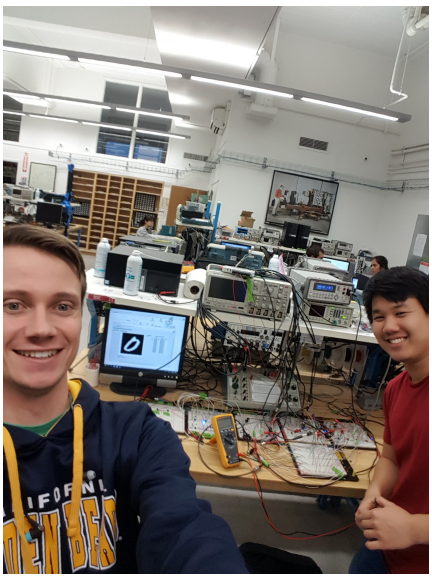
Analog Neural Network for Image Classification

Author - Sean Conlon, **Partner** - Stephen Zhang

December 19, 2019

Introduction		20
Diagrams		20
Description		40
Level of Functioning		40
Design Creativity		40
Measurements		10
Conclusions		10
References		5
Total		**185**



1 Introduction

The purpose of this project was to build an analog neural network that successfully classifies handwritten digits from the MNIST dataset. This was accomplished in three steps. First, a series of network architectures were simulated digitally to test their classification accuracy. Second, the weights from the best network architecture were then used to build the analog network. Third, the accuracy of the analog network was measured by classifying one hundred images randomly selected from the test set. The analog neural network was tested to be 83 percent accurate.

1.1 Theory

The goal of a classifying neural network is to take a vector object $\boldsymbol{X}$ and assign to it a class y_i from the set of classes $\boldsymbol{Y}$. In the case of image classification, the objects are images and the classes are the digits 0 through 9. The network accomplishes this by approximating a function $\mathcal{F}$ which given a object vector $\boldsymbol{X}$, outputs a set of values, $\boldsymbol{A}$, which are proportional to the probability that the given object belongs to the particular class.

$$\mathcal{F}(\boldsymbol{X}) = \boldsymbol{A}$$

where $A_i = P(Y = y_i|\boldsymbol{X})$. Therefore, classification amounts to selecting the class with the highest output from the network.

The network approximates the function $\mathcal{F}$ by taking advantage of the Universal Approximation theorem[1] which states that a two layer neural network with a sufficient number of neurons can approximate any continuous, real valued function. The structure of such an arbitrary network is shown in Fig. 1. Each neuron takes as inputs the values output by the previous layer. The neuron then applies a linear transformation followed by an activation function. As shown explicitly below,

$$f(\boldsymbol{x}) = ReLU(\boldsymbol{x} \cdot \boldsymbol{w} + b)$$

$$ReLU(x) = max(0, x)$$

where $\boldsymbol{w}$ is a vector of weights, b is a bias term, and $ReLU$ is the activation function. Neurons in the output layer do not have activation functions. The activation function is vital in the hidden layer because it is the source of non-linearity in the network. As a general rule of thumb, the more neurons a network has in its hidden layer the higher the order of relations it will be able to recognize.

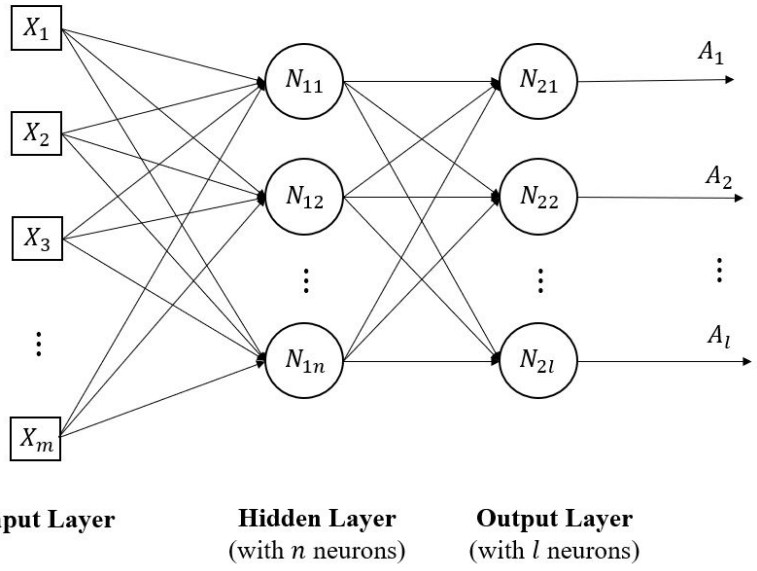


Figure 1: Diagram of a general two layer neural network with m inputs, n hidden layer neurons, and l classes.

