Sudarshan Pillai

11/14/20

Ali Minai-EECE 6036

HW # 4

1.1 System Description

Number of hidden layers - 1
Number of hidden neurons – 100
Learning rate – 0.01 (tested 0.05, 0.001, 0.1)
Momentum – 0.01 (tested 0.05, 0.001, 0.1)
Output thresholds – 1 is >= 0.75, 0 is < 0.25
Hidden & Output layer activation - Sigmoid
Rule for choosing initial weights – random values between range of -1 to 1
Epochs – 300, this gave the best results for training
Criterion to stop training – if loss function (MSE difference) is less than 0.01
f0 – 0.4 (of pixels to 0)
f1 – 0.05 (of pixels to 1)

1.2 Results

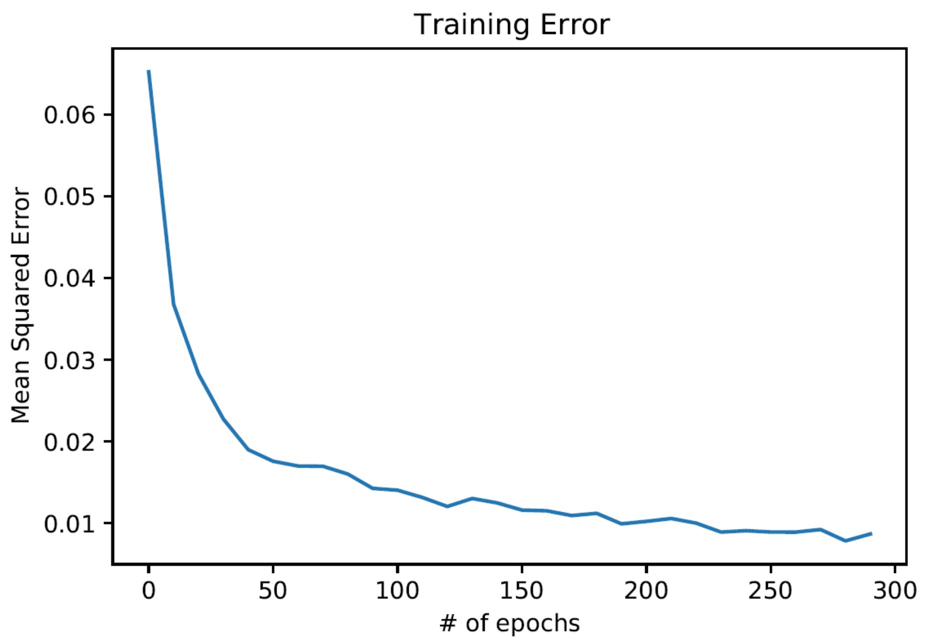


Figure 1.1 – Training error for the Autoencoder with noise

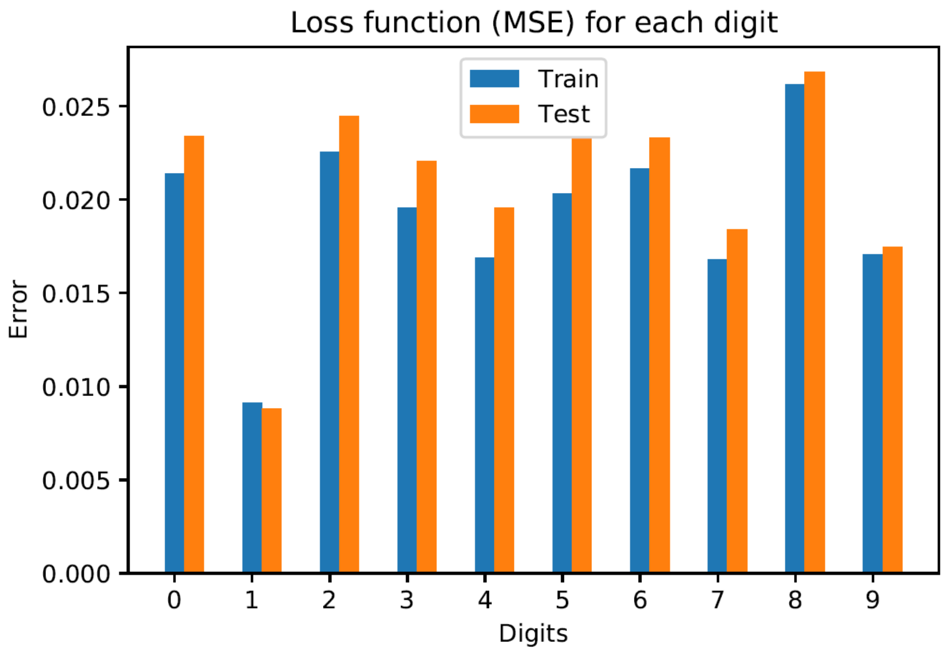
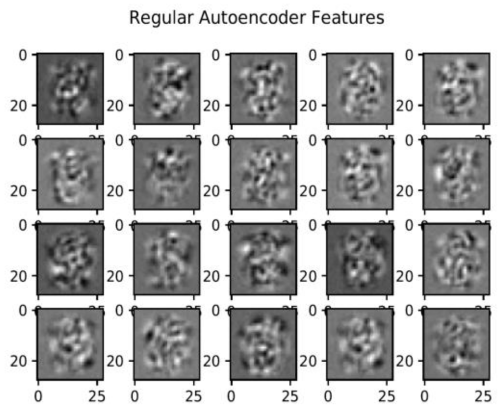


