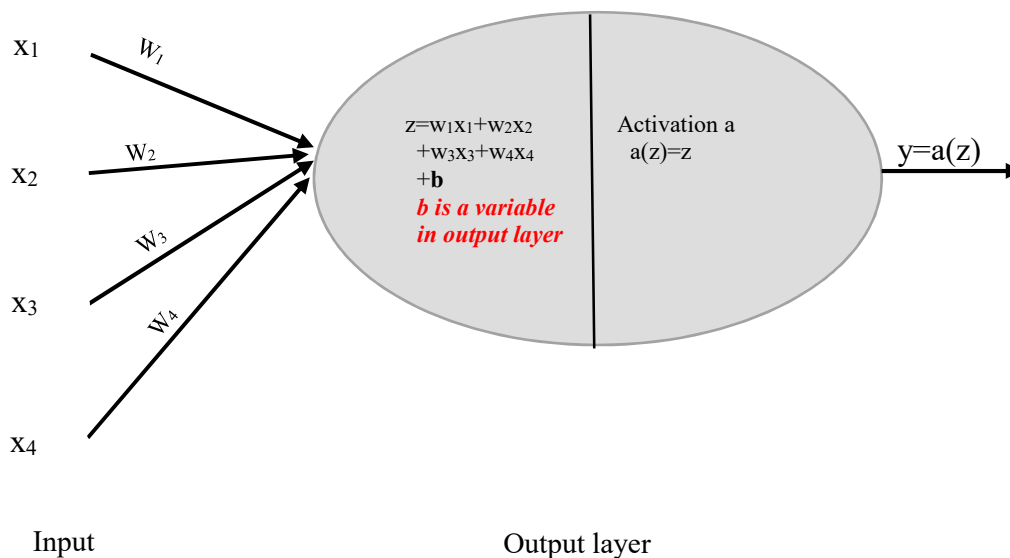
From Coursera, Andrew Ng

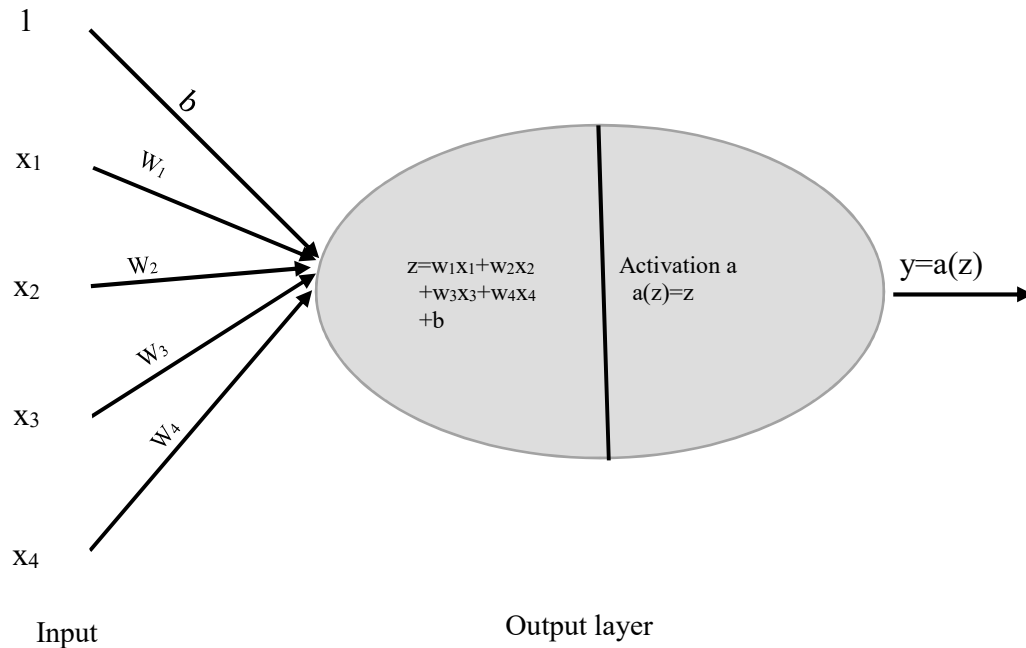
Above picture shows a real neuron of our brain and an artificial neuron.

We can represent the multivariate linear function (also called a **machine learning model**),

$$y = m_1 * x_1 + m_2 * x_2 + m_3 * x_3 + m_4 * x_4 + b$$

as a network. It is a **one layer (the output layer) neural network**. There are two ways to handle the parameter b as shown in the two graphs below.





