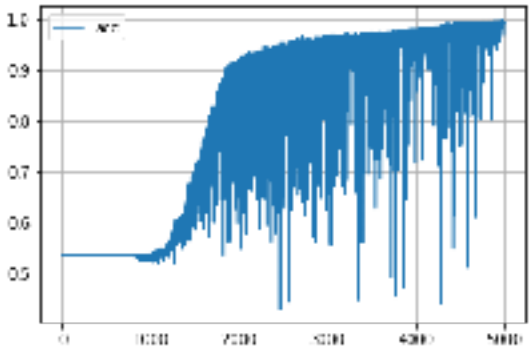
Introduction

In this project, we have implemented the FizzBuzz program using a conventional method using if-else statements under v1.0, and then built this software using machine learning under v2.0. For v2.0, we used neural networks and tried various configurations and hyper parameter values to finally choose the model that performs best for this problem.

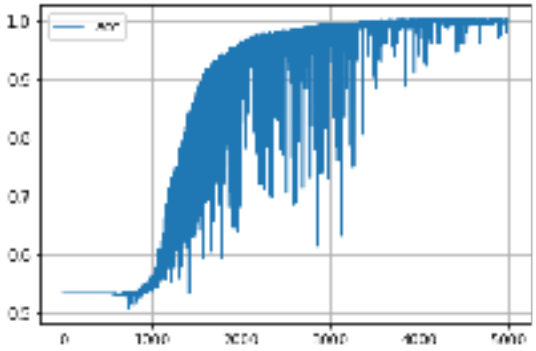
The performance of our model under different configurations is depicted by the graphs below:

Accuracy plot when 1 hidden layer with 100 neurons and relu activation is used:



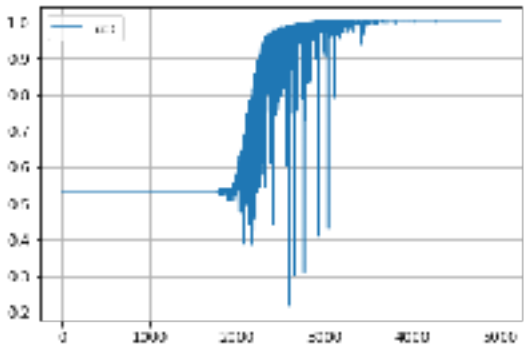
```
Errors: 4  Correct :96
Testing Accuracy: 96.0
```

Accuracy plot when 1 hidden layer with 300 neurons is used:



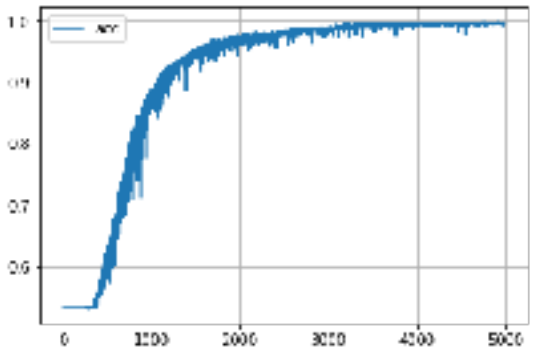
```
Errors: 4  Correct :96
Testing Accuracy: 96.0
```

Accuracy plot when 2 hidden layers with 300 neurons each are used:



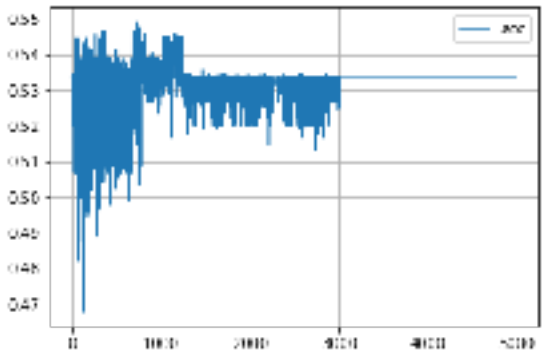
```
Errors: 2  Correct :98
Testing Accuracy: 98.0
```

Accuracy graph vs number of epochs for Adagrad Optimizer for 1 hidden layer 100 neurons.



