Lecture 17: Neural Networks

Instructor: Aditya Bhaskara *Scribe: Rahul Thomas Benny*

CS 5966/6966: Theory of Machine Learning

March 23rd, 2022

Abstract

This lecture deals with the history, fundamentals and concepts of neural networks and their significance in modelling a large family of functions and capturing a large variety of features within data.

1 Introduction

Earliest form of the neural networks was the perceptron which came about in the 1950s. A perceptron models a single biological neuron by taking in a weighted combination of inputs and firing when this combination is greater than a certain threshold. For example, considering that the perceptron has inputs $x_i, x_2...x_n$, the weighted combination of these inputs would be $a_1x_1 + a_2x_2 + ...a_nx_n$. The output of the perceptron would be represented as $\sigma(a_1x_1 + a_2x_2 + ...a_nx_n)$ where $\sigma()$ is a threshold function. It can be seen that a perceptron generalizes the linear threshold class of functions. This can also be seen as a logic circuit due to it's binary nature. Perceptrons cannot themselves be used to express a large class of functions. Due to this reason, combinations of these perceptrons were explored since each of the perceptrons' outputs could be seen as detecting a basic feature in the data. So, a combination of perceptrons was expected to show better ability in capturing multiple features which led to the large neural networks we know today.

2 Artificial/Deep Neural Network

A deep neural network can be defined as a layered 'circuit' that takes a vector of input features x, produces output $y = F_r \circ F_{r-1} \circ \cdots \circ F_1(x)$, where each F_i is a function of the form $F_i(z) = \sigma(Az + b)$, for some activation function $\sigma()$(that acts coordinate-wise). As seen in Fig.1, each of the perceptrons in this network, fire based on whether the weighted combination of inputs to them is larger than the threshold or not. The outputs of the perceptrons in one layer become the inputs for the perceptrons in the next layer and so on. As shown in the image, if the first layer has n_1 perceptrons, F_1 produces a vector of size n_1 as output of the form $\sigma(Ax + b)$ where A is a matrix representing the weights and b is the bias vector. This then becomes the input for the next layer and so on leading to the notation $F_r \circ F_{r-1} \circ \cdots \circ F_1(x)$. Note here that $\sigma()$ is generally considered to be the zero-centered threshold function but it doesn't have to be. Some of the other popular activation functions are:

1. Sigmoid

2. ReLU

3. Tanh

These non-linear functions are used to introduce non-linearities and increase the expressability of the neural networks thereby not limiting it to expressing linear functions only.

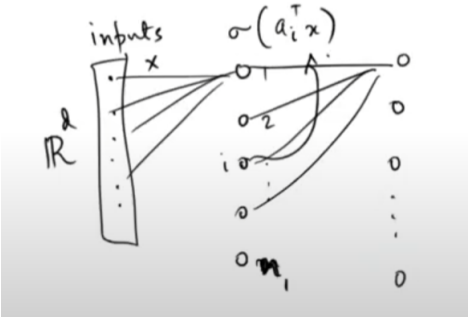

Figure 1: Diagram of the layers of a neural network

3 Basic Goal of Neural Networks

Neural networks are basically a fairly complex hypothesis class since it takes an input x and produces an output y. For eg: Class of all neural networks with r layers and n_1 neurons in the first layer, n_2 neurons in the second layer and so on. Supervised learning can be performed with this hypothesis class. The basic ERM problem is to find the best fit classifier in this hypothesis class given the data. This process can also be called neural network training in this context. This is a non-linear non convex problem that is done using a loss function such as square loss since the ERM problem itself is NP-hard.

3.1 *Expressibility*

All functions can be approximated via neural networks with basically any non-linear function σ. This means this class of functions is very expressible. This is why they are called universal approximators.
Barron's Theorem:

3.1 definition. If you have a square integrable continuous function f, $f : \mathbb{R}^d \to \mathbb{R}$, and it has a 'niceness parameter' C, then for any $\epsilon > 0$, there exists a neural net(2 layers) with $\approx \frac{C^2}{\epsilon}$ parameters such that $\int |f - h|^2 dx \leq \epsilon$ where h is the output of the neural network. This is true when activation functions are sigmoid or threshold.

Cybenko's Theorem:

3.2 definition. Any continuous function can be approximated(even point-wise) by a neural net with 2 layers with width depending exponentially on dimension, ϵ parameters. This holds for any non-linear activation function.

3.2 *Training Complexity and Training Dynamics for Gradient Descent and variants*

This is a highly researched area related to neural networks. This has to do with the complexity of training neural networks. While the basic ERM problem itself is NP-hard, using a loss function to train these networks produces impressive results. Multiple other methods than Gradient Descent has been tested to study the training dynamics but mostly gradient descent seems to continue to out perform them and is still one of the best neural network training algorithms.

3.3 *Generalization*

Generalization is the most desired property just like it is for most machine learning tasks. We know that VC-dimension $\approx$ number of parameters which is true for neural nets also. This helps us create a generalization bound if the ERM problem is solved well but this is a catch since we do not know how well we can solve ERM. These bounds are then fairly weak. Number of samples required to get generalization error of $\epsilon = Dlog(D/\epsilon)/\epsilon^2$ where D is the VD-dimension. This means that the dataset size should be considerably larger than the VC-dimension. In practice this is rarely realizable since the number of parameters in most neural networks are almost same or more than the number of samples. So, there is a requirement for a stronger bound than the one given by the VC-dimension. This is one of the important research areas within the field of deep learning.

4 Hand wavy Proof for Cybenko's Theorem

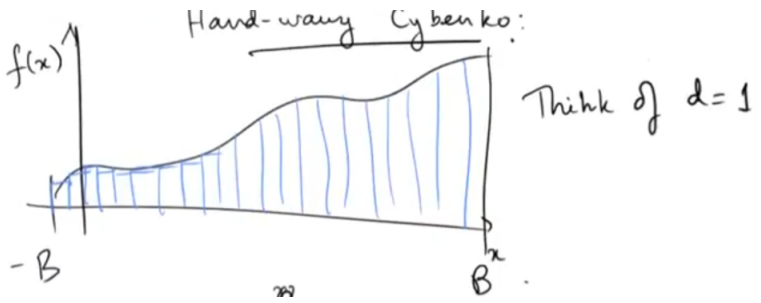

Figure 2: Function approximation

Approximating a continuous function $f(x)$ using rectangles, we get something like Fig.2. If the width of each of those rectangles are considered to be intervals($I_1, I_2, ...$), the domain $[-B, B] = I_1 \cup I_2 ... \cup I_n$
So, $f(x) \approx \sum_{j=1}^{n} f(x_i)\mathbb{1}(x \in I_j)$ where $x_i \in I_i$ This approximation is good because f is continuous Now, it only remains to show that the indicator function in the above approximation can be expressed in terms of the activation function used. This can be shown for different activation functions. For eg: For threshold functions, $\sigma(x-a) - \sigma(x-a-\delta) = \mathbb{1}(x \in (a, a+\delta))$ This can be shown for any of the non-linearities. This is a simpler proof for Cybenko's Theorem.