Project 2 - EEL 6814
Due December 2, 2024 --- Team of two students.

1- design a stacked autoencoder network (SAE)

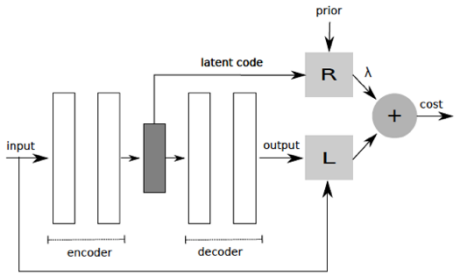
SAE projects data to a subspace to obtain features (codes) that can be then used for classification, similar to PCA. You will use again the Kuzushiji-MNIST from project 1. You will compare two different approaches: First train a SAE (5 hidden layers) with MSE. Once the SAE is trained, use the bottleneck layer outputs (features) as inputs to a Support Vector Machine (SVM) or MLP classifier. I suggest the SAE layers to be 800-200-XXX, i.e. the XXX is the number of units in the bottleneck layer selected by you. Present your strategy to find the best size of the bottleneck layer (as well as for the other hyper parameters) and support it with experiments.

Use the MLP classifier results from project 1 as a comparison to evaluate the obtained classification results. Compare the network sizes of each approach (MLP versus SAE+Classifier) and correlate the performance drop with the size of the bottleneck layer using confusion matrices. Explain the results.

Another application of SAE is to create new images only with the decoder. Freeze the SAE weights. From a training set image, store one code for each of the classes and add zero mean Gaussian noise to it, with varying variance. Generate 1,000 images. Test the generalization of Project 1 classifier in these images artificially created by the SAE as a function of the noise variance. If you use the 1,000 images to augment the training set for the classifier does performance improve? Show a confusion matrix.

2- design a penalty function that enhances discrimination in the latent space.

With this goal in mind, we are going to add a regularizer to the SAE reconstruction cost function, based on distances. Use the same SAE bottleneck layer size as part 1. Since we know the number of classes and the label for each input, we can create a set of target points (a prior) in the space of the codes. The goal is to design targets to be maximally discriminative i.e. a set of directions (one per class) to help organize the codes when we train the SAE. We are using more information to train the SAE, so classification should improve. Since during training we know the class of each input, we can measure the distance of the current code to its target and create a penalty for the SAE cost. To implement this idea, we will create a penalty (R in the figure) as $R = \frac{1}{K}\sum_{i=1}^{K} d_i$, where K is the batch size and d is the Euclidean distance between the current code and the current target. The new cost function for SAE becomes $J_{new} = L + \lambda R$, where L is the MSE reconstruction cost. This is a "mental model" that you must fill-in details with your team member.



The details are: the definition of the target directions in the space of the codes, the size of the batch K, the distance measure you will use, the selection of λ, the order of presentation of samples and the integration with the training of the SAE. Evaluate R first in the training set to quantify the quality of the constraint in the classification stage. To check the quality of your targets, select XXX as 10 such that you can compare the quality of your targets with the direct use of the class assignments as targets for the codes. The comparison should be in terms of the classification accuracy (of the MLP / SVM) in the training set. Validate the selection of λ, R together in a 3D space in training. You will have to retrain the SAE with the new cost: use the same bottleneck size and classifier as in part 1 to compare the classification results. Comment on the quality of this modification for classification.

Remember that the goal is to present a comprehensive comparison amongst all these ML algorithms, so use confusion matrices for each method, and quantify the computation time in your computer. The project was thought out for two students, so mention who did what. The report format is as in Project 1.