Project 2

EEL 6814

Due December 4, 2019

This project will design an autoencoder network, which projects data to subspaces to obtain features that can be then used for classification. You will use again the fashion MNIST that you used for project 1. We will compare two different approaches: First the stacked auto encoder – SAE (5 hidden layers) trained with MSE as found in the literature. Then you can use an MLP or a SVM as the classifier for the bottleneck layer outputs (features) of the trained SAE. Both should be compared with a CNN classifier. I suggest the SAE layers to be 500-200-XXX, i.e. the number of units of the bottleneck layer (which selects the dimensionality of the feature space) is selected by you, and it should be as small as possible for good generalization. You should experiment with several bottleneck layer sizes.

Second, we will use just the encoding part (the layers up to the bottleneck layer) of the SAE network, which is a 3 layer MLP as defined above (# of units per layer should be the same to allow for a comparison of cost methods), and train it with information losses: a) maximum cross entropy (XEnt); and b) maximum mutual information (MMI) –EXTRA POINTS. To classify the outputs of the MLP trained with ITL, use a maximum a posteriori (MAP) rule using the outputs of the trained MLP as the prior, which is simple and should be sufficient for good results.

The two ITL criteria still need class information. The labels exist in 10 dimensional space as points (1 hot encoding). But I suggest that you use a 10 dimensional Gaussian distribution at each location of the “1”, with a small bandwidth compared with the inter-point distance, instead of the exact target value of “1”. This avoids mixing continuous and discrete r.v. which do not have the same measure (minimum entropy for discrete r.v. is 0, while minimum entropy for continuous r.v. is minus infinite….).

Note that you must estimate XEnt and MMI with kernels, as explained in chapter 2 of the ITL book. The last homework had you write the code for MEE, and the estimators are basically the same (i.e. sums of Gaussian terms). Recall that the only term that changes is dJ/de, which gives the injected error for backprop. The XEnt can be estimated using the simple cross information potential , where y is the output of the MLP and d is the desired target. This is a minor modification to the MEE algorithm you already coded. The only thing you have to think about is how to perform the estimation if the size of the bottleneck layer is different from 10.

To estimate Mutual Information (extra points) I suggest you use the quadratic mutual information (QMI). The estimators are presented in the ch 2 (2.9 and 2.10) of my ITL. This is more involved to code, but the estimators are all double sums of Gaussians for which you already have the basic code. But it should give you better results.

Remember that the goal is to present a comprehensive comparison amongst all these ML algorithms, so use confusion matrices for each method, and also quantify the computation time in your processor.