Lab 2

Sina Erndwein
Samuel Leonardo Gracio

December 23, 2018

1 Short Explanation of the Lab

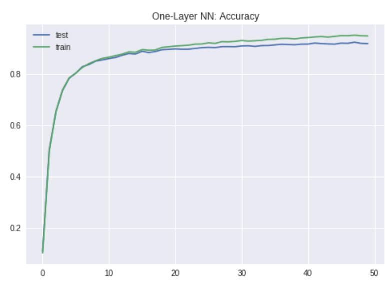
The MNIST data set is used for this lab, which is a collection of handwritten digits. Different neural networks are trained for recognizing the digits including single and multilayer neuworks and convolutional neural network (CNN). In the end, two famous architectures (LeNet5 CNN and AlexNET CNN) are built.

2 Results

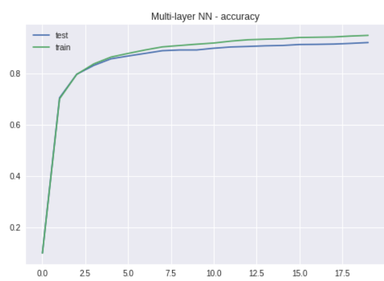
2.1 Single- and Multi layer Network

The single layer network uses zeros as initial weights and biases. Further, a softmax layer builds the model and *GradientDescentOptimizer* is used for minimizing the cross-entropy. The final model accuracy on the test data is **92.17%** with a minimized cost function at about **14480**. The result for accuracy of training and testing data can be seen in figure 1a

The multilayer network has 5 fully connected layers (neurons: 200, 100, 60, 30, 10), the first four using a sigmoid function and the last layer using again the softmax function for output. Otherwise there were no changes except for random initial weights compared to the single layer network. The final cost function after reduction is at about **9500**. Its accuracy on the test data set is about **92%** depending on the initial random weights.



(a) One Layer Network



(b) Multi Layer Network

Figure 1: Accuracy for train and test data for different neural networks

In order to increase performance of the Multi layer neural network, the sigmoid functions are switched for a ReLU function, the *GradientDescentOptimizer* is changed to the *AdamOptimizer*. Further, biases and weights are initialized in a better way (weights = random between -0.2 and 0.2 and bias = 0.1). The learning rate is updated during iterations to get the best learning rate for the dataset. With a final learning rate of **0.00057** the accuracy on the test data is at a value of **97.99%**.

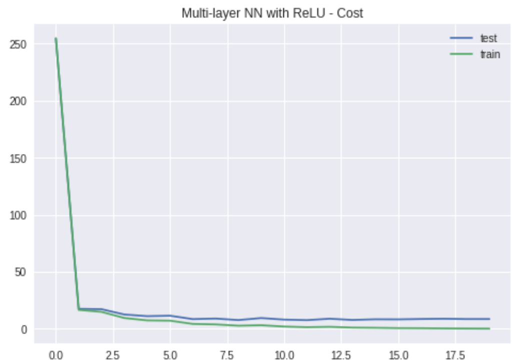
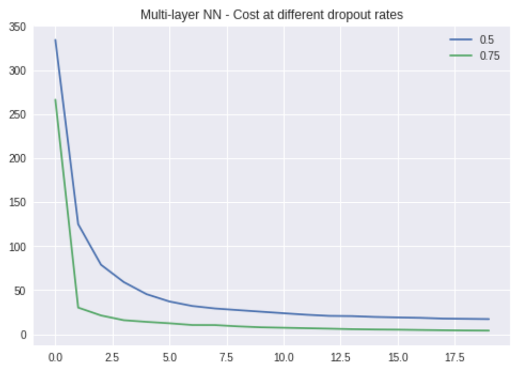


Figure 2: Overfitting can be seen as a difference in cost function of test and training data at high iterations

The question is, can a multi layer network reach even higher accuracies. To avoid overfitting, which can be seen in high iterations at figure **??**, two different dropout rates (0.25 and 0.5) are used during training. With a neuron keeping rate of 50%, the final test accuracy is at **95.12%** and with a keep rate of 75% the final accuracy is at **97.68%**. Interestingly, this did not help to increase the accuracy. Performance differences of the two dropout rates can be seen in figure 3.

(a) Test Data Accuracy

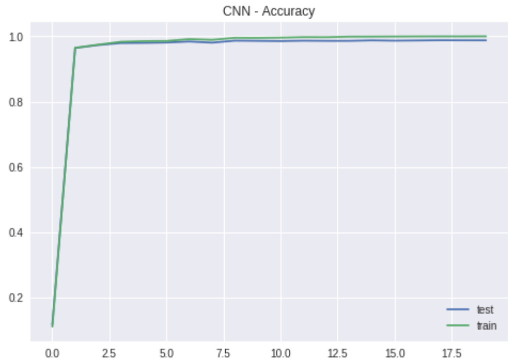


(b) Cost

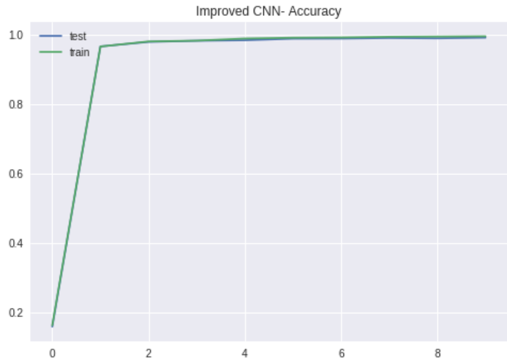
Figure 3: Comparison of Performance Measures of 0.5 and 0.75 keep rate for a Multilayer Network

