ENGR 105 Final Project Write-Up

The GUI that I created employs a neural network to recognize handwritten digits. It allows the user to choose a number that contains between 1 and 10 digits and them draw the digits using - most conveniently - a mouse or a trackpad. Then, the user can save the number to a .mat file for further processing if desired.

The usage of the GUI is fairly intuitive. There are multiple .m and .mat files that are needed for proper functioning of the GUI and those can all be found in the file uploaded to Canvas. However, the only script that needs to be ran is MainScreen.m. All the other files are either called up by MainScreen.m or subsidiaries of the functions/scripts that it calls.

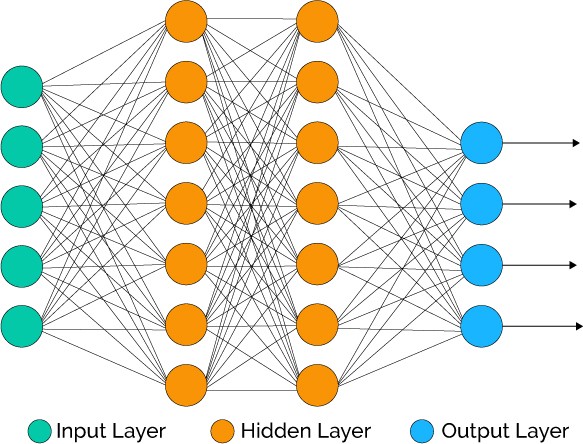
Basic Usage/Features: Once MainScreen.m is called there is a slider that can be employed to choose the number of digits in the number that you want to translate from writing to a number stored within the computer. Once the number of digits is chosen, a checkbox is employed to confirm the choice. Once that is clicked, the writing screen will automatically open and there will be a region – enclosed by a white square – where the numbers can be written. Only one number can be written in the box at a time, so the entire number is created by recognizing a series of single digits. To draw the image, the left-click on the mouse or the trackpad has to be clicked down and then, with the left-click/trackpad still held down, the number can be drawn by dragging the mouse or your finger on the trackpad. When the left-click on the mouse or trackpad is released, a dialogue will pop-up and ask whether the number displayed at the bottom of the figure is the correct one that was drawn. If the “Correct” option is chosen, the figure will be cleared and a new blank screen will pop-up to allow for the next digit to be written. If “Incorrect” is chosen the number will be erased and you will be able to write the number again. As a disclaimer, I trained the network with around 1,500 training digits samples that I drew by hand. I tried to incorporate as many styles of the digits as possible (such as the different types of 4’s and 2’s with a loop and without a loop), but I may have left out some specific styles. It should still be fairly good at recognizing the digits, but if it is inaccurate it is best to slow down the speed of drawing and/or increase the size of the digit drawn within the box. After the entire number is finished being drawn, a screen will pop-up that allows the user to exit, enter a new number, or store the number just found to a .mat file. If the user chooses to save to a .mat file, a new pushbutton appears that allows the user to delete the .mat just created if the button was chosen by accident/for a number of other reasons.

**\*Note:** neural\_network.m, costfunction\_backpropagation.m and fmincg.m are not called by the GUI, but are integral to finding the correct weights and biases that allow for actually detecting the digit, so I have included them for completeness. Also, to be clear, I did not create fmincg.m. It simulates the function of the built-in fminunc command in Matlab, but allows for very large input sets that the built-in function cannot process. It is called within neural\_network.m and since it is specialized and not built-in I have included it if you wanted to run neural\_network.m. Those files are the coolest/most interesting part of the project in my opinion as it is the actual neural network implementation and it took the longest for me to do. Running neural\_network.m will just out the cost function values being minimized to the command window.

