**EEN614: Neural Networks  
Final Project**

Abstract:

     For this project the creation of an authentication system will be attempted for an aerial drone. You may have heard of Amazon’s one day shipping which intends to use drones to deliver packages within a certain distance of their storage facilities. I want to create a neural network that would accurately recognize a person to deliver a package. In retrospect, this would also be a great application to deliver necessities (food, water, medicine, clothes) to individuals who live in natural disaster zones, remote or difficult to reach areas. In addition, the drone could also use validation of handwritten signatures to verify delivery. I hope to successfully implement the facial scanning software with DLib/OpenCV. The target persons face would be preloaded into the model. The program would then scan and match the stored facial data with the scanned facial image using a camera. For face detection I will be done using Haar Cascade classifier.

Introduction:

The goal of this project is an authentication and verification system that can be utilized by drones to successfully deliver packages. Initially, a face detection algorithm was implemented to register a user’s preloaded face. The facial detection was implemented successfully however, the neural network used was already designed by the creators of the OpenCV python library. The successful implementation was spot on for this project but not working on the architecture didn’t match up with the prompt.

To illustrate the design of a network the verification for signatures was created in Google Colab. The training was done using the MNIST dataset. The MNIST database contains handwritten digits with a training set of 60,000 examples, and a test set of 10,000 examples.

Below is a plot the first 5 images to sample what kind of images are included in the dataset.A picture containing object

Description automatically generated

Facial Recognition:

Detection\_Storage-2

A screenshot of a cell phone

Description automatically generated

*Figure 1: Face Detection and storage code*

A person looking at the camera

Description automatically generated

The detection storage code takes pictures of the user’s face using the web camera. These photos are saved in the folder “datasets” under the folder with your stored name. When the detection code is executed it will reference the detection storage database and store those pictures in the user’s folder.

Recognition

A screenshot of a cell phone

Description automatically generated

A person brushing his teeth

Description automatically generated

Our User Interface consists of a tracking box and label. The tracking box is drawn when a face is detected and the label is drawn when a face is recognized, in this case the face is labeled as ‘Brandon’, or unrecognized, which would label the face as ‘Unknown’. Multiple stored faces can be saved in this folder and the appropriate name will populate when it is matched.

Handwritten Signature Verification:

The created ANN architecture wasn’t as complex as previous assignments. The training models’ parameters were as follows: epochs=6, batch size=128, 20% validation data and, adam optimizer. Efficiency and accuracy were paramount in the design. The complexity was repeatedly downsized. However, the number of training parameters suffered as well. Despite this the validation accuracy was 96% and took 20 seconds to train in total.

A close up of text on a white background

Description automatically generated

A screenshot of a cell phone

Description automatically generated

There were only 2 hidden layers of size 32 and 10 utilized. The first layer consisted of a relu activation function with no regularizers. With 53,000 parameters to train the dataset didn’t train the most features for validation.

A screenshot of text

Description automatically generated

Discussion:

The facial detection/recognition part of the project was difficult to implement. There were various components to getting it to use the web cam and more importantly utilize it for the detection. There was a major learning code for Haar Cascade. Haar Cascade is a machine learning object detection algorithm used to identify objects in an image or video. The classifier consists of several stages that are applied systematically to a region of interest until at some stage the object/image is rejected, or all the stages are passed. The code has comments on each line detailing the steps done. Lastly, its true A text editor sublime was installed to compile and run the code.

The ANN implementation for handwriting classification went smoothly. The network created was simpler compared to the multiple layer’s others have done. The training data was accurately classified while the validation was even better. The dropout layers helped make the connections better. Regularizers were tried but it did nothing to improve the validation or reduce the loss. With the following architecture a loss of 11% was reached with an accuracy of 96%. There was an overfitting issue with more epochs present. The training was held at 6 epochs to reach maximum effectiveness before the issue corrupted the model. The first stage of the project took more time than anticipated. Making perfection of the model difficult. For future work, some optimization could be made. With more time its anticipated that this addition could max out the accuracy to 98-99%.

Conclusion:

Facial detection and facial recognition systems were implemented and successfully classified Brandon’s face. In addition, a second system for signature validation was designed and implemented reaching an accuracy of 90%. This system can be utilized by any package delivery company (Amazon, UPS, FEDEX) trying to boost efficiency by creating a drone fleet. The facial recognition of a preloaded face could be done from any phone using the same facial recognition system that opens it. The validation system would act as a two-factor system accepting signatures and making sure it’s the desired person’s name.

In addition, the system could be implemented into the store fronts of USPS scanning shipping forms and populating the text into a digital file. Saving time and the necessity of retyping addresses and user error.