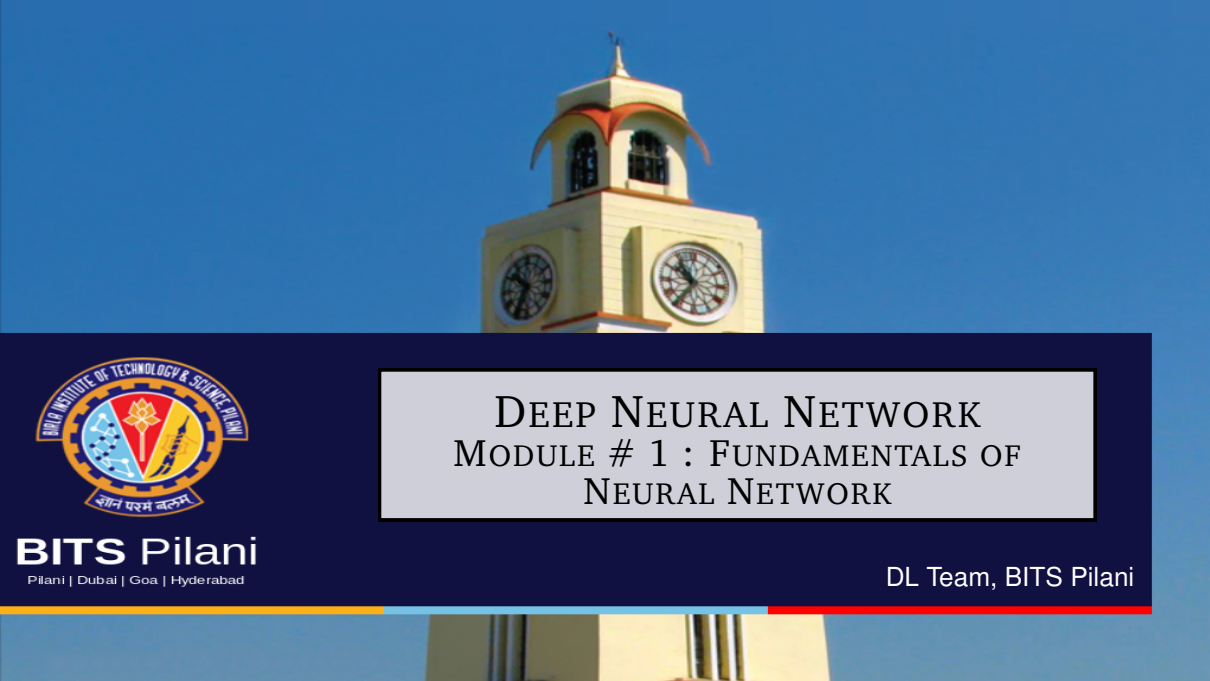


Deep Neural Network

Module # 1 : Fundamentals of Neural Network

BITS Pilani

Pilani | Dubai | Goa | Hyderabad

DL Team, BITS Pilani

The author of this deck, Prof. Seetha Parameswaran,
is gratefully acknowledging the authors
who made their course materials freely available online.

Table of Contents

What we Learn...

1. Fundamentals of Neural Network and Multilayer Perceptron
2. Deep Feed-forward Neural Network
3. Improve its performance by Optimization
4. Improve its performance by Regularization
5. Convolutional Neural Networks
6. Sequence Models
7. Attention Mechanism
8. Neural Network search
9. Time series Modelling and Forecasting
10. Other Learning Techniques

Text and Reference Books

Text Books

T1	Dive into Deep Learning by Aston Zhang, Zack C. Lipton, Mu Li, Alex J. Smola. https://d2l.ai/chapter_introduction/index.html
R1	Deep Learning by Ian Goodfellow, YoshuaBengio, Aaron Courville https://www.deeplearningbook.org/

Evaluation Schedule

No	Name	Type	Duration	Weight	Day, Date, Time
EC1	Quiz I	Online	0.5 hr	5 %	Check Canvas
	Quiz II	Online	0.5 hr	5 %	Check Canvas
	Assignment I	Online	4 weeks	15 %	Check Canvas
	Assignment II	Online	4 weeks	15 %	Check Canvas
EC2	Mid-sem Regular	Closed book	2 hrs	30%	Check Calendar
EC3	Compre-sem Regular	Open book	2.5 hrs	30%	Check Calendar

Lab Sessions

1. Python Libraries – Keras and Tensorflow

L1 Introduction to Tensorflow and Keras

L2 Deep Neural Network with Back-propagation and optimization

L3 CNN

L4 RNN

L5 LSTM

L6 Auto-encoders

LMS

Refer Canvas for the following

- Handout
- Schedule for Webinars
- Schedule of Quiz, and Assignments.
- Evaluation scheme
- Session Slide Deck
- Demo Lab Sheets
- Quiz-I, Quiz-II
- Assignment-I, Assignment-II
- Sample QPs

The Lecture Recordings are made available on Microsoft Teams.

Honor Code

All submissions for graded components must be the result of your original effort. It is strictly prohibited to copy and paste verbatim from any sources, whether online or from your peers. The use of unauthorized sources or materials, as well as collusion or unauthorized collaboration to gain an unfair advantage, is also strictly prohibited. Please note that we will not distinguish between the person sharing their resources and the one receiving them for plagiarism, and the consequences will apply to both parties equally.

In cases where suspicious circumstances arise, such as identical verbatim answers or a significant overlap of unreasonable similarities in a set of submissions, will be investigated, and severe punishments will be imposed on all those found guilty of plagiarism.

In case of Queries regarding the course ...

Step 1 : Post in the discussion forum.

Read through the existing posts and if you find any topic similar to your concern, add on to the existing discussion.

Avoid duplication of queries or issues. This is highly appreciated.

Step 2 : Email to the IC at seetha.p@pilani.bits-pilani.ac.in if the query or issue is not resolved within one weeks' time. Turn around for response to the email is 48hrs.

In the subject please mention the phrase **"DNN"** clearly.

Use BITS email id for correspondence. Emails from personal emails will be ignored without any reply.

PATIENCE is highly APPRECIATED :)

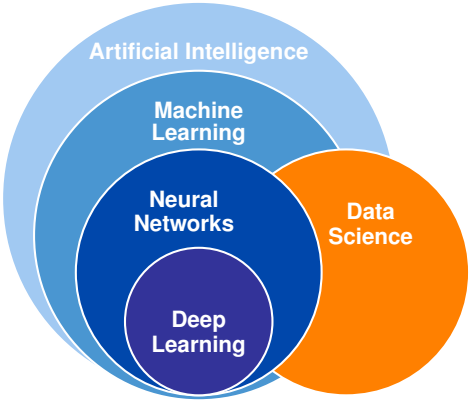
Table of Contents

What is Deep Learning?

