Neural Networks

John Tan Chong Min

About Myself

- Year 3 PhD Student at NUS
- Published a paper on Neural Network structure pruning in Year 1 (DropNet, ICML 2020)
- Current focus on Reinforcement Learning methods like AlphaZero and generalization methods
- Active Twitch Streamer for Programming/Game Creation:
  - https://www.twitch.tv/johncm99

Program Overview

- Week 1: Overview
- Week 2: Multi-Layer Perceptron
- Week 3: Convolutional Neural Networks (CNN)
- Week 4: Recurrent Neural Networks (RNN)

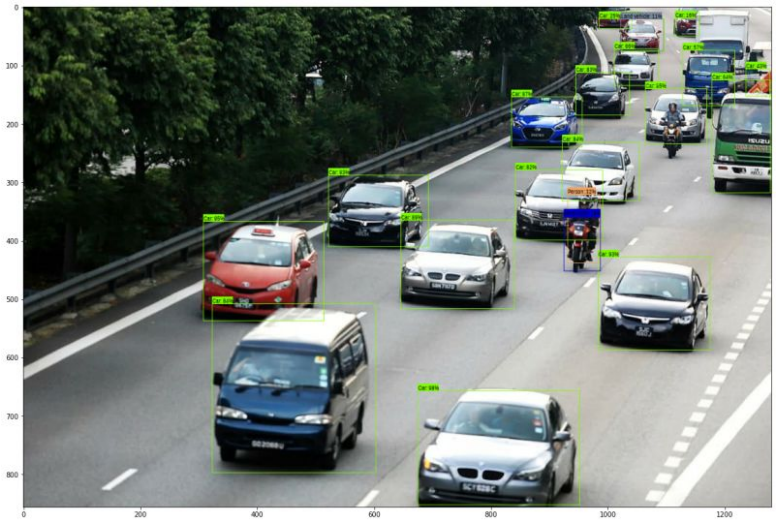
What to focus on

- Focus more on the concepts, less on the math
- No need to worry about the math equations
- Ask questions anytime: All questions are good. Questions help with understanding
- Every week: Theory first, then coding practical

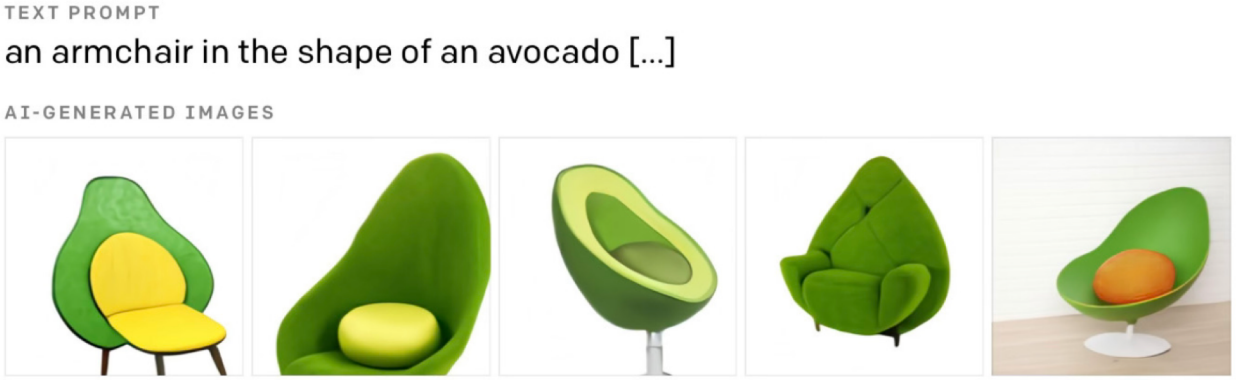
Why learn neural networks?

Practical Applications (Image & Text)

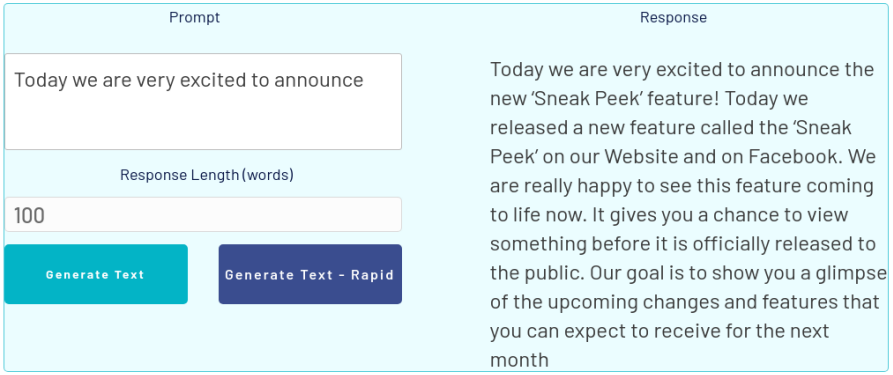
- Just feed in data, and it learns the relations itself!



Object Detection

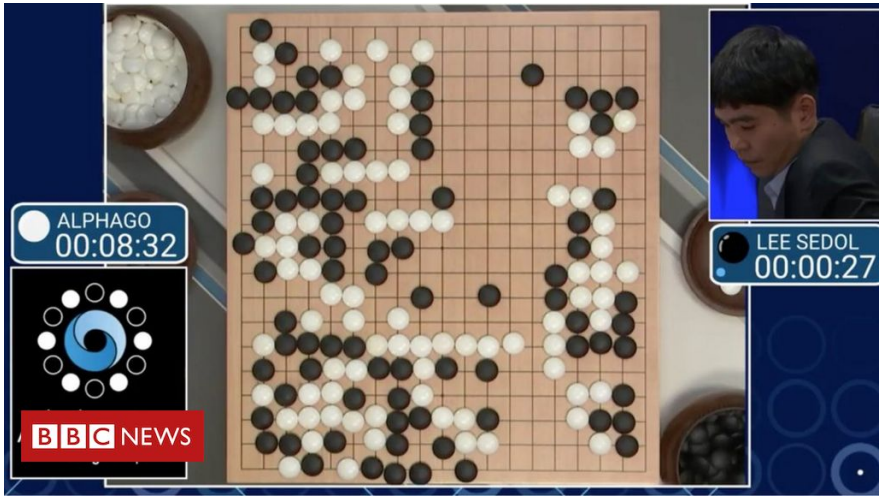


DALL-E



GPT-3

Practical Applications (Reinforcement Learning)

