Deep Learning - Assignment 5

Transfer Learning

Introduction

This project will explore autoencoders using a tutorial from the Keras development team on how to build autoencoders with Keras and tensorflow background in python. The encoders are built using a random seed number generator. This seed is support to be able to be manipulated to have consistent results by using numpy.random.seed(1234), but there is a known issue with using tensorflow backend that does not set this seed preventing us from obtaining consistent results. The data set used was Fashion MNIST, the same data set in Assignment 4.

The following five encoders were built using the Fashion MNIST dataset: a simple autoencoder based on a fully-connected layer, a sparse autoencoder, a deep fully-connected autoencoder, a deep convolutional autoencoder, and an image denoising model. The input and output images used to test were recorded and a model loss plot was made.

Results

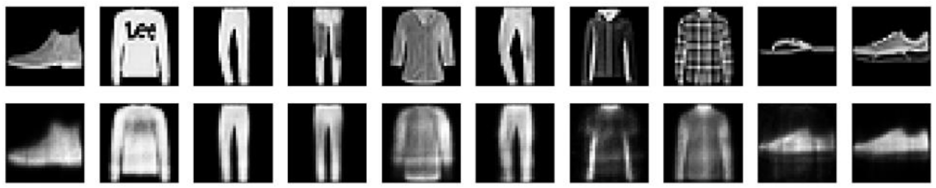


Figure 1: Results of simple autoencoder based on a fully-connected layer

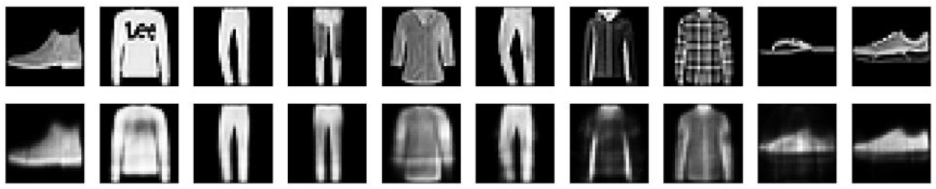


Figure 2: Results of a sparse autoencoder

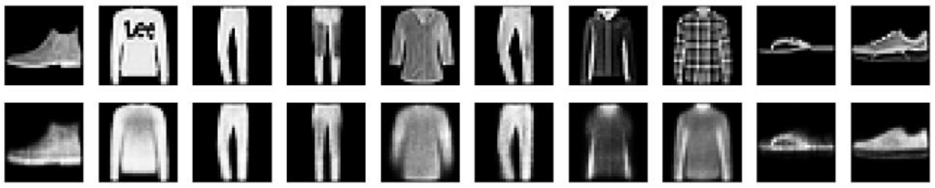


Figure 3: Results of a deep fully-connected autoencoder

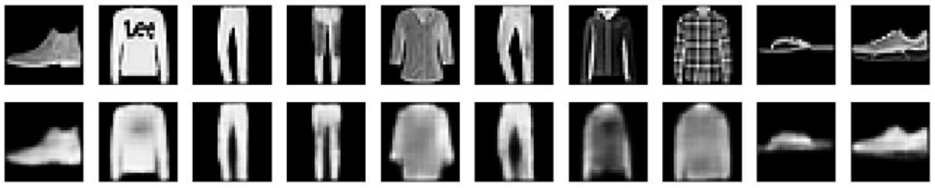


Figure 4: Results of a deep convolutional autoencoder

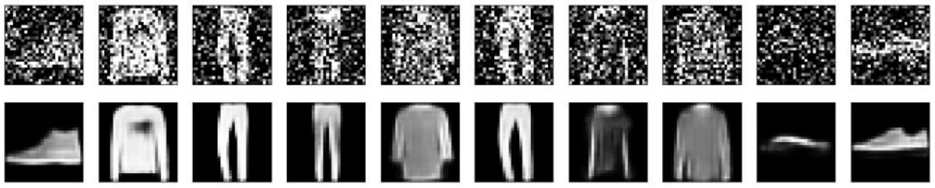
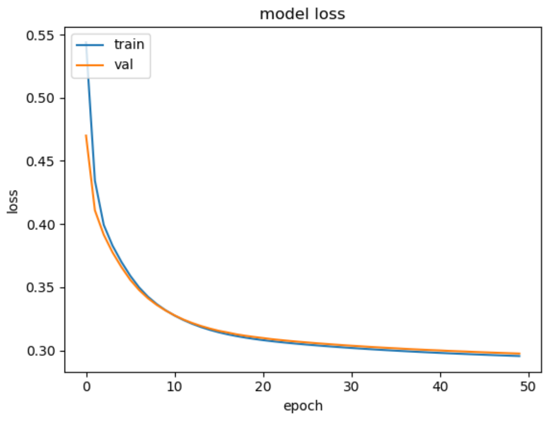