- Deep Learning is a type of **machine learning** based on **artificial neural networks** in which multiple layers of processing are used to extract progressively higher level features from data.
- Deep learning is a method in artificial intelligence (AI) that teaches computers to process data in a way that is **inspired by the human brain**.
- Deep learning is a machine learning technique that teaches computers to do what comes naturally to humans: **learn by example**.
- Deep learning is a subset of machine learning, which is essentially a neural network with **three or more layers**.
- Deep Learning gets its name from the fact that we add more **Layers** to learn from the data.

Where in AI sits DL?

- AI is a general field that encompasses machine learning and deep learning, but that also includes many more approaches that don't involve any learning.



AI – ML – DL

AI : Artificial intelligence is the **science** of making things smart. The aim is make machines perform human tasks. Eg: Robot cleaning a room.

ML : Machine learning is an **approach** to AI. The machine learns or perform tasks through learning by experience.

DL : Deep Learning is a **technique** for implementing machine learning to recognise patterns.

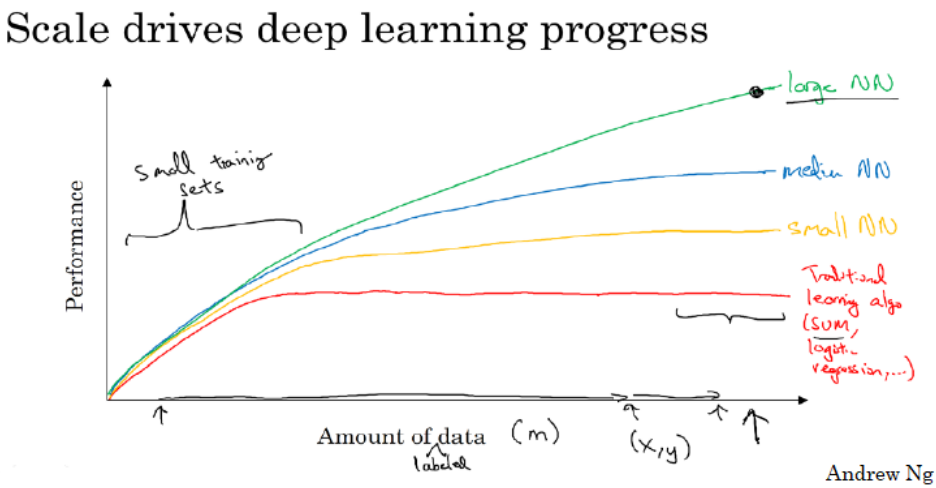
Deep (Machine) Learning

- Deep learning is a specific subfield of machine learning.
- Learning representations from data that puts an emphasis on learning successive layers of increasingly meaningful representations.
- The **deep** in deep learning stands for this idea of successive layers of representations.
- The number of layers that contribute to model the data is called the **depth** of the model.
- In deep learning, the layered representations are learned via models called **neural networks**, structured in literal layers stacked on top of each other.

Why Deep Learning?

- Large amounts of data
- Lots and lots of unstructured data like images, text, audio, video
- Cheap, high-quality sensors
- Cheap computation - CPU, GPU, Distributed clusters
- Cheap data storage
- Learn by examples
- Automated feature generation
- Better learning capabilities
- Scalability
- Advance analytics can be applied

Why Deep Learning?



Deep Learning: Timeline

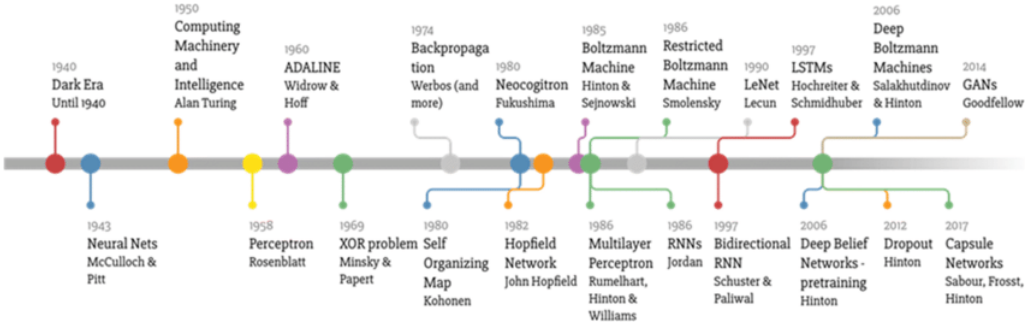


Table of Contents

Breakthroughs with Neural Networks

TECHNEWSWORLD EMERGING TECH SEARCH

Computing Internet IT Mobile Tech Reviews Security Technology Tech Blog Reader Service

Microsoft AI Beats Humans at Speech Recognition

By Richard Adhikari
Oct 20, 2016 11:40 AM PT

Print Email

Image: Adobe Stock

Microsoft's Artificial Intelligence and Research Unit earlier this week reported that its speech recognition technology had surpassed the performance of human transcriptionists.

Most Popular Newsletters News Aler

How do you feel about Black Friday and Cyber Monday?

- They're great -- I get a lot of bargains!
- The deals are too spread out -- I'd prefer just one day.
- They're a fun way to kick off the holiday season.
- I don't like the commercialization of Thanksgiving Day.
- They're crucial for the retail industry and the economy.
- The deals typically aren't that good.

Vote to See Results

E-Commerce Times

- Black Friday Shoppers Hungry for New Experiences, New Tech
- Pay TV's Newest Innovation: Giving Users Control
- Apple Celebrates Itself in $300 Coffee Table Tome
- AWS Enjoys Top Perch in IaaS, PaaS Markets
- US Comptroller Gears Up for Blockchain and

Breakthroughs with Neural Networks

Image Segmentation and Recognition

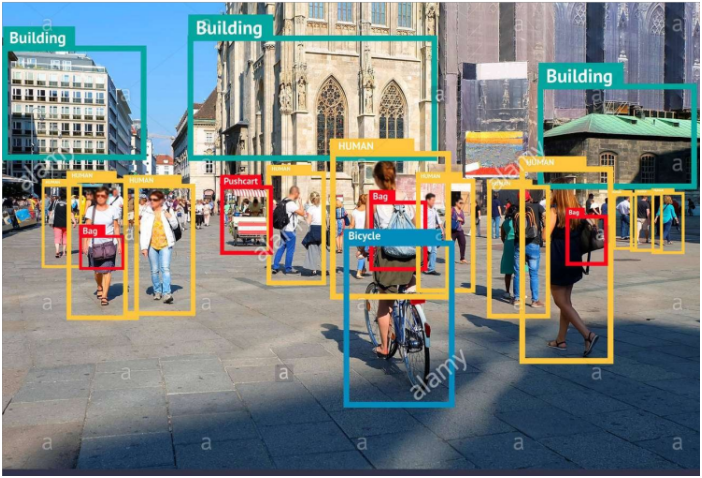
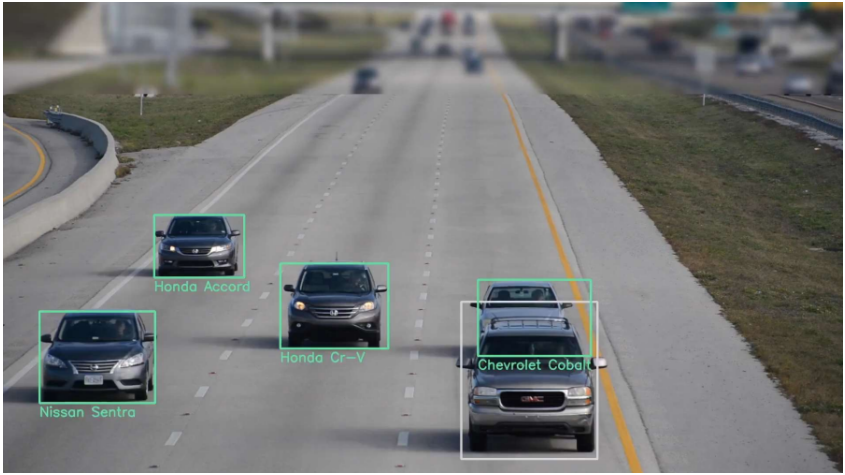


