Software 1.0 versus Software 2.0

Dhayanidhi Gunasekaran
Department of Computer Science
University at Buffalo
Buffalo, NY 14214
dhayanid@buffalo.edu

Abstract

To compare the two problem solving approaches to software development: the logic-based approach and the machine learning approach.

1 Problem Statement

To implement the task of Fizzbuzz using python and machine learning. Fizzbuzz problem can be explained as follows. An integer divisible by 3 is printed as “Fizz”, and an integer divisible by 5 is printed as “Buzz”. An integer divisible by both 3 and 5 is printed as “Fizzbuzz”. If an integer is not divisible by 3, 5 or 15, then the integer should be printed as it is.

2 Solution

2.1 Software 1.0

Software version 1 is a typical programming language. In this problem we use python to solve the Fizzbuzz problem. If-then-else conditional statements are used to check the divisibility conditions. The input integer is checked with divisibility conditions and the respective label is printed. If any of the conditions is not satisfied then the input is printed.

2.2 Software 2.0

Software version 2 is the machine learning implementation of the given problem. First step is preparing and processing the data required to train the model. A simple Neural Network with a single hidden layer is used as a model for solving this problem. In this solution, numbers from 101 to 1000 is used as training data and numbers from 1 – 100 is used for testing the model. The training data is then binary encoded to a 10bit array which is the shape of input layer. There is only one hidden layer in this model in which the number of neurons in this layer can be changed according to the model’s performance. And the shape of output layer is 4 since the solution has to be categorized in to 4 groups such as Fizz, Buzz, Fizzbuzz and Other. The weights for each layer are randomly initialized. Softmax cross entropy is used for determining the errors. Gradient descent is used as an optimizer for reducing the errors. Before training the model, the training data is grouped randomly into batches and are trained based on the number of epochs. After the training is completed, the model is tested with the test data and the accuracy is determined. Based on the accuracy the hyper parameters such as Learning rate, number of epochs, number of neurons and batch size can be changed to tweak the performance of the model.

The following figure is the pictorial representation of the Neural Network model used for solving the Fizzbuzz problem.

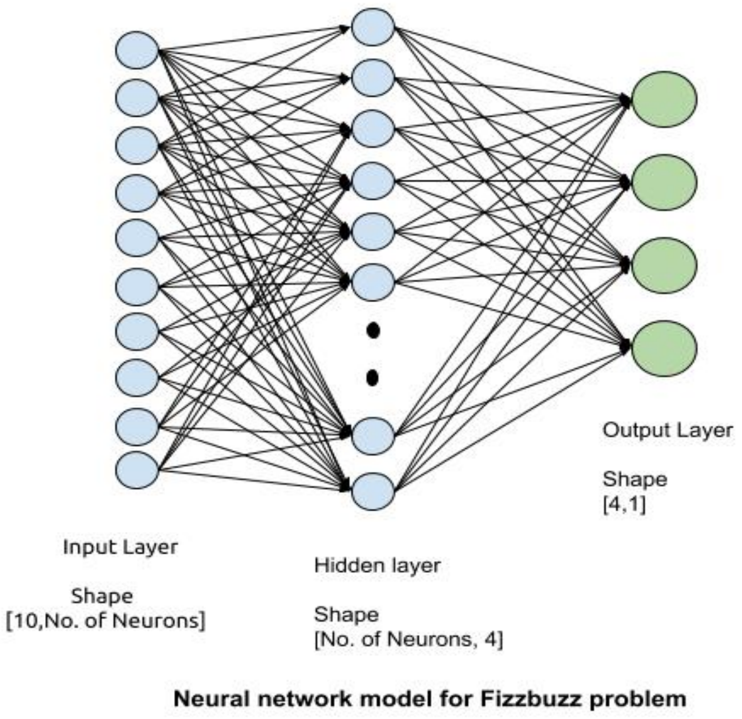

Figure 1: Neural Network Model

3 Performance

The performance of the solution is calculated based on the number of correct fizz-buzz predictions. It depends on the various hyper parameters such as Learning rate, number of epochs, number of neurons in hidden layer, etc.