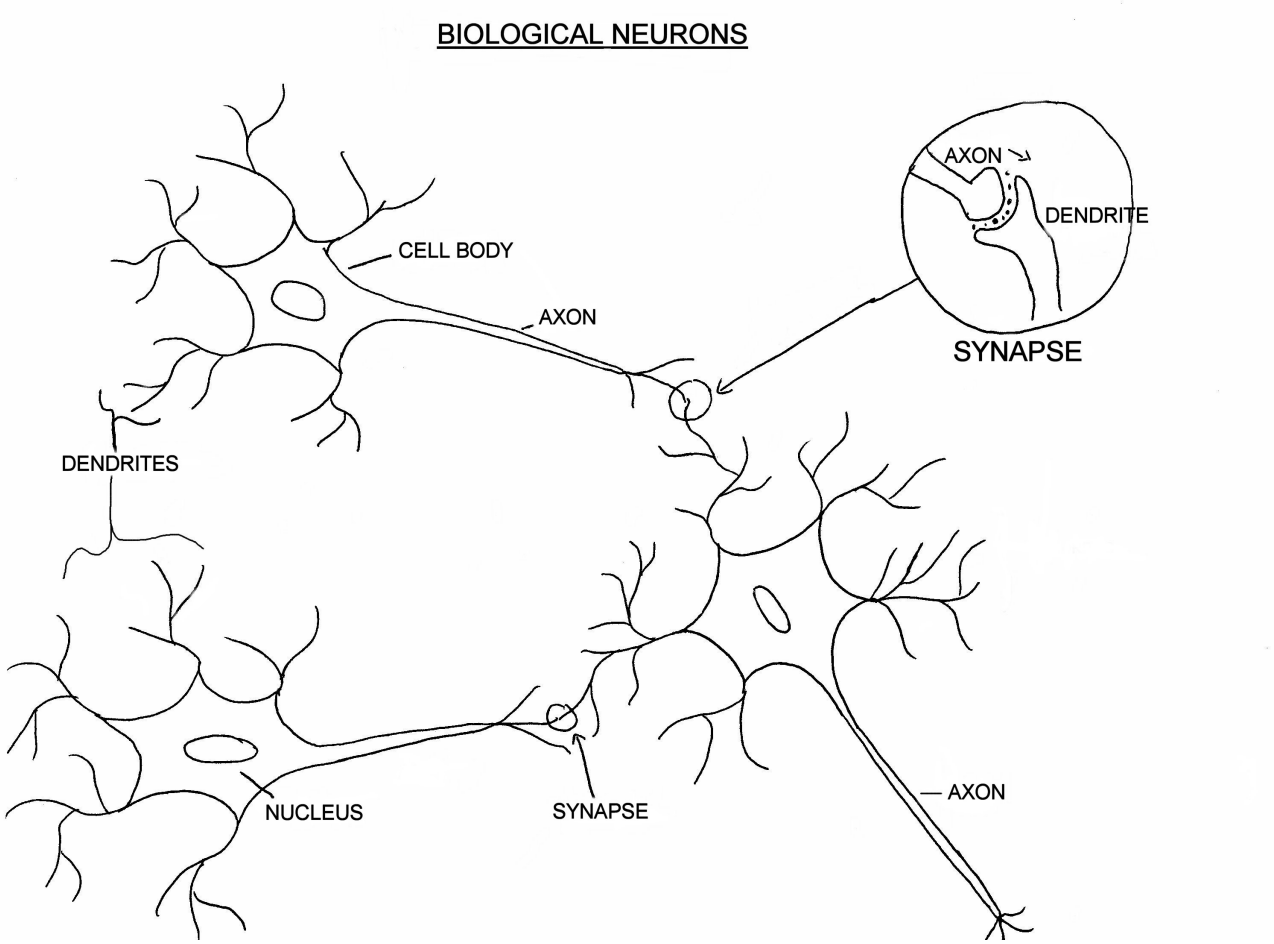


DeepMind: AlphaGo

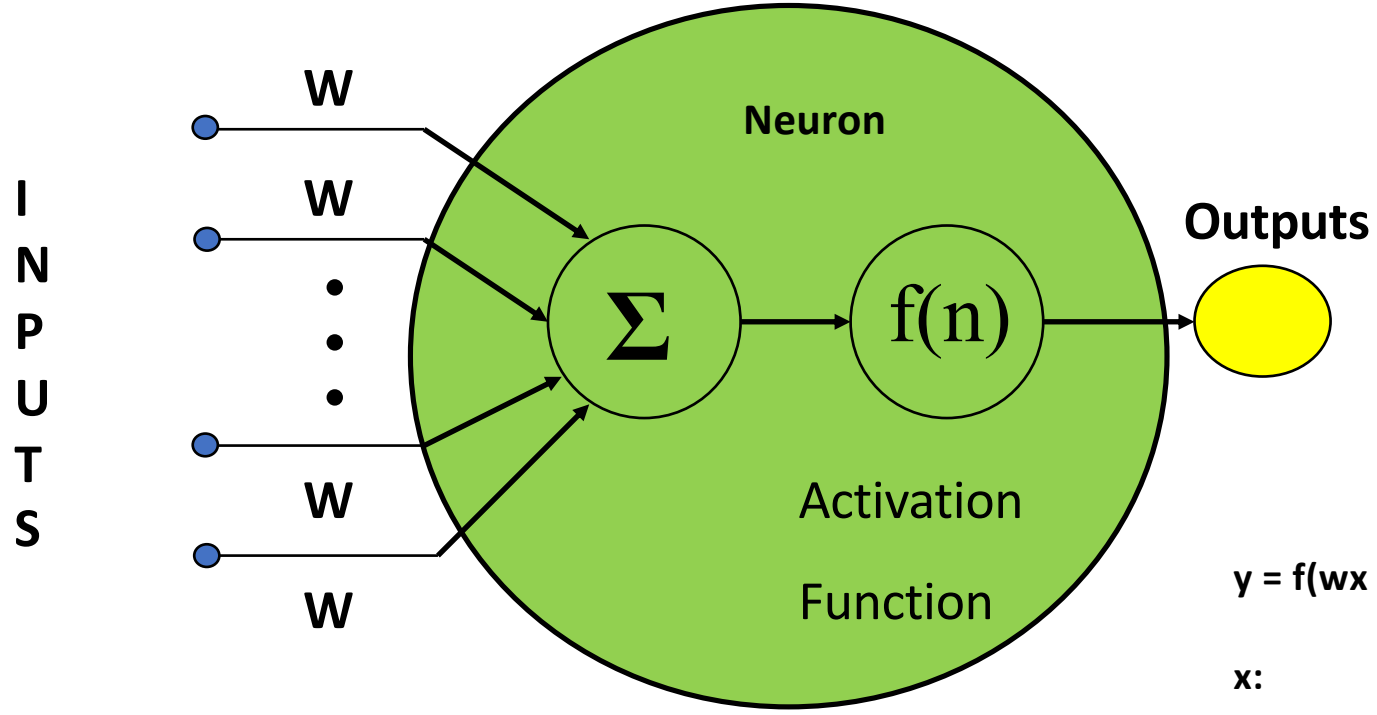
OpenAI Dota 2

So, what is a neuron?

Biological Neuron



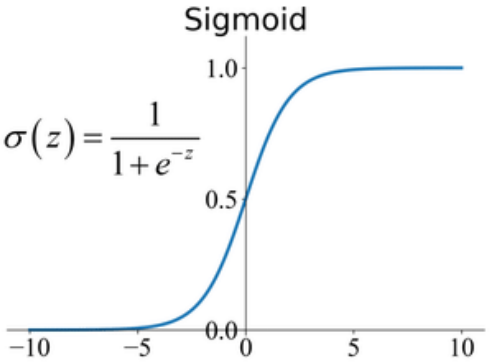
Artificial Neuron



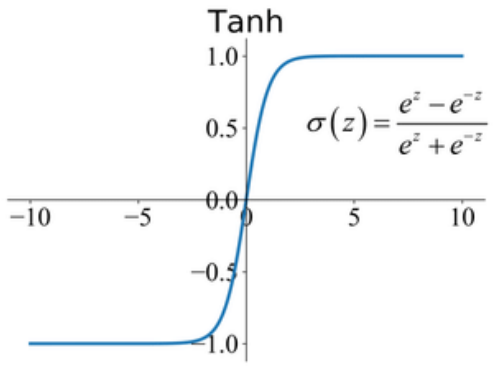
y = f(wx + b)

x:	**inputs**
y:	**output**
w:	**weights**
f(.):	**Activation function**

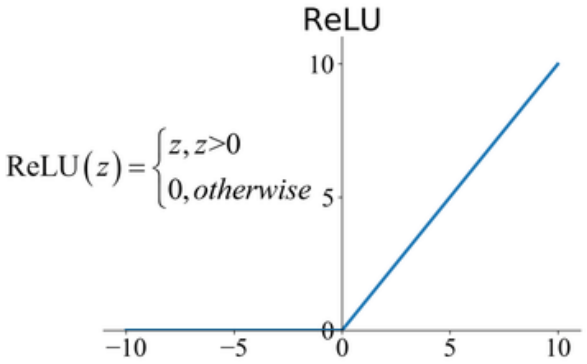
Common Activation Functions



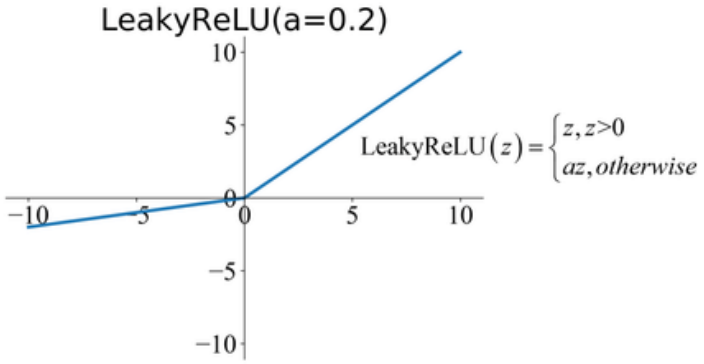
(a)



(b)



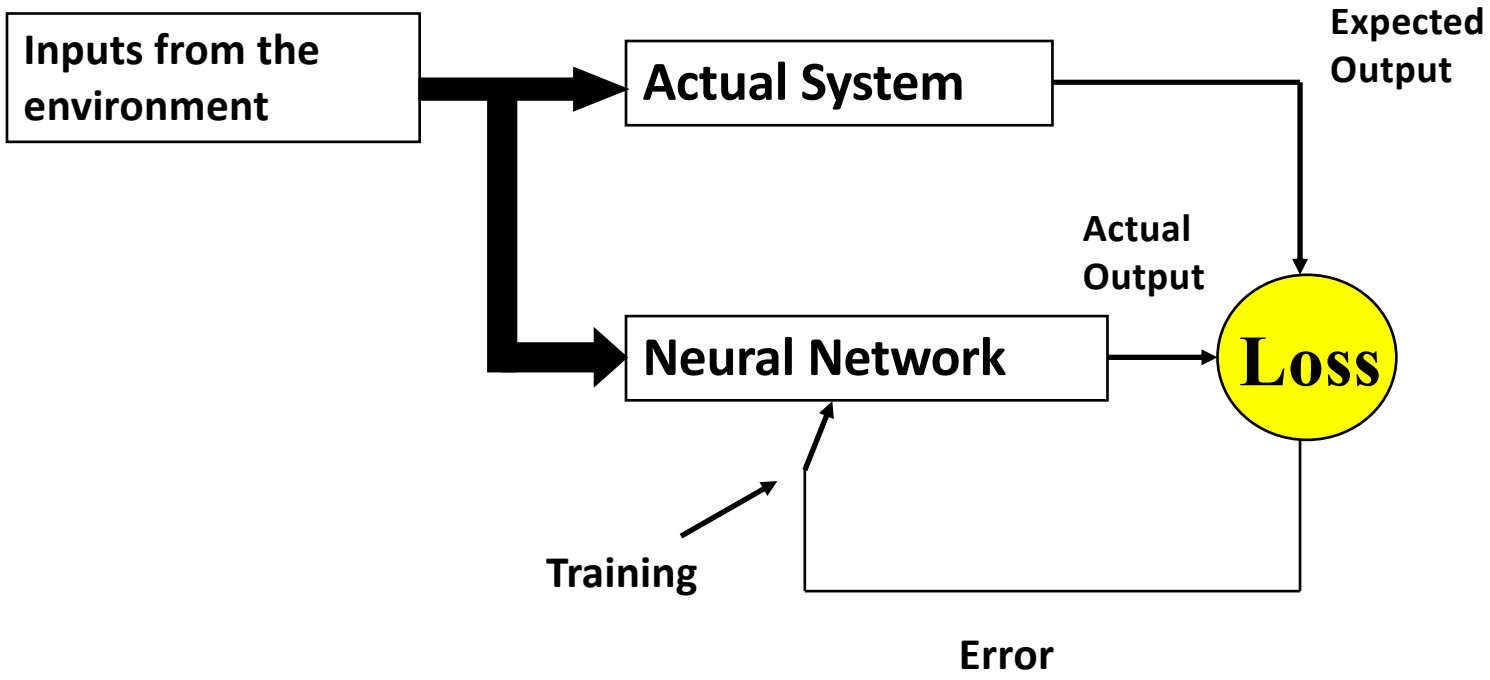
(c)