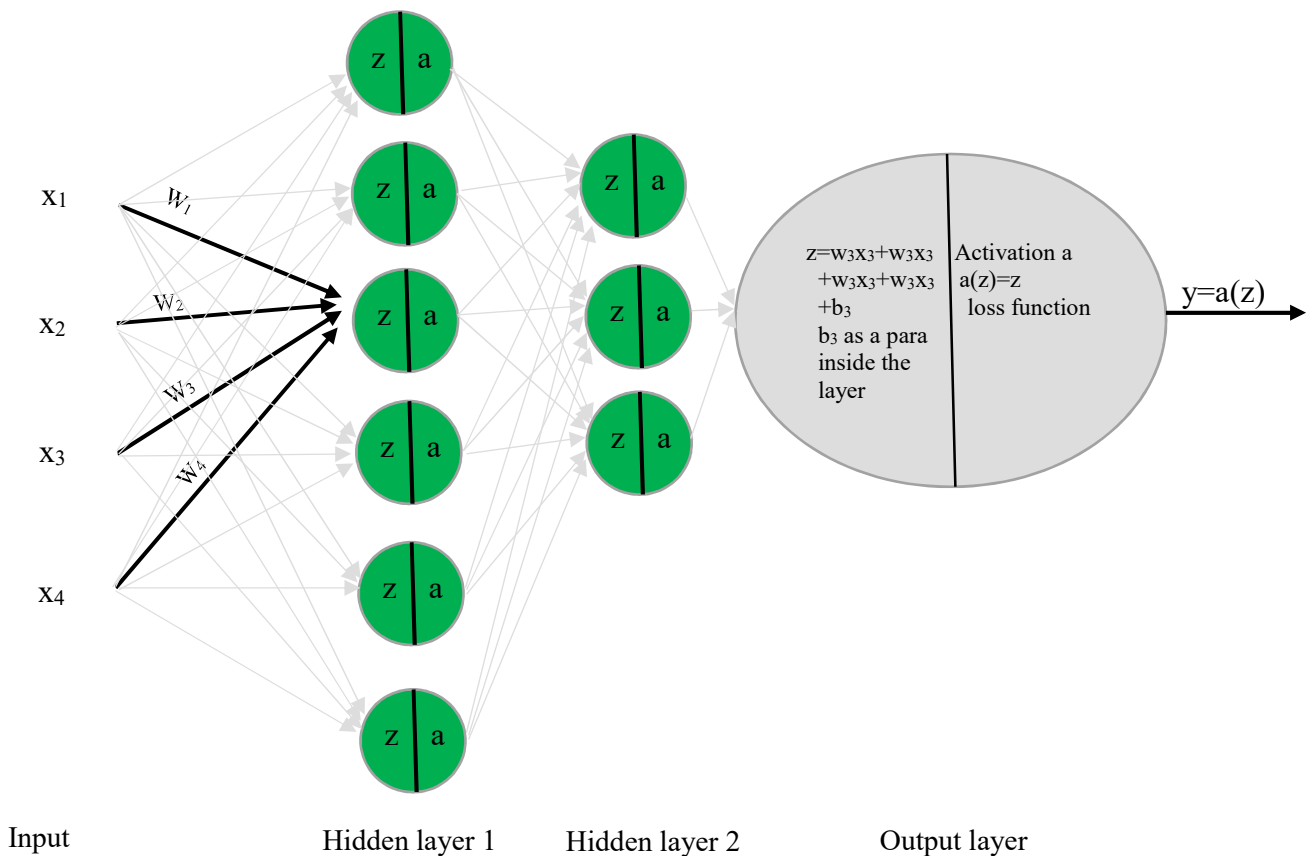
Input or input layer?

Usually, the input(s) are not considered as one layer of the neural network, because in human brain, the inputs to human neurons come from eyes, ears, hands, toes, skin and other nerve system around our body. Hence the input(s) are not part of the human brain. However, people do frequently call the input(s) as input layer when talking about artificial neural network. When that happens, please consider it as a convenient way to describe the input(s). The input(s) are still not counted as one layer.

Section 3.6 Deep Learning Neural Networks

A deep learning neural network is a standard neural network plus 1 or more hidden layers.

It looks like the following **three layers neural network (two hidden layers and one output layer)**.



