The Life of an Unemployed Postgrad Trying to Design a Neural Network from Scratch

Peter Daniel George Davidson

December 12, 2024

Abstract

This paper chronicles the brave and questionable journey of an unemployed postgrad attempting to design a neural network from scratch. With little more than a vague recollection of neural network theory and a questionable amount of caffeine, the author embarks on an intellectual quest to make a machine "learn" from data, only to realize that, much like their job search, it might take longer than expected (and involve more bugs).

1 Introduction

It's a Thursday morning. I've been unemployed for six months. My resume is as fresh as my mind after woking 3 days in the postoffice with a pass remarcable chinese women called Mae who cant speek english yet most of our conversation includes her searing at me in cantonese. So....

On my day off I've got nothing to do except binge-watch coding tutorials on YouTube, because, clearly, learning to build a neural network from scratch is a *great* use of my time.

But here I am, attempting the impossible. With no job prospects in sight, apart from a pending inerview from Thales, I'm deciding to give the world of machine learning a shot. After all, how hard could it be to design a neural network? Sure, I may have a fleeting recollection of backpropagation from a lecture I barely stayed awake for (malware analysis @ QUB 2024), but what's the worst that could happen?

Spoiler alert: I have no idea what I'm doing.

This paper will explore the ongoing (and often laughable) journey of creating a neural network from the ground up. Through trial, error, and frequent existential crises, I will attempt to build something that can learn, predict, and maybe—just maybe—make me look like I have my life together. But first, I'll need to figure out what the heck a "neuron" is again.

2 Lets try and start

2.1 Day 1

I have started off my watching a YouTube video on the bacis of neual networks from what I understand neural networks are made up of four things; Nodes, Layers, weights and biases.

1. **Neurons (Nodes)**:
   Neurons are the basic building blocks of a neural network, and they're kind of like the brain's neurons, just in a digital form. Each neuron takes in information (called inputs), does some basic math with it, and then spits out a result. This result is sent to other neurons for further processing. The math usually involves adding up the inputs, multiplying them by certain values (called weights), and then running the result through a function to decide what the neuron will output.

2. **Layers**:
   A neural network is organized into layers of neurons. There are three main types of layers:

   - **Input Layer**: The layer that receives the initial data or features of the problem you're trying to solve (e.g., pixel values of an image, features of a dataset).
   - **Hidden Layers**: Layers between the input and output that perform most of the computation. A neural network can have one or more hidden layers.
   - **Output Layer**: The layer that produces the final result, such as a classification label or a predicted value.

   Analogy: Think of layers as different stages in a factory assembly line. Each layer takes in something, processes it, and passes it on to the next stage for further refinement.

3. **Weights and Biases**:
   Weights and biases are parameters that control the learning process of a neural network.

   - **Weights**: Each connection between neurons has an associated weight, which determines the importance of the input in the decision-making process.
   - **Biases**: Bias terms allow the network to make predictions even when all the inputs are zero. It helps adjust the output independently of the input.

   During training, the network adjusts the weights and biases to minimize the error in its predictions. This is done through a process called *backpropagation* and *gradient descent*.
   Analogy: Weights are like the volume knobs for each connection—how much influence does this connection have on the output? Biases are like the "base" level setting—where the system starts, even if there are no inputs.

After resurching some incredibly complacated formulas the task ahead looked, for lack of a better word terrible. thankfully when we convert these formulas into python, they become wayyyyy less daunting.

From here I began to code....

he makes a .py file just after typing this

Simulated the poutput of one sigle neuron, this could be from origional values of something we are trying to model, or could be the outputs from nodes in past layers.