Machuria Johnson
ITAI 2376
Module 1
Adam Blank

Module 1 – Python Lab Assignment: Building a Basic Neural Network

I have created a basic neural networking using the MNIST dataset for digit classification. This dataset was created in 1994 as a combination of NIST Special Database 1 and Special Database 3. Special Database 1 consists of digits written by high school students and Special Database 2 consists of digits written by members of US Census Bureau. I am using Keras to load the dataset and then create a test/train dataset. The test and train datasets are normalized to create a consistent scale, and speed up training by changing from a 0-255 integer value to a 0.0-1.0 float value, the expected input format for most neural networks.

After loading the dataset, preparing the test and train datasets, we start building the model.

First we flatten the model in the input layer. This reshapes the data to a 28x28 pixel array which will be used in subsequent layers. Then we run the first hidden layer, dense, this creates 128 neurons which are fully connected, receiving input from all features in the array and processes them using ReLU. This is used because is mathematically simpler than other options like sigmoid or tanh. The second hidden layer is dropout, this randomly sets 20% of the inputs to 0 to prevent over fitting. Finally Dense is the output layer. This has 10 neurons related to the 10 classes of the MNIST dataset using softmax to determine the probability across the classes. After this the model is compiled using the "adam" optimizer, this is used configure the model for training. The optimizer is set to reduce loss during training, and the loss function is defined to determine how the model is performing during training. Metrics are determined to "accuracy" calculating how often the predictions equal the label, looking at the fraction of images that are correctly classified. Once the model has been configured for training, the training can begin with model.fit. We specify the training datasets x_train and y_train (x are the images in the MNIST dataset, y is the label 0-9 in the MNIST dataset) and the number of epochs to train the model. When training the model there is a balance between over training and undertraining the model. If you undertrain, the model will not learn enough to be accurate. Overtraining can lead to the model being very good at identifying the training data but will have problems identifying data outside the provided training data. After training the model, we evaluate by comparing the test data, and looking at the loss ratio and accuracy of the results. We can adjust our hyperparameters to see how they affect the accuracy of the model.

I also added a check, where I created an image of the number 5 in MS Paint. With black text on a white background and used that to check that the model could predict the class of the number and check how confident it was in that prediction.

I believe the model could be made more accurate using a different activation in the first hidden layer, or by adjusting the dropout in the second hidden layer, or by changing the activation in the third hidden layer. Other methods of improving the model would be to write a for loop that incremented the training epochs while analyzing the model evaluation, stopping when the accuracy stops increasing, or begins to decline.