Training the network then amounts to fine-tuning the weight vectors for each neuron such that the outputs of the network correctly classify the given object. Exactly how this was done is outside the scope of the project. However, it is important to note for purposes of understanding error tolerances inside the analog network that the weights are chosen in such a fashion to optimize the distance between the correct output, A_i, and the incorrect outputs. In other words, excluding edge cases in which there are multiple equally likely classes, the output of the winning neuron should be much higher than the rest.

1.2 The Data Set and Preprocessing

The MNIST handwritten database[2] was chosen based on its ubiquity in machine learning literature. It consists of 70,000 black and white images of handwritten digits. For the purposes of the project, 14,000 images were selected at random from the database. Eighty percent were then chosen at random to be included in the training set and the rest were placed in the test set.

Traditionally, image classification networks use the values of each pixel as the set of inputs. However, due the limitations associated with building an analog neural network, 784 inputs was unrealistic. Voltage dividers in tandem with a power supply were initially tested as a method for generating input voltages because it would allow for an arbitrary number of voltages to be input. This method was rejected because of its inherent imprecision and lack of flexibility, testing different images would require changing resistors. Instead, the two analog output ports on the DAQ were used as inputs.

The dimensionality of the images was reduced from 784 to 2 using Uniform Manifold Approximation and Projection (UMAP)[3] which is a common technique for general non-linear dimensionality reduction. Its purpose is to map the images onto a lower dimensional space while also preserving the key features of each image. This is analogous to clustering images with similar features. A plot visualizing this is shown in Fig. 2. It is apparent from the plot that the diminsion reduction is not perfect. But, if the images were not separable, then classification would at best require a much more sophisticated neural architecture and at worst be impossible. Finally, the last preprocessing step is to scale the input features to be between plus and minus 10, the output voltage range of the DAQ.

Figure 2: Plot of the test set after UMAP dimension reduction; each color represents a different digit.

1.3 Simulating the Analog Network

Simulating the analog neural network served three purposes: help optimize the model parameters, train the network, and test the network's response to imprecise weights.

Possible model parameter's to explore were the number of input features, the number of hidden layers, and the number of neurons per hidden layers. As was discussed in the preprocessing section, the number of input features was limited by the number of DAQ output ports. One hidden layer was chosen because it offered simplicity without a substantial loss in accuracy. Likewise, five hidden layer neurons were chosen because it was the model with the lowest number of hidden layer neurons that maintained a test set classification accuracy of over 90 percent. In total, this model architecture was tested to classify at 93.6% accuracy.

$$\mathcal{F}(\boldsymbol{X}) = \boldsymbol{W_2} \cdot ReLU(\boldsymbol{W_1} \cdot \boldsymbol{X} + \boldsymbol{b_1}) + \boldsymbol{b_2} = \boldsymbol{A}$$

Using the model as described above, the neural network is equivalent to the above function. Where $\boldsymbol{W_1}$ is a (2×5) matrix where the i^{th} column is the weight vector of the i^{th} neuron in the hidden layer, and $\boldsymbol{b_1}$ is a (5×1) vector where the i^th element corresponds to the bias of the i^{th} neuron in the hidden layer. Similarly, $\boldsymbol{W_2}$ is a (5×10) matrix whose rows are comprised of the weight vectors of the output layer neurons, and $\boldsymbol{b_2}$ is a (10×1) vector of biases for the output layer neurons. The dot products in the formula are applied row wise. After training, the weights and biases were found to be as follows.

$$\boldsymbol{W_1} = \begin{pmatrix} -0.64 & -1.85 & 0.44 & 0.22 & 1.50 \\ -1.51 & 0.75 & 0.64 & 2.12 & -0.35 \end{pmatrix}$$

$$\boldsymbol{W_2} = \begin{pmatrix} -1.16 & 1.15 & 0.64 & -0.72 & -0.08 & -0.47 & 0.93 & 0.24 & 0.34 & -0.56 \\ 1.20 & -1.91 & -0.81 & 0.36 & -0.31 & 1.42 & -0.08 & -0.57 & -0.39 & 0.79 \\ -1.19 & 0.57 & 0.25 & 0.76 & -1.53 & -0.77 & -0.63 & 0.25 & 2.07 & 0.61 \\ -0.13 & 0.46 & 1.38 & 0.41 & -0.29 & -1.79 & -0.14 & 1.01 & -0.18 & -0.47 \\ -0.35 & -0.16 & -0.75 & -0.11 & 1.23 & 0.69 & -0.67 & 0.34 & -1.26 & 0.64 \end{pmatrix}$$