Figure 5: Results of an image denoising model



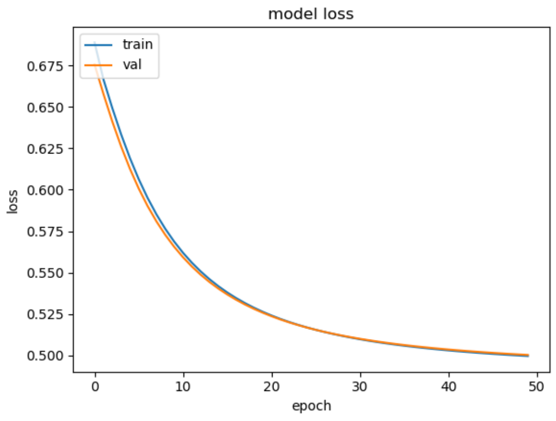


Figure 6: Model Loss of a simple autoencoder(AE) *Figure 7: Model Loss of a sparse autoencoder(AE)*

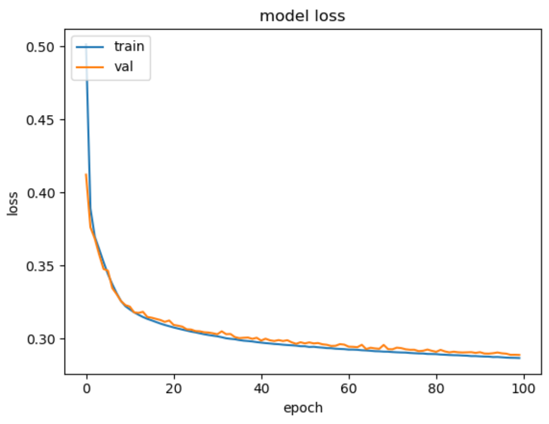


Figure 8: Model Loss of a deep fully-connected AE

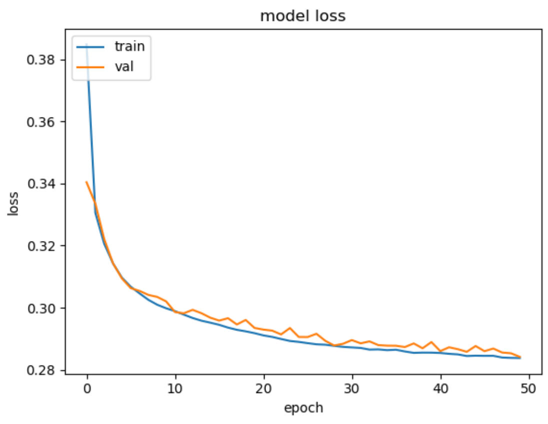


Figure 9: Model Loss of a deep convolutional AE

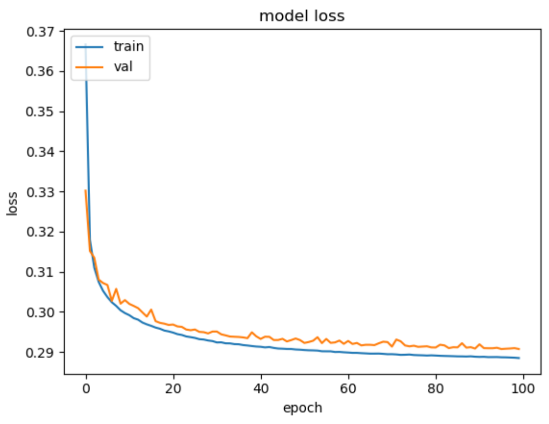


Figure 10: Model Loss of an image denoising model

Autoencoders	Loss	Accuracy
Simple	0.2955	0.2975
Sparse	0.2899	0.2921
Deep Connected	0.2867	0.2886
Deep Convolutional	0.2838	0.2841
Denoising	0.2885	0.2907

Figure 11: Model Results

Conclusion

All autoencoders performed relatively equal. According to the results the simple autoencoder had the highest loss and highest accuracy. By the pictures, its seemed that the best autoencoder was either deep fully-connected autoencoder. The denoising model had the best results in that the encoded image was the least blurry.

The tutorial had a lot of issues with the code they provided. To get the sparse autoencoder to work the activity regulator had to be 10^{-9}. Separating the encoder layer and the decoder layer was not necessary in parts 1 through 3. This caused part 3 to not work right as it had 3 layers for its decoder. The deep convolutional auto encoder and the denoising model took the longest to build. It took about 2 hours for the deep convolutional model and 5 hours for the denoising model. The best way to build this was to have it build overnight.