Figure 1.2 – Training and test set error for all digits

1.3 Features



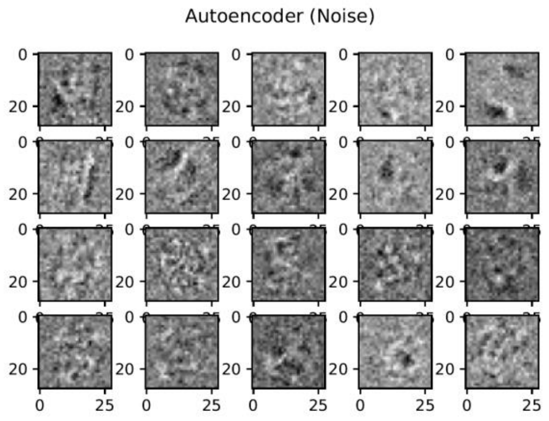


Figure 1.3 – autoencoder features for 20 random neurons from the hidden layer

Figure 1.4 – autoencoder with noise features for 20 random neurons from the hidden layer

Looking at figure 1.3 & 1.4, we can see that both the hidden layer features (autoencoder and autoencoder with noise) are very random, and it is hard to interpret what they are. Although, it is probably expected to at least see some sort of pattern or features.

Finally, there is some difference between both the hidden layer features; the autoencoder with noise feature map seems to have a bit more noise than the regular autoencoder as expected.

1.4 Sample Outputs

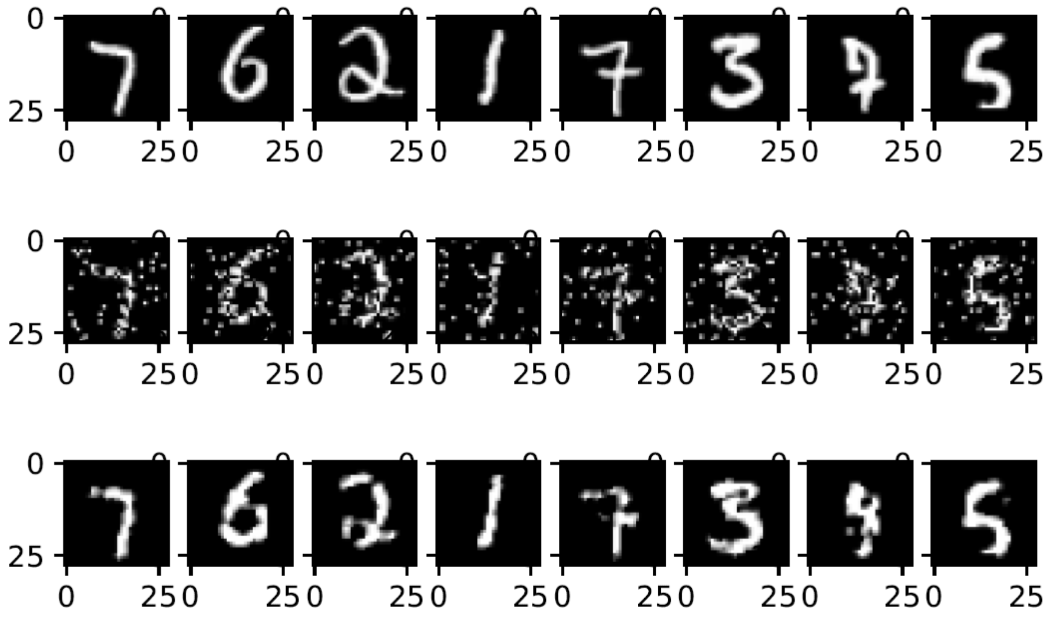


Figure 1.5 - Upper image is true values, middle image is the autoencoder with noise, and bottom image is the image produced from autoencoder