$$\boldsymbol{b_1^\top} = \begin{pmatrix} -0.93 & 1.49 & 3.08 & -0.03 & -1.09 \end{pmatrix}$$

$$\boldsymbol{b_1^\top} = \begin{pmatrix} -0.64 & 0.71 & -0.48 & -0.07 & -0.55 & 1.24 & -2.01 & -0.79 & 2.75 & -0.15 \end{pmatrix}$$

Finally, the precision of the weights were rounded from Python's default precision to only two decimal places, as shown above. This change made no substantial change to the accuracy of the model. This indicates that the analog network will be able to withstand some levels of error in the neuron outputs.

2 Circuit Design

2.1 Analog Neuron

The function of a individual neuron is to take the dot product between the input voltages and the set of weights. This can be achieved with an operational amplifier as is shown in the figure below. The output voltage of a general neuron in the hidden layer is

$$-V_{out} = V_1 \left(\frac{50k\Omega}{R_1}\right) + V_2 \left(\frac{50k\Omega}{R_2}\right) + 1V \left(\frac{50k\Omega}{R_{bias}}\right)$$

Likewise, the output of a general neuron in the output layer will follow the same calculation using five inputs and 5 weights. The resistor values were calculated using the following formula:

$$R_n = \frac{100k\Omega}{w_n}$$

where w_n is the weight which was calculated during the digital simulation. A $100k\Omega$ resistor value is used in the formula so that the analog neuron outputs half the voltage as its digital counter part. This was done to prevent the operational amplifiers from railing. Because the voltage will be halved by each layer, the final outputs of the analog network will be multiplied by a corrective factor of 4. The accuracy of the neurons are predicated on the accuracy of the resistor values. For this reason, each resistor was measured with a digital multimeter to be with in 100Ω of the ideal value.

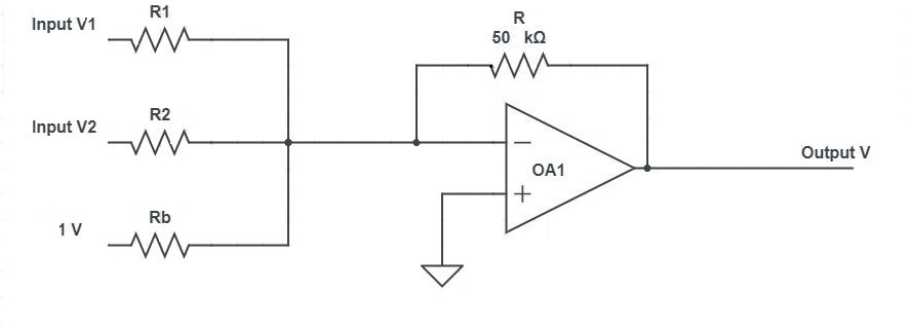
Rather than build negative resistor weights, the input voltages corresponding to negative weights were flipped using a inverted follower circuit. This is shown in more detail in the analog network diagram.

2.2 Analog ReLU Function

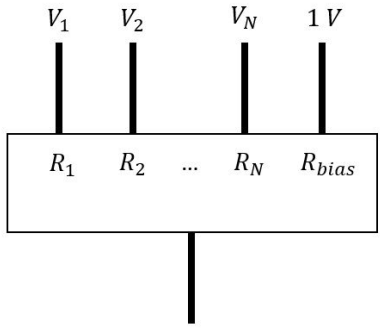
The output of the analog neuron is opposite in sign to the digital neuron. Therefore, the analog ReLU function must ideally transmit negative voltages without loss and block positive voltages entirely.This was done using a pair of oppositely biased diodes, shown below. When the input voltage is positive, the diode connected to V_{out} is reverse biased and does not transmit the signal. And, the diode connected to ground is forward biased and transmits the signal to ground. When the input voltage is negative, the diode to V_{out} passes the voltage through.

