Programming Assignment-4: Autoencoders

Nimisha T M and Sharath M S, ee13d037@ee.iitm.ac.in and ee15s050@ee.iitm.ac.in

Due date: 31/10/2017 11:55 PM

Note:

1. For any questions, please schedule a time with TAs before deadline according to their convenience. Please use moodle dicussion threads for posting your doubts and also check it before mailing to TAs, if the same question has been asked earlier.
2. Submit a single zip file in the moodle named as PA1_Rollno.zip containing report and folders containing corresponding codes.
3. Read the problem fully to understand the whole procedure.
4. Late submissions will be evaluated for reduced marks and for each day after the deadline we will reduce the weightage by 10%.

---

1 Experiments

1. PCA and AE comparison:

   In this experiment, we will try to see how a deep autoencoder with non-linearities is better than a PCA. Load the MNIST data and do a PCA over it. Take only the first 30 highest eigen values and corresponding eigen vectors. Now, project the data to these eigen vectors and reconstruct back from the 30 dimensional representation. For more details about PCA read this artice.

   Now, construct a deep autoencoder with layers dimension as (1000—500 —250 — 30 — 250 — 500 — 1000) (i.e the latent representaion to be of 30-dimensional) Train the network on MNIST data with suitable learning rate till convergence. Reconstruct the test data and see the reconstruction accuracy. Compare the reconstruction accuracy with that of PCA.

2. Standard Autoencoder:

   Design a under-complete autoencoder with just one hidden layer that acts as dimensionality reduction for MNIST dataset. Keep the dimension of hidden layer as a variable and train the network for different hidden unit dimensions. Check the reconstruction.

   (a) Pass a hand-written digit image into the auto-encoder and generate the reconstruction. What is the quality of the reconstruction?

   (b) What kind of reconstructions do you get when you pass other types of digit images, non-digit images or even noise as input to the auto-encoder?

   (c) The weight vectors associated with each hidden node is called a filter. Try to visualize the learned filters of the standard auto-encoder as images. Does their structure make any sense to you? What do you think they represent?

3. Design an over-complete autoencoder with Sparsity regularization (Check L1Penalty in torch). We impose sparsity by adding L1 penalty on the hidden layer activation. L1 penalty is nothing but L1 norm on the output of hidden layer. Here, the parameter controls the degree of sparsity (the one you pass to L1 Penalty function while defining the model). Higher the value, more sparser the activations are. You can vary the value of this parameter and observe the change in filter visualizations. Also, if the sparsity is too much, it could lead to bad reconstruction error.

   (a) Compare the average hidden layer activations of the Sparse Auto-Encoder with that of the Standard Auto-Encoder (in the above question). What difference do you observe between the two?

   (b) Now, try to visualize the learned filters of this Sparse Auto-Encoder as images. What difference do you observe in the structure of these filters from the ones you learned using the Standard Auto-Encoder?

4. Design a denoising autoencoder with just one hidden unit.

   (a) What happens when you pass images corrupted with noise to the previously trained Standard Auto-Encoder ?

   (b) Change the noise level (typical values: 0.3, 0.5, 0.8, 0.9) and repeat the above experiments. What kind of variations do you observe in the results.

   (c) Visualize the learned filters for Denoising Auto-Encoders. Compare it with that of Standard Auto-Encoder. What difference do you observe between them?

5. Manifold learning
   As discussed in class, AE learns the manifold on the data. In this experiment, we will try to represent the MNIST data with an AE and try to check what the representation space is learning.

   (a) Take an input data from MNIST. Try moving in random directions (i.e add random noise to it). This implies in a 784-dimensional space, if you randomly sample or randomly move in different direction you end up not getting a valid digit. Why is it so?

   (b) Now train and AE with the following hidden units (782–64—8 –64—782). Once the network is converged pass a digit in test set. Add noise to the representation and try to reconstruct the data. What do you observe and why? Relate with manifold learning.

6. Convolutional Autoencoders
   As discussed in class, AE can also be implemented as fully convolutional networks with the decoder consisting of upsampling operation that can have any of the following operation - i) Unpooling or ii) Unpooling + Deconvolution or iii) Deconvolution.

(a) Train a Convolutional AE for the MNIST data with 3 convolutional layers for encoder and the decoder being the mirror of encoder (i.e a total of 7 convolutional layers for AE with the final convolutional layer mapping to the output).

Architecture for the encoder part: Input-Conv1(8 3x3 filters with stride 1)-2x2 Maxpooling - Conv2(16 3x3 filters with stride 1)-2x2 Maxpooling - Conv3(16 3x3 filters with stride 1)-2x2 Maxpooling (Encoded Representation).

This needs to followed by the decoder network. Experiment with all the three types of upsampling. Refer to this report for good description of convolutional arithmetic in deep learning.

Keeping all the other parameters the same, report on reconstruction error and convergence with the different types of upsampling. Also visualize the decoder weights for the three cases. What do you observe?

2 Submissions

For each question, submit your code and results. Attach a pdf with the convergence plots and your acquired results for each of the subsection. Comment on your observations.