Ben Prescott
MSDS 458 2021SU
7/09/2021

A.1 First Research/Programming Assignment

Abstract

This research focuses on hand written digit classification, leveraging five different experiments with a single hidden layer fully-connected neural network. A total of five networks were trained to classify the digits from the MNIST dataset, evaluating their performance and the impact to their individual node activations with each model.

The Experiment #1 focuses on a very basic network consisting of a single hidden layer with a single node, while Experiment #2 adds a node to the hidden layer, for a total of two nodes. Experiment #3 provided a more flexible approach, allowing for any number of nodes in the hidden layer, but still using the constraint of a single hidden layer. This experiment resulted in the best possible model given the constraints.

Experiment #4 leveraged Principal Component Analysis as a form of dimensionality reduction. The PCA data was then used to re-train the same model architecture found in Experiment #3, and results compared. The final experiment, Experiment #5, used a Random Forest Classifier to perform dimensionality reduction, re-training the model from Experiment #3 and comparing the results of standard pre-processing techniques.

Introduction

The purpose of these experiments are to gain a deeper understanding of how deep neural networks work, and how activation values of nodes shift with more or less nodes in the hidden layer. The ultimate goal was a deeper understanding, while providing experiments that becoming more refined as they progress. Prior to experimentation, the assumption was made that a neural network with a single hidden layer and single node would perform poorly, with activation values not providing much clarity until a second node (or more) was added.

Based on previous experience, an additional assumption was made that performing dimensionality reduction may not improve model performance much. A strong classifier can be built using a rather simple network architecture with high accuracy. These two assumptions would be tested throughout this research.

Literature Review

Based on my research, there are many articles published exploring the right selection of activation functions for neural networks, as well as the impact of different activation functions on the execution times of the algorithm and the post-training weights. However, I was not able to find much (or anything) on evaluating the specific node activation values and how different hyperparameters may impact them.

One paper, titled *Review and Comparison of Commonly Used Activation Functions for Deep Neural Networks,* Tomasz Szandala discusses many activation functions that are used in neural network architectures, including the one I chose for my experiments – ReLU or Rectified Linear Unit. The paper also describes the vanishing gradient problem and dying ReLU problem, which were useful given I chose ReLU as my activation function.

Methods

This research started with the available MNIST dataset using the TensorFlow Datasets library. Each image in the dataset is 28x28 pixels, consisting of 784 individual pixels. The dataset was initially split into training and testing data, with 60,000 samples for training and 10,000 for testing. The 60,000 samples were further split to provide 55,000 samples for training and 5,000 samples for a validation set.

Basic pre-processing was done on the images, such as rescaling the X data to reside between 0 and 1 rather than the original pixel values. This was done to reduce the variance

between data, thus reducing the distance between pixel values for the network. The Y data was also processed to encode the labels into binary representations.

Experiment #1 consisted of defining a basic neural network, consisting of 784 input nodes, one hidden node, and ten output nodes. An ‘Early Stopping’ callback was added to the model to help with reducing computation runtime, as well as helping reduce the risk of overfitting. This model was then trained on the original normalized dataset, resulting in an accuracy value of ~40% on the test data. The activation values from Experiment #1 were shown in a table, as well as in a visual boxplot, both showing significant overlap in nearly all possible outcomes. However, the labels “2” and “6” provided the greatest number of consistent activation values, and largest activation values, with “7” and “9” providing a pattern of no activation at all.

Experiment #2 followed suit with Experiment #1, only changing the hidden layer nodes from one to two. Overall accuracy of the model improved from ~40% to ~70% with no other changes to the data or architecture. However, the spread of activation values changed dramatically, with “6” providing nearly no activation on the nodes, and “7” and “9” providing the greatest activation values and consistency. All output labels activated the nodes more consistently than with a single node, with a more consistent range of activation values.

Experiment #3 focused on shifting the number of nodes in the single hidden layer, improving the model to create the “best model” for future use. Many training cycles were leveraged, stopping on 30 nodes in the hidden layer, with all other hyperparameters remaining the same. This model resulted in ~97% accuracy with the same input data. An interesting result of this experiment was the 30 node activation values. The range of activation values decreased, with the largest value across all values being 12.7, compared to 26.5 in Experiment #2. The

ranges were also more consistent across all labels, with most labels having more consistent activation values than prior experiments.