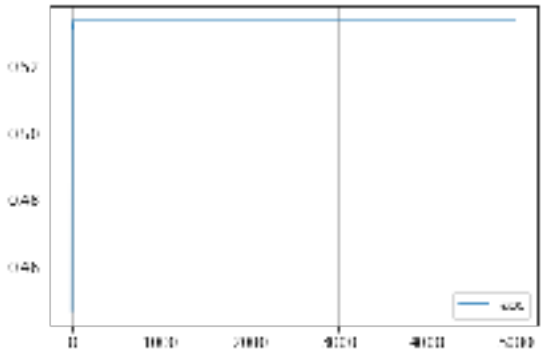
```
Errors: 3  Correct :97
Testing Accuracy: 97.0
```

Accuracy graph vs number of epochs for Adam Optimizer for 1 hidden layer 100 neurons.



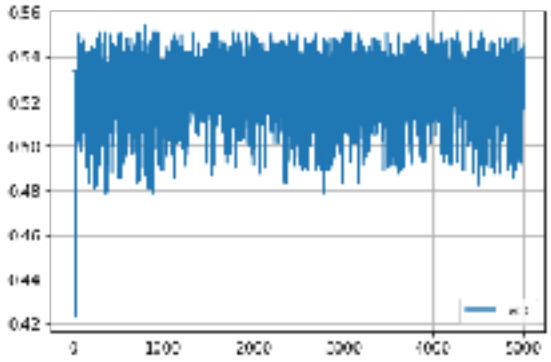
```
Errors: 47  Correct :53
Testing Accuracy: 53.0
```

Accuracy graph vs number of epochs for Adadelta Optimizer for 1 hidden layer 100 neurons.



```
Errors: 47  Correct :53
Testing Accuracy: 53.0
```

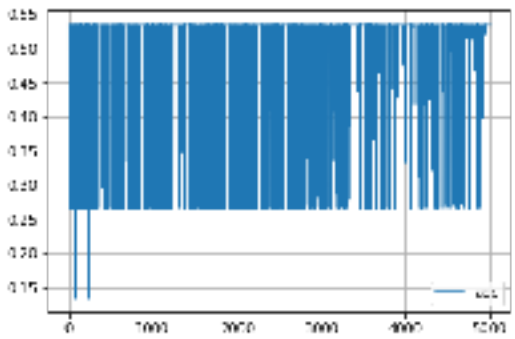
Accuracy graph vs number of epochs for RMSProp Optimizer for 1 hidden layer 100 neurons.




```
Errors: 47  Correct :53
Testing Accuracy: 53.0
```

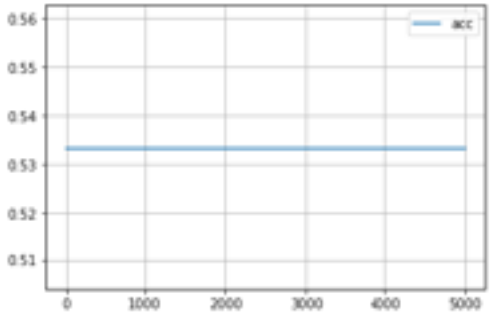
Using Different Activation Functions:

Train Accuracy graph vs number of epochs when Sigmoid activation function is used for 1 hidden layer 100 neurons.



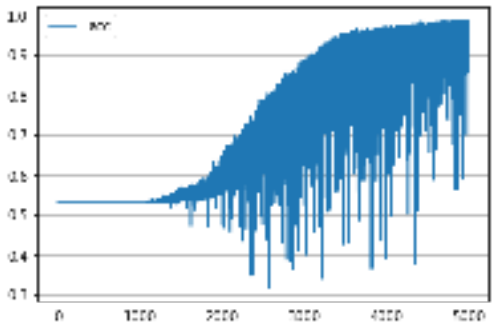
Errors: 47 Correct :53
Testing Accuracy: 53.0

Train Accuracy graph vs number of epochs when Tanh activation function is used for 1 hidden layer 100 neurons.



