A Report on varying hyperparameters for a FizzBuzz problem

Miki Padhiary
Person Number : 50286289

September 17, 2018

Abstract

In this experiment we studied how to create densely connected neural network using sequential model. This experiment also helped in understanding various aspects of hyperparameters. Hyperparameter is a parameter whose value is set before the learning process begins and tuning hyperparameters can have an impact on how the model learns. If hyperparameters are poorly chosen, then the network will learn slowly or might not learn at all.

1 Introduction

Our problem statement was to implement FizzBuzz problem using Machine Learning algorithms.
Problem Definition: If an integer is divisible by 3, then the output should be Fizz, and if the integer divisible by 5 the output should be Buzz. An integer divisible by both 3 and 5 should return an output of FizzBuzz. If an integer is not divisible by 3 or 5 or 15, it should simply print Other. For the above problem, we had created a Densely Connected neural network using the sequential model offered by keras. Sequential model is a linear stack of layers, and multiple layers can be added to the model.

2 Experiments

Below are the list of various network tunings done:

- Case 1: Number of nodes in hidden layer
- Case 2: Number of layers in neural network
- Case 3: Change in Drop out
- Case 4: Change in activation function
- Case 5: Change in optimizers

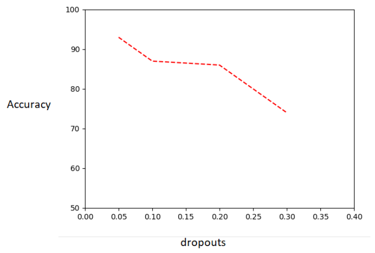
- Case 6: Change in the batch size
- Case 7: Change in the validation data split

3 Explanations

Hidden Layer: A Hidden layer is one which transform the inputs into something that the output layer can use.

Layers in Neural Network : A neural network is formed in three layers, called the input layer, hidden layer, and output layer. Each layer consists of one or more nodes. The lines between the nodes indicate the flow of information from one node to the next.

Drop out : Dropout is a regularization technique for reducing overfitting in neural networks.



Graph of Dropout vs. accuracy

Activation Functions : The activation function of a node defines the output of that node given an input or set of inputs.

Optimizers : An optimizer helps us to minimize (or maximize) an Objective function.

Batch Size : This is the size of input data which is fed at a time

Validation Data Split : This defines the way how a training data is splitted into training and validation sets. Here 0.2 specifies 80% of the data will be used as training and rest 20% as validation set.

4 Results

- **Case 1:** Increasing the number of neurons to 290 in the hidden layer increases the accuracy of the model. However, after a certain value for example 320, the accuracy remains constant. *Refer Figure 1 : case 1*
- **Case 2:** Currently number of layers in our neural network is 2. If it is increased to 3 with number of neurons to either 110 or 256, the accuracy calculated is 79% and 90% respectively, which is lesser than the accuracy predicted with 2 layers. *Refer Figure 2 : case 2*

- **Case 3:** Reducing the dropout to 0.05 helps in increasing the accuracy. *Refer Figure 3 : case 3*
- **Case 4:** When sigmoid function is applied instead of softmax at the output layer the accuracy was slightly lesser. For tanh, the accuracy went far below to 7%. For ReLU the accuracy was 98%. *Refer Figure 4 : case 4*
- **Case 5:** For Adam, the accuracy was found out to be 90%. For RMSProp, Adagrad , AdaDelta and SGD the accuracy was 92%, 94%, 99% and 53% respectively. *Refer Figure 5 : case 5*
- **Case 6:** If the batch size is reduced to 20, the accuracy went up to 94%. *Refer Figure 6 : case 6*
- **Case 7:** Changing the validation split to 0.3, provides an accuracy of 100%. Decreasing it to validation split to 0.1, provides an accuracy of 94% *Refer Figure 7 : case 7*

5 Analysis

Below is the tabular form of representation of values which was already provided in the code and the values which landed upto an accuracy of 100% . *Refer Figure 8 : case 8*

Hyperparameters	Default Values	Final Values
Case 1: Number of nodes in hidden layer	256	290
Case 2: Number of layers in neural network	2	2
Case 3: Change in Drop out	0.2	0.05
Case 4: Change in activation function	ReLU	ReLU
Case 5: Change in optimizers	RMSProp	AdaDelta
Case 6: Changing the batch size	128	20
Case 7: Changing the validation split	0.2	0.3