Image Recognition



Breakthroughs with Neural Networks

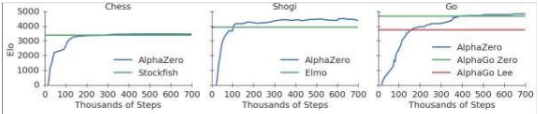
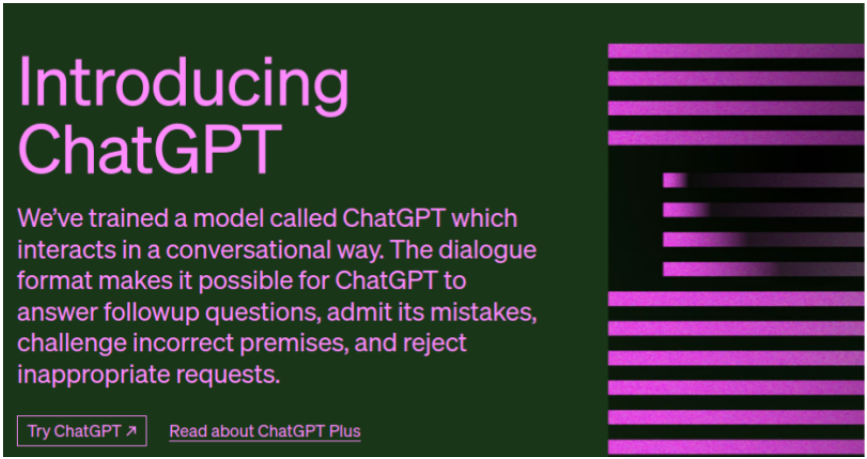


Figure 1: Training *AlphaZero* for 700,000 steps. Elo ratings were computed from evaluation games between different players when given one second per move. **a** Performance of *AlphaZero* in chess, compared to 2016 TCEC world-champion program *Stockfish*. **b** Performance of *AlphaZero* in shogi, compared to 2017 CSA world-champion program *Elmo*. **c** Performance of *AlphaZero* in Go, compared to *AlphaGo Lee* and *AlphaGo Zero* (20 block / 3 day) (*29*).

Breakthroughs with Neural Networks



Applications of Deep Learning

Application	Input	Output	Neural Network
Real Estate	House features	House Price	Std NN
Photo Tagging	Image	Text	CNN
Object detection	Image	Bounding box	CNN
Speech Recognition	Audio	Text transcript	RNN
Translation	English Text	French Text	RNN
Autonomous driving	Image, Sensors Radars	Position of other cars, objects, signals	Hybrid NN

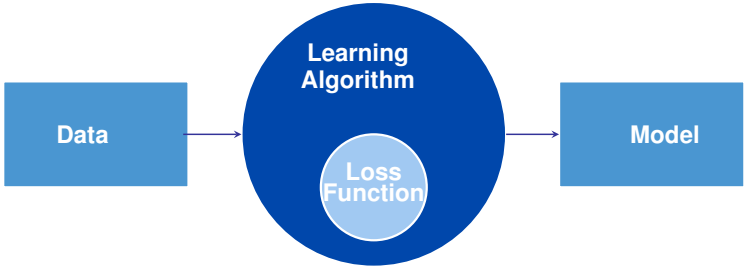
Many more applications...

- a program that predicts tomorrow's weather given geographic information, satellite images, and a trailing window of past weather.
- a program that takes in a question, expressed in free-form text, and answers it correctly.
- a program that given an image can identify all the people it contains, drawing outlines around each.
- a program that presents users with products that they are likely to enjoy but unlikely, in the natural course of browsing, to encounter.

Table of Contents

Core Components of DL Problem

- The **data** that we can learn from.
- A **model** of how to transform the data.
- An **objective function** that quantifies how well (or badly) the model is doing.
- An **algorithm** to adjust the model's parameters to optimize the objective function.



1. Data

- Collection of examples.
- Data has to be converted to an useful and a suitable numerical **representation**.
- Each example (or data point, data instance, sample) typically consists of a set of attributes called **features** (or covariates), from which the model must make its predictions.
- In the supervised learning problems, the attribute to be predicted is designated as the **label** (or target).
- Mathematically, a set of *m* examples,

$$Data = \mathcal{D} = \{X, t\}$$

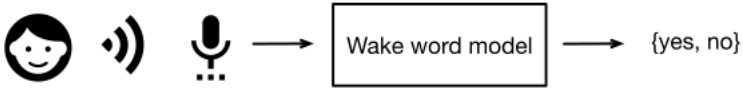
- We need right data.

1. Data

- Dimensionality of data
  - Each example has the same number of numerical values. This data consist of fixed-length vectors. Eg: Image
  - The constant length of the vectors as the dimensionality of the data.
  - Text data has varying-length data.

2. Model

- Model denotes the computational machinery for ingesting data of one type, and spitting out predictions of a possibly different type.
- Deep learning models consist of many successive transformations of the data that are chained together top to bottom, thus the name deep learning.



3. Objective Function

- **Learning means improving at some task over time.**
- A formal mathematical system of learning machines is defined using formal measures of how good (or bad) the models are. These formal measures are called as **objective functions**.
- By convention, objective functions are defined so that lower is better. Because lower is better, these functions are sometimes called **loss functions**.

3. Loss Functions

- To predict numerical values (regression), the most common loss function is squared error.
- For classification, the most common objective is to minimize error rate, i.e., the fraction of examples on which our predictions disagree with the ground truth.

