Deep Learning: Basics and Intuition

Teeradaj Racharak (เอ็กซ์)
r.teeradaj@gmail.com

Introduction

- Deep learning involves the training of a so-called neural networks.
- In this course, we'll study how neural networks (including their history) and understand basic neural network learning algorithms.
- After that, we'll cover some of the modern neural network architectures that attract people's interest nowadays.

Some History about Brain and Computation

Aristotle
Greek Philosopher

Santiago Ramón y Cajal
Nobel prize in Physiology (1906)

ARISTOTLE'S RADIATOR

by DWAYNE GODWIN & JORGE CHAM

The Brain is Neural Circuit

Santiago Ramón y Cajal
Spanish Neuroscientist and Pathologist
(Received Nobel prize in 1906 - joint work with Golgi)

- A neuron is a discrete functional unit of the nervous system
- Neurons communicate with each other via specialized junctions, or spaces, between cells — neural circuit !

Brain's Neurons *vs.* Computational Neurons

William James

American Philosopher and Psychologist

With a greater understanding of the basic elements of the brain, efforts were made to describe how basic neurons could result in overt behaviors, to which William James was a prominent theoretical contributor.

Some History about Neural Circuits and Logic

Warren Sturgis McCulloch
American Neurophysiologist

Walter Pitts
American Logician

Being Homeless and Interdisciplinary Research

Neurophysiologist

Logician

McCulloch arrived in early 1942 to the University of Chicago and invited Pitts, who was homeless, to live with his family.

In the evenings, McCulloch and Pitts collaborated. Pitts was familiar with the work of Leibniz on computing. They considered the question of whether the nervous system is a kind of universal computing device as described by Leibniz.

This led to their 1943 seminal neural networks paper *i.e.*
A Logical Calculus of Ideas Immanent in Nervous Activity.

Impact

Neurophysiologist
(1899 - 1969)

Logician
(1923 - 1969)

The bulletin of mathematical biophysics
December 1943, Volume 5, Issue 4, pp 115–133 | Cite as

A logical calculus of the ideas immanent in nervous activity

Neural networks and Logic

Time

Memory

Threshold Logic and Learning

State Machines

Neuron's Definition of McCulloch and Pitts

- The McCulloch-Pitts neuron worked by inputting either 1 or 0 (where 1 represents true and 0 otherwise).
- Likewise, the threshold given as a real number was used to output either 1 or 0 if the threshold was met or exceeded.

Input x_1	**Input x_2**	**Output**
0	0	0
0	1	0
1	0	0
1	1	1

($\wedge$, threshold = 2.0)

Input x_1	**Input x_2**	**Output**
0	0	0
0	1	1
1	0	1
1	1	1

($\vee$, threshold = 1.0)

Revolutionizing Neurons

Donald O. Hebb
Canadian Psychologist

In 1949, Donald Hebb would help to revolutionize the way that artificial neurons were perceived.

He stated in his book (titled 'The Organization of Behavior') that:

"when two neurons fire together, the connection between the neurons is strengthened"

- The original statement was as follows:

"When an axon of cell A is near enough to excite a cell B and repeatedly or persistently takes part in firing it, some growth process or metabolic change takes place in one or both cells such that A's efficiency, as one of the cells firing B, is increased"

195X - Rosenblatt (Perceptron)

Some may say that Perceptron is the first neural network, but some say that it is McCulloch-Pitts neuron

- Rosenblatt developed the first neural network model called ‘Perceptron’ using the McCulloch-Pitts neuron and the finding of Hebb.
- The activation function used by McCulloch & Pitts was the threshold step function; however, others (*e.g.* sigmoid, piecewise linear, Gaussian) may be used.
- Many of neural network today stem from Rosenblatt’s perceptron.

Brief History of Deep Learning

- **1943 - McCulloch and Pitts** published the seminal paper titled ‘A Logical Calculus of Ideas Immanent in Nervous Activity’.
- **1950 - Alan Turing** published a paper ‘Computing Machinery and Intelligence’ while he was working at the University of Manchester. In his paper, he left a question ‘Can machines think?’.
- **1957 - Rosenblatt** developed the perceptron.
- **195X - Muroga** introduced Linear-Threshold gate (LT gate).
- **1962 - Hubel and Wiesel** published a paper ‘Receptive fields, binocular interaction and functional architecture in the cat's visual cortex’.

