CSE574 Introduction to Machine Learning

Project 1.1: Software 1.0 Versus Software 2.0

Project Report

Submitted By

Karan Manchandia

Abstract

This project aims to compare the Logic Based Approach (Software 1.0) and The Machine Learning Approach (Software 2.0) for Software Development, in order to solve a particular problem. The task is a Fizzbuzz problem.

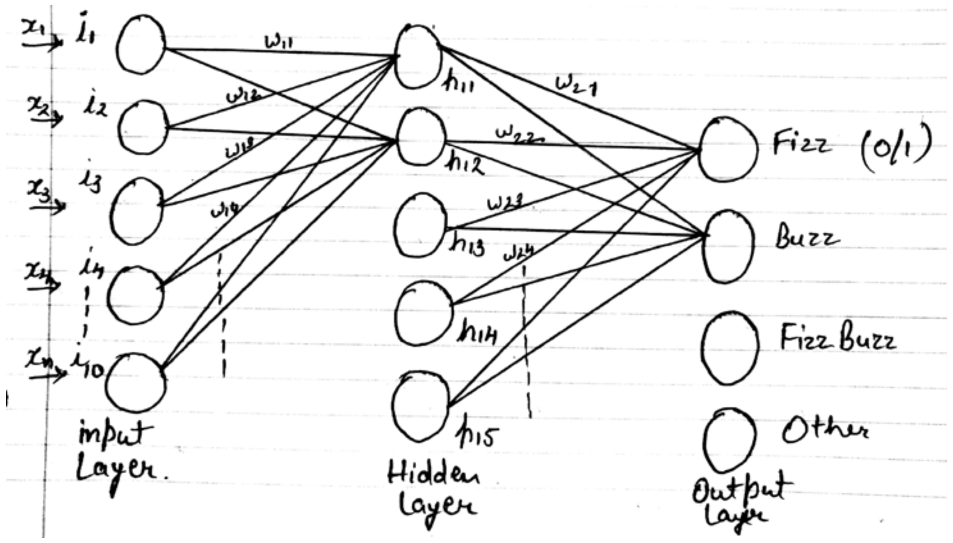
Software 1.0 solves the Fizzbuzz problem using a standard logic in python (code attached) and Software 2.0 solves the same problem by using a Neural Network Model (Tensorflow) that would classify the integers in four buckets Fizz, Buzz, Fizzbuzz and other.

Introduction

In software 2.0 we have understood the code for implementing a three layer Neural Network model using the tensorflow library. The Neural Network Model has 10 input neurons, 4 output neurons and a hidden neuron layer between the input and the output layer.

Why 10 input neurons and 4 output neurons?

Ten input neurons are required because the input decimal intigers can vary from 1-1000 and it requires 10 bits input when converted to binary. Four output neurons are required because the input integers are to be classified in four buckets. The figure below shows the block diagram of the Neural Network Model.



I have tuned the values of hyperparameters in order to find the most optimal value for these hyperparameters, so as to get the most accurate results.