3. Loss Functions

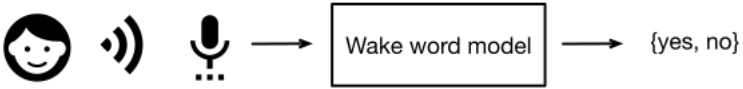
- Loss function is defined
  - with respect to the model's parameters
  - depends upon the dataset.
- We learn the best values of our model's parameters by **minimizing the loss** incurred on a set consisting of some number of examples collected for training. However, doing well on the training data does not guarantee that we will do well on unseen data. i.e **Model has to generalize better.**
- When a model performs well on the training set but fails to generalize to unseen data, we say that it is **overfitting**.

4. Optimization Algorithms

- Optimization Algorithm is an algorithm capable of **searching for the best possible parameters for minimizing the loss function**.
- Popular optimization algorithms for deep learning are based on an approach called **gradient descent**.

Example of the Framework

- We have to tell a computer explicitly how to map from inputs to outputs.
- We have to define the problem precisely, pinning down the exact nature of the inputs and outputs, and choosing an appropriate model family.
- Collect a huge dataset containing examples of audio and label those that do and that do not contain the wake word.



Example of the Framework

- Create a Model
  - Define a flexible program whose behavior is determined by a number of parameters.
  - To determine the best possible set of parameters, use the data. The parameters should improve the performance of the program with respect to some measure of performance on the task of interest.
  - After fixing the parameters, we call the program a model.
    Eg: The model receives a snippet of audio as input, and the model generates a selection among yes, no as output.
  - The set of all distinct programs (input-output mappings) that we can produce just by manipulating the parameters is called a family of models.
    Eg: We expect that the same model family should be suitable for ”Alexa” recognition and ”Hey Siri” recognition because they seem, intuitively, to be similar tasks.

Example of the Framework

- The meta-program that uses our dataset to choose the parameters is called a learning algorithm.
- In machine learning, the learning is the process by which we discover the right setting of the parameter coercing the desired behavior from our model.
- **Train the model with data.**

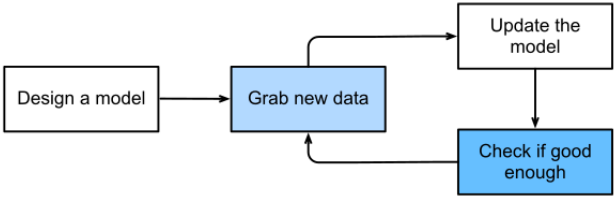
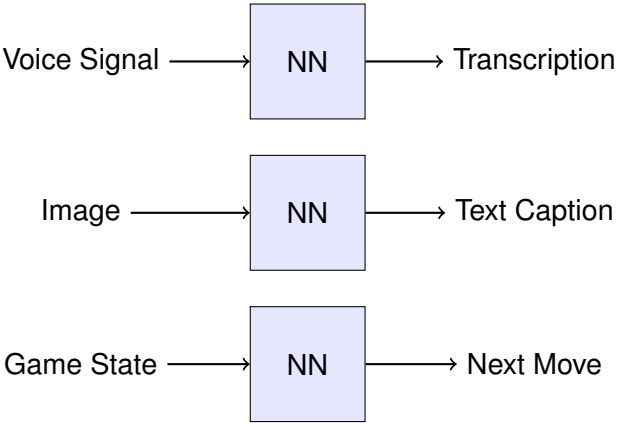


Fig. 1.1.2: A typical training process.

Table of Contents

What are Neural Networks??



What are Neural Networks???

It begins with brain.

- Humans learn, solve problems, recognize patterns, create, think deeply about something, meditate and many many more.....
- Humans learn through association. [Refer to Associationism for more details.]

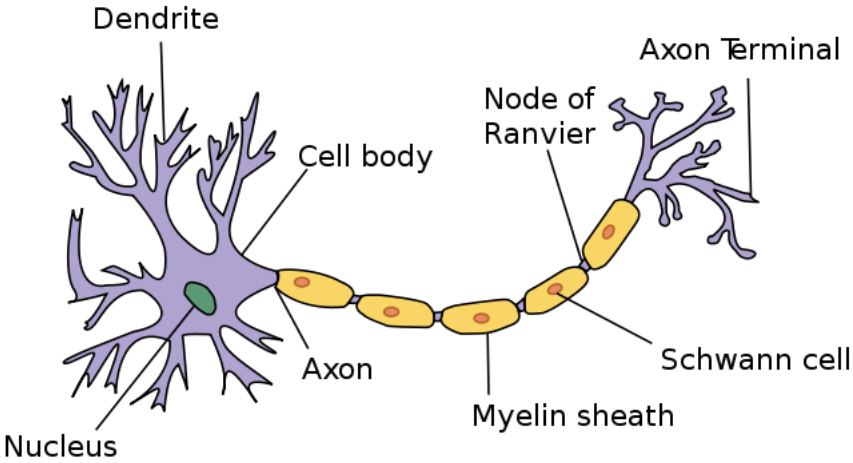
Observation: The Brain

- The brain is a mass of interconnected neurons.
- Number of neurons is approximately 10^{10}.
- Connections per neuron is approximately $10^{(4\ to\ 5)}$.
- Neuron switching time is approximately 0.001 second.
- Scene recognition time is 1 second.
- 100 inference steps doesn't seem like enough. Lot of parallel computation.

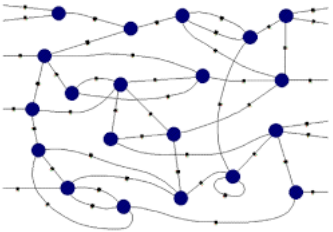
Brain: Interconnected Neurons

- Many neurons connect **in** to each neuron.
- Each neuron connects **out** to many neurons.

Biological Neuron



Connectionist Machines



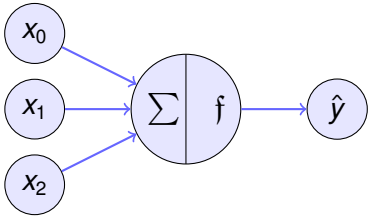
[1]

- Network of processing elements, called artificial neural unit.
- The neurons are interconnected to form a network.
- All world knowledge is stored in the connections between these elements.
- Neural networks are connectionist machines.

[1] alanturing.net

Artificial Neuron

- Neuron is a processing element inspired by how the brain works.
- Similar to biological neuron, each artificial neuron will be do some computation. Each neuron is interconnected to other neurons.
- Similar to brain, the interconnections between neurons store the knowledge it learns. The knowledge is stored as parameters.



Properties of Artificial Neural Nets (ANNs)

- Many neuron-like threshold switching units.
- Many weighted interconnections among units.
- Highly parallel, distributed process.
- Emphasis on tuning parameters or weights automatically.

