Lecture 16 : Neural Networks - Introduction

Instructor: Aditya Bhaskara *Scribe: Sachin Kumar Singh*

CS 5966/6966: Theory of Machine Learning

March 15th, 2022

Abstract

This lecture quickly summarize Optimization, Improvements, Generalizations and dive into the general idea and notations of neural networks. It discuss expressibility from theory of deep learning(supervised).

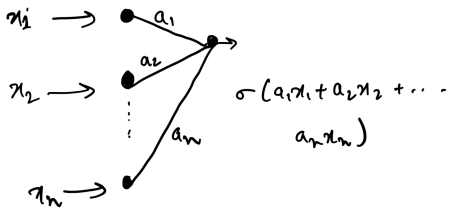
1 Quick recap and Summary

We explored convex optimization, and various settings which guarantees convergence. For smooth functions we can do something more complex than regular gradient descent like Polyak's heavy ball method, second order methods. The main idea is to make faster progress in gradient decent and attempt to slip out of saddle points. If you are not making much progress via gradient decent just make a random jump to neighborhood.

Distributed optimization, Asynchronous parallel gradient decent, online optimization(closely related to stability, regularization) are very close to what we discuss and these can be the direction for further reading and projects.

2 Neural Networks

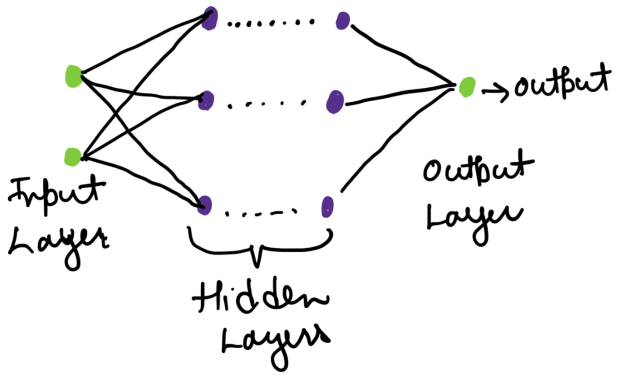
The earliest neural net was perceptron(1950s) it was based on biological neuron.



For some feature or inputs $x_1, x_2, x_3...x_n$, perceptron takes these input and compute $a_1x_1, a_2x_2, a_3x_3...a_nx_n$, where $a_1, a_2, ..a_n$ as weight of the connection or the strength of the connection. This is feed to a threshold function $\sigma()$, where $F(z) = \sigma(Az + b)$ it fires or produce result when a threshold is cross. We can also visualize it as a logic circuit (not Boolean).

In general complex concepts can be expressed with the help of simple combination of basis concepts. For example edge detector or finding different edges can be termed as a basic concept which can be used to build up a alphabet or shape detector by detecting multiple edges.

The neural net consist of input layer, output layer as well as hidden layers. The input layer receives the input form outside. The input is then passed through



one or more hidden layers, which transform the input into data that is fed to output layer. Finally, the output layer provides an output.

Neural networks possess different layers which consist of neurons, neurons is two adjacent layers are connected with a certain weight w. For each neuron the input is multiplied by the strength of the connection. Neural networks take a input vector R_d, feed them through the sequence of these layers and then generate an output. The output of a particular layer can be of any dimension and is independent of the input dimension d.

For a neuron with input vector $x = [x_1, x_2, ...x_n]$, weight $w = [a_1, a_2, ...a_n]$ with function F produces the output $F(a_1x_1 + a_2x_2.....a_nx_n)$. This output is feed forward to the next neuron. For the output y which is produced after r layers will be $y = F_r(F_{r-1}(F_{r-2}....F_1(x)))$.

The activation function calculates a weighted total and then adds bias to it to determine whether a neuron should be activated or not. The activation function's goal is to introduce non-linearity into a neuron's output. Some of the common activation functions are :

- Threshold
- Sigmoid
- ReLU
- Tanh

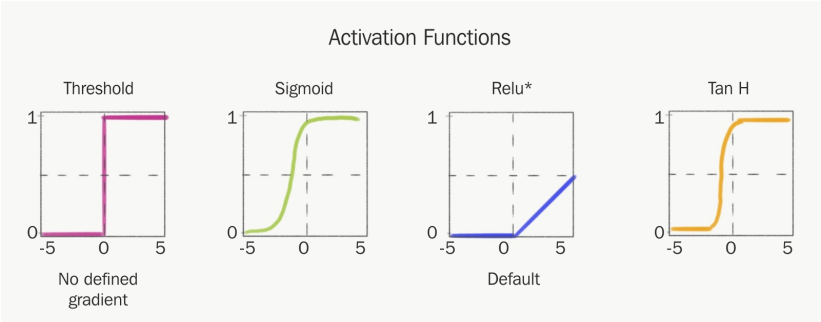


Figure 1: Various Activation Function(Source : Internet)

Neural networks are basically a (fairly complex) hypothesis class – takes input x, produces y. For given data $(x_1, y_1), (x_2, y_2)$... from some distribution D, find

h in this class that minimizes the risk such that $h(x_i) \approx y_i, \forall i$. ERM problem usually called neural network "training" – given data, find best fit classifier. Non-convex optimization problem (due to non-linearity), NP-hard even in very simple cases but a loss function can be approximated such as square loss.

3 Theory of Deep Learning (Supervised)

We can classify all the research in this area into three categories - Expressibility, Training complexity and training dynamics for GD and variants Complexity of associated ERM problem and Generalization.

Expressibility - The main motivation is that if we define a particular hypothesis class, what is special about this class and why not some other hypothesis class. This allow to show that all function can be approximated via neural nets with any non-linearity. There are two theorems related to expressibility Barron's theorem and Cybenkos's theorem.

3.1 THEOREM. *According to Barron's Theorem if you have a square integrable continuous function $f : R_d \to R$, and it has a niceness parameter C, then for any $\epsilon > 0, \exists$ a 2 layer neural network with C^2/ϵ internal nodes such that -*

$$\int |f - h|^2 dx \leq \epsilon \text{ :(where h is the output)}$$

In other words, any continuous function f that satisfies an appropriate "niceness" condition (parametrized by C) can be approximated to error ϵ by a 2-layer NN with C^2/ϵ internal nodes.

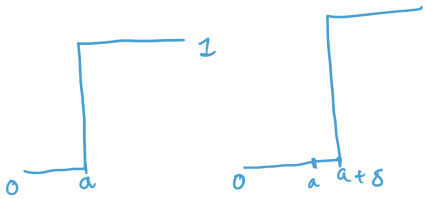
3.2 THEOREM. *According to Cybenko's Theorem any continuous function f can be approximated(even pointwise) by a NN with 2 layers with width depending exponentially on dimension and ϵ parameters. This make fewer assumptions as compared to barron's.*

Let say you have one dimensional function in a finite domain D. It has been divided into non-intersecting intervals $I_1, I_2 ... I_n$ so domain will be $I_1 \cup I_2 ..... \cup I_n$.

graph

$$f(x) \approx \sum_{j=1} n f(x_i) 1(x \in I_j)$$

Further $1(x \in I_j)$(1 is the indicator) can be expressed in terms of σ, depending upon σ.



For example when σ is threshold, for two functions one which gets triggered at a point a and other at point $a + \delta$ it can be expressed as -

$$\sigma(x - a) - \sigma(x - a - \delta) = 1(x \in (a, a + \delta))$$