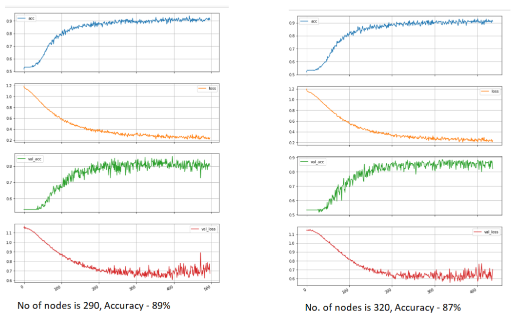


Figure 1: case 1

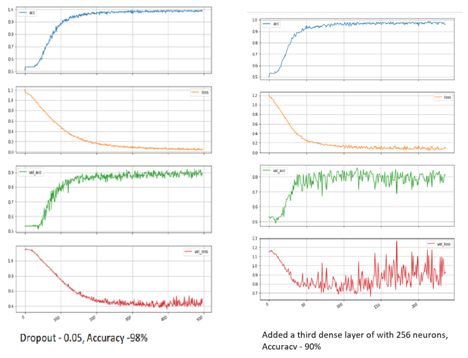


Figure 2: case 2

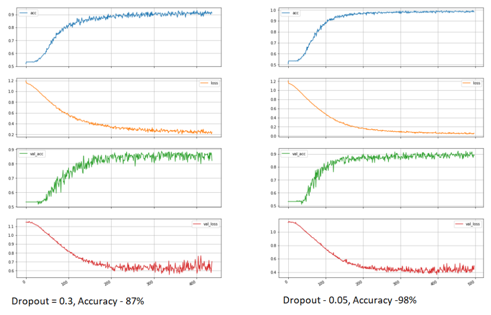


Figure 3: case 3

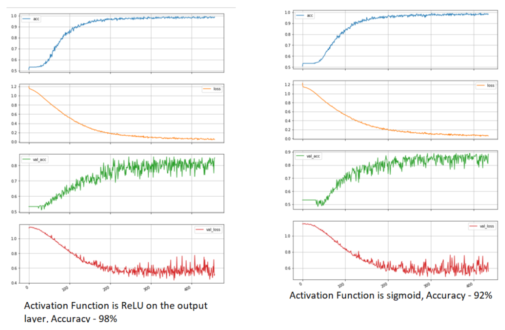


Figure 4: case 4

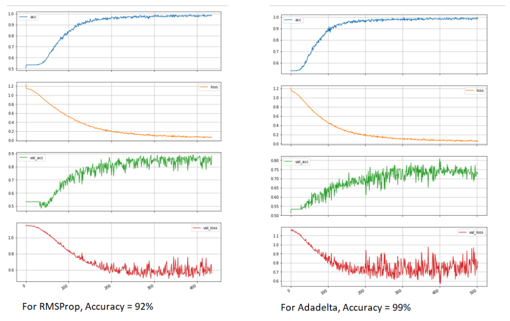


Figure 5: case 5

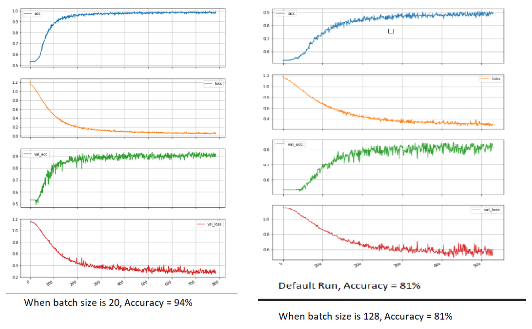


Figure 6: case 6

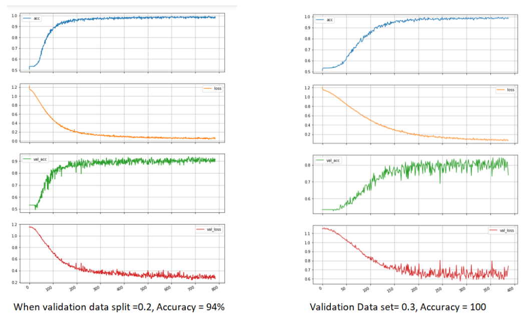


Figure 7: case 7

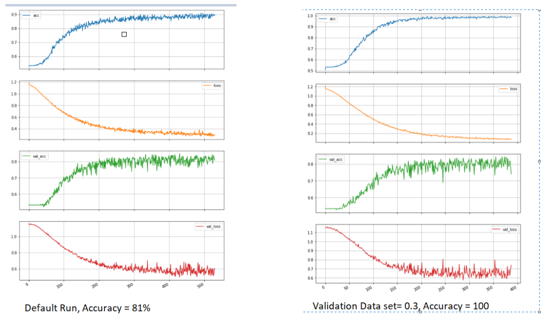


Figure 8: Default and Final Values

Figure 9: Graphs for modifications in hyperparameters

6 Conclusion

From the above experiment, we can conclude that the best parameters for tuning the neural network is the **Final Values** column as specified in Section 5, Analysis.

References

[1] FizzBuzz Keras implementation code posted in UB Learns.

[2] https://keras.io/

[3] Pattern Recognition and Machine Learning *Christopher M Bishop*