The following table shows the final hyper parameters values with which the maximum efficiency is obtained.

Table 1: Final Hyper Parameter values.

Hyper Parameter	**Value**
Learning rate	0.1
Number of epochs	4000
Batch size	128
Number of neurons in Hidden layer	1000
Activation function	relu
Accuracy	**99% (Based on average of 5 results)**

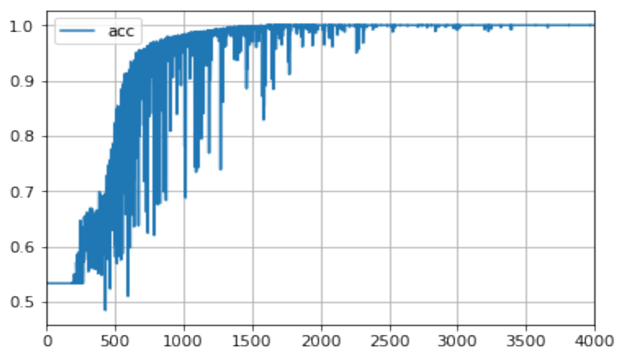

Figure 2: Accuracy vs No. of epochs

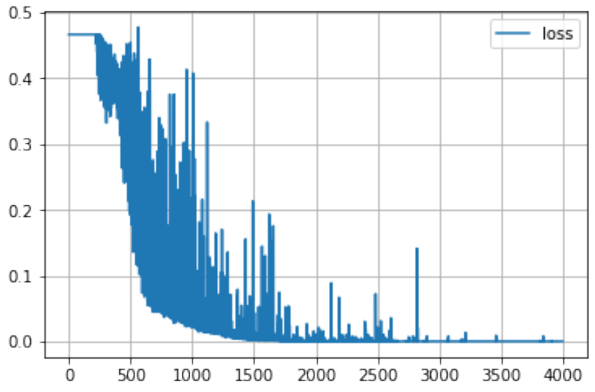

Figure 3: Loss vs No. of epochs

The accuracy of the individual categories is determined and explained in the below table.

Table 2: Accuracy of Fizz, Buzz, Fizzbuzz

Categories	**Accuracy (percentage)**
Fizz	100
Buzz	100
Fizzbuzz	100
Other	98.11

3.1 Learning rate

Gradient descent determines the weight after so many iterations in order to minimize the cost function. In order to achieve better results with gradient descent we need to set a value called learning rate. Learning rate determines how fast or slow the optimal weights are achieved. If it is too small it would require too many iterations to converge to the best values. If the learning rate is too large the optimal solution would be missed. Hence it is important to find the learning rate. Learning rate vs accuracy is provide in table 3.

Table 3: Learning rate vs Accuracy (Epochs = 4000)

Learning rate	Accuracy (percentage)
0.001	53
0.05	93
0.1	99
0.5	96
1	94

3.2 Number of epochs

An epoch is defined as a single pass through which the entire training data is passed to the training model. In this solution, the training data is divided into batches. A single epoch is said to be completed when all the batches of training data is fed in to the model. Usually with large number of epochs, the accuracy would be higher. While smaller number of epochs may result in poor performance. In this solution, number of epochs is set as 4000.

Table 4: Number of epochs vs Accuracy (Learning rate = 0.1)

Number of epochs	Accuracy (percentage)
1000	53
3000	90
4000	99
5000	96
10000	92

3.3 Batch Size

Batch size is referred as the number of training data used in a single iteration. Higher the batch size, more memory space is required for computation. In this solution, Since the number of training data is comparatively less, the batch size is set as 128, batches are generated based on different permutations of the training data.

3.4 Activation function

Activation function in a Neural network decides whether the particular neuron is activated or not. There are many activation functions such as Sigmoid Function, tanh, Rectified linear units(relu), Leaky relu and step functions. Depending upon the problem, activation functions can be selected. Activation functions usually does the job for categorizing a particular neuron using the probability distribution. In a problem like Fizzbuzz where the neurons need to be categorized into multiple classes such as Fizz, buzz and Fizzbuzz, activation functions like relu, leaky-relu works really well. Below graphs show the different activation functions used in the solution and its accuracy for the model.