There are practical limitations to the accuracy of this design however. First, the diodes have a constant voltage drop across them of roughly $0.6\ V$ which contributed systematic error. Second, the diodes have a break down voltage at which a reversed bias voltage will be transmitted. This break down voltage is less than $12\ V$, the approximate maximum output of the analog neuron. Therefore, on the order of one or two positive volts were transmitted by the ReLU for neurons with strongly positive outputs. As a result, this is the main source of error for images which have strong positive outputs in the hidden layer.

An alternative design using a comparator and a JFET was also tested. In this design, the input to the ReLU function, V_{in}, was connected to the input to the comparator which would

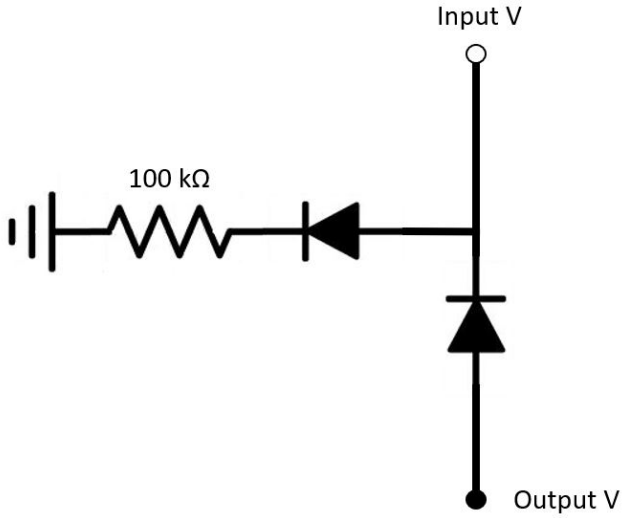


(a) Circuit diagram for a neuron in the hidden layer with arbitrary weights.

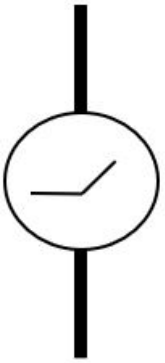


(b) Symbol for a neuron with N number of inputs and arbitrary weights.

compare the input voltage to ground. Then, the output from the comparator was connected to the gate of the JFET. The source and drain of the JFET were connected to V_{in} and V_{out} respectively. This design however suffered from similar problems as the two diode scheme. The voltage drop across the JFET was non-negligible and the comparator was unreliable for small input voltages close to zero.