1.5 Analysis of Results

The learning rate was determined by trial and error after running through all the epochs and the final value which gave the best results for this algorithm was *lr = 0.01*. The momentum used for this algorithm after trial and error was *momentum = 0.*01, and the number of hidden layers neurons used was 100. Figure 1.1 shows the overall performance of the network over 300 epochs, and it can be noticed that the error rate decreases as the training continues and eventually stays constant around the 230th epoch. This was due to the condition that the training stops once it reaches an error less than 1%.

Looking at figure 1.2, we can see that the model did a good job on classifying all the numbers. However, it performed the best on predicting the 1s which is probably due to the fact that the shape of a 1 is just a line. Also, looking at the train and test set error for each digit we can see that they are relatively close to each other besides the digit 2 and 5. This is probably due to the fact that 2 and 5 can have many variations which may cause the error to be high.

Figure 1.3 and 1.4 gives the feature images captured by the weight of random 20 neurons and for both the regular autoencoder and the autoencoder with noise. The features of these algorithms are produced by the weights of the hidden layers.

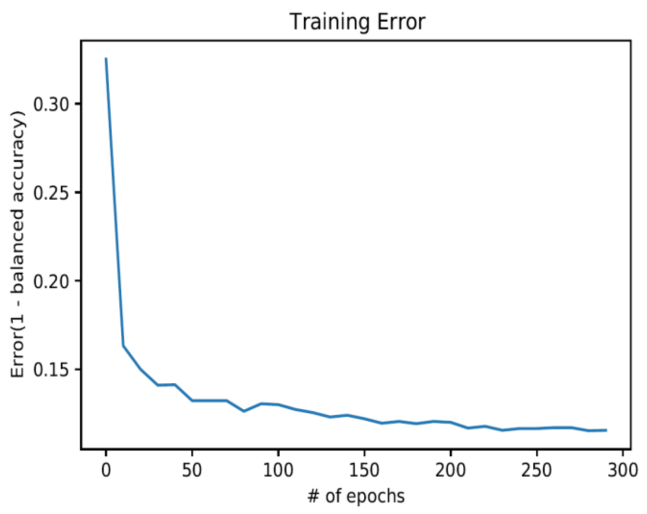
Finally, figure 1.5 shows the 8 different random outputs reconstructed to show the true image, autoencoder with noise, and regular autoencoder. The autoencoder algorithm is good enough to produce the original image with not much error, it shows a true representation as how the naked eye would see it. The autoencoder with noise algorithm was sufficient to produce the original image but as you can see there is some error which may cause some confusion. For this algorithm, f0 and f1 were selected from trial and error. Overall, even the autoencoder with noise was able to produce the original image to almost match the true image.

In conclusion, when comparing the learning task for the regular autoencoder and the autoencoder with noise; it can be concluded that the autoencoder with noise does a much better job learning and capturing better features as it has a much harder task than the regular autoencoder does.

2.1 System Description

Number of hidden layers - 1
Number of hidden neurons – 100
Learning rate – 0.01 (tested 0.05, 0.001, 0.1)
Momentum – 0.01 (tested 0.05, 0.001, 0.1)
Output thresholds – 1 is >= 0.75, 0 is < 0.25
Hidden & Output layer activation - Sigmoid
Rule for choosing initial weights – case1: weights from regular autoencoder &
case2: weights from autoencoder w/noise
Criterion to stop training – if loss function is less than 0.01
Epochs – 300, this gave the best results for training