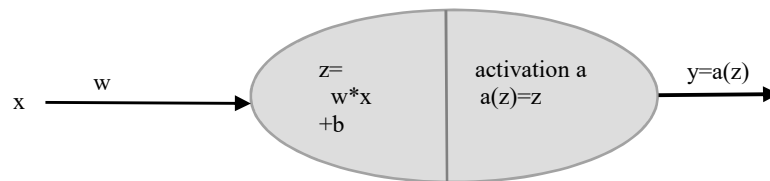
The layers marked in green color are called hidden layers. Inputs are fully connected to the first hidden layers and a hidden layer is fully connected to the next hidden layers. Finally, the last hidden layer connects to the output.

There are several types of Deep Learning Models. Yes, LLMs (large language models) are much more complicate machine learning models. For our camp, we will train two simple deep learning models. We will discuss below.

A sequence model is a deep learning neural network that processes sequential data usually in order such as time, e.g., video frames, audio sequence, language translation.

A sequential model is a deep learning neural network that contains layers connected from left to right, but no return (or cycles). Above graph of deep learning neural network is a sequential model. It may be also used as a sequence model when the input data has an order. Non-sequential models are more advanced, e.g., LLMs. Such model may have layers where a node in layer 5, for example, is connected back to a node in layer 2.

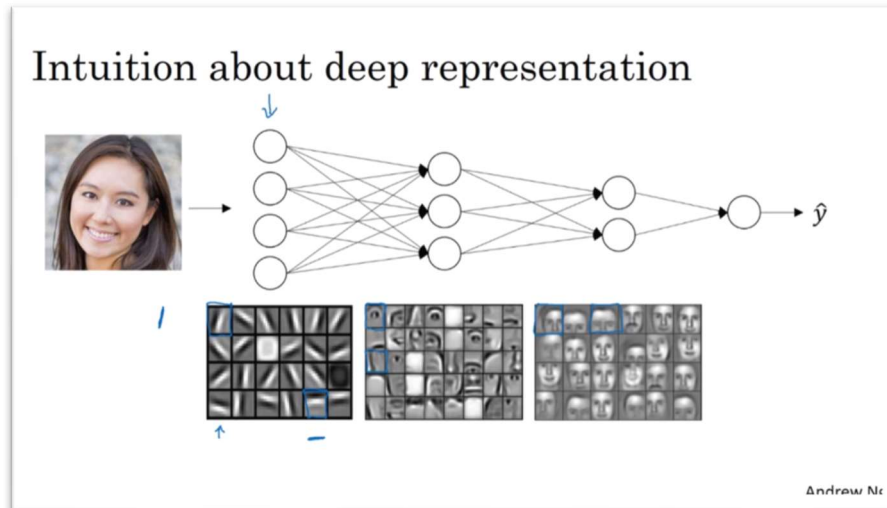
In our camp, we will train the following *sequential model* $y = wx + b$ using a dataset from google.



Section 3.7 Convolution Neural Networks

We will also train a convolutional neural network that can recognize hand-written digits.

A convolution model is essential for learning from images.



Question: What is a convolution?

Answer: It contains two types of layer

- a. Convolution and pooling layer
 1. Convolution to detect edge, patterns
 2. Pooling to down sample the size of data for fast computations
- b. Fully connected layers that flats multi-dimensional dataset to one dimensional so that it can connect fully to the last layer, i.e., the output layer.

Convolution and pooling layer -- part 1

– convolution to do edge detection using filters (also called kernels)

Vertical edge detection

3	0	1	2	7	4
1	5	8	9	3	1
2	7	2	5	1	3
0	1	3	1	7	8
4	2	1	6	2	8
2	4	5	2	3	9

6x6

"convolution"

1	0	-1
1	0	-1
1	0	-1

3x3 filter

*

4x4

=

Andrew Ng

Vertical edge detection

10	10	10	0	0	0
10	10	10	0	0	0
10	10	10	0	0	0
10	10	10	0	0	0
10	10	10	0	0	0
10	10	10	0	0	0