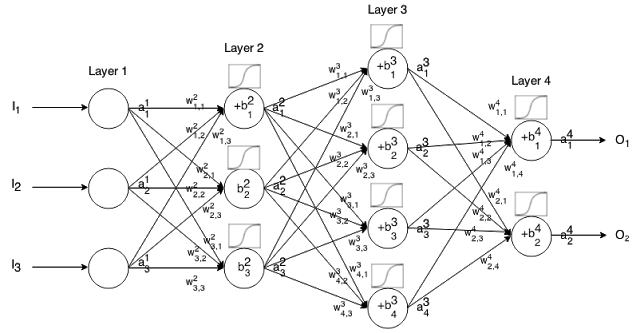
This project was truly a daunting task at first, but the knowledge that I have gained has led me even deeper into a field I am very passionate about: artificial intelligence. I had a limited understanding of how machine learning actually worked, but now I know that it is much less magical than I believed at first. The computer is learning, but not in the sense that I thought at first. Essentially, it uses the training data to minimize a function. However, I believe the beauty is in how the network is structured. The layers and connections between “neurons” very much mimic the human brain. The network looks complicated and that is kind of reassuring because it makes sense that recognizing digits, a task we do every day, is not simple. If it really was that easy that would say something about our brains and our intelligence. But the complexity of the problem emphasizes how the process of evolution has led to our brains that can recognize digits almost without thinking at all. As I kept thinking about how the network is solely using calculus and a bunch of basic math to compute the output, I started to doubt how intelligent the network really was and how it didn’t feel as intelligent as I expected. But then I had a realization, and began to question what was so intelligent about us humans. We may have more neurons than the model that I created, but it is modeled after our brains. Maybe our brains aren’t that special after all? All we do is taking information from our environment (like the training samples to the network) and iterate and change our output depending on the various inputs. Aren’t we not just machines also? We just have many more pieces of training data to learn from and more neurons to make more complex calculations. But if a non-living computer that was just created around half a decade ago can begin to rival our intelligence, that may be reason for questioning out intelligence that has had to evolve over hundreds of millions of years to get to the point we are at today. I would be naïve to say that humans’ intelligence is unique and cannot be surpassed/replicated. Computers have already passed us in terms of computational power and I believe it is not much longer until they have surpassed us entirely. If we can create general artificial intelligence that is able to create its own neural networks/artificial intelligence it will truly be capable of achieving intelligence equal to and much greater than human intelligence. That is a future I am very excited, but also slightly scared to see. I guess only time will tell.

Through this project I have learned not only about coding, but I have also started to learn about the future of humans and intelligence in the universe, which is quite powerful I believe. Also, I realized how much I could already do with the Matlab knowledge I had. I had to employ my experience with image processing, animation, the imfreehand command used in the 5th homework assignment for tracing images (this was probably the most central command to the project because it was the perfect form for the interaction between the user and the computer), the GUI elements, indexing and vectorization, for/while loops and iterating, matrix manipulations, plotting, and much more I am probably forgetting. I saw how even very complex tasks are just combinations of the building blocks that we have learned throughout the course. I didn’t really have to learn new information, I just had to apply the knowledge (such as in the form of commands and structure) I already knew in a different/unique way.

I have really enjoyed this class and I believe it is going to be an essential building block in my journey through Penn and beyond. I appreciate all the effort that went into teaching the class and I truly think I learned much more than how to code.



**Figure 1:** Simplified neural network that shows the connections between each of the layers.



**Figure 2:** A little more in-depth image that shows some of the variables associated with the network. (squiggle above each neuron/perceptron shows the sigmoid function that is implemented in the GUI)

**Acknowledgments**

<https://www.youtube.com/watch?v=aircAruvnKk>

**Link 1:** This video, along with its subsequent parts, was invaluable to me for understanding neural networks. It is really well made and the explanations are very thorough, but easily understandable.

<http://neuralnetworksanddeeplearning.com/>

**Link 2:** This website also gave me really useful information about neural networks and their implementation

I also wanted to thank Eric Eaton - a professor here at Penn that teaches a class about neural networks and artificial intelligence – he gave me some great resources from his class that were very useful.