Experiment #4 and #5 worked with reducing the number of dimensions using Principal Component Analysis and a Random Forest Classifier, respectively. Performing PCA on the data reduced the dimensionality from 784 input values to 154. However, the model's performance when trained on this data was nearly identical to that of Experiment #3, with an accuracy of ~97%. The Random Forest Classifier was used to identify the 70 most important features of the data. These features were then used to train the model from Experiment #3. While the model performance was good – around 92% accuracy – it was lower than that of Experiment #3 and #4.

However, the activation values for Experiment #4 and #5 were interesting, as the PCA resulted in noticeably more consistent activations across all output labels, showing even more overlap in values between the nodes. The RFC activation values actually showed the least amount of overlap between node activations, providing more consistency in activation values and no labels that caused the majority of nodes to not activate.

Results

The results of the different experiments displayed a few main outputs: that complex networks are not necessary for basic digit classification, and that activation values of nodes can be influenced by both the number of nodes in the network, as well as the number of input features to the hidden layer.

The first model from Experiment #1 performed poorly and didn't have enough of a comparison for activation values, as no other nodes existed. However, it did showcase that a single node may tend to activate more often for a smaller subset of values, which would help

explain the poor accuracy of the model. The later experiments described how adding one additional node can drastically improve performance, as well as provide more clarity into whether one node would activate over the other, or both activate. It seemed that the higher the output label, such as “9” compared to “1”, the more likely both nodes would activate with similar values. The smaller the digit the more likely one node would activate and the other wouldn’t. However, more testing would be needed to confirm if this holds true.

While the model performance did not increase while using dimensionally-reduced data from PCA or RFC, it did show how lower dimensional data can provide similar results to that of higher dimensional data, at least in this context. The RFC model using only 70 features, down from the original 784, shows impressive performance for the drastic drop in features. This alludes to the power of the most important features of an image and the benefit of using RFC for identifying them.

Conclusions

This research was conducted to gain insight into how deep learning works, including how different data transformations and dimensionality reduction may impact model performance. While keeping all hyperparameters consistent, with the exception of number of hidden layer nodes, the experiments show that nodes “share” the activations as the network size grows, where-as a single node demonstrates inconsistency in activation values, contributing to the overall poor performance. Additional research can be conducted using different activation functions to see how the activation values may shift, and if new patterns may emerge.

The results from the PCA and RFC experiments also show the power of feature importance and feasibility of reducing overall dimensions. This approach would scale to other

more complex image classification problems, such as the potential to use RFC to identify important features for cancer identification scenarios. Both PCA and RFC would be recommended to help reduce the overall complexity of the model, in turn reducing training time and necessary compute resources.