(a) Circuit diagram for the analog ReLU function

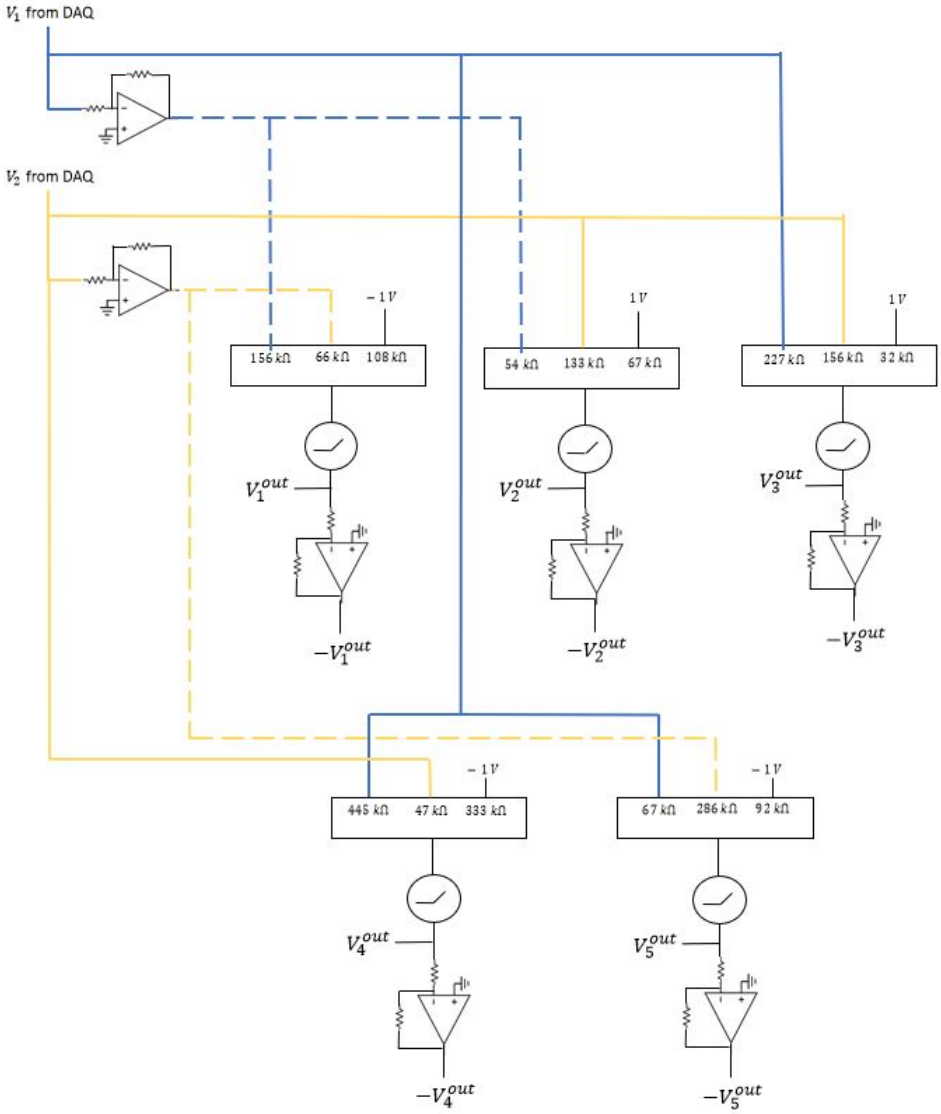


(b) Symbol for the ReLU function

2.3 Analog Neural Network

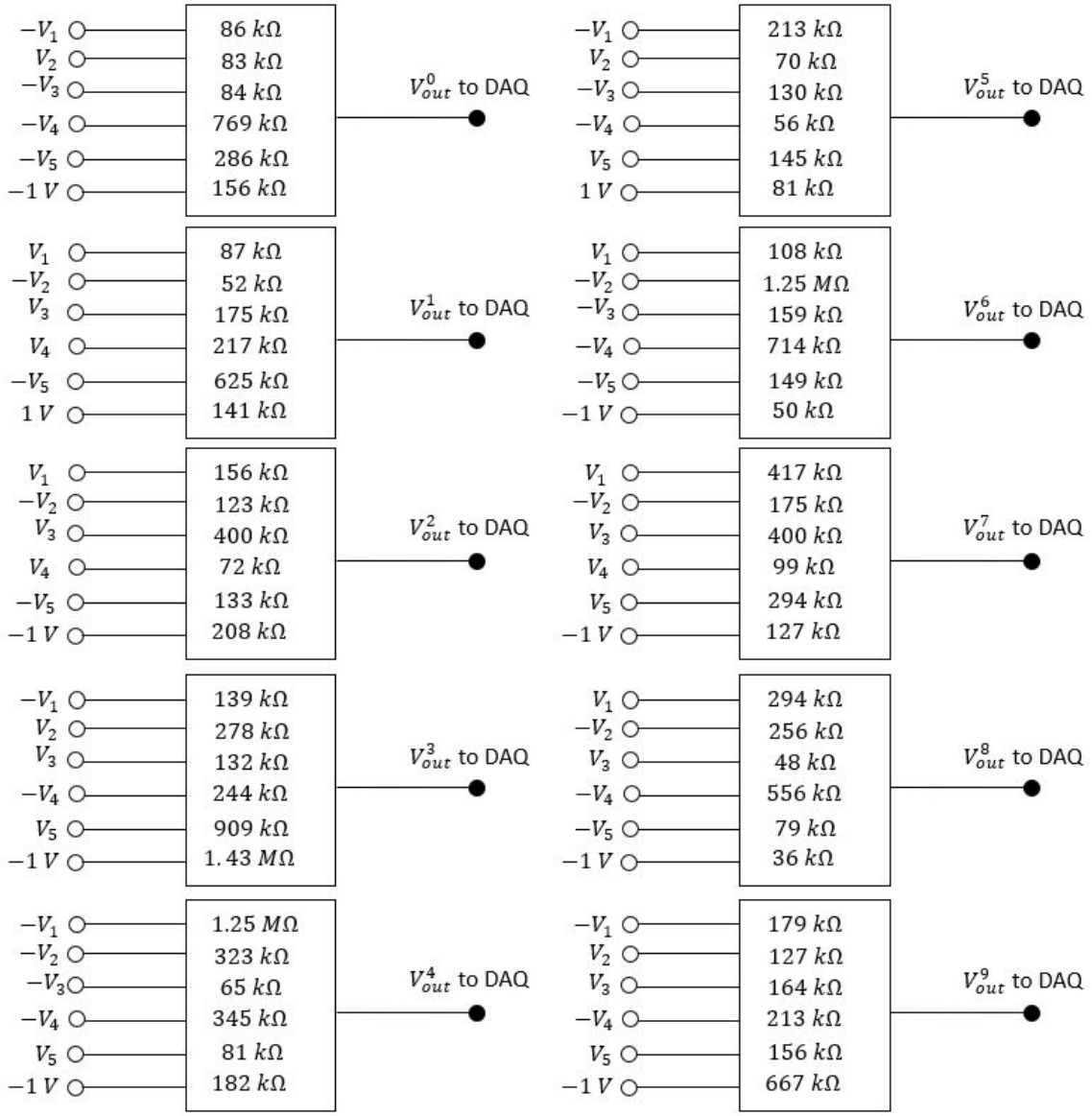
2.3.1 Hidden Layer

The hidden layer takes as inputs V_1 and V_2 from the DAQ and outputs the positive and negative activations for the five neurons. Blue and yellow wires correspond to voltages of V_1 and V_2 respectively. Dashed wires indicate that the sign of the voltage has been flipped due to a negative weight in the neuron.


V_1 from DAQ
V_2 from DAQ
−1 V
156 kΩ
66 kΩ
108 kΩ
1 V
54 kΩ
133 kΩ
67 kΩ
1 V
227 kΩ
156 kΩ
32 kΩ
V_1^{out}
$-V_1^{out}$
V_2^{out}
$-V_2^{out}$
V_3^{out}
$-V_3^{out}$
−1 V
445 kΩ
47 kΩ
333 kΩ
−1 V
67 kΩ
286 kΩ
92 kΩ
V_4^{out}
$-V_4^{out}$
V_5^{out}
$-V_5^{out}$

2.3.2 Output Layer

The neurons in the output layer take the five outputs from the hidden layer as inputs. The sign of these input voltages is determined by the sign of the corresponding weight in the neuron.



2.4 Graphical User Interface

The graphical user interface was built using Lab View and designed to run the network from end to end. The interface takes the index of an image in the test data set as input. It then