CS 5830 - Project 8

Leo Beal & Chris Brown

INTRODUCTION

In this short, exploratory project, we found a classification dataset and applied both decision trees and neural networks to it. The dataset used described a set of ten English letters using attributes such as the cartesian coordinates of the starting and ending points of the letters. Our decision trees and neural nets both classified the letters using the six provided attributes in the dataset. Higher depth decision trees obtained a better F1 score and found the vertical portion of the cartesian coordinate attributes to be the most important attributes. The neural networks performed better the larger they were, but once they no longer improved, trade-offs had to be made to choose which characters are the most important to classify correctly.

DATASET

We acquired our Artificial Characters Dataset from the University of California, Irvine. The dataset was 30,654 samples with 25,545 split into a predetermined training set and 5,109 split into the test set. There are seven attributes and one target. One attribute is used to describe the data type, and as we are using Python for our analyses, we dropped this attribute. The remaining six attributes include: the initial and final coordinates of a character segment in the cartesian plane, the length of a segment computed using the geometric distance between the initial and final coordinates, and the length of the diagonal of the smallest rectangle which includes the picture of the character. A category is a number from 1 to 10, representing a character in [A, C, D, E, F, G, H, L, P, R]. There are no missing or bad values in the provided datasets and all values are continuous and positive. We normalized all six attributes since neural network activation functions perform best with normalized values. We also one-hot encoded the category so we could use a multi-class cost function. We finally combined all the data into one train file and one test file as the dataset split all the pieces of data into separate files.