2.2 Results



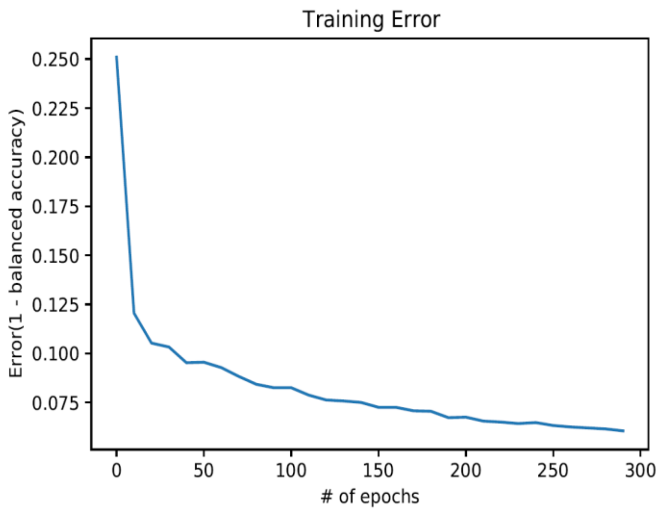
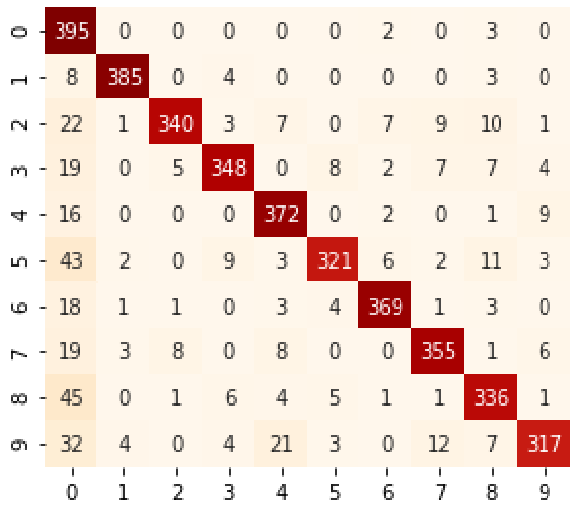
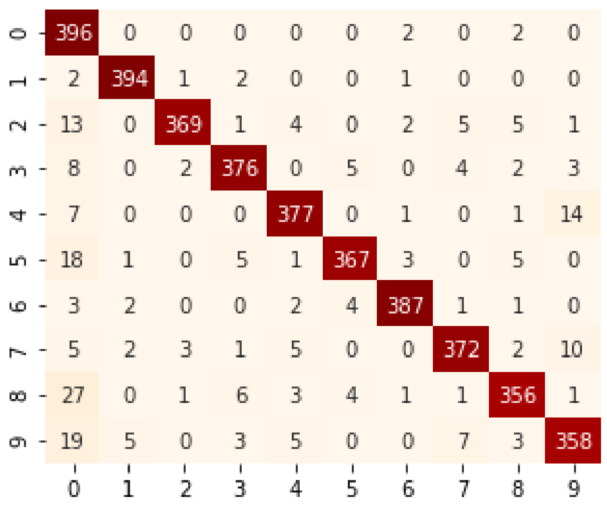


Figure 2.1 – case 1 training error (autoencoder final weights from input to hidden layer)

Figure 2.2 – case 2 training error (autoencoder w/noise final weights from input to hidden layer)



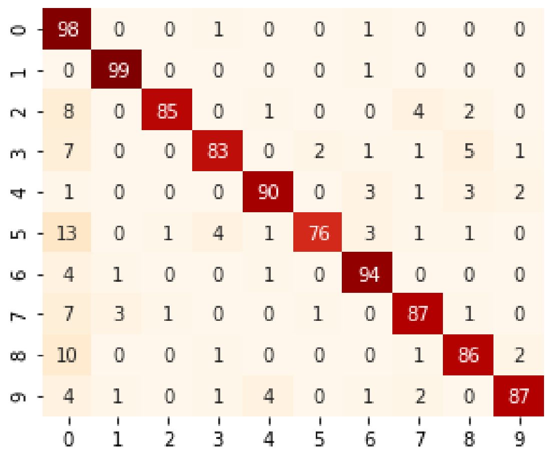
	0	1	2	3	4	5	6	7	8	9
0	395	0	0	0	0	0	2	0	3	0
1	8	385	0	4	0	0	0	0	3	0
2	22	1	340	3	7	0	7	9	10	1
3	19	0	5	348	0	8	2	7	7	4
4	16	0	0	0	372	0	2	0	1	9
5	43	2	0	9	3	321	6	2	11	3
6	18	1	1	0	3	4	369	1	3	0
7	19	3	8	0	8	0	0	355	1	6
8	45	0	1	6	4	5	1	1	336	1
9	32	4	0	4	21	3	0	12	7	317