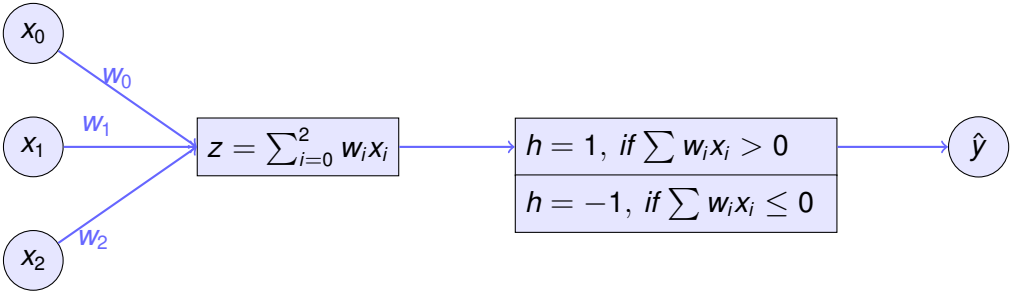
When to consider Neural Networks?

- Input is high-dimensional discrete or real-valued (e.g. raw sensor input).
- Possibly noisy data. Data has lots of errors.
- Output is discrete or real valued or a vector of values.
- Form of target function is unknown.
- Human readability, in other words, explainability, of result is unimportant.
- Examples:
  - Speech phoneme recognition
  - Image classification
  - Financial prediction

Table of Contents

Perceptron

- One type of ANN system is based on a unit called a perceptron.
- **A perceptron takes a vector of real-valued inputs, calculates a linear combination of these inputs, then outputs a 1 if the result is greater than some threshold and -1 otherwise.**



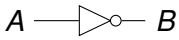
Representing Logic Gates using Perceptron

- Any linear decision can be represented by either a linear regression equation or a Boolean equation.
- The quest is to find out how to represent the above using Perceptron.
- For this we test whether each of the logic gates can be represented by a Perceptron.

NOT Gate

Question:

- How to represent NOT gate using a Perceptron?
- What are the parameters for the NOT Perceptron?
- Data is given below.



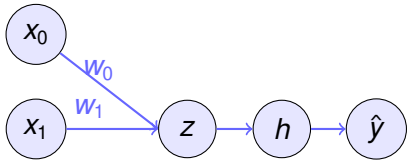
A	B
0	1
1	0

Rewrite as

x_1	t
-1	1
1	-1

Perceptron for NOT gate

- Perceptron equation is $\hat{y} = w_0 x_0 + w_1 x_1$.
- $x_0 = 1$ always.
- $h > 0$ for output to be 1.
- For each row of truth table, the equations are given.
- One solution is $w_0 = 1$ and $w_1 = -1$. (Intuitive solution)
- This give a beautiful linear decision boundary.

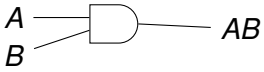


x_1	t_1	h	h
-1	1	$w_0 x_0 + w_1(-1) > 0$	$w_0 - w_1 > 0$
1	-1	$w_0 x_0 + w_1(1) < 0$	$w_0 + w_1 < 0$

AND Logic Gate

Question:

- How to represent AND gate using a Perceptron?
- What are the parameters for the AND Perceptron?
- Data is given below.



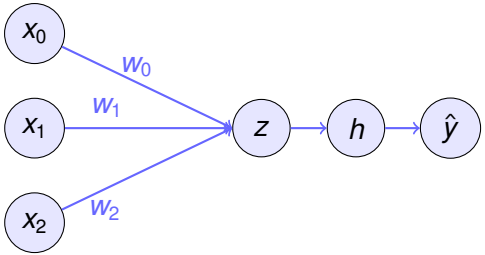
A	B	AB
0	0	0
0	1	0
1	0	0
1	1	1

Rewrite as

x_1	x_2	t
-1	-1	-1
-1	1	-1
1	-1	-1
1	1	1

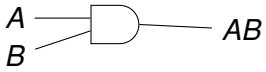
Perceptron for AND gate

- Perceptron equation is $\hat{y} = w_0 x_0 + w_1 x_1 + w_2 x_2$.
- $h > 0$ for output to be 1.
- For each row of the truth table, the equations are given.
- Solve for the inequalities.
- One solution is $w_1 = w_2 = 2, w_0 = (-1)$.
- This give a beautiful linear decision boundary.

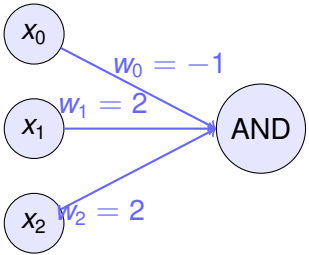


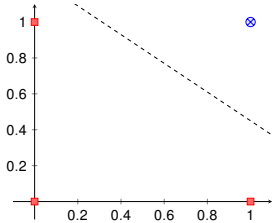
x_1	x_2	t	h
-1	-1	-1	$w_0 + w_1(-1) + w_2(-1) < 0$
-1	1	-1	$w_0 + w_1(-1) + w_2(1) < 0$
1	-1	-1	$w_0 + w_1(1) + w_2(-1) < 0$
1	1	1	$w_0 + w_1(1) + w_2(1) > 0$

Perceptron for AND gate



A	B	AB
0	0	0
0	1	0
1	0	0
1	1	1





Perceptron and Decision boundary for AND gate

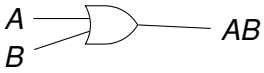
Exercises

- Represent OR gate using Perceptron. Compute the parameters of the Perceptron.
- Represent NOR gate using Perceptron. Compute the parameters of the Perceptron.
- Represent NAND gate using Perceptron. Compute the parameters of the Perceptron.

OR Logic Gate

Question:

- How to represent OR gate using a Perceptron?
- What are the parameters for the OR Perceptron?
- Data is given below.



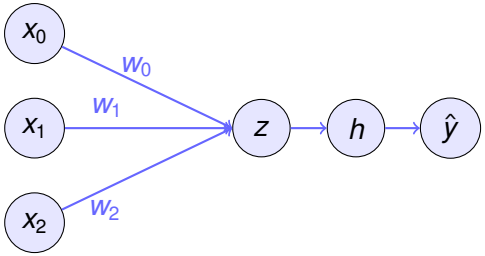
A	B	AB
0	0	0
0	1	1
1	0	1
1	1	1

Rewrite as

x_1	x_2	t
-1	-1	-1
-1	1	1
1	-1	1
1	1	1

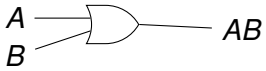
Perceptron for OR gate

- Perceptron equation is $\hat{y} = w_0 x_0 + w_1 x_1 + w_2 x_2$.
- $h > 0$ for output to be 1.
- For each row of the truth table, the equations are given.
- Solve for the inequalities.
- One solution is $w_0 = w_1 = w_2 = 2$.
- This give a beautiful linear decision boundary.

