EE6132 Advanced Topics in Signal Processing - Assignment No 4

Name: Atishay Ganesh
Roll Number: EE17B155

November 6, 2019

1 Abstract

The goal of this assignment is the following.

- To experiment with Autoencoders.
- To explore the use of Autoencoders in denoising.
- To explore dimensionality reduction using autoencoders (manifold learning).

2 Assignment

2.1 Part 0

Implementation Details: We use Pytorch in Python 3.6 for implementing the Autoencoders Networks. We use torch as the Deep Learning Framework, Tensorboard for plotting the graphs and torchvision for the MNIST dataset and the appropriate transforms.

We write code to load the MNIST Data and split it into train, validation and test sets. (50000-10000-10000 split of images) We normalise the images to 0 to 1 for convenience.

2.2 Comparing PCA and Autoencoders

In this subsection we compare Principal Component Analysis versus Autoencoders for the task of dimensionality reduction. We perform PCA taking only the first 30 eigenvalues and using those to reconstruct the images. We then train an autoencoder to perform dimensionality reduction. We use ReLU activation, with a latent dimension of 30.

The reconstruction Accuracy was:
Using Principal Component Analysis:
MSE Reconstruction Accuracy = 0.017
Using Autoencoders:
MSE Reconstruction Accuracy = 0.0153

2.3 Experimenting with hidden units of varying sizes

In this subsection we compare autoencoders with different sizes of hidden units. We compare the quality of reconstruction visually and using MSE Loss. We vary the latent dimension from 64 to 128 to 256.
The reconstruction Accuracy was:
Latent Dimension 64 - 0.0102
Latent Dimension 128 - 0.0041
Latent Dimension 256- 0.0020
It is observed that the error decreases as the capacity of the network increases. This will only hold good for a while though. The output for random noise is mostly random, but you can see parts of digits in the output as well.

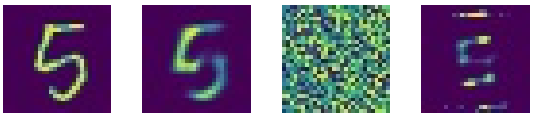

Figure 1: Latent Dimension 64: Original image and reconstruction for an input image and noise, respectively.

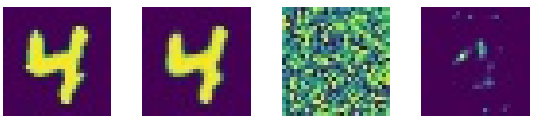

Figure 2: Latent Dimension 128: Original image and reconstruction for an input image and noise, respectively.

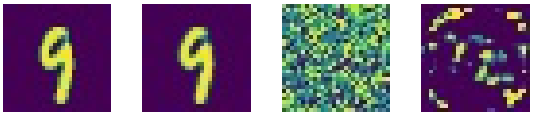

Figure 3: Latent Dimension 256: Original image and reconstruction for an input image and noise, respectively.

2.4 Sparse Autoencoder

In this subsection we regularise an autoencoder with Sparsity regularisation using L1 Penalty. We compare the average hidden layer activations of the two networks (i.e with and without sparsity). We keep the hidden layer size as 32 in both the cases.
The RMS value of hidden layer activation is-
Without enforcing sparsity: 4.347062
After Enforcing sparsity: 2.218107

Crossentropy Loss metric:
Without Sparsity: 0.1118
With Sparsity: 0.1097

2.5 Denoising Autoencoder

In this subsection we train an autoencoder to remove noise from images, in particular, we use salt and pepper noise. We vary the levels of noise as well.

We observe that the original autoencoder does quite poorly at creating a lower dimensional representation of the image. The reconstruction of the image is very bad. The more the noise, the worse it performs, quite obviously. The denoising autoencoder does much better at removing the noise, and works even with higher amounts of noise. With a salt and pepper noise of 0.1, the regular autoencoder network is still able to produce something resembling a digit, but the noise can still be seen. With higher probabilities of noise the standard autoencoder completely fails, while the denoising autoencoder works well. At a noise of 0.8, the number is sometimes barely visible, and sometimes not visible at all.////

MSE Reconstruction Accuracy of regular and denoising autoencoder:

Noise Level	Regular	Denoising
0.1	0.4646	0.2282
0.3	0.9079	0.4227
0.5	1.4033	0.5486
0.8	2.3543	0.6554

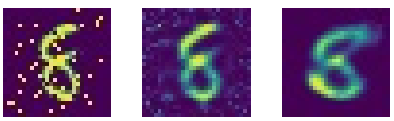

Figure 4: Prob:0.1 Image with noise, Reconstruction after denoising, Regular reconstruction