(d)

How to Learn?

- Supervised Learning:



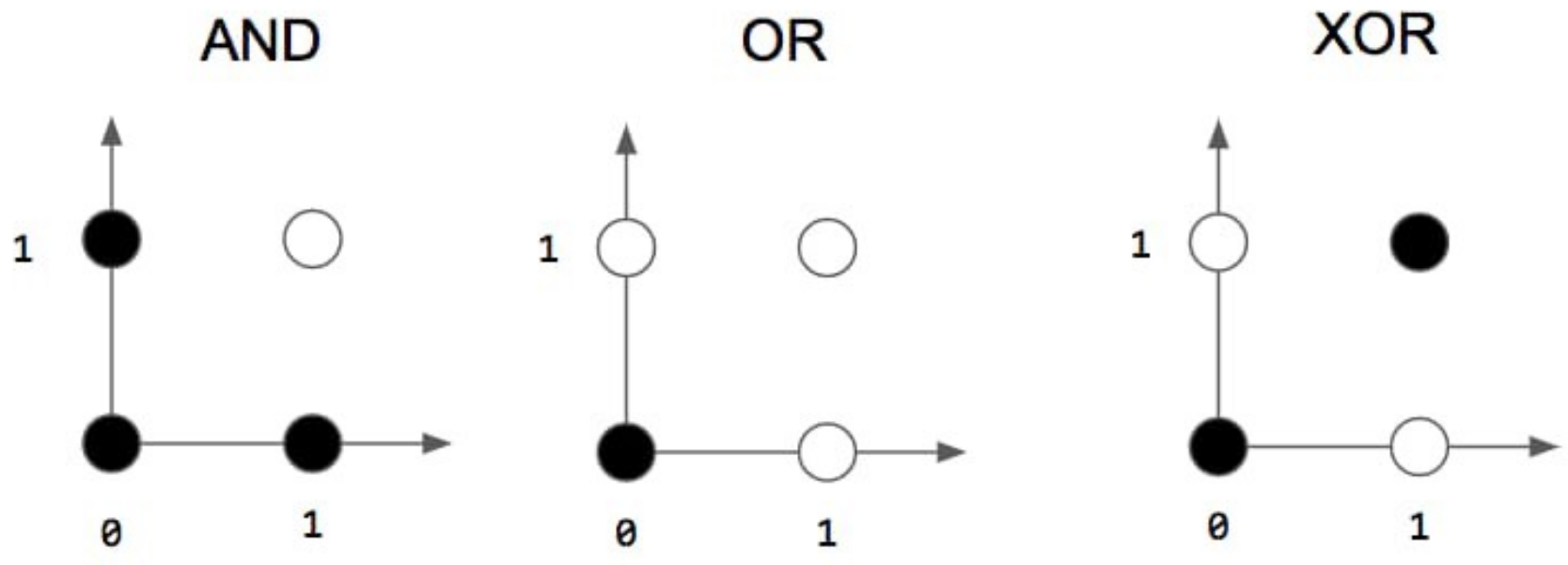
Common Loss Functions

- Regression Problems
  - Mean Squared Error
    - $\frac{1}{2}(y - \hat{y})^2$
  - Mean Absolute Error
    - $|y - \hat{y}|$
- Classification Problems
  - Cross-Entropy
    - $-\sum y \, log \hat{y}$
    - y: actual output probabilities
    - $\hat{y}$: predicted output probabilities

Multi-Layer Perceptron

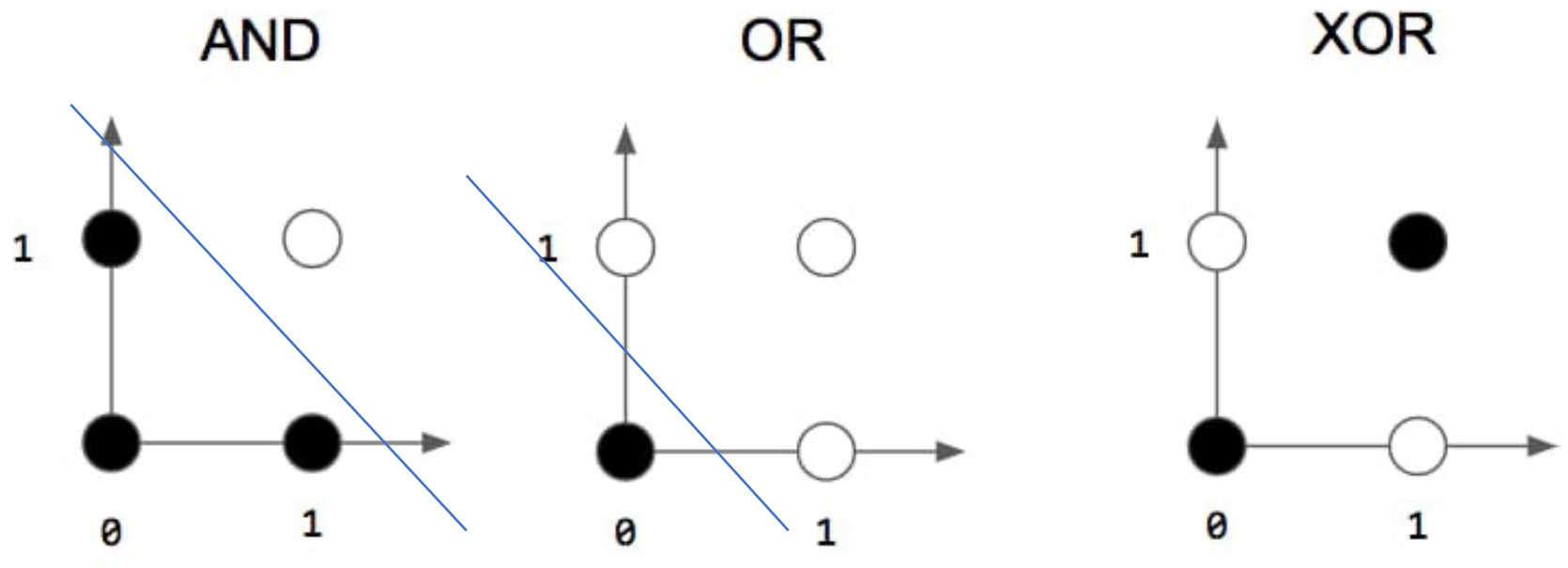
The XOR problem

- Try to draw a line to separate the black and white circles

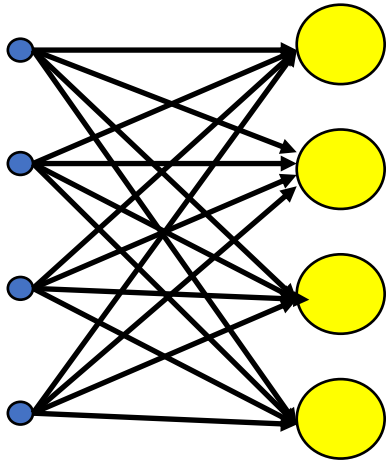


The XOR problem

- Try to draw a line to separate the black and white circles

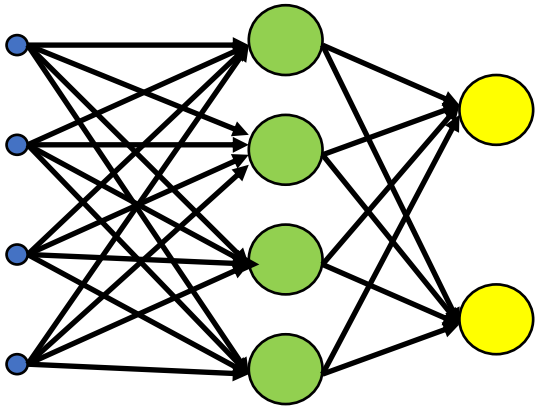


Multi-Layer Perceptron



Single Layer

(Perceptron)



Multiple Layers

(Multi-Layer Perceptron)

Think about it

- Why can’t activation functions in hidden layers be linear?
- Linear Activation: $f(x) = x$
- What is $f(wg + h)$, when $g = f(wx + b)$?

