Neural Network for Real-Time Handwriting Recognition

Group Members :

Nomaira Javaed, Roman Jennings, Mory Kaba, Emmanuel Kareem, Noah Khamliche, Luke Kim, Meenakshi Kommineni, Jianbin Lao

Real-time Handwriting Recognition :

We have chosen proposal 9, Neural Network for Real-Time Handwriting Recognition, for our project. Real time handwriting recognition is an important subset of machine learning, as it provides many solutions to some of the world's problems. From a security standpoint, real time handwriting recognition can be used to stop counterfeit checks, forged signatures, help find evidence of a crime. In terms of quality of life, we could improve handwritten conversion to text, create signed logins unique to each user, create documents from speech that look like they were handwritten. There are a plethora of applications for a handwriting specific neural network.

Implementation :

For our implementation of this project, we will be using a relatively new type of neural network, Capsule Networks. Capsule Networks try to perform inverse graphics based on whatever they are fed. Capsule networks, as the name suggests, are made up of many different capsules, which are essentially evaluation functions. These capsule functions try to predict the presence and the instantiation parameters of an object at a given location. Capsule networks output

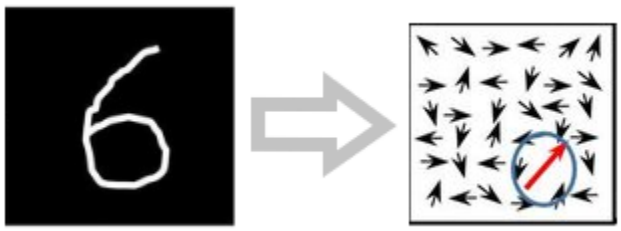

Figure 1

looks like the image (*Figure 1*) on the right. This is a map of where this capsule thinks that the input, the number 6 is.

Our Implementation will use a UI System that allows a user to draw in the window. The user will then type the letter + “Shift” for a capital letter, and simply the letter for the lowercase letter. This allows the user to swiftly enter the characters with one hand and draw with the other, using a mouse or digital pen. The user will enter each letter one by one during the data collection period. The users input data will be passed to the Capsule Neural Network, which will compute the geometry of the image, and map it to the current user, creating a user specific model. These models will be saved and retrieved for each user. Once the user finishes the training period, the user will be recognized after submitting a writing sample in the application.

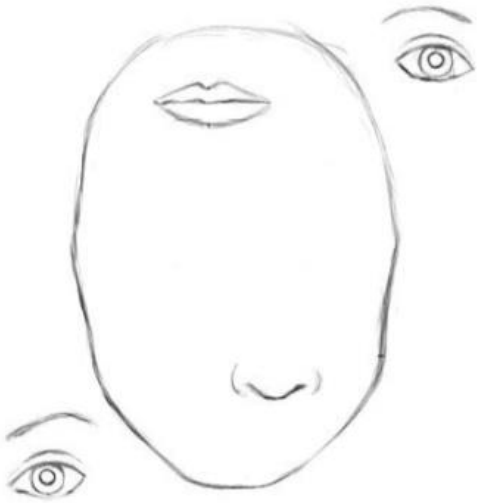

Figure 2

Why Capsule Networks? :

We chose to use a Capsule Networks specifically because they are more powerful than conventional Convolution Neural Networks. One of the main capabilities of a capsule network is to detect order. Below is a simple example where CNN’s fail to recognize correctly but Capsule Networks recognize the image as correct.

Figure 2 image is of a face that is deconstructed. To a CNN, this image has the correct face shape, two eyes, a mouth, a nose. All of the things that a face has. Even the best of CNN facial recognition networks will fail some of the time on images such as this one. Capsule Networks mitigate this hiccup by keeping track of the individual position and order of these elements, rather than if they exist in the image or not. This

will help with the variablility in rotation and scale of writing. Using a traditional CNN, we would need to line up the images every time we needed to run an image through the network. Using Capsule Networks allows us to screencapture the GUI rather than fit a specific image everytime.

Training, Validation, and Error Analysis:

Training will be tailored to the individual user, as required by the project guidelines. To do this, we will be using a combination of mouse, and USB drawing tablet as input. Validation of our models will be kept track of behind the scenes, logging each user's recognition and transcription success, that is was the letter we identified correct. These hits/misses will be used for cross validation and model enhancements utilizing a popular technique, K-Folds Cross Validation. This technique allows each individual element training/testing to be seen by the Capsule Network. This is beneficial as normal Training/Testing data splits often overfit with small sets of data, similar to what we will be working with.

Error analysis will be completed once the user has done multiple samples of writing. With the techniques we will implement in this project, we can expect 90% or more in accuracy. Some similar research systems have recorded 99.7% accuracy for their systems, so 90% is a feasible value for success.