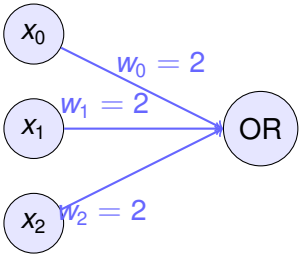


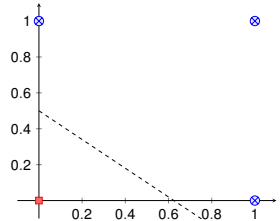
x_1	x_2	t	h
-1	-1	-1	$w_0 + w_1(-1) + w_2(-1) < 0$
-1	1	-1	$w_0 + w_1(-1) + w_2(1) > 0$
1	-1	-1	$w_0 + w_1(1) + w_2(-1) > 0$
1	1	1	$w_0 + w_1(1) + w_2(1) > 0$

Perceptron for OR gate



A	B	AB
0	0	0
0	1	1
1	0	1
1	1	1





Perceptron and Decision boundary for OR gate

Perceptron Learning Algorithm

$$w_i \leftarrow w_i + \Delta w_i$$

$$\Delta w_i = \eta(t - \hat{y})x_i$$

where :

t = target value

$\hat{y}$ = perceptron output

η = learning rate

Convergence of Perceptron Learning Algorithm

It can be proved that the algorithm will converge

- If training data is linearly separable.
- Learning rate is sufficiently small.
  - The role of the learning rate is to moderate the degree to which weights are changed at each step.
  - It is usually set to some small value (e.g., 0.1) and is sometimes made to decay as the number of weight-tuning iterations increases.

Perceptron Learning for NOT gate

- Equations are

$$\begin{aligned}
\text{Hypothesis} \quad & z = w_0 + w_1 x_1 \\
\text{Activation} \quad & h = sign(z) \\
\text{Compute output} \quad & \hat{y} = h = sign(w_0 + w_1 x_1) \\
\text{Compute } \Delta w \quad & = \eta(t - \hat{y})x \\
\text{Update } w_{new} \quad & \leftarrow w_{old} + \Delta w
\end{aligned}$$

Perceptron Learning for NOT gate

- Assume $w_0 = w_1 = 0$. Let the learning rate$= \eta = 1$.
- Epoch 1

x_1	t	w_1	w_0	z	h	$isequal(t, h)$	Δw	New w
0	1	0	0	$0+0=0$	-1	Not equal	$\Delta w_1 = 1(1-(-1))0 = 0$ $\Delta w_0 = 1(1-(-1)) = 2$	$w_1 \leftarrow 0+0=0$ $w_0 \leftarrow 0+2=2$
1	-1	0	2	$2+0=2$	1	Not equal	$\Delta w_1 = 1(-1-1)1 = -2$ $\Delta w_0 = 1(-1-1) = -2$	$w_1 \leftarrow 0+-2=-2$ $w_0 \leftarrow 2+-2=0$

- Epoch 2

x_1	t	w_1	w_0	z	h	$isequal(t, h)$	Δw	New w
0	1	-2	0	$0+0=0$	-1	Not equal	$\Delta w_1 = 1(1-(-1))0 = 0$ $\Delta w_0 = 1(1-(-1)) = 2$	$w_1 \leftarrow -2+0=-2$ $w_0 \leftarrow 0+2=2$
1	-1	-2	2	$2+-2=0$	-1	Equal		

- Algorithm Converges.

Demo Python Code

- https://colab.research.google.com/drive/1DUVcOoUIWhl8GQKc6AWR1wi0LaMeNkgD?usp=sharing

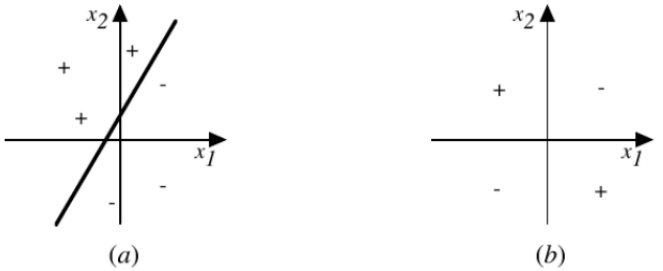
Student pl note:
The Python notebook is shared for anyone who has access to the link and the access is restricted to use BITS email id. So please do not access from non-BITS email id and send requests for access. Access for non-BITS email id will nto be granted.

Exercise

- Represent OR gate using Perceptron. Compute the parameters of the Perceptron using Perceptron learning algorithm.
- Represent AND gate using Perceptron. Compute the parameters of the Perceptron using Perceptron learning algorithm.

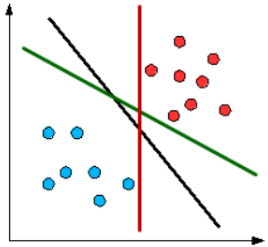
Representational Power of Perceptron

- A Perceptron represents a hyperplane decision surface in the n-dimensional space of examples.
- The Perceptron outputs a 1 for examples lying on one side of the hyperplane and outputs a -1 for examples lying on the other side.

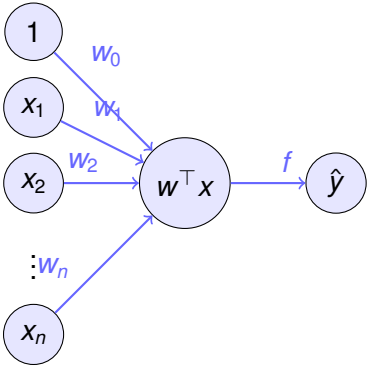


Linearly Separable Data

- Two sets of data points in a two dimensional space are said to be linearly separable when they can be completely separable by a single straight line.
- A straight line can be drawn to separate all the data examples belonging to class +1 from all the examples belonging to the class -1. Then the two-dimensional data are clearly linearly separable.
- An infinite number of straight lines can be drawn to separate the class +1 from the class -1.



Perceptron for Linearly Separable Data



$$x = \begin{bmatrix} x_0 \\ x_1 \\ \vdots \\ x_n \end{bmatrix} \qquad w = \begin{bmatrix} w_0 \\ w_1 \\ \vdots \\ w_n \end{bmatrix}$$

$$\sum = w_0 + w_1 x_1 + \cdots + w_n x_n$$

$$= w^\top x$$

$$\hat{y} = \begin{cases} 1 & \text{if } w^\top x > 0 \\ -1 & \text{otherwise.} \end{cases}$$

Challenge: How to learn these n parameters $w_0 \ldots w_n$?

Solution: Use Perceptron learning algorithm.

Perceptron Training

$$w_i \leftarrow w_i + \Delta w_i$$

where

$$\Delta w_i = \eta(t - o)x_i$$

where:

- $t = c(\vec{x})$ is target value
- o is perceptron output
- η is small constant (e.g., .1) called *learning rate*

Perceptron training rule will converge

- If training data is linearly separable
- η sufficiently small

Perceptron and its Learning - Review

- Data
  - Truth tables or set of examples
- Model
  - Perceptron
- Objective Function
  - Deviation of desired output t and the computed output $\hat{y}$
- Learning Algorithm
  - Perceptron Learning algorithm

Table of Contents

Linear Regression Example

- Suppose that we wish to estimate the prices of houses (in dollars) based on their area (in square feet) and age (in years).
- The linearity assumption just says that the target (price) can be expressed as a weighted sum of the features (area and age):

$$price = w_{area} * area + w_{age} * age + b$$

  w_{area} and w_{age} are called weights, and b is called a bias.
- The weights determine the influence of each feature on our prediction and the bias just says what value the predicted price should take when all of the features take value 0.

