Modified National Institute of Standards and Technology database (MNIST) contains 60,000 training images and 10,000 testing images of 28x28 handwritten numbers. It was created in the late nineties to build and test neural nets. Once an evolutionary feat, implementing a convolutional neural net on MNIST data is now a simple tutorial into neural nets (LeCun, et al.). This data set was used to compare the performance of 16 different neural network architectures. This was achieved by a combination of 4 parameters being varied between 2 values each. The parameters that differed were learning rate (0.001 or 0.01), dropout (0.5 or 0.9), the number of convolutional layers (1 or 2), and the number of fully connected layers (1 or 2). Overall, the structure was the convolutional layer(s) followed by the fully connected layer(s). Dropout took place before the final fully connected layer. Each neural network architecture ran for 2000 steps. This neural net was created using the TensorFlow library with Python and analyzed with TensorBoard ("TensorFlow").

The accuracies of the top 14 architectures are likely statistically similar. Correct classification rates of 0.995 to 0.959 gives a difference of about 3.8% between the best and worst performers, which is nearly insignificant. When briefly looking at the accuracies of each model based on dropout, the models with a dropout rate of 10% consistently have slightly better accuracies than the models with a dropout rate of 50%. The spread is, again, nearly insignificant. More tests would have to be ran with varied dropouts in order to find how that affects the classification results. A more significant parameter worth noting is the computation time of each model. With larger datasets, the computation can become expensive and may be something to consider when building a neural network. The fastest performer of this set was about 2.5 minutes of computation. These were the simplest models with one convolutional and connected layer each. The changes in dropout did not seem to affect the computational time. The longest runtime at fifteen minutes was, unsurprisingly, the most complicated net. This net had two convolutional layers followed by two fully connected layers. Given the relatively insignificant difference in accuracy between the faster and slower training models, it may be beneficial to use the smaller architecture. This is also just testing over one data set. Testing on a different data set has the potential to show stronger performance from the faster trained models.

Not every set seemed to perform well. Two architectures did not learn anything, and swayed around 0.1 correctly classified, which is the result of just randomly assigned parameters. Two of the nets with an architecture involving a learning rate of 0.01, drop out of 0.9, two fully connected layers, and different convolutional layers can be seen at the bottom of the graph in Figure 1. The reasoning behind this is not apparent, and is something to consider in future work.

Figure 1. Graph of test accuracy vs. step number for all 16 models

Interestingly, the learning rate seemed to affect the distribution of the weights. With a higher learning rate, the weights seemed to vary more and have a larger distribution. Conversely, a smaller, slower learning rate had the opposite effect. There was a smaller distribution of weights. This can be seen in Figure 2. This makes sense considering that with a smaller learning rate, the changes to the weights and other parameters, are not updated as quickly as compared to a faster learning rate.

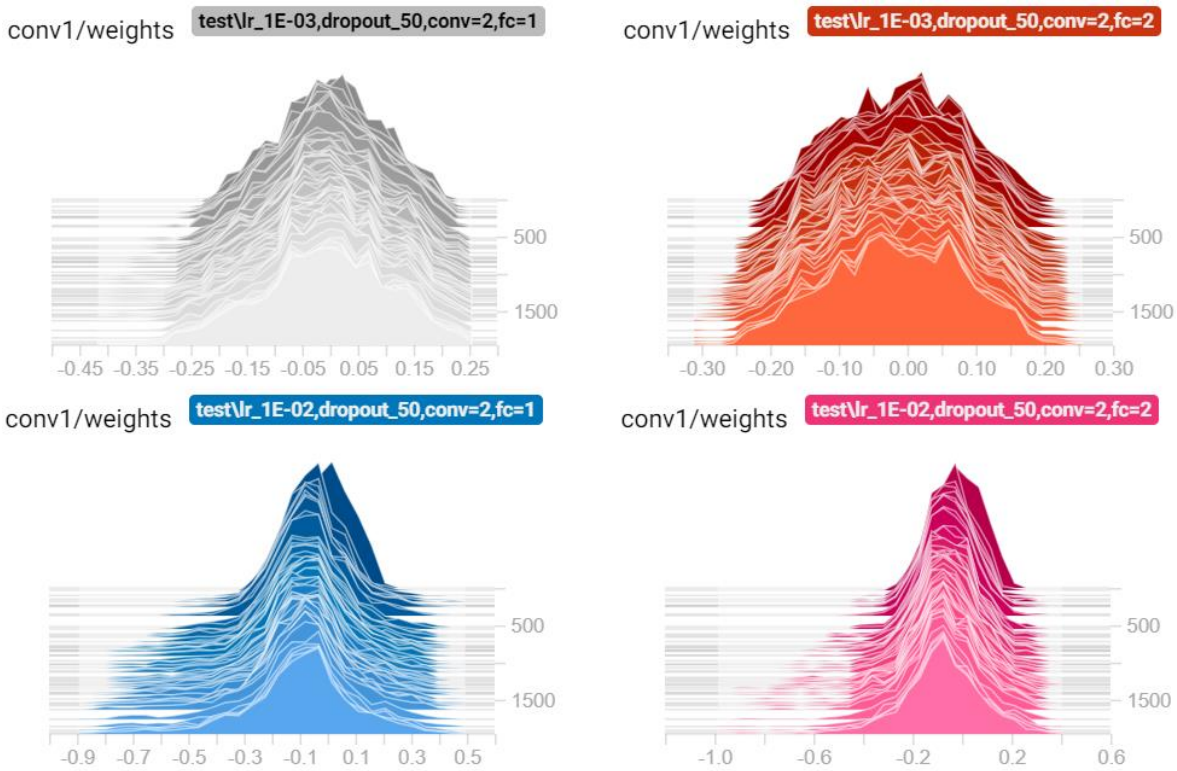

Figure 2. Distribution of weights over time in the 1st convolutional layer for 4 models

Although there may be good accuracy for a set of training data, a system can be judged solely on the training data. Testing data must be considered to see how the system reacts to new data and ensure a sufficient model. From testing 16 different trained models on MNIST, several lessons were learned. Not every model is going to work. The size of the neural network can dramatically affect the time it takes to train. Learning rate can play a part in how distributed a layer's weights become. Though not conclusive, there is some support to indicate that too much dropout can negatively affect the classification accuracy of a model. Finally, and most importantly, nothing is guaranteed with neural networks.

Works Cited

"TensorFlow." *TensorFlow.org*, 1.8, Google Brain Team, 11 Nov. 2015, www.tensorflow.org/.

LeCun, Yann, et al. "Image Database." National Institute of Standards and Technology, 1998.