Think about it

- Why can’t activation functions in hidden layers be linear?

- Linear Activation: $f(x) = x$

- What is $f(wg + h)$, when $g = f(wx + b)$?

- **Answer:**
- $f(wg + h) = w(wx + b) + h = (ww)x + (wb + h)$
- It is still linear!

The Credit Assignment Problem

Problem: Credit Assignment

- You are the boss in charge of different workers
- Each worker contributes to the end product, with different amounts of contribution
- At the end, the product is evaluated (by a third party) and you are given a profit based on how well the product is
- **Question: How much to reward each worker?**

Solution: Credit Assignment

- **Solution:**
  - Hypothetically ask how much would changing the worker's output affect the overall product.
  - If increasing his/her output makes the product worse, decrease his/her output, and vice versa

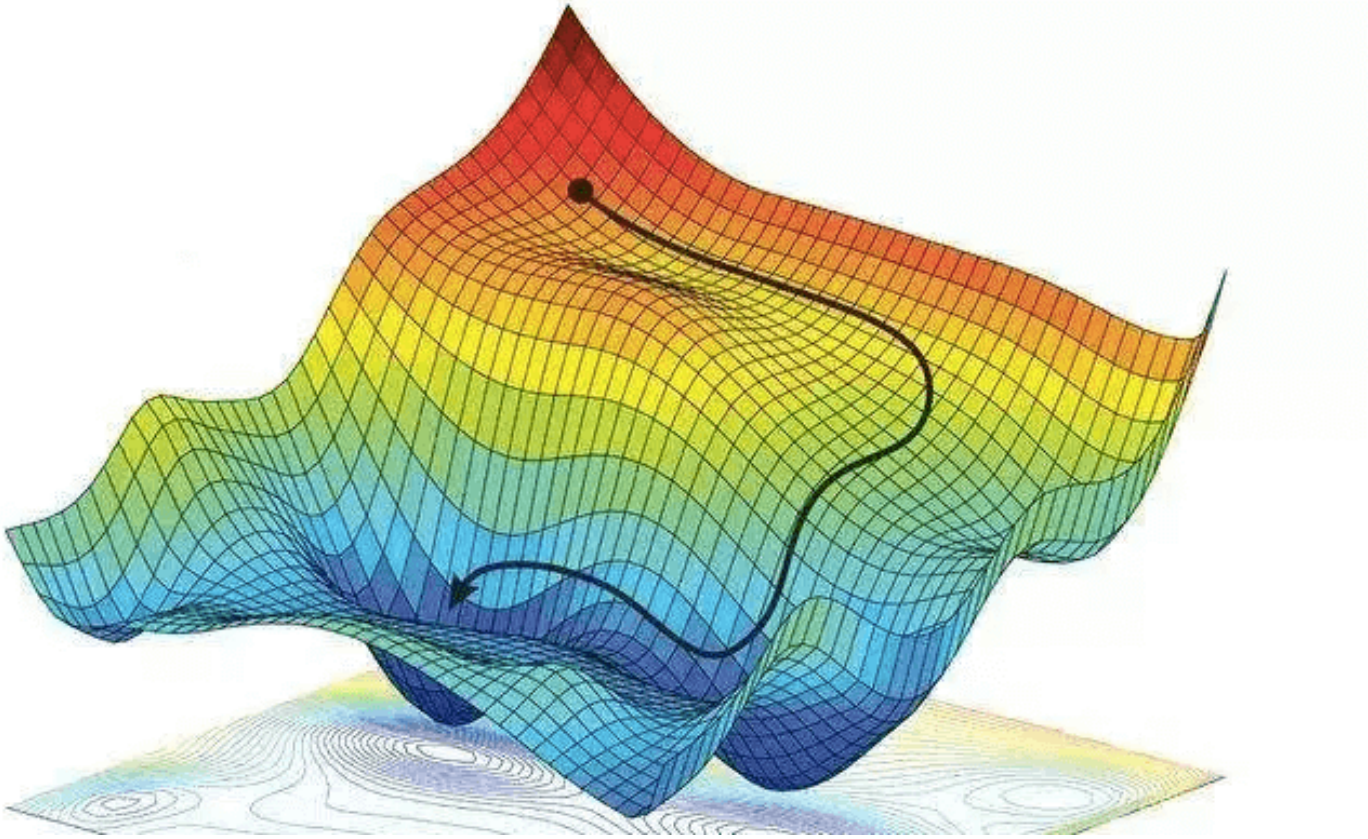
Try it out!

- Try to get the right w, b for
  - $y = wx + b$

- We have 5 pairs of (x,y) given
  - (2, 3)
  - (3, 4)
  - (4, 5)
  - (5, 6)
  - (6, 7)

Gradient Descent

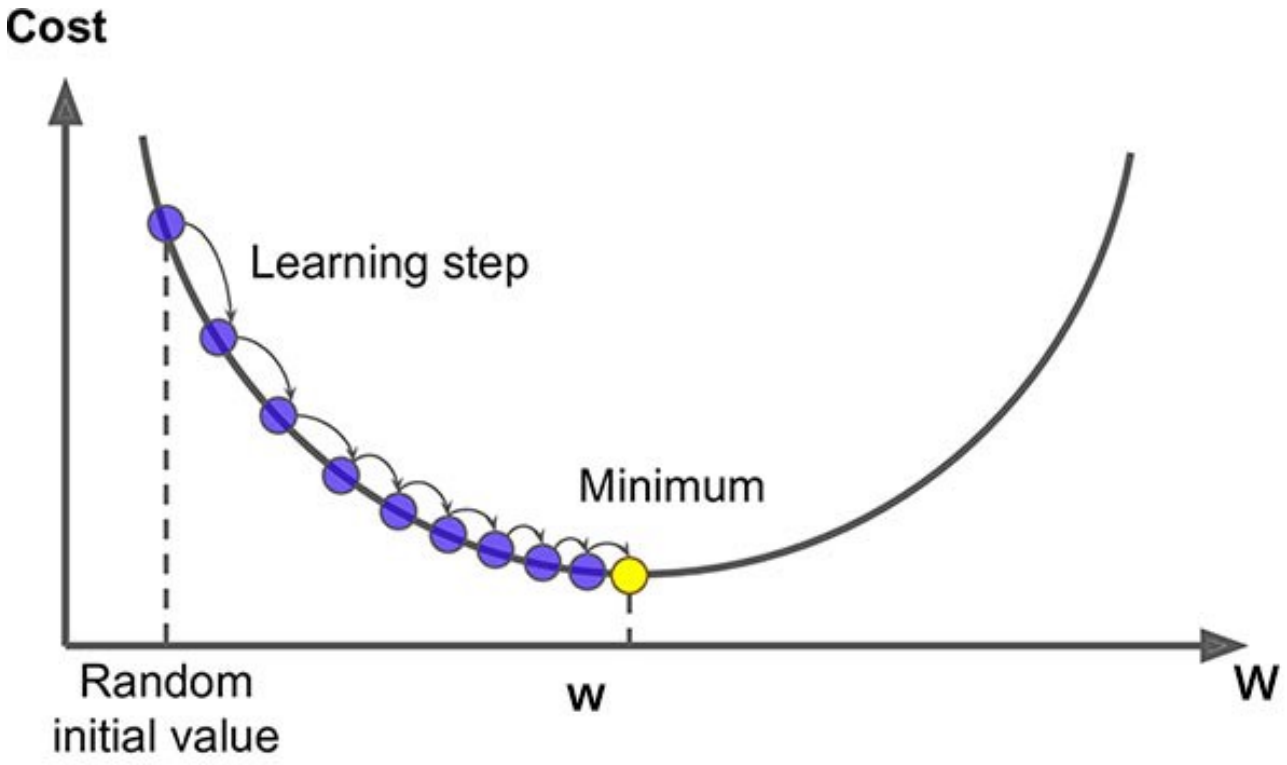
Loss Landscape



If we want to go down a hill, without knowing the full landscape but only the surrounding area.