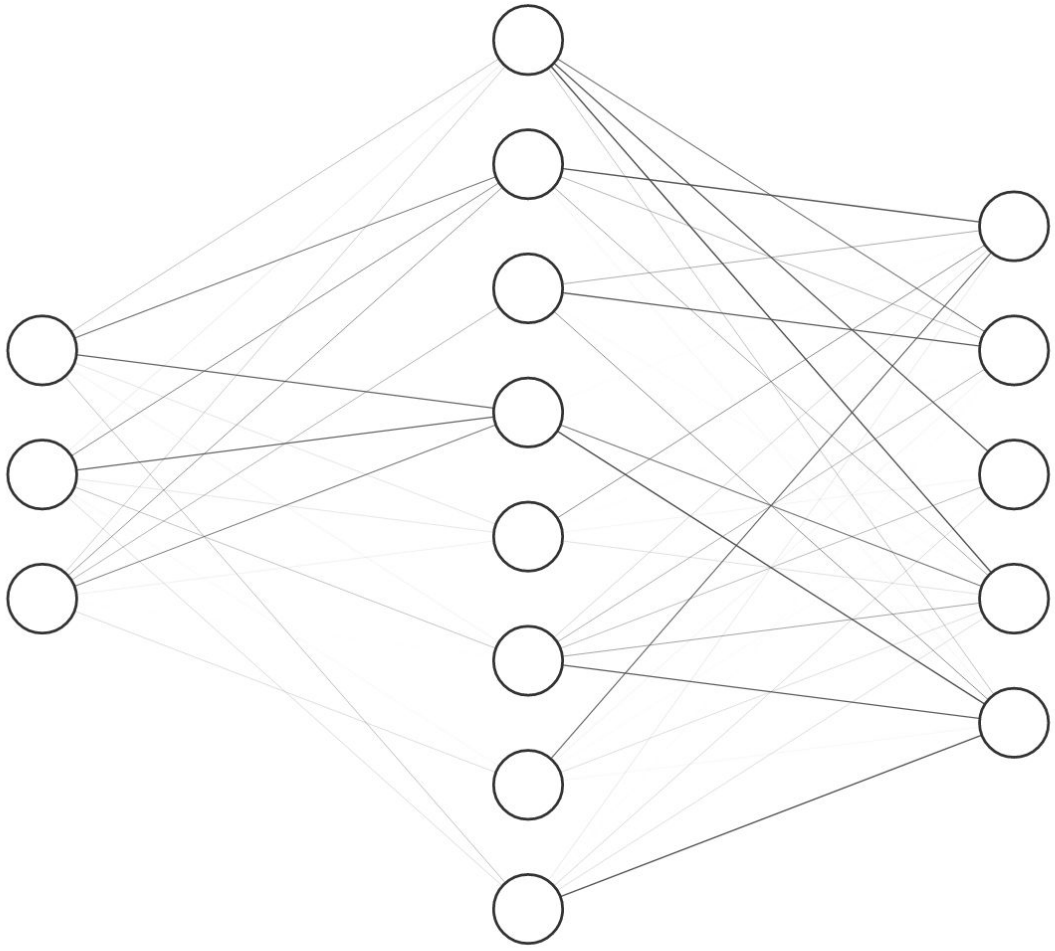
ANALYSIS TECHNIQUE

The decision trees were put together with the sklearn tree package. We began by making a test tree with a max depth of 5 using all 6 prediction attributes and predicting all 10 letters. Some other preliminary testing was done with max depths of 7 and 9. After that, similar trees were made using only 4 of the 6 attributes. After this, both types of trees were created with max depths of 5 to 19. The report back included the precision, recall, f-score, and the importance of each used attribute.

CS 5830 - Project 8

Leo Beal & Chris Brown

To construct our Neural Networks, we used Tensorflow 2.0 with Keras. We decided to use these packages as we are most familiar with them and find them to be the most intuitive. We first had to load and format the data as described in the **dataset** section. We then built a test neural network of just three layers with an input layer of size 6, a hidden layer of size 32, and an output layer of size 10. We also used a categorical cross-entropy loss function as we one-hot encoded our target. Our mini-batch size was 16 and our maximum number of epochs was 250. Using this test network we verified our training and testing process worked and our network could achieve an accuracy greater than randomly guessing. We then modified the hidden layer to have size 256, added regularization and dropout, and created a custom learning rate optimizer. We then used the same framework, but increased the hidden layer size to 1024 neurons. We then took this network and added a second hidden layer with 256 nodes, then 1024 nodes. As our neural networks are too large to visualize concisely, we have included a graphic on a smaller version of a network with 3 nodes in the input layer, 8 in the hidden, and 5 in the output. The darkness of the edges indicates how important they are.



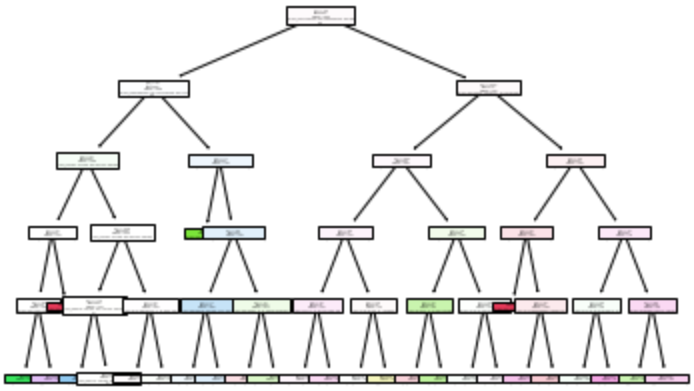
Input Layer $\in \mathbb{R}^3$ Hidden Layer $\in \mathbb{R}^8$ Output Layer $\in \mathbb{R}^5$

CS 5830 - Project 8

Leo Beal & Chris Brown

RESULTS

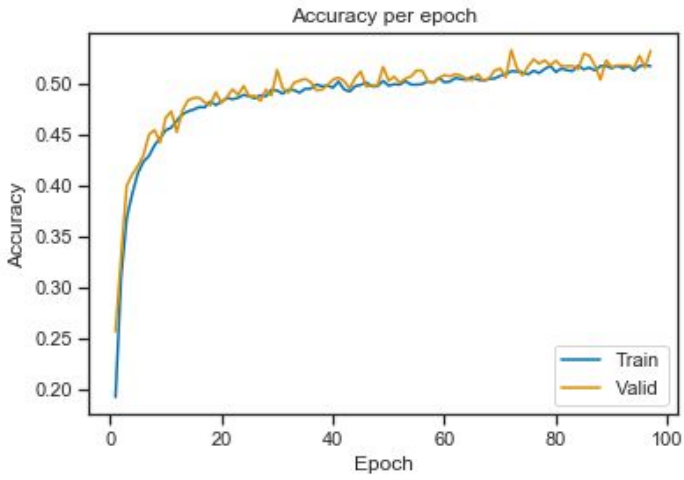
The first noticeable trait of the decision trees was that as the maximum depth increased, the f-score for each letter increased dramatically. At about a depth of 15, all f-scores were above .5, up from the .2 to .4 range at a max depth of 5. At a depth of 19, all f-scores were in the range of .7 to .9. However, this comes at both a computational cost and a complexity cost, possibly leading to overfitting the data. The most important features of the data were the y1 and y2 values, both with importance values around .2.