Brief History of Deep Learning

- **1974 - Werbos** introduced ‘learning through backpropagation’.
- **1980 - Fukushima** proposed a hierarchical multilayered neural network for handwritten character recognition. This work is also served as the inspiration for convolutional neural network.
- **1986 - Rumelhart, Hinton & Williams** rediscovered backpropagation and published a paper ‘Learning Representations by Backpropagating errors’.
- **1995 - Lecun and Bengio** published a paper ‘Convolutional networks for images, speech, and time series’.
- **1997 - IBM** developed Deep Blue *i.e.* a chess-playing computer.
- **2011 - IBM** developed Watson *i.e.* a question-answering system.
- **2012 - Krizhevsky and Hinton** developed AlexNet and won ImageNet (ILSVRC) competition with substantially higher accuracy
- **2016 - Google Deepmind** developed AlphaGo.

Why should we use Neural Network?

Solving Non-linearly Separable Problems

Non-linear classification

$$g(\theta_0 + \theta_1 x_1 + \theta_2 x_2 + \theta_3 x_1 x_2 + \theta_4 x_1^2 x_2 + \theta_5 x_1^3 x_2 + \theta_6 x_1 x_2^2 + \ldots)$$

This is possible; but, is this a good solution?
If not, why is this not a good one?

Solving Non-linearly Separable Problems

Non-linear classification

$$\left.\begin{array}{l} x_1 = \text{size} \\ x_2 = \text{\#bedrooms} \\ x_3 = \text{\#floor} \\ x_4 = \text{age} \\ \vdots \\ x_{100} \end{array}\right\} \text{100 features}$$

If we want to model a polynomial of degree 2, how many terms do we need in the equation?

$$\left.\begin{array}{l} x_1^2, x_1x_2, x_1x_3, x_1x_4, \ldots, x_1x_{100} \\ x_2^2, x_2x_3, x_2x_4, \ldots, x_2x_{100} \\ \vdots \end{array}\right\} \approx 5000 \text{ terms}$$

This number grows in $\mathcal{O}(n^2)$

Solving Non-linearly Separable Problems

Non-linear classification

$$\left.\begin{array}{l} x_1 = \text{size} \\ x_2 = \text{\#bedrooms} \\ x_3 = \text{\#floor} \\ x_4 = \text{age} \\ \vdots \\ x_{100} \end{array}\right\} \text{100 features}$$

If we want to model a polynomial of degree 2, how many terms do we need in the equation?

What if $x_1x_2x_3, x_1^2x_3, x_{10}x_{11}x_{17}, \ldots$?

$\approx$ 170,000 features *i.e.* $\mathcal{O}(n^3)$

Problems in Computer Vision

Humans see this:

brightness of a pixel in the image
(*aka.* intensity value)

But the camera sees this:

194	210	201	212	199	213	215	195	178	158	182	209
180	189	190	221	209	205	191	167	147	115	129	163
114	126	140	188	176	165	152	140	170	106	78	88
87	103	115	154	143	142	149	153	173	101	57	57
102	112	106	131	122	138	152	147	128	84	58	66
94	95	79	104	105	124	129	113	107	87	69	67
68	71	69	98	89	92	98	95	89	88	76	67
41	56	68	99	63	45	60	82	58	76	75	65
20	43	69	75	56	41	51	73	55	70	63	44
50	50	57	69	75	75	73	74	53	68	59	37
72	59	53	66	84	92	84	74	57	72	63	42
67	61	58	65	75	78	76	73	59	75	69	50

Question

- Suppose you are learning to recognize cars from 100 x 100 pixel images. Let the features be pixel intensity values. If you train logistic regression including all the quadratic terms ($x_i x_j$) as features, about how many features will you have?

(i) 5,000
(ii) 100,000
(iii) 50 million (5×10^7)
(iv) 5 billion (5×10^9)

Hint: $C(n, r) = \dfrac{n!}{r!(n - r!)}$

Car Detection

Label examples of ‘cars’

Label examples of ‘not cars’

What is this?