2.2 Convolutional Neural Network

A CNN should have better performance compared to a multilayer network. For the CNN, a first structure of three convulational layers with 4,8 and 12 filters is used. Afterwards, a fully connected layer is used with a ReLU function and finally, a layer using the softmax function gives the final output. The learning rate is still updated in each iteration. The final test data accuracy is **98.8%**.



(a) Accuracy of a CNN



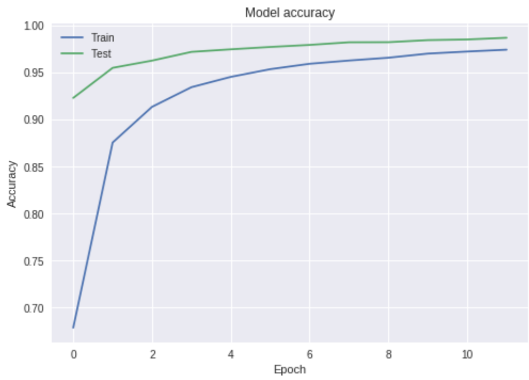
(b) Accuracy of a CNN with dropout rate of 30%

Figure 4: Accuracy for CNN without and with applied dropout

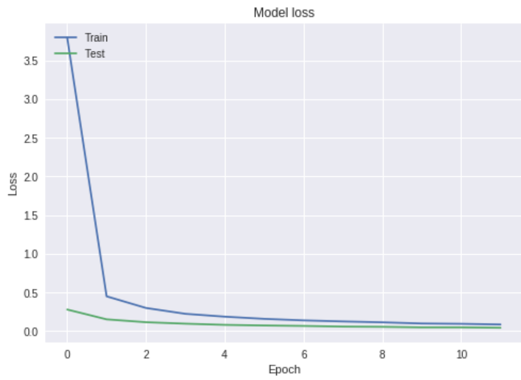
Some overfitting can be seen at high iterations. To improve the performance of the CNN, more filters are used (6, 12, 24) and dropout of 30% is applied to the fully connected layer. The dropout number was determined by tryouts with dropout values between 50 and 80%. After 2000 iterations, the results are a test data accuracy of **99.07%** and overfitting can be avoided as it can be seen in figure 4b.
The same model can be build using the TensorFlow Layers *tf.layers*. The final test accuracy using TensorFlow Layers is at **98.98%**.

2.3 Keras

Instead of Tensorflow, Keras is used in this part for the MNIST digit classification. The learning rate is set to 0.001 since this seemed to give good results and dropout is at 0.7. After 12 epochs, the final test accuracy is at **98.7%**. Plots for this network can be seen in



(a) Accuracy of a CNN built in Keras



(b) Cost of a CNN built in Keras

Figure 5: Performance measures for a CNN built in Keras

2.4 LeNet-5 CNN

For this part, the LeNet-5 architecture for a CNN is used to show its performance on the MNIST dataset. The structure is based on a published paper by LeCun et al. from 1998. By using 0.07 as dropout rate and 0.001 as learning rate as before, the final results for this network are a test data accuracy of **98.68%** after 2000 iterations.

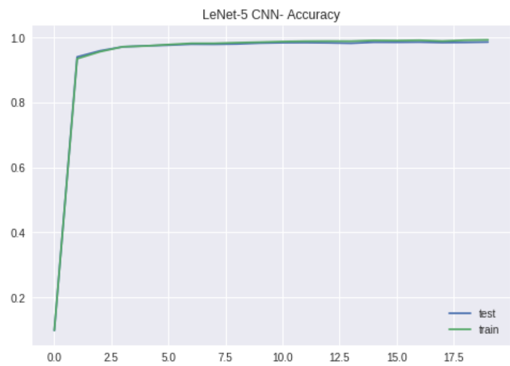


Figure 6: Accuracy of the LeNet-5 CNN

2.5 AlexNET

3 Summary

In the following table, performance measures can be seen for the different implemented neural networks.

Table 1: Performance Measures for Neural Networks

Neural Network	Min Train Loss	Test Accuracy [%]
Single-layer	14261	92.14
Multi-layer	9602	91.86
Multi-layer ReLu	0.2	98.06
Multi-layer ReLu, dropout 0.5	16.1	95.38
Multi-layer ReLu, dropout 0.75	3.58	97.74
CNN	0.14	98.65
Improved CNN	1.69	**99.07**
CNN tf.layers	1.53	98.98
CNN Keras	0.08	98.7
CNN LeNet-5	146	98.68