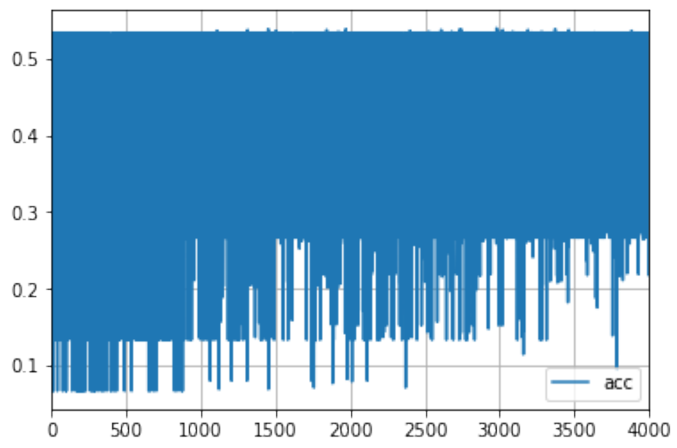

Figure 4: Accuracy graph in sigmoid function

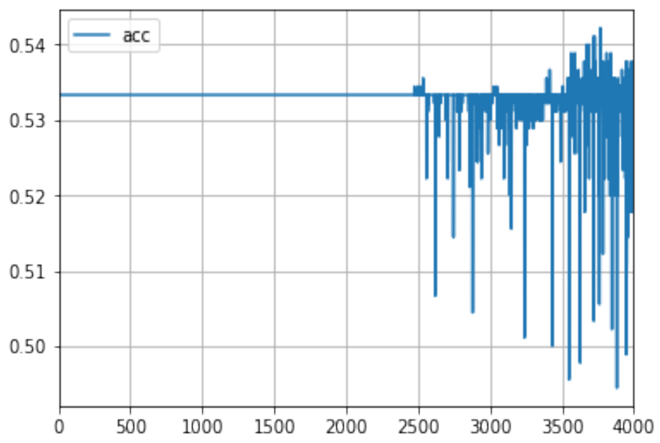

Figure 5: Accuracy graph in tanh function

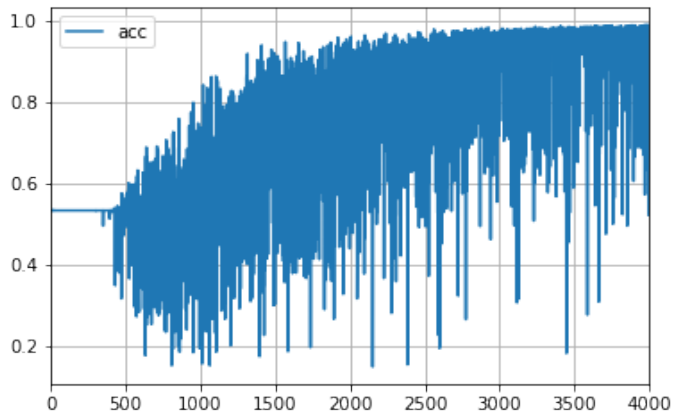

Figure 6: Accuracy graph in selu function

The below table shows the Accuracy for various activation functions.

Table 5: Activation function vs Accuracy

Activation function	**Accuracy (percentage)**
relu	99
Sigmoid	24
tanh	53
selu	96
Leaky-relu	98

4 Conclusion

Software 2.0 implementation of Fizzbuzz problem provides the insight of typical machine learning algorithms, creating and processing datasets, training the model and predicting the output. The solution implements Neural networks to train the model with rectified linear unit as an activation function. Softmax cross entropy function is used to determine the loss and Gradient descent optimization to minimize the loss producing accuracy of 99%.

References

[1] Tensorflow Documentation - https://www.tensorflow.org/api_docs/python/tf

[2] Python Documentation - https://docs.python.org/3/

[3] Pandas Documentation - http://pandas.pydata.org/pandas-docs/stable/

[4] Keras Documentation - https://keras.io/utils/