displays the image on the screen along with the two input voltages into the network, the image's class in the data set, and the digital network's outputs. The program then inputs the voltages into the network and collects the 10 output voltages using the DAQ. It then multiplies by the corrective factor of -4, discussed in the analog neuron section, and displays the final activations on screen. Then, the user can classify the image by selecting the class with the maximum voltage output displayed on screen from the analog network.

The second analog input on the DAQ was shorting the signal, so the digital multimeter was used to measure the voltage of the one's neuron.

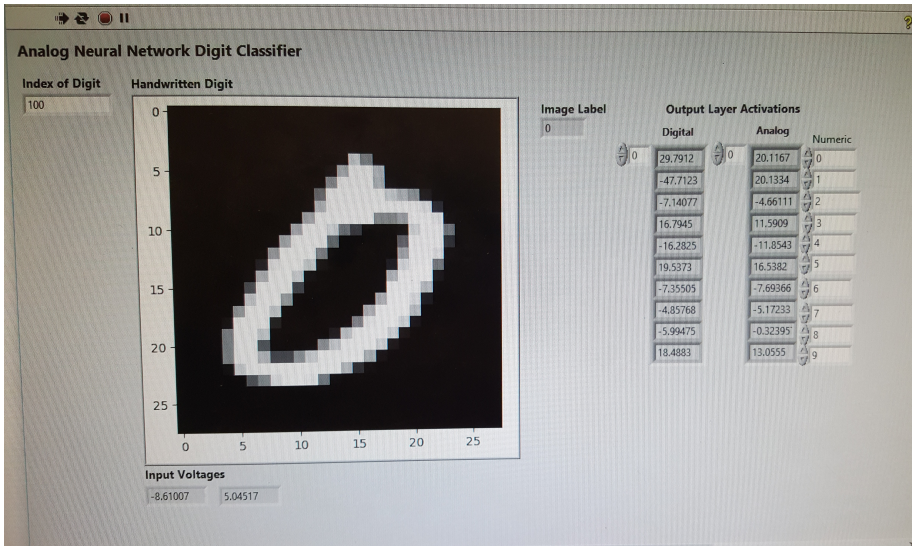


Figure 5: A screenshot of the GUI correctly classifying a zero in the dataset

3 Results

There are two metrics that could be used to judge the accuracy of the network: ability to classify images from the test set and ability to output the same activations as the digital network. The former was measured explicitly over the first 100 images of the test set. The latter was not explicitly quantified due to time constraints however one can make an estimate of the accuracy based on observations.