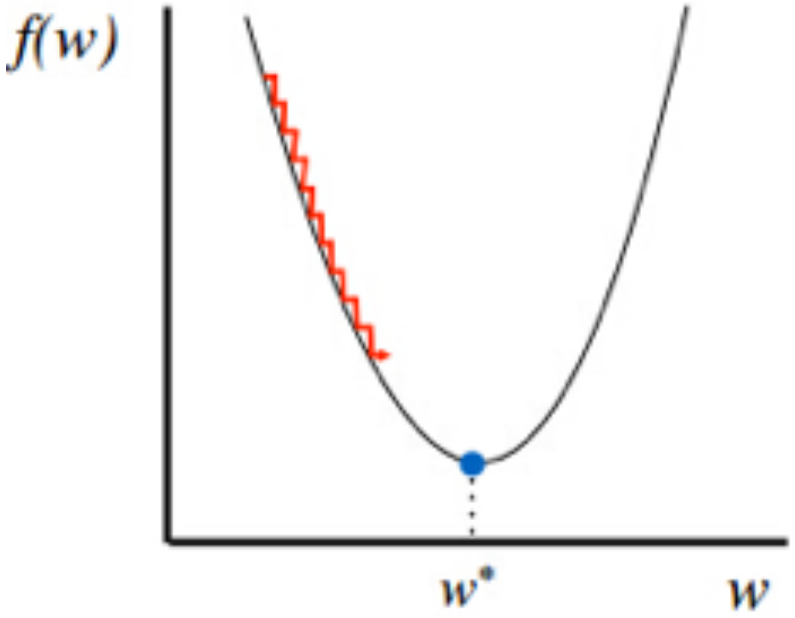
How do we go down?

Gradient Descent

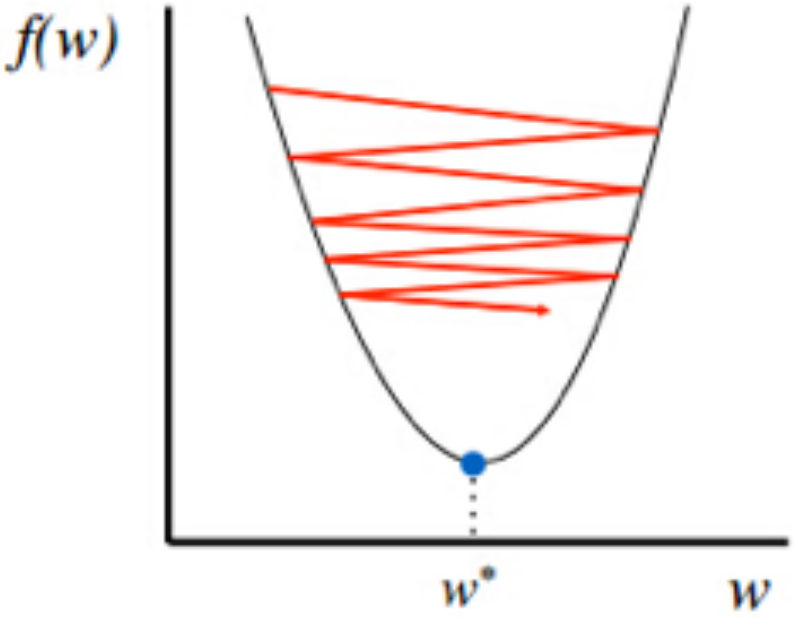


$$w = w - lr \cdot \frac{\partial L}{\partial w}$$

Gradient Descent: Learning Rate

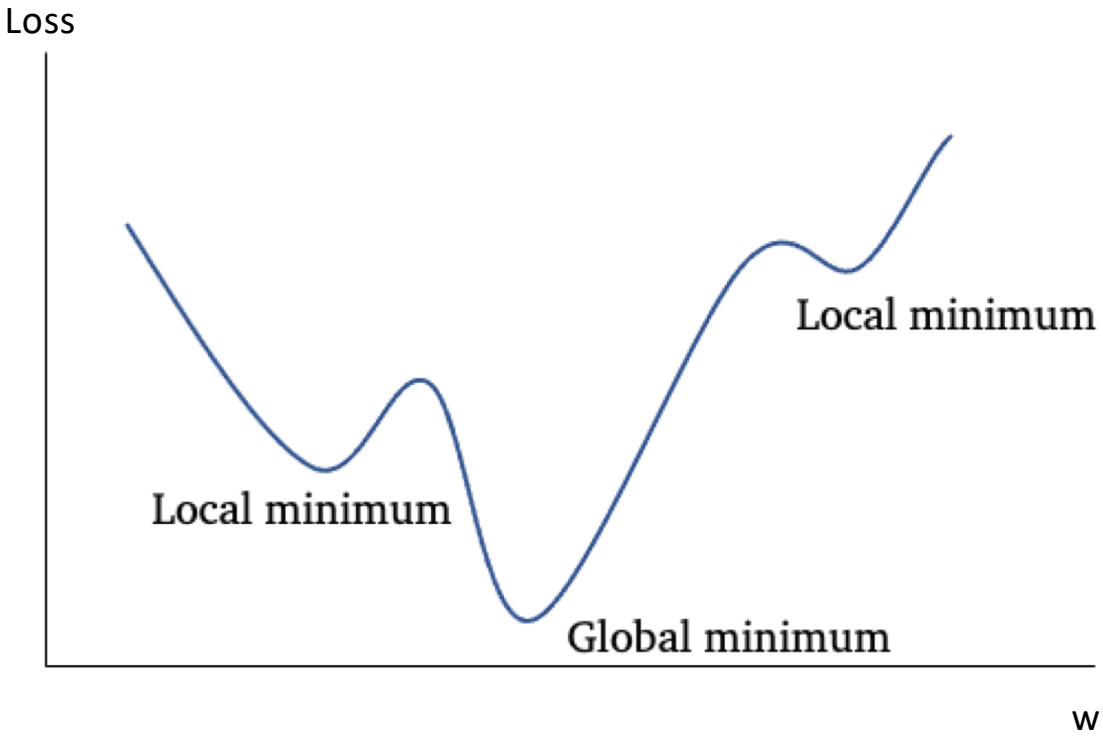


Too small: converge very slowly



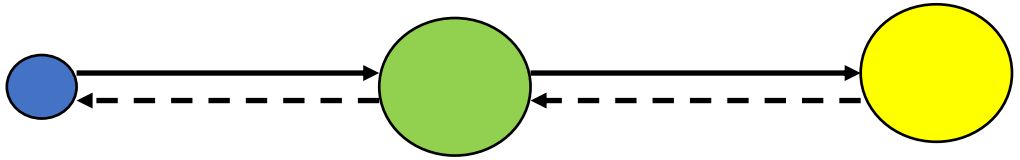
Too big: overshoot and even diverge

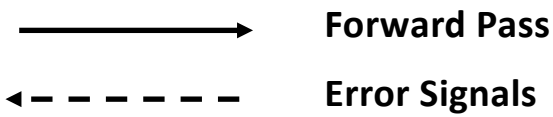
Requirements for Gradient Descent



- Smooth differentiable loss function
- (Hopefully) convex loss landscape
- Forward operations leading to loss function must be differentiable