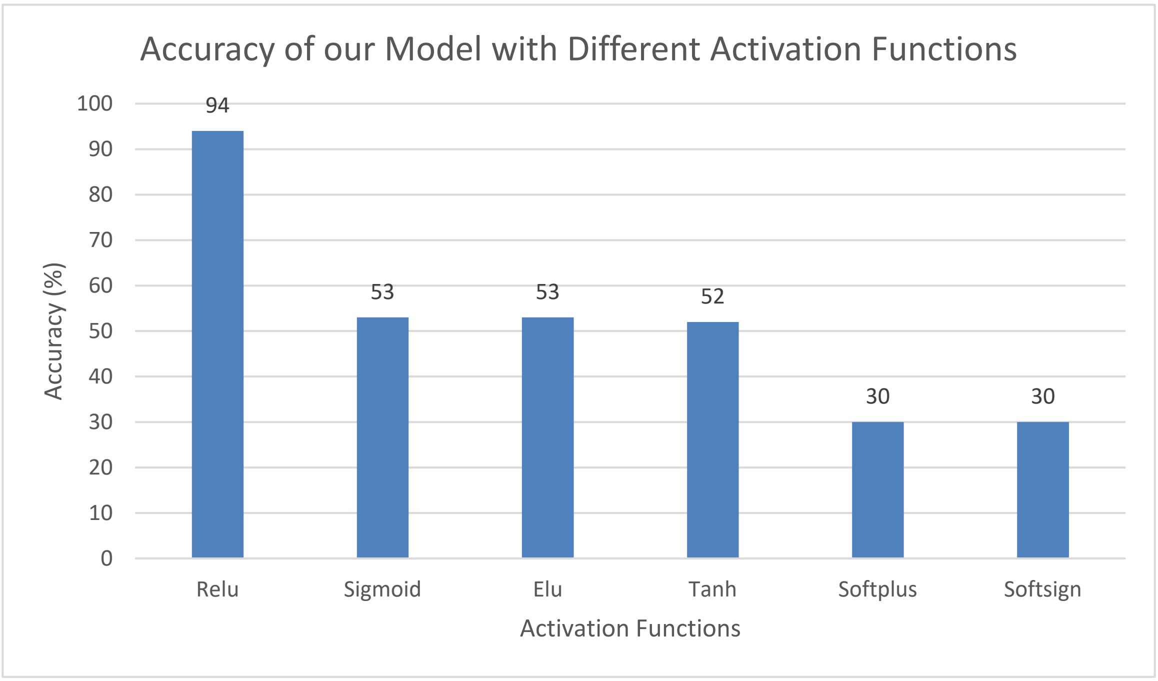
Changing the Hyper-parameters to get the most optimum values for best accuracy

The variables which determine the network structure along with the variables which determine how the network training is done are called hyper-parameters.

Examples: Activation Function, Learning Rate, Number of epochs, Batch size etc.

1. The relationship between Activation Function and the accuracy

The functions that are used to introduce non linearity to the model are called activation function. The following graph shows the accuracy of the Neural Network for different Activation Functions.



The highest accuracy obtained for the fizzbuzz problem is with Rectified Linear Unit Activation Function.

Why Relu?

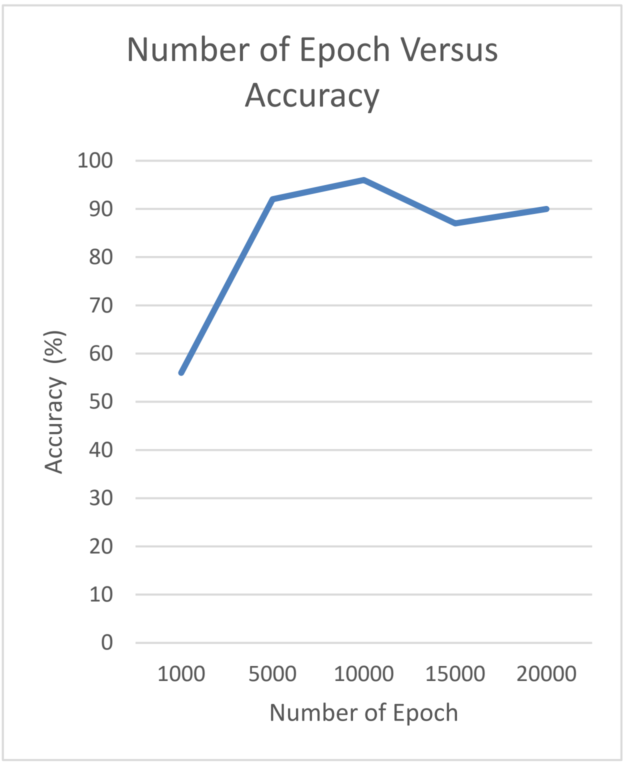
The following are the reasons, why the Relu Activation Function is the best fit for our Fizzbuzz problem:

- The gradient is 1 for all possible positive values of input. This has made the algorithm called Gradient Decent work much faster.
- Relu gives zero output for negative values and positive output for all the positive inputs.
- Learning rate is fast when using Relu as compared to sigmoid function.
- Relu gives the most accurate result in the fizzbuzz problem.

2. Effect of changing the number of epoch on Accuracy and the Training plus Testing time.

An Epoch is defined as a complete iteration of data set to be learned to a learning machine. As we increase the number of epoch, it is expected that accuracy would increase. Also, as the number of iterations of training data increases, the Training plus Testing time increases.