Data

- The dataset is called a **training dataset or training set**.
- Each row is called an **example** (or data point, data instance, sample).
- The thing we are trying to predict is called a **label** (or target).
- The independent variables upon which the predictions are based are called **features** (or covariates).

$$
\begin{aligned}
m &- \text{ number of training examples} \\
i &- \text{ index of } i^{th} \text{ example} \\
x^{(i)} &- \text{ features of } i^{th} \text{ example} \\
&= [x^{(1)}, x^{(2)}, x^{(3)}, \ldots, x^{(m)}] \\
y^{(i)} &- \text{ label corresponding to } i^{th} \text{ example} \\
&= [y^{(1)}, y^{(2)}, y^{(3)}, \ldots, y^{(m)}]
\end{aligned}
$$

Affine transformations and Linear Models

- Affine transformations are

$$\begin{aligned}\hat{y} &= w_1 x_1 + w_2 x_2 + \ldots + w_d x_d + b \\ &= w^T x + b \\ \hat{y} &= WX + b\end{aligned}$$

- $\hat{y} = WX + b$ is an **affine transformation** of input features, which is characterized by a linear transformation of features via weighted sum, combined with a translation via the added bias.
- Models whose output prediction is determined by the affine transformation of input features are **linear models**.
- The affine transformation is specified by the chosen **weights (w)** and **bias (b)**.

Loss Function

- Loss function is a quality measure for some given model or a measure of fitness.
- The loss function quantifies the distance between the real and predicted value of the target.
- The loss will usually be a non-negative number where smaller values are better and perfect predictions incur a loss of 0.
- The most popular loss function in regression problems is the squared error.

Squared Error Loss Function

- The most popular loss function in regression problems is the squared error.
- For each example,

$$loss^{(i)}(w, b) = \frac{1}{2}\left(\hat{y}^{(i)} - y^{(i)}\right)^2$$

- For the entire dataset of m examples average (or equivalently, sum) the losses

$$L(w, b) = \frac{1}{m}\sum_{i=1}^{m} loss^{(i)}(w, b) = \frac{1}{m}\sum_{i=1}^{m}\left[\frac{1}{2}\left(\hat{y}^{(i)} - y^{(i)}\right)^2\right]$$

- When training the model, find parameters (w_{opt}, b_{opt}) that minimize the total loss across all training examples.

$$w_{opt}, b_{opt} = argmin_{w,b} \quad L(w, b)$$

Single-Layer Neural Network

Linear regression is a single-layer neural network.

- Number of inputs (or feature dimensionality) in the input layer is d. The inputs are $x_1, \ldots, x_d$.
- Number of outputs in the output layer is 1. The output is $\hat{y}_1$.
- Number of layers for the neural network is 1. (conventionally we do not consider the input layer when counting layers.)
- Every input is connected to every output, This transformation is a **fully-connected layer or dense layer.**

Single-Layer Neural Network

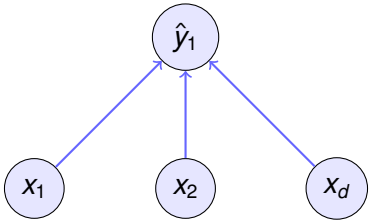
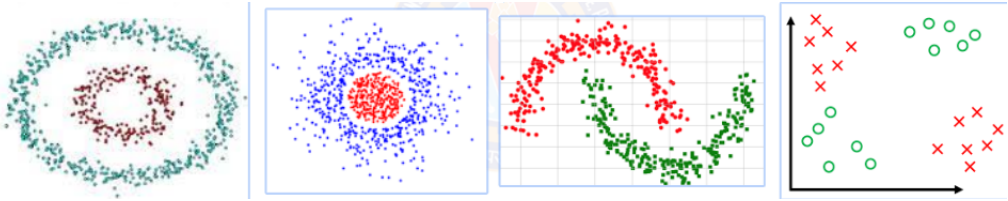


Table of Contents

Non-linearly Separable Data

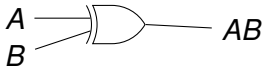
- Two groups of data points are non-linearly separable in a 2-dimensional space if they cannot be easily separated with a linear line.



XOR Logic Gate

Question:

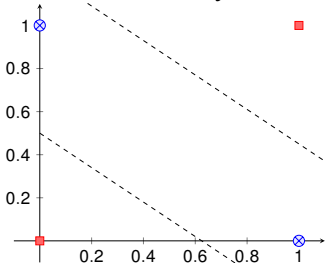
- How to represent XOR gate using a Perceptron?
- What are the parameters for the XOR Perceptron?
- Data is given below.



A	B	AB
0	0	0
0	1	1
1	0	1
1	1	0

Challenge: Data is non-linearly separable.

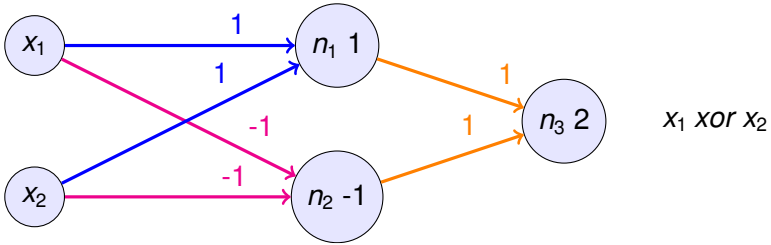
Decision boundary for XOR



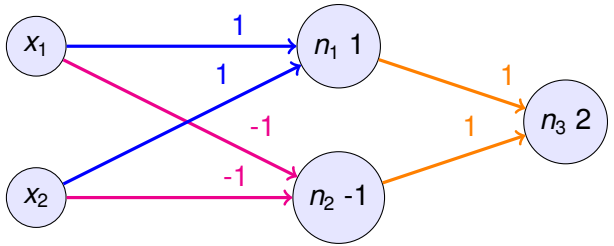
MLP for XOR gate

Qn: How to represent XOR gate using a Perceptron?

- Use Multilayer Perceptron (MLP)
- Introduce another layer in between the input and output.
- This in-between layer is called hidden layer.