Backpropagation in Practice

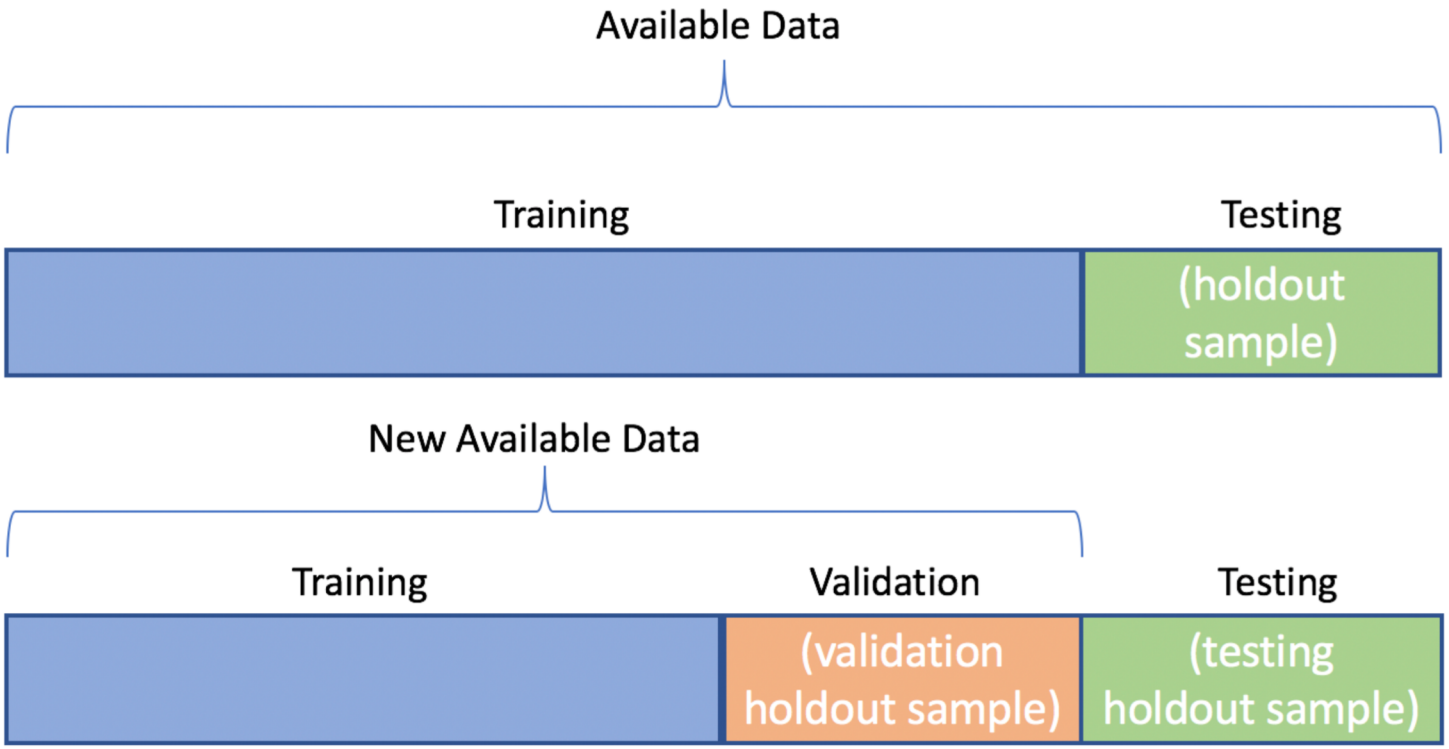




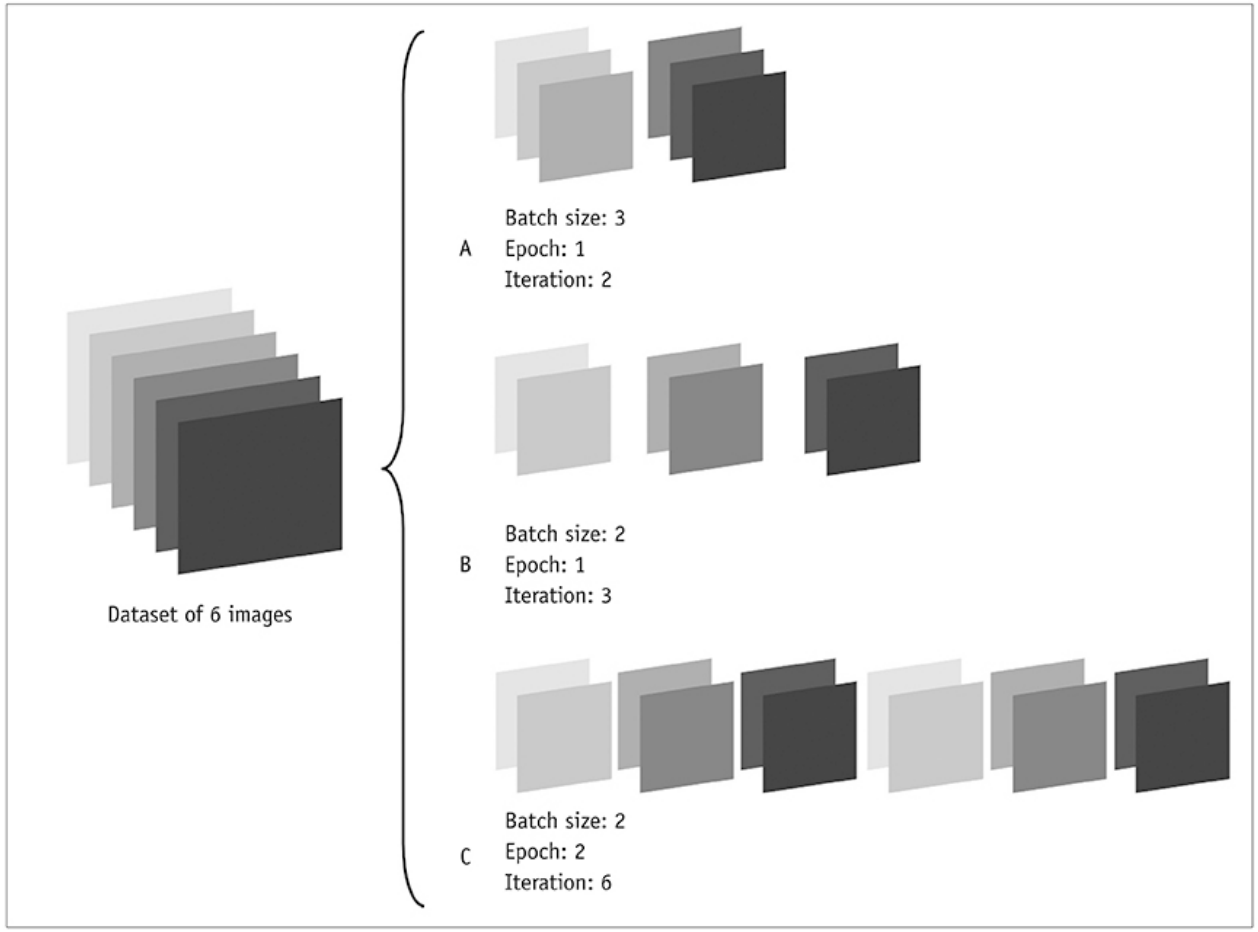
Good News

- All these gradient calculations are done for you automatically if you use automatic differentiation software