Number of Epoch	Accuracy (%)	Training plus Testing time (seconds)
1000	56	6
5000	92	31
10000	96	49
15000	87	79
20000	90	105

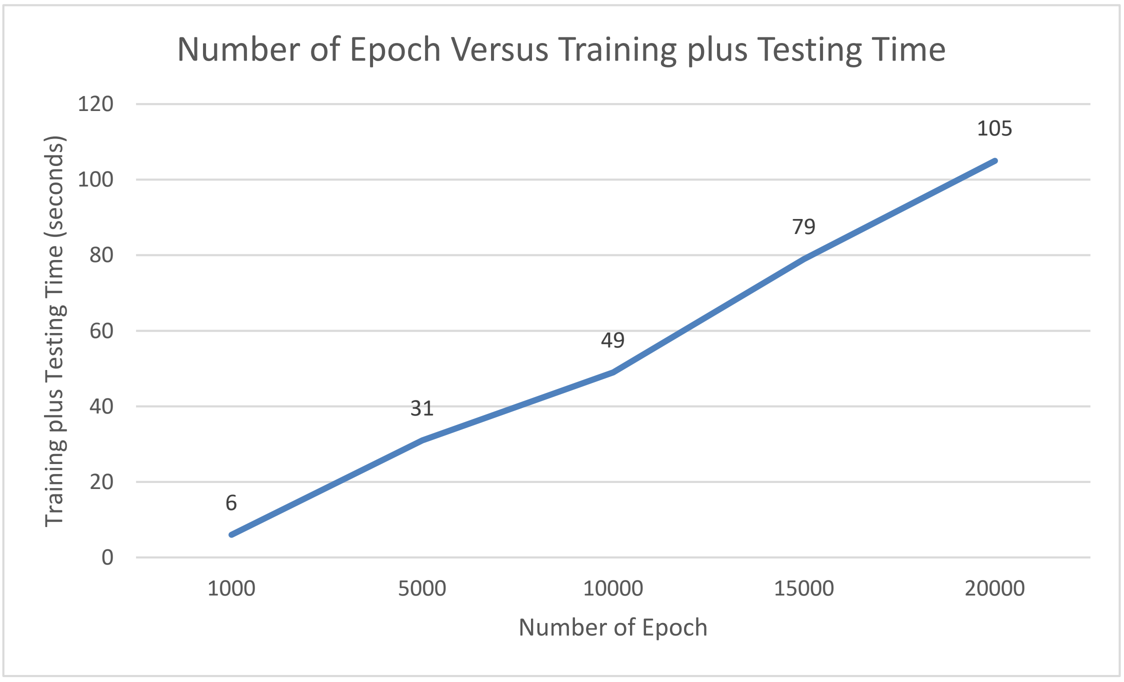


It can be seen from the data presented in the graph below that the accuracy decreases as we move from 10,000 to 15,000 epoch. This irregularity is because of a concept called overfitting.

Why Overfitting? Why overfitting is bad in Machine Learning?

When we set epochs to larger number without early stopping, it will cause the problem of overfitting. This will cause further classifications to be error-some. If you train your model to fit each and every data, the model would not have learnt, how to deal with data that it is not trained on.

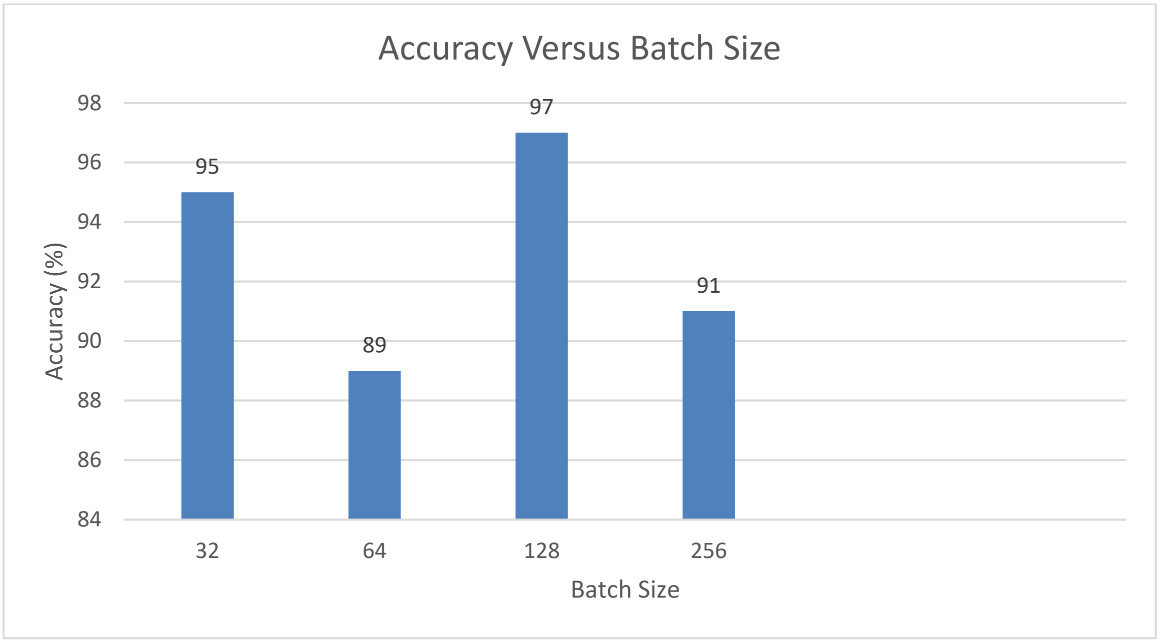
Overfitting is bad since this does not generalize the training model and accuracy on new test data may not be very great.