MLP for XOR gate



x_1 *xor* x_2

x_1	x_2	n_1	n_2	n_3
0	0	$0\cdot 1+0\cdot 1=0 \not\geq th$ $n_1=0$	$0\cdot(-1)+0\cdot(-1)=0>th$ $n_2=1$	$0\cdot 1+1\cdot 1=1 \not\geq th$ $n_3=0$
0	1	$0\cdot 1+1\cdot 1=1\geq th$ $n_1=1$	$0\cdot(-1)+1\cdot(-1)=-1\geq th$ $n_2=1$	$1\cdot 1+1\cdot 1=2\geq th$ $n_3=1$
1	0	$1\cdot 1+0\cdot 1=1\geq th$ $n_1=1$	$1\cdot(-1)+0\cdot(-1)=-1\geq th$ $n_2=1$	$1\cdot 1+1\cdot 1=2\geq th$ $n_3=1$
1	1	$1\cdot 1+1\cdot 1=2\geq th$ $n_1=1$	$1\cdot(-1)+1\cdot(-1)=-2 \not> th$ $n_2=0$	$1\cdot 1+0\cdot 1=1 \not\geq th$ $n_3=0$

Solution of XOR Data

- Data
  - Truth table
- Model
  - Multi-layered Perceptron
- Challenge
  - How to learn the parameters and threshold?
- Solution for learning
  - Use gradient descent algorithm

Gradient Descent

To understand, consider simpler *linear unit*, where

$$\hat{y} = w_0 + w_1 x_1 + \cdots + w_n x_n$$

Let's learn w_i's that minimize the squared error

$$E[\vec{w}] \equiv \frac{1}{2} \sum_{d \in D} (t_d - \hat{y}_d)^2$$

where D is set of training examples.

Gradient

$$\nabla E[\vec{w}] \equiv \left[\frac{\partial E}{\partial w_0}, \frac{\partial E}{\partial w_1}, \cdots \frac{\partial E}{\partial w_n} \right]$$

Training rule:

$$\Delta \vec{w} = -\eta \nabla E[\vec{w}] \quad \text{i.e.,} \quad \Delta w_i = -\eta \frac{\partial E}{\partial w_i}$$

Gradient Descent

$$
\begin{aligned}
\frac{\partial E}{\partial w_i} &= \frac{\partial}{\partial w_i}\frac{1}{2}\sum_d (t_d - \hat{y}_d)^2 \\
&= \frac{1}{2}\sum_d \frac{\partial}{\partial w_i}(t_d - \hat{y}_d)^2 \\
&= \frac{1}{2}\sum_d 2(t_d - \hat{y}_d)\frac{\partial}{\partial w_i}(t_d - \hat{y}_d) \\
&= \sum_d (t_d - \hat{y}_d)\frac{\partial}{\partial w_i}(t_d - \vec{w}\cdot\vec{x_d}) \\
\frac{\partial E}{\partial w_i} &= \sum_d (t_d - \hat{y}_d)(-x_{i,d})
\end{aligned}
$$

Gradient Descent Algorithm

Gradient-Descent(training examples $\langle \vec{x}, t \rangle$, learning rate η)

- Initialize each w_i to some small random value
- Until the termination condition is met, Do
  - Initialize each Δw_i to zero.
  - For each $\langle \vec{x}, t \rangle$ in *training_examples*, Do
    - Input the instance $\vec{x}$ to the unit and compute the output $\hat{y}$
    - For each linear unit weight w_i, Do

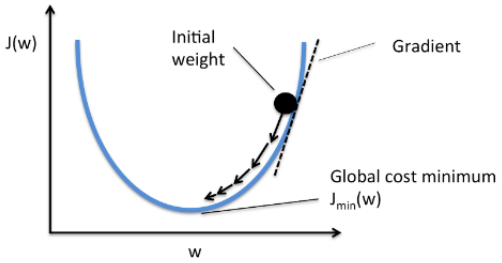
$$\frac{\partial E}{\partial w_i} = \sum_d (t_d - \hat{y}_d)(-x_{i,d})$$

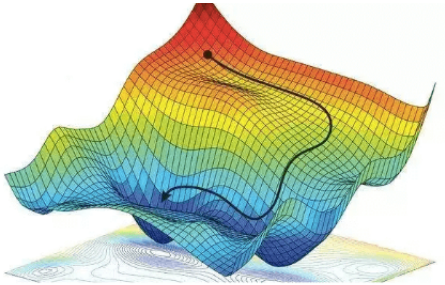
$$\Delta w_i = -\eta \frac{\partial E}{\partial w_i}$$

  - For each linear unit weight w_i, Do

$$w_i \leftarrow w_i + \Delta w_i$$

Gradient Descent Algorithm





Incremental Gradient Descent Algorithm

- Do until satisfied
  - For each training example d in D
    - Compute the gradient $\nabla E_d[\vec{w}]$
    - $\vec{w} \leftarrow \vec{w} - \eta \nabla E_d[\vec{w}]$

$$E_d[\vec{w}] \equiv \frac{1}{2}(t_d - \hat{y}_d)^2$$

Incremental Gradient Descent can approximate *Batch Gradient Descent* arbitrarily closely if η made small enough.

Minibatch Stochastic Gradient Descent (SGD)

- Apply Gradient descent algorithm on a random minibatch of examples every time we need to compute the update.
- In each iteration,
  1. Step 1: randomly sample a minibatch B consisting of a fixed number of training examples.
  2. Step 2: compute the derivative (gradient) of the average loss on the minibatch with regard to the model parameters.
  3. Step 3: multiply the gradient by a predetermined positive value η and subtract the resulting term from the current parameter values.

$$w \leftarrow w - \frac{\eta}{|B|} \sum_{i \in B} \partial_w loss^{(i)}(w, b) = w - \frac{\eta}{|B|} \sum_{i \in B} x^{(i)} \left(w^T x^{(i)} + b - y^{(i)} \right)$$

$$b \leftarrow b - \frac{\eta}{|B|} \sum_{i \in B} \partial_b loss^{(i)}(w, b) = b - \frac{\eta}{|B|} \sum_{i \in B} \left(w^T x^{(i)} + b - y^{(i)} \right)$$

Training SGD

- Initialize parameters (w, b)
- Repeat until done
  - compute gradient

$$g \leftarrow \partial_{(w,b)} \frac{1}{|B|} \sum_{i \in B} \left(w^T x^{(i)} + b - y^{(i)} \right)$$

  - update parameters

$$(w, b) \leftarrow (w, b) - \eta g$$

PS: The number of epochs and the learning rate are both hyperparameters. Setting hyperparameters requires some adjustment by trial and error.

Numerical Example of SGD

- Equation is $y = (x + 5)^2$.
- When will it be minimum?
- Use gradient descent algorithm .
- Assume starting point as 3 and Learning rate as 0.01.
- Equations:

$$\frac{dy}{dx} = \frac{d}{dx}((x + 5)^2) = 2(x + 5)$$

$$x \leftarrow x - \eta \cdot \frac{dy}{dx}$$

- Epoch 1:

$$x \leftarrow 3 - 0.01 * 2(3 + 5) = 2.84$$

- Epoch 2:

$$x \leftarrow 2.84 - 0.01 * 2(2.84 + 5) = 2.68$$

Further Reading

1. Chapter 1 of Dive into Deep Learning (T1)
2. Chapter 2 for Python Prelims, Linear Algebra, Calculus, Probability (T1)
3. Chapter 4 of Book: Machine Learning by Tom M. Mitchell

Thank You!