The results of the analog network's classification accuracy of the first 100 digits in the test set is shown below. The accuracy was measured to be 83%. The accuracy of the digital network over the same 100 images was measured to be 92%. The accuracy for each digit class are shown below. The most likely source of the 9% error is the analog ReLU performance at high positive voltage inputs. The network's relatively low accuracy classifying ones and fours is most likely due to those class's reliance on the ReLU function. The output of each neuron was tested individually to be below 5%; however, it is possible that parasitic oscillations and couplings amongst the 15 neurons created higher error in the neuron's outputs. There was a high variance in the accuracy of the output activations when compared to the digit activations. The percentage difference between the two values ranged from under 1 % up to roughly 15%. This variance is likely due to how well the ReLU was able to block the positive activations in the hidden layer for the given image.

Image Classification Accuracy by Digit Class											
Image Class	**0**	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**Total**
# Correctly Classified	5	11	10	12	4	14	7	10	7	3	83
Total in Data Set	7	16	11	14	6	14	9	12	8	3	100
Percent Accuracy	71.4%	68.8%	90.9%	85.7%	66.7%	100.0%	77.8%	83.3%	87.5%	100.0%	83.0%

4 Conclusion

The analog neural network was able to classify images of handwritten digit with 83% accuracy. This was done by first testing different network architectures by simulating their accuracy. A two layer architecture with four neurons in the hidden layer was chosen. Then, the analog network was built using the weight and bias values from the simulation. A graphical user interface was also built using Lab View to allow for easier testing of the network. Finally, the network's accuracy was tested on the first 100 images in the test set. Likely sources of error in the network are the non-linearity and break down voltage of the diodes in the analog ReLU circuit as well as parasitic capacitance in the fully connected network. The inaccuracies in the ReLU circuit can be mitigated by using a more accurate facsimile to the behavior of the analog ReLU in the digital simulation. This would allow for the network to tune the weights in such a manner to account for the behavior of the diodes rather than assume they behave ideally. The network could also be improved by incorporating the learning stage into the analog network. This would mean replacing the static resistor values used in this project with variable resistors that could be controlled with the DAQ. For the neural architecture used in this project, that would require a DAQ with 75 analog output voltage ports which is impractical. However, building variable resistors for the purpose of building analog neural networks is a field of active research. So, analog neural networks could play a role in the future of artificial intelligence and machine learning.

5 References

[1] *Universal approximation Theorem*, Wikipedia. 2014 Nov.
url(`https://en.wikipedia.org/wiki/Universal_approximation_theorem`)

[2]McInnes, L, Healy, J, *UMAP: Uniform Manifold Approximation and Projection for Dimension Reduction*, ArXiv e-prints 1802.03426, 2018
(url: `https://umap-learn.readthedocs.io/en/latest/`)

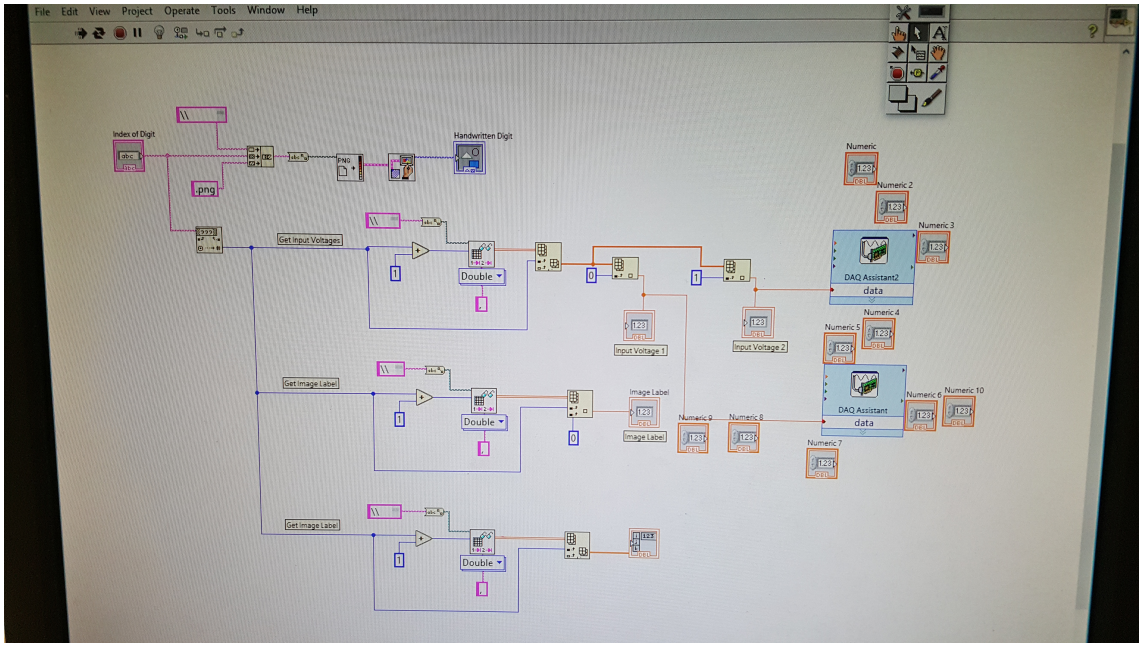
[3]Y.LeCun, C.Cortes, C. Burges. The MNIST Database of Handwritten Digits, 2019.
(url:`http://yann.lecun.com/exdb/mnist/`)