This graph shows that the Training plus Testing Time increases as the number of epoch increases.

3. Effect of changing the Batch Size on the accuracy of the model

The effect of changing the batch size cannot be understood correctly with the FizzBuzz problem. It has been generally observed that with a larger batch size, The generalization ability of the model greatly decreases. It can be clearly seen in the graph below that we are getting maximum accuracy for Batch Size = 128.

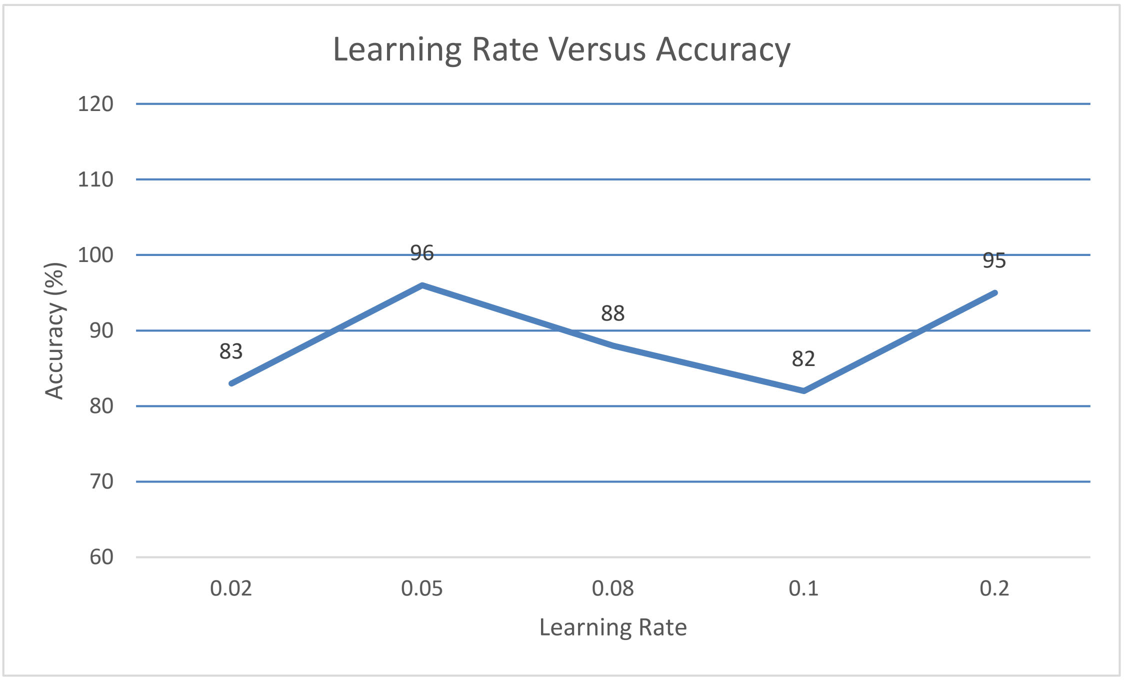


What is Generalization?

It is the ability of the model to correctly predict in the case of odd circumstances, in which the model has not been trained on. For Example, a face recognization system should be able to correctly predict a human face living on earth and an alien that is to land on earth in the year 2050.