Car Detection

Car Detection

Learning Algorithm

Car Detection

Learning Algorithm

Q: What is the dimension of the feature space?

A: 50×50 pixel images

$\implies$ 2,500 pixels ($\therefore n = 2500$)

Quadratic features (*i.e* $x_i \times x_j$)

$\approx 3 \times 10^6$ features

Basics of Neural Network

Examples and Intuition

Many of neural network today stem from Rosenblatt's perceptron

Threshold Step

Sigmoid

Piecewise Linear

Gaussian

Examples and Intuition

1) Logical AND

Let $x_1, x_2 \in \{0,1\}$ Goal: $y = x_1$ **AND** x_2

Examples and Intuition

1) Logical AND

Let $x_1, x_2 \in \{0,1\}$ Goal: $y = x_1$ **AND** x_2

i.e. $h_\Theta(x) = g(-30 + 20x_1 + 20x_2)$

x_1	x_2	$h_\Theta(x)$
0	0	$g(-30) \approx 0$
0	1	$g(-10) \approx 0$
1	0	$g(-10) \approx 0$
1	1	$g(10) \approx 1$

Question

Suppose $x1$ and $x2$ are binary values (0 or 1). What boolean function does the network shown below compute?

Examples and Intuition

2) Logical NEGATION

x_1	$h_\Theta(x)$
0	$g(+10) \approx 1$
1	$g(-10) \approx 0$

$$i.e.\ h_\Theta(x) = g(+10 - 20x_1)$$

Question

Suppose that $x1$ and $x2$ are binary values (0 or 1).
Which of the following networks computes the boolean function
(**NOT** x_1) **AND** (**NOT** x_2) ?

Examples and Intuition

3) Logical XNOR

x_1, x_2 are binary (0 or 1)

Building XNOR

'Putting them together'

x_1 **AND** x_2

(**NOT** x_1) **AND** (**NOT** x_2)

x_1 **OR** x_2

x_1	x_2	$a_1^{(2)}$	$a_2^{(2)}$	$h_\Theta(x)$
0	0			
0	1			
1	0			
1	1			

Building XNOR

'Putting them together'

x_1 **AND** x_2

(**NOT** x_1) **AND** (**NOT** x_2)

x_1 **OR** x_2

x_1	x_2	$a_1^{(2)}$	$a_2^{(2)}$	$h_\Theta(x)$
0	0			
0	1			
1	0			
1	1			

Building XNOR

'Putting them together'

x_1 **AND** x_2

(**NOT** x_1) **AND** (**NOT** x_2)

x_1 **OR** x_2

x_1	x_2	$a_1^{(2)}$	$a_2^{(2)}$	$h_\Theta(x)$
0	0			
0	1			
1	0			
1	1			

Building XNOR

'Putting them together'

x_1 **AND** x_2

(**NOT** x_1) **AND** (**NOT** x_2)

x_1 **OR** x_2

x_1	x_2	$a_1^{(2)}$	$a_2^{(2)}$	$h_\Theta(x)$
0	0	0	1	
0	1	0	0	
1	0	0	0	
1	1	1	0	

Building XNOR

'Putting them together'

x_1 **AND** x_2

(**NOT** x_1) **AND** (**NOT** x_2)

x_1 **OR** x_2

input **hidden** **output**

x_1	x_2	$a_1^{(2)}$	$a_2^{(2)}$	$h_\Theta(x)$
0	0	0	1	1
0	1	0	0	0
1	0	0	0	0
1	1	1	0	1

k-classes Classification

Pedestrian

Car

Motorcycle

Trunk

Fully-connected Neural Network

$$h_\Theta(x) \in \mathbb{R}^4$$

$$h_\Theta(x) \approx \begin{bmatrix} 1 \\ 0 \\ 0 \\ 0 \end{bmatrix}$$

$$h_\Theta(x) \approx \begin{bmatrix} 0 \\ 1 \\ 0 \\ 0 \end{bmatrix}$$

$$h_\Theta(x) \approx \begin{bmatrix} 0 \\ 0 \\ 1 \\ 0 \end{bmatrix}$$

$$h_\Theta(x) \approx \begin{bmatrix} 0 \\ 0 \\ 0 \\ 1 \end{bmatrix}$$

k-classes Classification

$h_\Theta(x) \in \mathbb{R}^4$

Goal:

$h_\Theta(x) \approx \begin{bmatrix} 1 \\ 0 \\ 0 \\ 0 \end{bmatrix}$

$h_\Theta(x) \approx \begin{bmatrix} 0 \\ 1 \\ 0 \\ 0 \end{bmatrix}$

$h_\Theta(x) \approx \begin{bmatrix} 0 \\ 0 \\ 1 \\ 0 \end{bmatrix}$

$h_\Theta(x) \approx \begin{bmatrix} 0 \\ 0 \\ 0 \\ 1 \end{bmatrix}$

Training set:

$$(x^{(1)}, y^{(1)}), (x^{(2)}, y^{(2)}), \ldots, (x^{(m)}, y^{(m)})$$

where $y^{(i)}$ is one of:

$$\begin{bmatrix} 1 \\ 0 \\ 0 \\ 0 \end{bmatrix}, \begin{bmatrix} 0 \\ 1 \\ 0 \\ 0 \end{bmatrix}, \begin{bmatrix} 0 \\ 0 \\ 1 \\ 0 \end{bmatrix}, \begin{bmatrix} 0 \\ 0 \\ 0 \\ 1 \end{bmatrix}$$

Question

Suppose you have a multi-class classification problem with 10 classes. Your neural network has 3 layers, and the hidden layer (layer 2) has 5 units. How many elements does $\Theta^{(2)}$ have?

(i) 50
(ii) 55
(iii) 60
(iv) 66

Let’s summarize what we have learnt so far

Neural Network's Intuition

- In neural network, we want to learn a function $f : x \mapsto y$ in which:
  - $f(x)$ can be considered as a neuron or unit;
  - A neural network can be simple *i.e.* consisting of only 1 neuron; or,
  - A neural network can be complex by stacking many units so that one passes its output to another.

input **hidden** **hidden** **output**

Neural Network's Intuition

- Let's try to construct a neural network for univariate regression setting.
- In this setting, a function $f : x \mapsto y$ can be represented by a single neuron that computes:

$$f(x) = \max(ax + b, 0)$$

for some fixed coefficients a and b. (why is this reasonable?)

- This particular unit is called a ReLU (rectified linear unit).

Stacking Units to Yield Complex Functions

- We have seen previously that stacking units can represent XNOR.
- Now, let's consider a more realistic example in which we want to predict house price given size, number of bedrooms, postal code, and wealth of the neighborhood the house is in.
- To address this problem, we may stack units as follows:
  - Compute a 'family size' variable based on size and #bedrooms;
  - Compute how 'walkable' the neighborhood is based on postal code;
  - Compute 'school quality' based on postal code and wealth of the neighborhood.

Stacking Units to Yield Complex Functions

Finally, we might decide the price depends on these three derived features *viz.* family size, walkable, and school quality.

(From Ng (2017), CS229 deep learning lecture notes)

End-to-end Learning

- Have you ever seen any problems with this architecture as we described so far?
- Luckily, we don't need to solve these problems as neural network's learning is an end-to-end learning.
- That means the network figures out for itself what intermediate features are best for the task at hand.

Hidden Units

- Intermediate units or hidden units are neurons between the raw inputs and the output.
- Suppose we have:
  - Four inputs *viz.* x_1, x_2, x_3, and x_4
  - Three hidden units
  - A single output y

- The goal of the network will be to find intermediate features that will best predict each $y^{(i)}$ from the corresponding $\boldsymbol{x}^{(i)}$.
- It may be difficult to understand the 'meaning' of the intermediate features thus induced. Neural networks are thus called black boxes.

Terminology Summary

- A neuron or unit applied some functions (*e.g.* ReLU, Sigmoid) to its input in order to generate an output.
- Units may be composed (or stacked) into neural networks.
- Input features are sometimes represented by units called input units organized into an input layer.
- One or more outputs comprise the output layer.
- Intermediate units are called hidden units and may be organized into zero or more hidden layers.
  - Why is this reasonable? *cf.* slide#49.