[4]D.Yun, et.al, *An Analog Neural Network Computing Engine using CMOS-Compatible Charge-Trap-Transistor (CTT)*, ArXiv e-prints 1709.06614,
url(`https://arxiv.org/ftp/arxiv/papers/1709/1709.06614.pdf`)

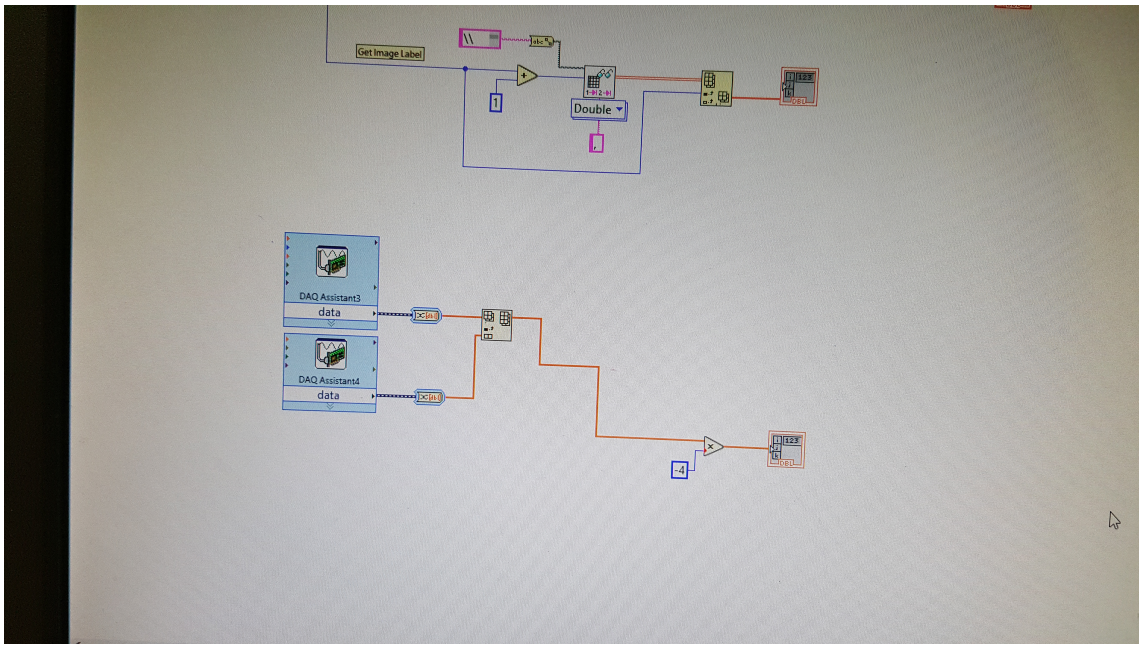
[5]Agrawal, A. *Building Neural Network from scratch*, Towards Data Science. 2018 Jan.
(url: `https://towardsdatascience.com/building-neural-network-from-scratch-9c88535bf8e9`)

6 Appendix A - Lab View GUI

6.1 Data Output



6.2 Data Input



7 Appendix B - Jupyter Notebook

The following pages are print outs of the python notebook used to test the neural networks as well as do some pre and post processing tasks such as convert weights to resistor values. The code for the Network class object is from a Towards Data Science[5] article.