6x6

1	0	-1
1	0	-1
1	0	-1


3x3


*


0	30	30	0
0	30	30	0
0	30	30	0
0	30	30	0

4x4

=







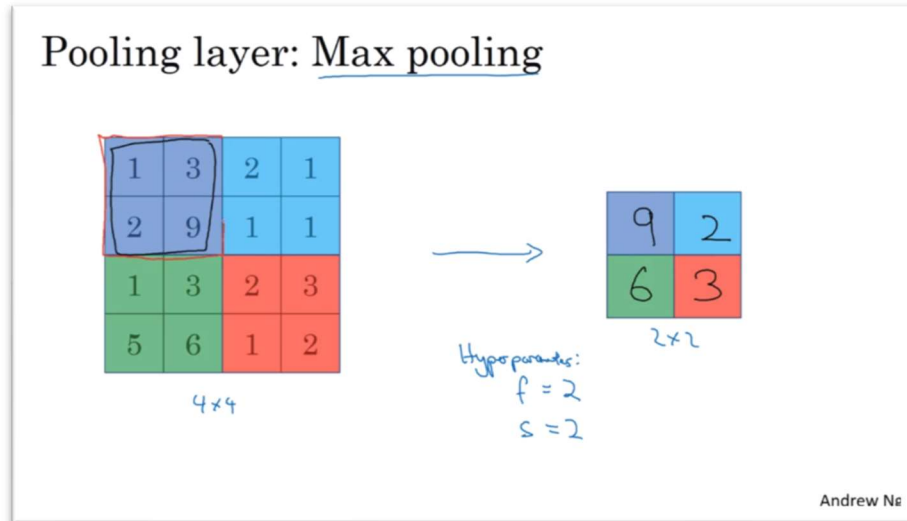
Andrew Ng

Usually, we apply other filters to detect edges at different angles.

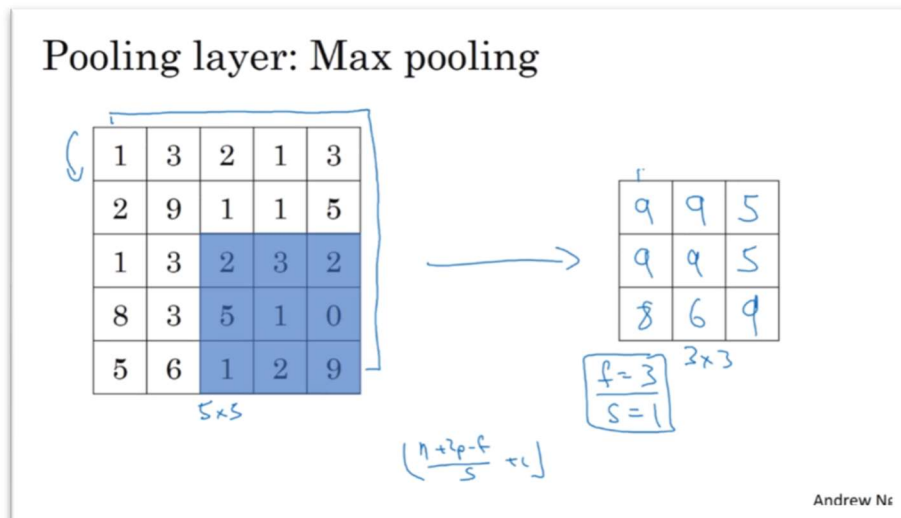
Convolution and pooling layer -- part 2

--Pooling to reduce the size of dataset for fast computation

Max pooling with filter=2, stride=2



Max pooling with filter=3, stride=1



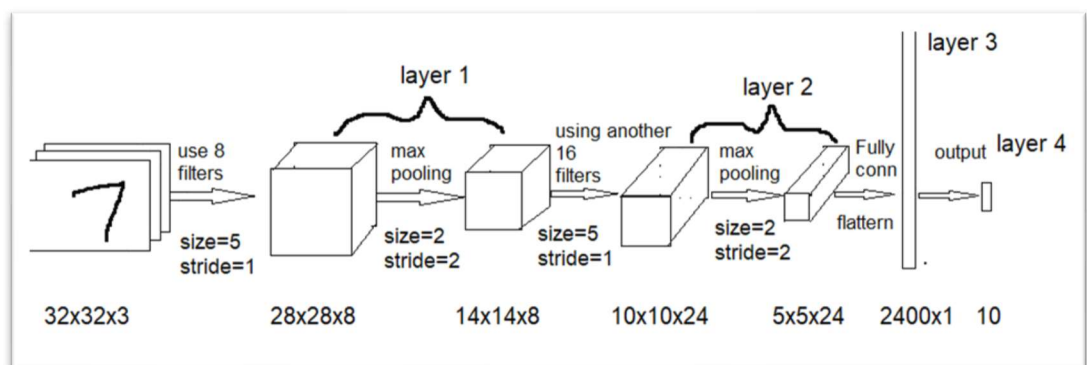
Fully connected Layer

Flat a multi-dimensional dataset to 1 dimensional dataset, so that it can be fully connected to the last layer, the output layer, where predictions or classifications are made.

In the graph below, it is the layer 3.

Putting together

The follow graph shows a simple convolutional neural network (CNN) that can recognize hand written digits. It is a 4-layer CNN.



We will train above CNN model using a dataset from Google.