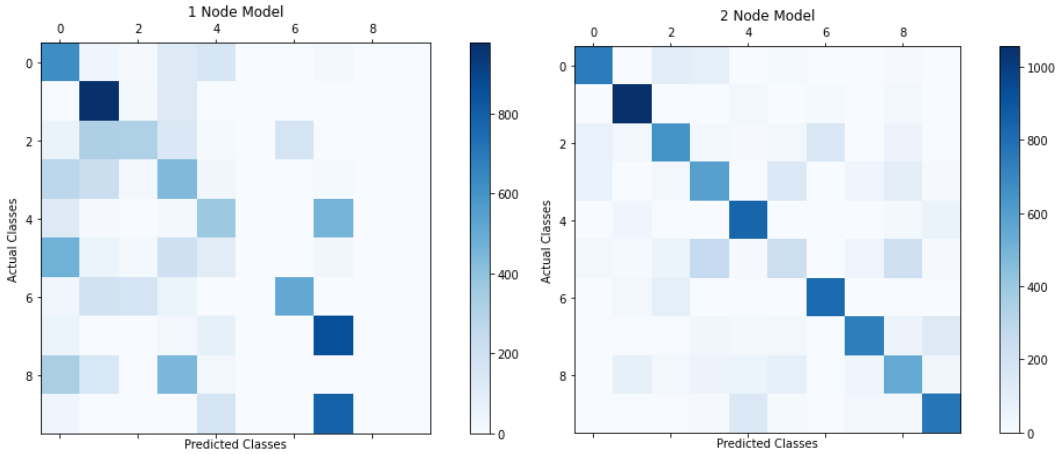
References

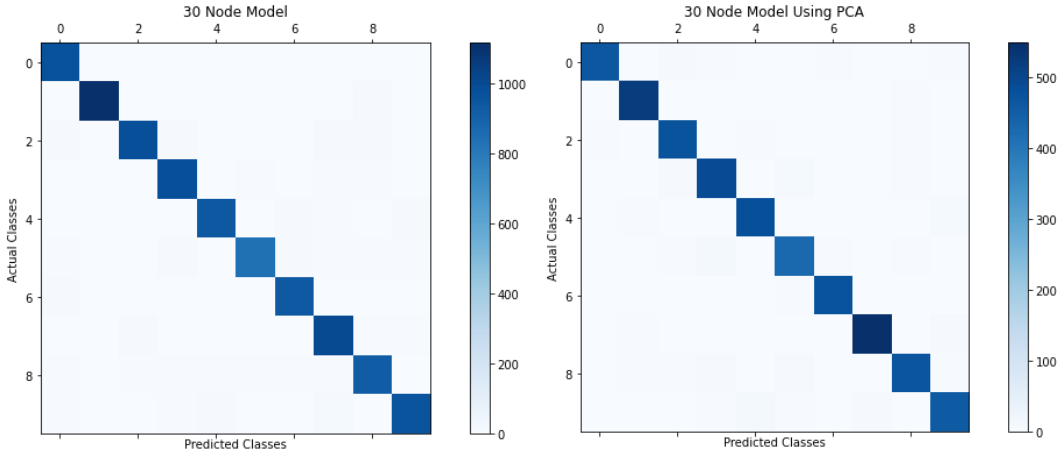
Chollet, F. (2021). *Deep Learning With Python*. S.l.: O'Reilly Media.

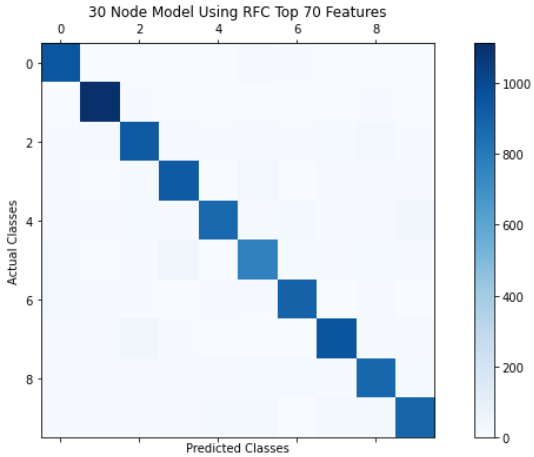
Szandała, T. (2020). Review and Comparison of Commonly Used Activation Functions for Deep Neural Networks. Bio-Inspired Neurocomputing, 203–224. https://doi.org/10.1007/978-981-15-5495-7_11

Appendix

Model Confusion Matrices:

