GR5241 Spring 2022

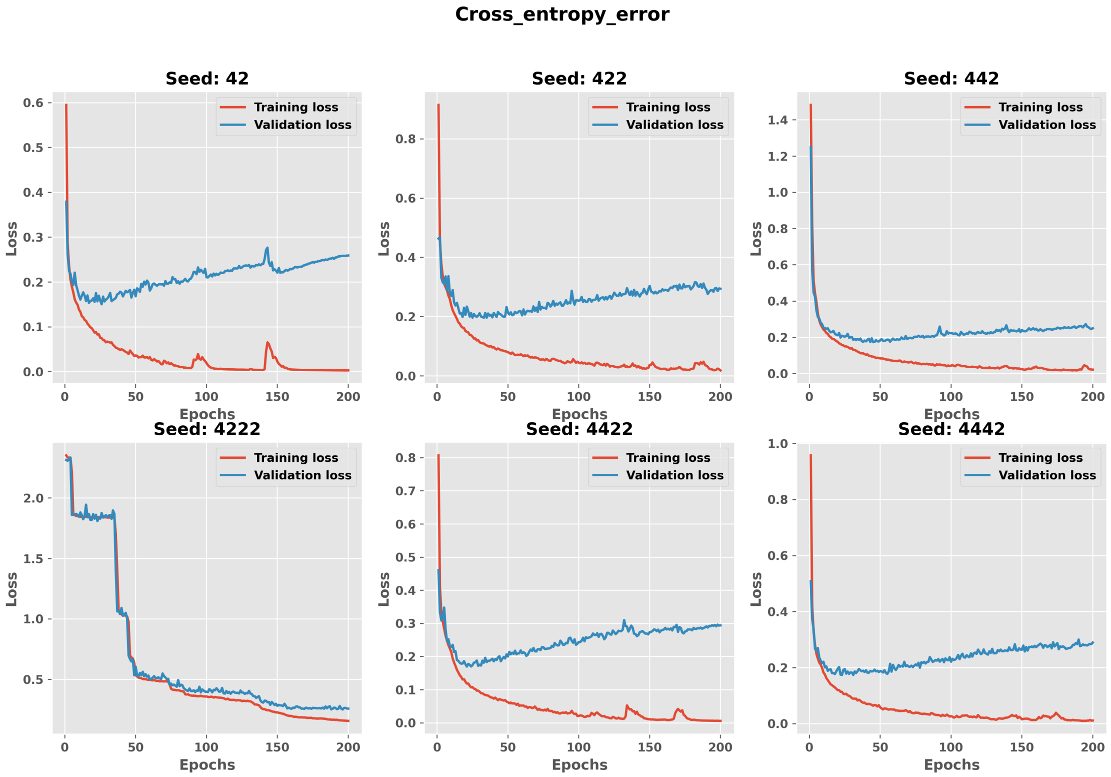
Project Milestone 2: Deep Learning Part

Name: Kangshuo Li UNI: kl3259

Part 4: Deep Learning

3. (5 points) Train a single layer neural network with 100 hidden units (e.g. with architecture: 784 → 100 → 10). You should use the initialization scheme discussed in class and choose a reasonable learning rate (i.e. 0.1). Train the network repeatedly (more than 5 times) using different random seeds, so that each time, you start with a slightly different initialization of the weights. Run the optimization for at least 150 epochs each time. If you observe underfitting, continue training the network for more epochs until you start seeing overfitting.

(a) Plot the average training cross-entropy error (sum of the cross-entropy error terms over the training dataset divided by the total number of training example) on the y-axis vs. the epoch number (x-axis). On the same figure, plot the average validation cross-entropy error function. Examine the plots of training error and validation/test error (generalization). How does the network's performance differ on the training set versus the validation set during learning? Use the plot of training and testing error curves to support your argument.