Train-Test Split



Batch, Epoch, Iteration

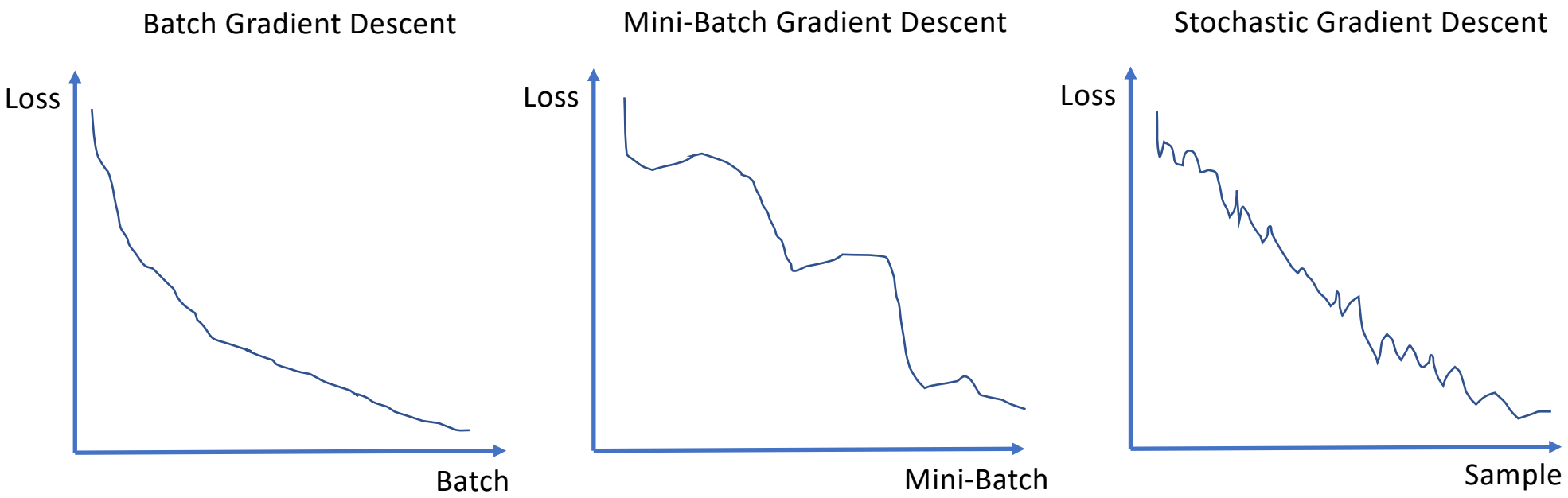


Batch size: Number of samples per pass through neural network

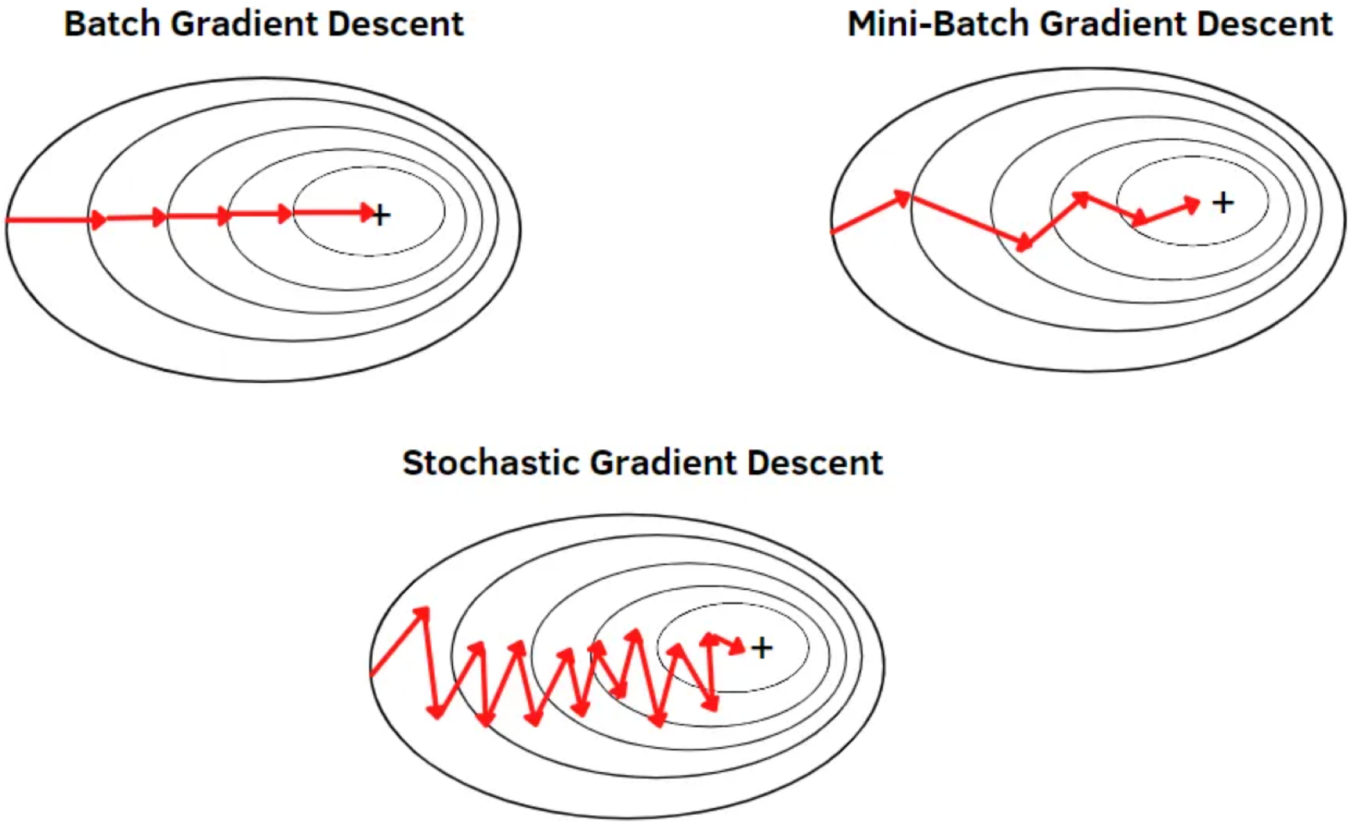
Epoch: Number of times we pass the entire batch of samples