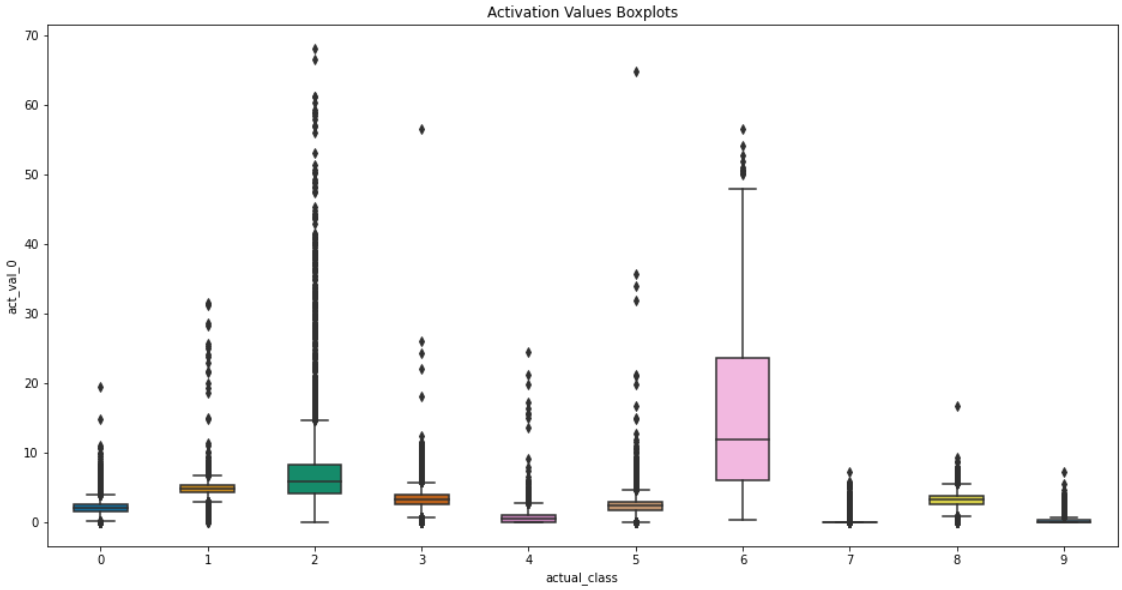




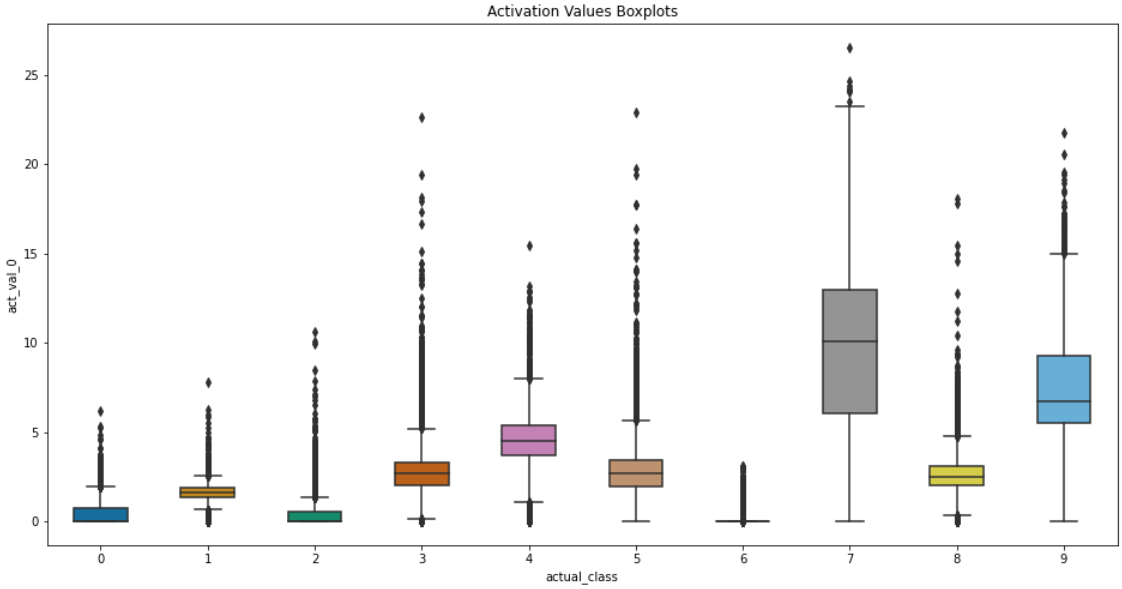


Node Activation Boxplots:

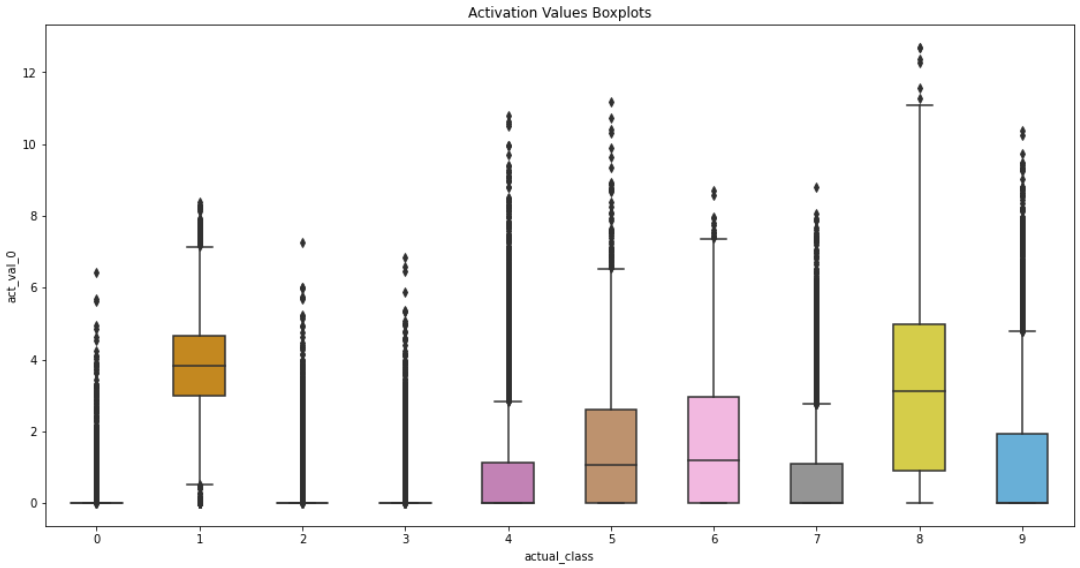
Model #1



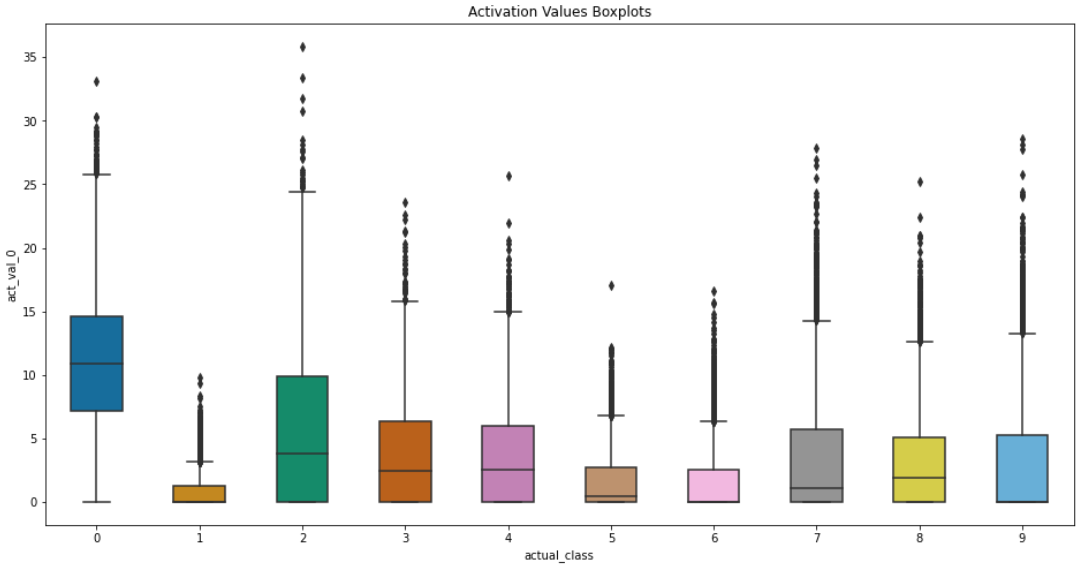
Model #2



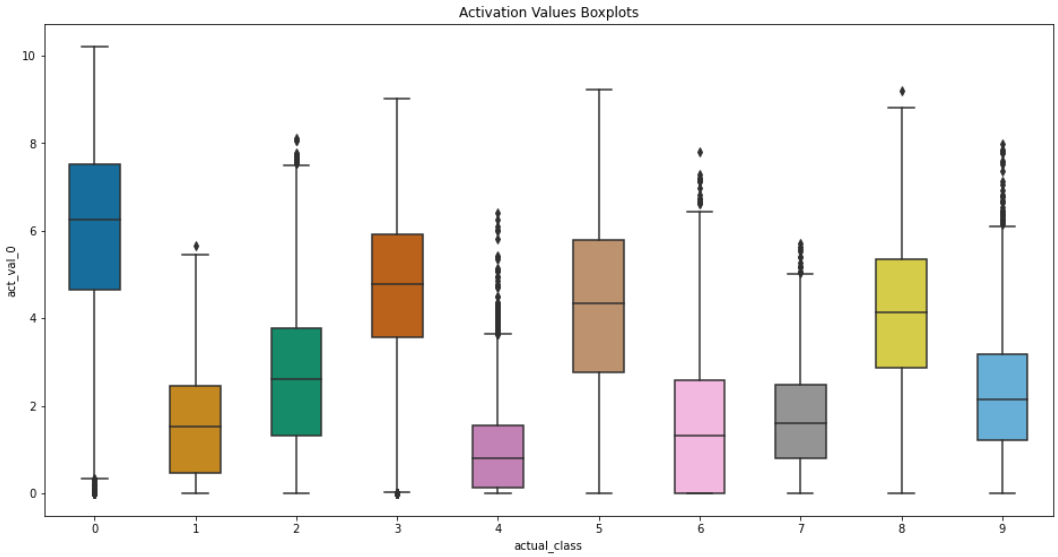
Model #3



Model #4



Model #5


Activation Values Boxplots
act_val_0
actual_class
0
1
2
3
4
5
6
7
8
9
0
2
4
6
8
10