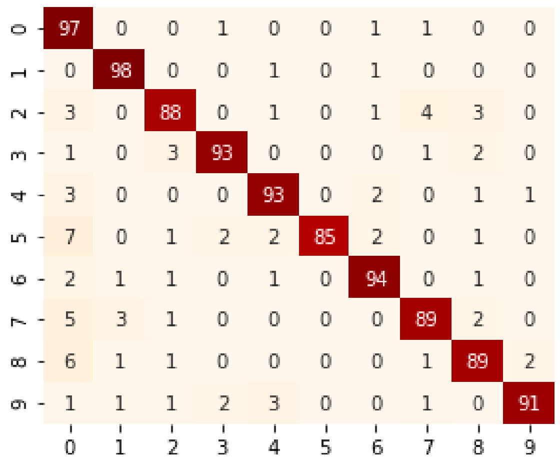
	0	1	2	3	4	5	6	7	8	9
0	396	0	0	0	0	0	2	0	2	0
1	2	394	1	2	0	0	1	0	0	0
2	13	0	369	1	4	0	2	5	5	1
3	8	0	2	376	0	5	0	4	2	3
4	7	0	0	0	377	0	1	0	1	14
5	18	1	0	5	1	367	3	0	5	0
6	3	2	0	0	2	4	387	1	1	0
7	5	2	3	1	5	0	0	372	2	10
8	27	0	1	6	3	4	1	1	356	1
9	19	5	0	3	5	0	0	7	3	358

Figure 2.3 – case 1 confusion matrix for training set (autoencoder final weights from input to hidden layer)

Figure 2.4 – case 2 confusion matrix for training set (autoencoder w/noise final weights from input to hidden layer)



	0	1	2	3	4	5	6	7	8	9
0	98	0	0	1	0	0	1	0	0	0
1	0	99	0	0	0	0	1	0	0	0
2	8	0	85	0	1	0	0	4	2	0
3	7	0	0	83	0	2	1	1	5	1
4	1	0	0	0	90	0	3	1	3	2
5	13	0	1	4	1	76	3	1	1	0
6	4	1	0	0	1	0	94	0	0	0
7	7	3	1	0	0	1	0	87	1	0
8	10	0	0	1	0	0	0	1	86	2
9	4	1	0	1	4	0	1	2	0	87



	0	1	2	3	4	5	6	7	8	9
0	97	0	0	1	0	0	1	1	0	0
1	0	98	0	0	1	0	1	0	0	0
2	3	0	88	0	1	0	1	4	3	0
3	1	0	3	93	0	0	0	1	2	0
4	3	0	0	0	93	0	2	0	1	1
5	7	0	1	2	2	85	2	0	1	0
6	2	1	1	0	1	0	94	0	1	0
7	5	3	1	0	0	0	0	89	2	0
8	6	1	1	0	0	0	0	1	89	2
9	1	1	1	2	3	0	0	1	0	91

Figure 2.5 – case 1 confusion matrix for test set (autoencoder final weights from input to hidden layer)

Figure 2.6 – case 2 confusion matrix for test set (autoencoder w/noise final weights from input to hidden layer)

2.3 Analysis of Results

The learning rate was determined by trial and error after running through all the epochs and the final value which gave the best results for this algorithm was *lr = 0.01*. The momentum used for this algorithm after trial and error was *momentum = 0.*01, and the number of hidden layers neurons used was 100. Figure 2.1 and 2.2 shows the overall performance of the network over 300 epochs, and it can be noticed that the error rate decreases as the training continues and eventually stays constant around the 230th epoch. Also, looking at both the graphs you can see that the training error at the beginning for case 2 is lower than that of case 1. This is probably due to the fact that the autoencoder with noise has better weights than that of the regular autoencoder which is what case 1 uses.

Figure 2.3 and 2.4 gives the confusion matrix for training set for both the cases. It can be noted that both the model (case 1 and case 2) does extremely good on recognizing the digit 1. This is probably because the digit 1 is much easier to detect since it is a straight line with not much loops or curves like the other digits.

Figure 2.5 and 2.6 gives the confusion matrix for the test set of both the cases. It can be noted that overall, the classification of the digits works better for case 2 which utilized the autoencoder with noise weights than case 1 which used the regular autoencoder weights. This is probably because the autoencoder with noise has a harder task to learn so its weights are more precise than those of the regular autoencoder.

3 Program Appendix

“No code needs to be submitted with this homework because the programs used in this homework are the same as those used in homework 3”