For the second run, only the x and y values were used. While this did not have a significant change on the f-scores, it did change the importance of y1 and y2 to about .3, leaving them the more important features again.

Using our first neural network with a size of 3 layers and 256 hidden neurons, we achieved a test accuracy of 50.48%. Our second network with 1024 hidden neurons had an accuracy of 53.75%. Our third and fourth networks had an accuracy of 55.33% and 55.12%, respectively. The best overall network was the fourth network which had 1024 hidden neurons in 2 hidden layers. Although the third network had a slightly higher accuracy, the fourth network did better at classifying most of the characters, but since it struggled with the 10th character (R) in particular, its overall accuracy was slightly slower. In this situation, an ensemble of networks may prove to be ideal as certain networks do better at classifying certain letters.

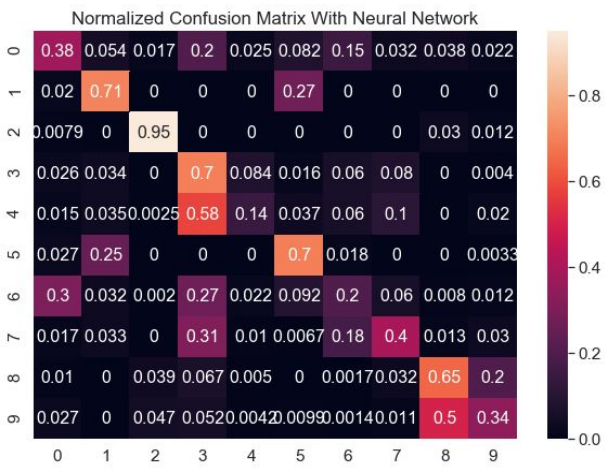
As for training speeds, the fourth network trained the fastest by only requiring 97 epochs while the other three networks consistently required over 200 epochs. A figure has been included to show the accuracy increasing of network four over its ~100 epochs.



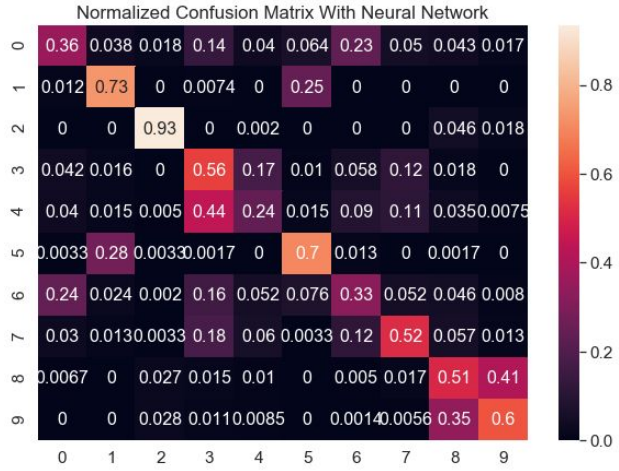
CS 5830 - Project 8

Leo Beal & Chris Brown

To illustrate the difference in performance between the third and fourth network, two confusion-matrices are shown below. It is interesting to observe the tradeoff one must make when choosing between the two neural networks. As a reminder, the numbers 0 through 9 represent their respective character in the list [A, C, D, E, F, G, H, L, P, R] with A being 0 and R being 9.



2 Hidden Layers - 1024 Neurons



2 Hidden Layers - 256 Neurons

An interesting note is that both the neural networks and the decision trees always predicted the letter D the best. This is likely because it is rather dissimilar to the other letters in the set while letters such as E and F share a lot of characteristics.

SLIDES

https://docs.google.com/presentation/d/1vUS322fElTUISxdlcfaF7uANrVVtNvXVaZIL6fNvHTI/edit?usp=sharing