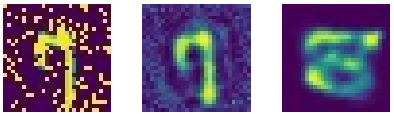

Figure 5: Prob:0.3 Image with noise, Reconstruction after denoising, Regular reconstruction

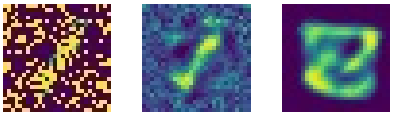

Figure 6: Prob:0.5 Image with noise, Reconstruction after denoising, Regular reconstruction

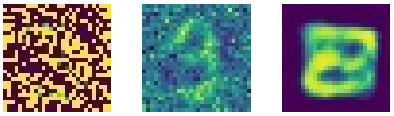

Figure 7: Prob:0.8 Image with noise, Reconstruction after denoising, Regular reconstruction

2.6 Visualing Filters of Various Autoencoders

We compare the filters of the various types of autoencoders, ie Standard, Sparse and Denoising.

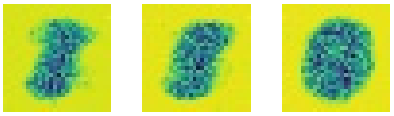

Figure 8: Some Encoding Filters of the Standard Autoencoder

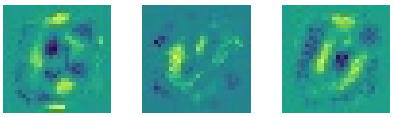

Figure 9: Some Encoding Filters of the Sparse Autoencoder

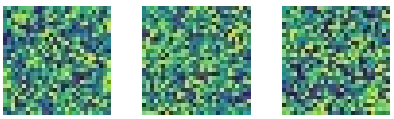

Figure 10: Some Encoding Filters of the Denoising Autoencoder

2.7 Manifold Learning

In this subsection we explore the fact that the Autoencoder learns the lower dimensional manifold that data belongs to.

Moving in random directions from a MNIST image in the 784- dimensional space yields random images. This is because the mnist data occupies a very low dimensional manifold in the 784 dimensional space.

After training an autoencoder and moving slight distances from the representation for a certain input, we observe that the output image is still a number. Some of the times it is the same number and some of the times it is a different number. This is because the autoencoder has learnt the manifold to which the mnist dataset belongs to.

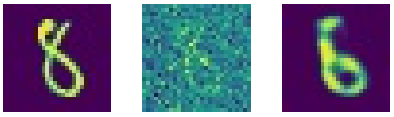

Figure 11: Clean Image, Image after noise is added, and image after noise added in the manifold

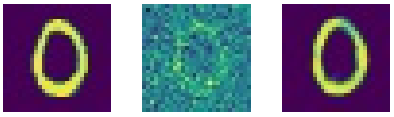

Figure 12: Clean Image, Image after noise is added, and image after noise added in the manifold

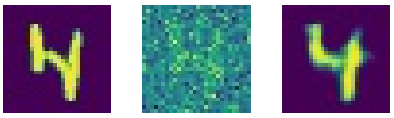

Figure 13: Clean Image, Image after noise is added, and image after noise added in the manifold

2.8 Convolutional Autoencoder

In this subsection we train an Convolutional Autoencoder with 3 convolutional layers for the encoder.

We use Transpose convolution and Max Unpooling and a combination of both the methods. Max Unpooling seems to perform better than the other two methods. Performance of various Decoder Models (MSE of Accuracy):

Unpooling: 0.0033 Transpose Convolution: 0.0150 Unpooling+ Transpose Convolution: 0.0134

Figure 14: Convolutional Filters

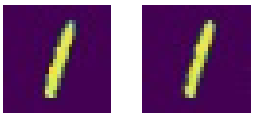

Figure 15: Convoutional Autoencoder with Unpooling Decoder

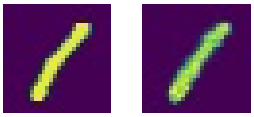

Figure 16: Convoutional Autoencoder with Transpose Convolution Decoder

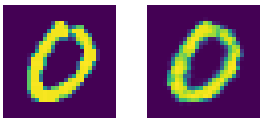

Figure 17: Convoutional Autoencoder with a combination of both in the decoder

3 Conclusions

- We experimented with Autoencoders.
- We compared PCA with Autoencoders
- We studied Convolutional Autoencoders, Sparse Autoencoders and Denoising Autoencoders.
- We explored the ability of the Autoencoder to learn the lower dimensional manifold the data lies in.