Iteration: Number of passes through neural network

Batch vs Stochastic Gradient Descent

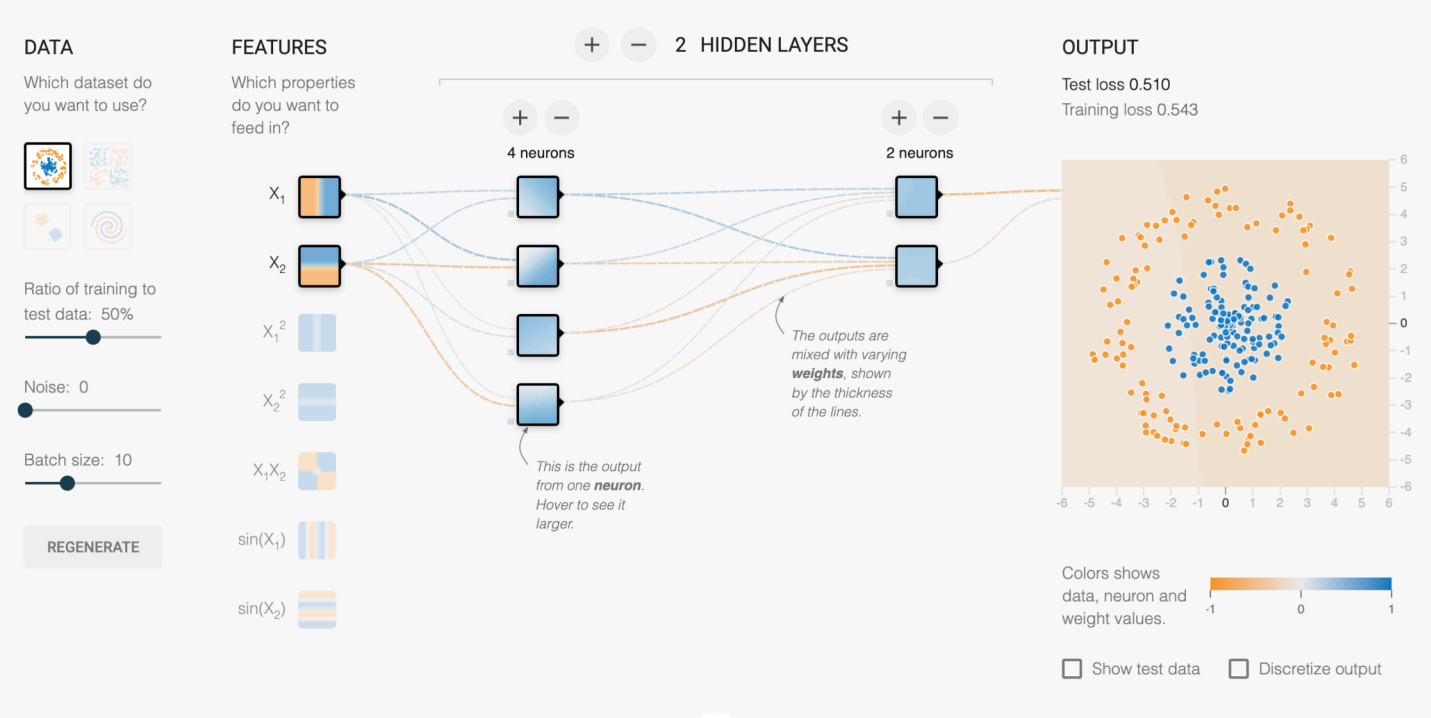


Batch vs Stochastic Gradient Descent



TensorFlow PlayGround

- https://playground.tensorflow.org/



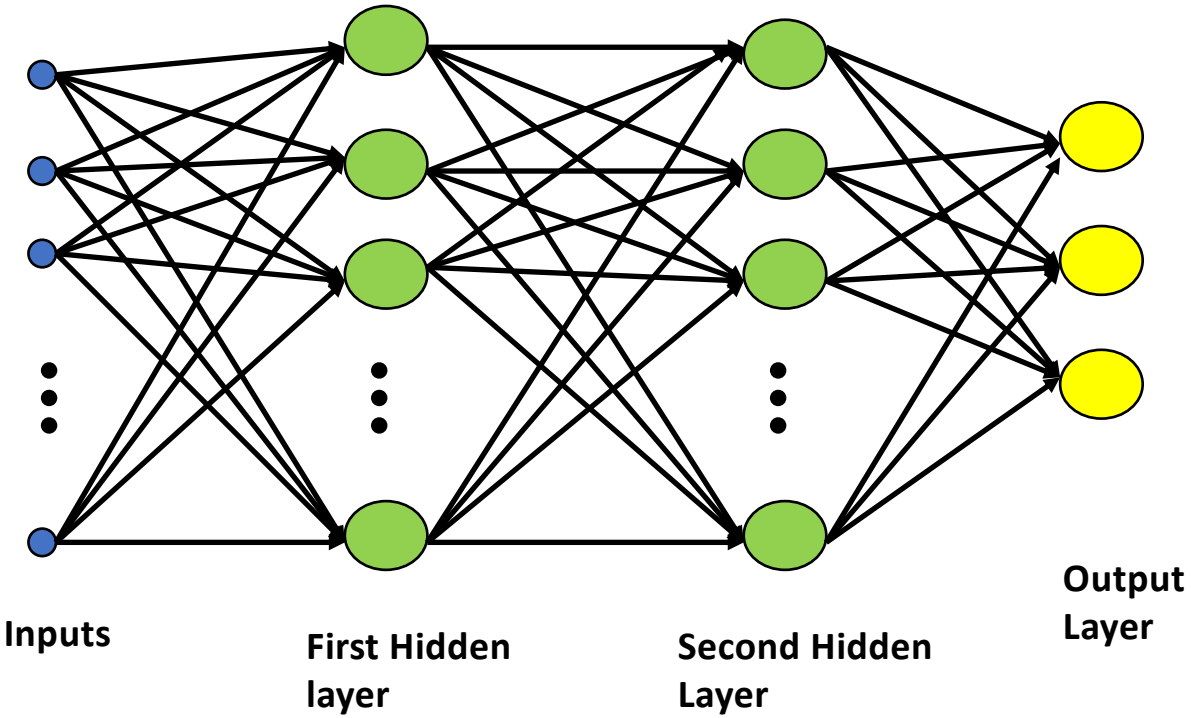
Get Yourself Ready

- Google Colaboratory
  - https://colab.research.google.com/

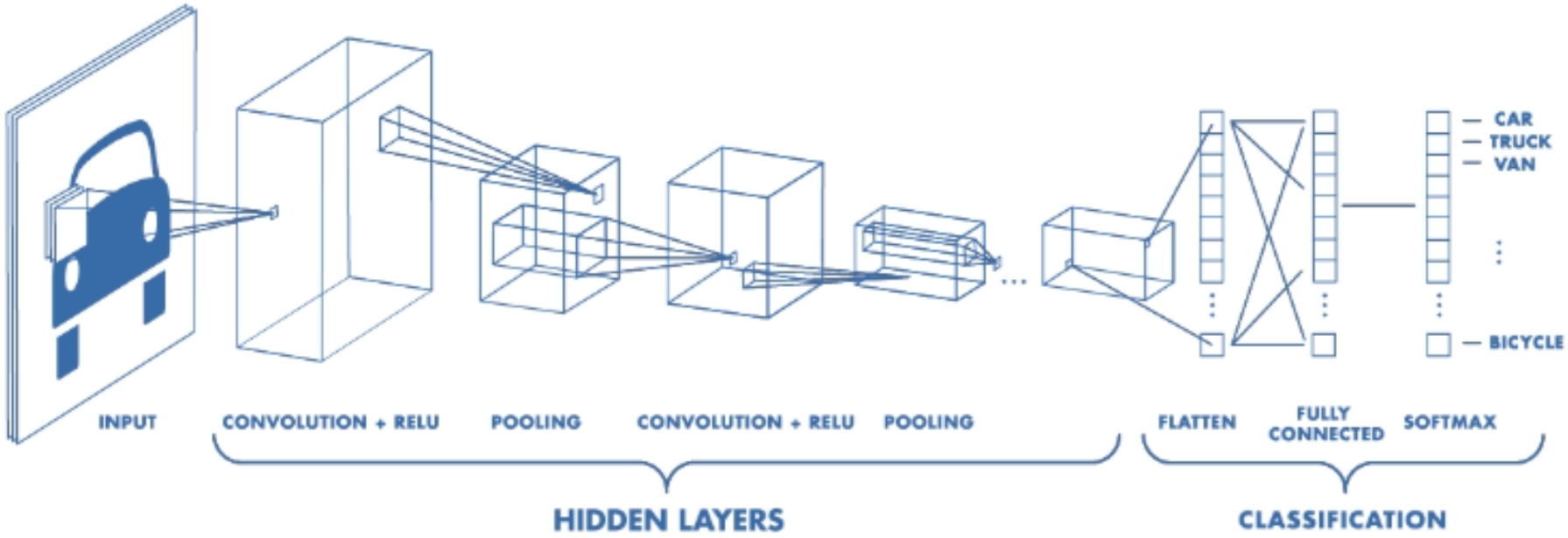
- Next week (Coding Session):
  - Practical coding of MLPs on Fashion MNIST dataset
  - Learn to code using TensorFlow

Types of Neural Networks

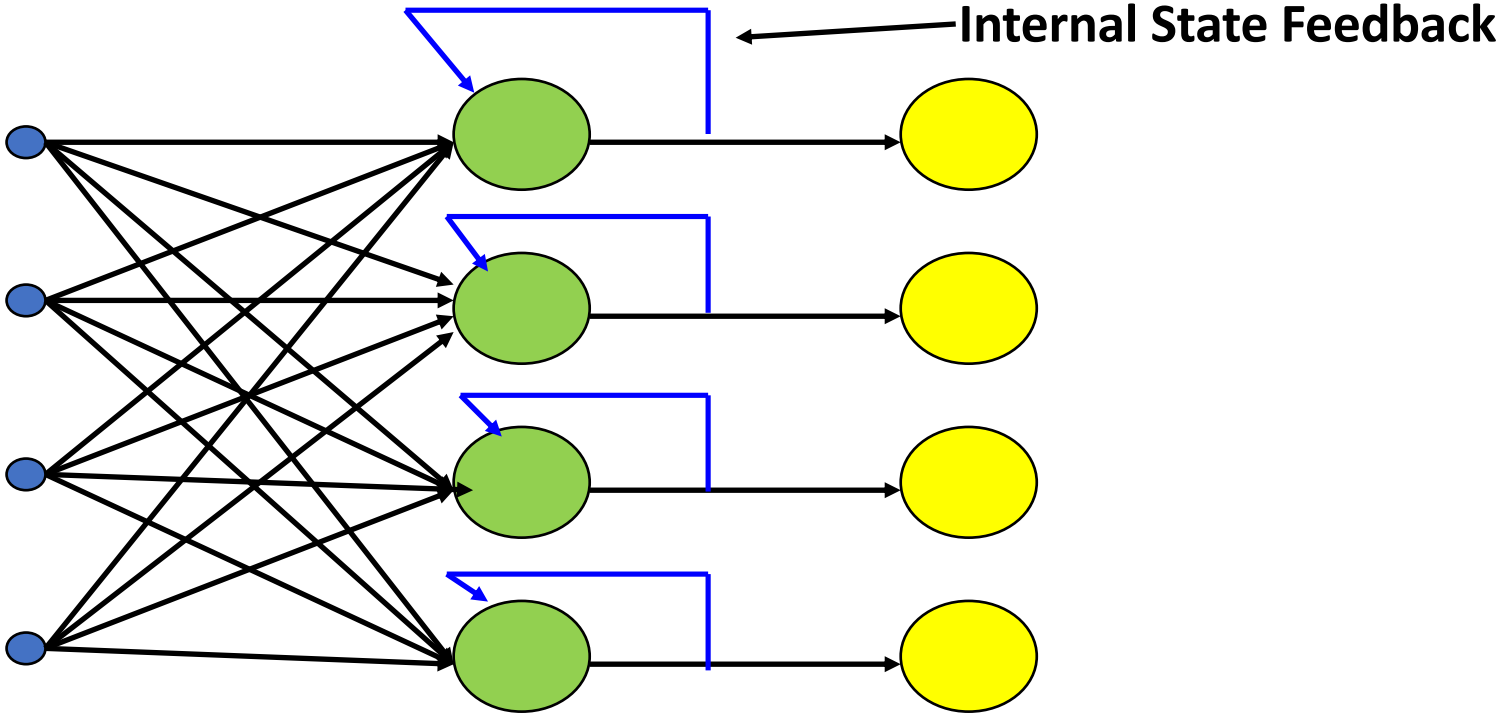
Week 2: Multilayer Perceptron



Week 3: Convolutional Neural Networks



Week 4: Recurrent Neural Network



Recurrent Networks

Potentially Future Session: Transformers

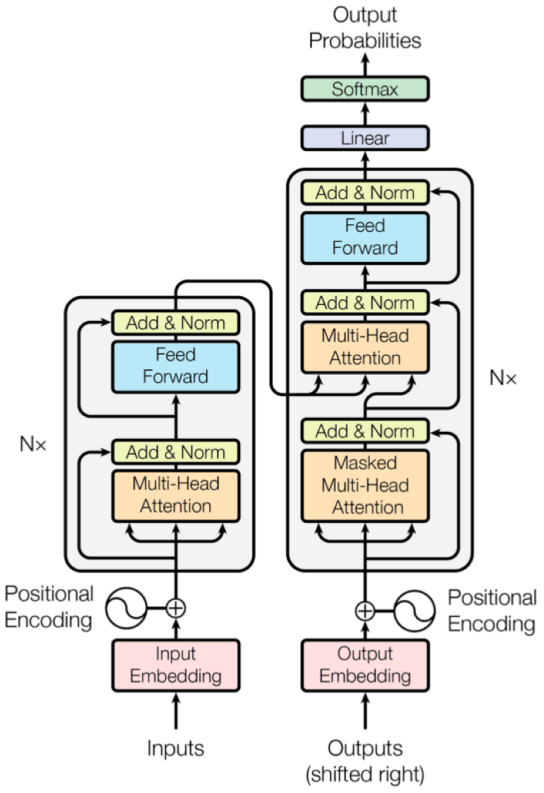


Figure 1: The Transformer - model architecture.

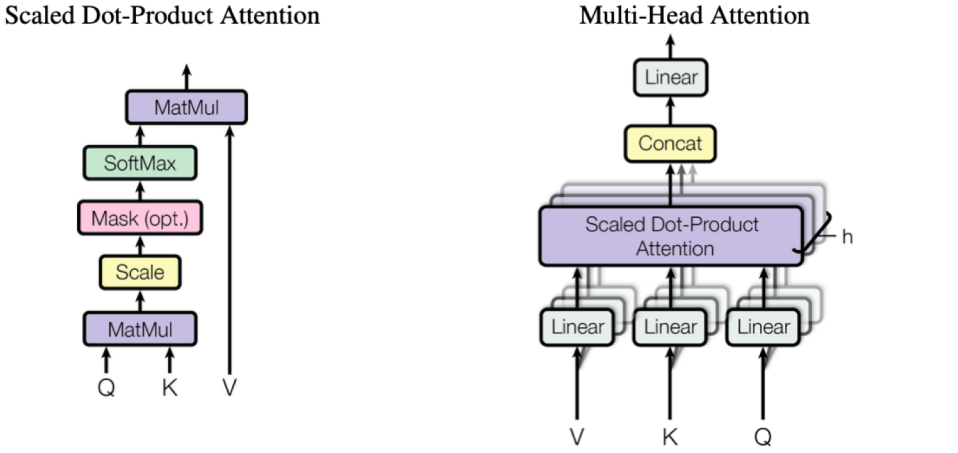


Figure 2: (left) Scaled Dot-Product Attention. (right) Multi-Head Attention consists of several attention layers running in parallel.

$$\text{Attention}(Q, K, V) = \text{softmax}(\frac{QK^T}{\sqrt{d_k}})V$$

Taken from: Attention is all you need (Vaswani et al, 2017)

Questions?