4. Effect of changing the Learning Rate on the accuracy of the model

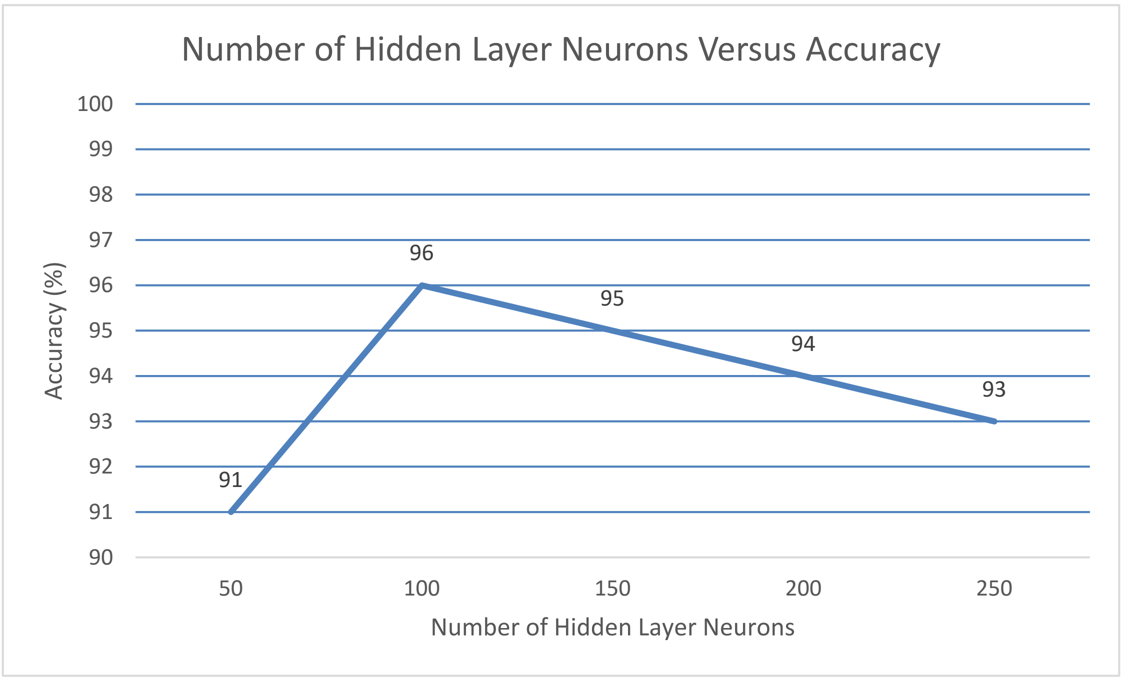
As the name suggests, Learning Rate defines, how quickly a model updates its parameters. We get different accuracy as we update the Learning Rate, which is depicted in the figure below.



The highest accuracy obtained is at Learning Rate = 0.05.

5. Effect of changing the Number of Neurons in the Hidden Layer on the accuracy of the model

Conceptually speaking, as we increase the number of neurons in the hidden layer, the accuracy of the Neural Network Model should increase. This is because of the increase in the complexity and precision of the model. The graph for accuracy versus a number of hidden layer neurons is shown below. It can be seen from the graph below that with an increasing number of hidden neurons, there is an increase in accuracy till a particular point and after this point, there is no significant change in the accuracy.



Selection of the most optimal Hyperparameters:

As presented in the above paragraphs and the graphs, the following are the most optimal values for the hyperparameters and the accuracy values are presented in the table below:

Hyper-parameters	Value / Type	Accuracy (%)
Activation Function	Relu	94
Number of Epoch	5000	92
Batch Size	128	97
Learning Rate	0.05	96
Hidden Layer Neurons	100	96

Conclusion

For solving the FizzBuzz problem, using the neural network model, the maximum accuracy achieved is 98% by using the hyper-parameters value as mentioned in the table above.

References

- https://towardsdatascience.com/epoch-vs-iterations-vs-batch-size-4dfb9c7ce9c9
- www.fon.hum.uva.nl/praat/manual/epoch.html

- https://towardsdatascience.com/what-are-hyperparameters-and-how-to-tune-the-hyperparameters-in-a-deep-neural-network-d0604917584a
- https://www.tensorflow.org/api_guides/python/nn