Errors: 47 Correct :53
Testing Accuracy: 53.0

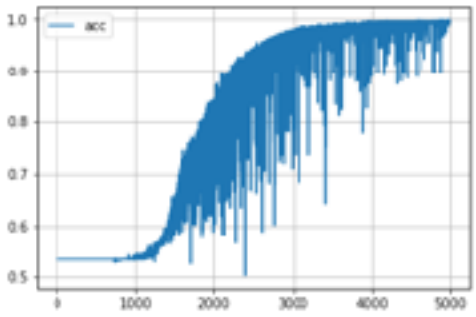
Train Accuracy graph vs number of epochs when leaky-relu activation function is used for 1 hidden layer 100 neurons.



Errors: 13 Correct :87
Testing Accuracy: 87.0

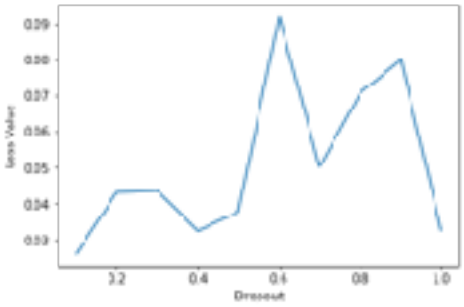
Analysis after adding Dropout:

Training accuracy graph vs. number of epochs after adding a dropout of 0.5 to a network with 1 hidden layer of 100 neurons using relu.



Errors: 3 Correct :99
Testing Accuracy: 99.0

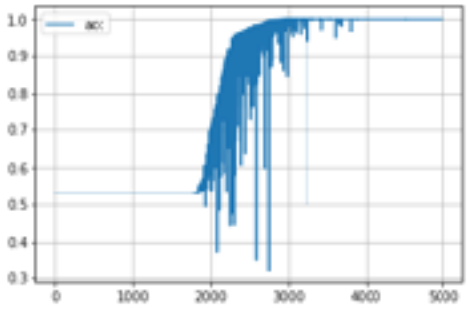
Analysis of cross entropy loss function against dropout values from 0.1 to 1.
We also tried adding a range of dropout values from 0.1 to 1 and plotted the loss function against that to check the effect of dropout on loss function and hence on the bias. Here, we lay emphasis on bias instead of overfitting because the size of training data is comparitively small.



Best Performing Neural Network Configuration:

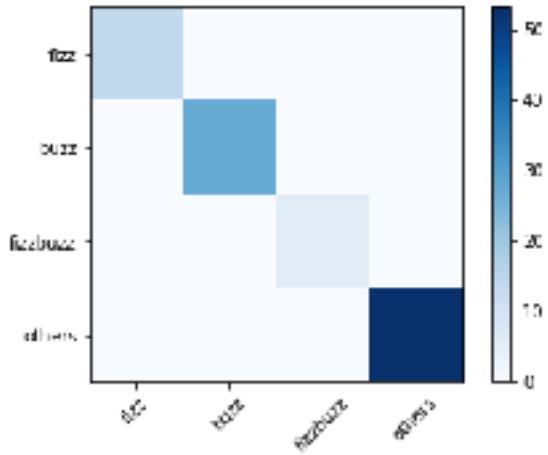
From the experiments done above with the neural network configurations, we came up with the following model which performed best on our test dataset:

Number of Hidden Layers: 2
Number of Neurons: 300 neurons in each layer.
Dropout: 0
Activation Function: RELU in both hidden layers.
Optimization: GradientDescent
Epochs: 5000
Learning rate: 0.5



Errors: 0 Correct :100
Testing Accuracy: 100.0

Confusion Matrix:



The confusion matrix shows the proportionality of predicted labels to actual labels across each class. As seen, a strong diagonal intensity implies high accuracy.

Conclusion:

As seen, the software 1.0 will always generate the correct label as it is pre-instructed. But such systems fail when they need to capture a new pattern in the incoming data and thus, need manual modification. The neural network model trained in software 2.0 has a very high accuracy as well, but might not always be 100% accurate. But it is designed to learn these patterns over time and will thereby evolve on its own.