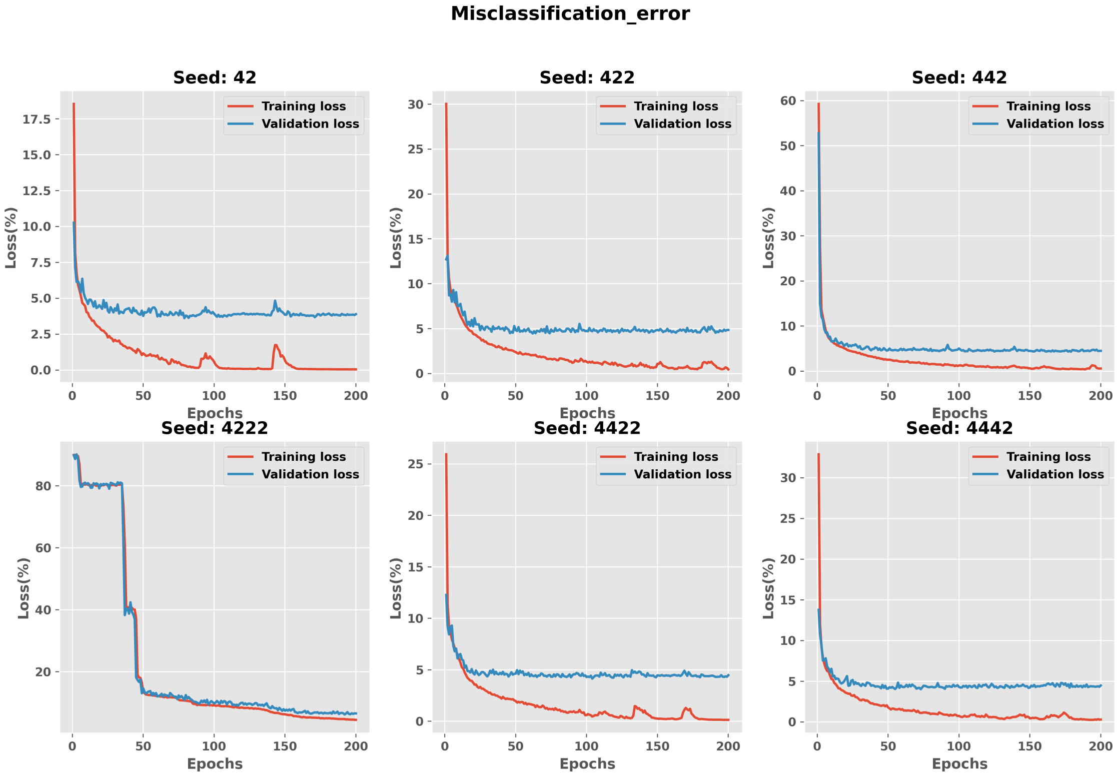


The leanring curves with respect to cross-entropy error of all 6 random initializations are shown above.

In this question, we set the learning rate $l = 0.1$, and the number of epochs equals to 200 with batch size $N_{batch} = 64$. Since we used a setup that requires sufficient epochs to attain a status like overfitting, we can note that most of the training process have a pattern of overfitting. In axes of seed 42, 422, 442, 4422, and 4442, the cross-entropy loss started from different values with different initializations and decreased quickly in the first 25 epochs, then as the number of training epochs increased, the cross-entropy error slowly decreased to 0 with some fluctuations while the validation(we use test set here) cross-entropy error slightly increased from 0.2 to around 0.3, so these plots show that their corresponding training process involves overfitting. In other words, the single neural networks began to become weaker for generalization while still working well on the training set.

Also note that the training process with seed 4222 got unique learning curve, which includes a much higher initial cross-entropy loss and the learning process has several cliff drops and seems to have no significant overfitting pattern. This could be the result of a relatively unlucky initialization with parameters that were far from the local optimal point. Finally it also got the training error close to 0 and validation error close to 0.3, but the time cost to attained same performance is higher than the other random initializations.

(b) We could implement an alternative performance measure to the cross entropy, the mean miss-classification error. We can consider the output correct if the correct label is given a higher probability than the incorrect label, then count up the total number of examples that are classified incorrectly (divided by the total number of examples) according to this criterion for training and validation respectively, and maintain this statistic at the end of each epoch. Plot the classification error (in percentage) vs. number of epochs, for both training and testing. Do you observe a different behavior compared to the behavior of the cross-entropy error function?

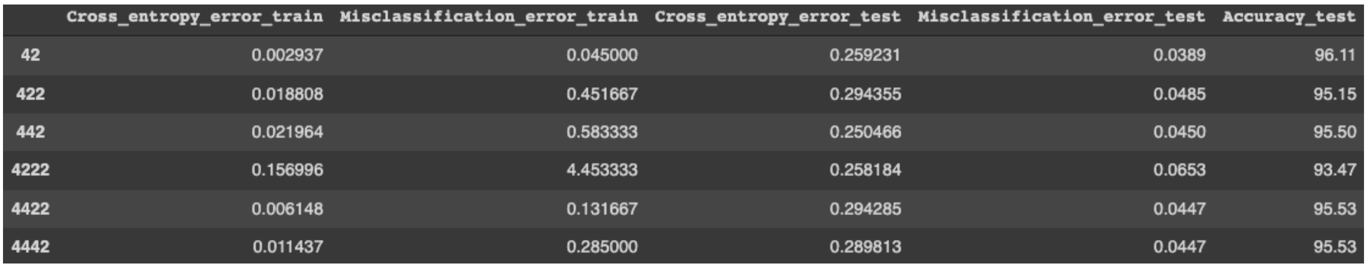


The leanring curves with respect to misclassification error of all 6 random initializations are shown above.

The learning curves of average misclassification error are pretty similar to learning curves based on cross-entropy error. They have the same quick decreasing section in the first 25 epochs and an overfitting section corresponding to the remaining epochs, and the fluctuation is at the same pace. Seed 4222 also got cliff drops in this learning curve and takes longer time to earn the same performance as other neural networks.

However, note that the overfitting parts of these plots are different from those in cross-entropy loss, the validation means misclassification error with seeds 42, 422, 442, 4422, and 4442 nearly maintaining the same value with some little noise in the overfitting phase, and the training loss of mean misclassification keep decreasing to 0. In previous question, the cross-entropy error on test set was slowly increasing. If we measure the ability of generalization of the neural networks by using mean misclassification error, then this ability is not damaged by overfitting.

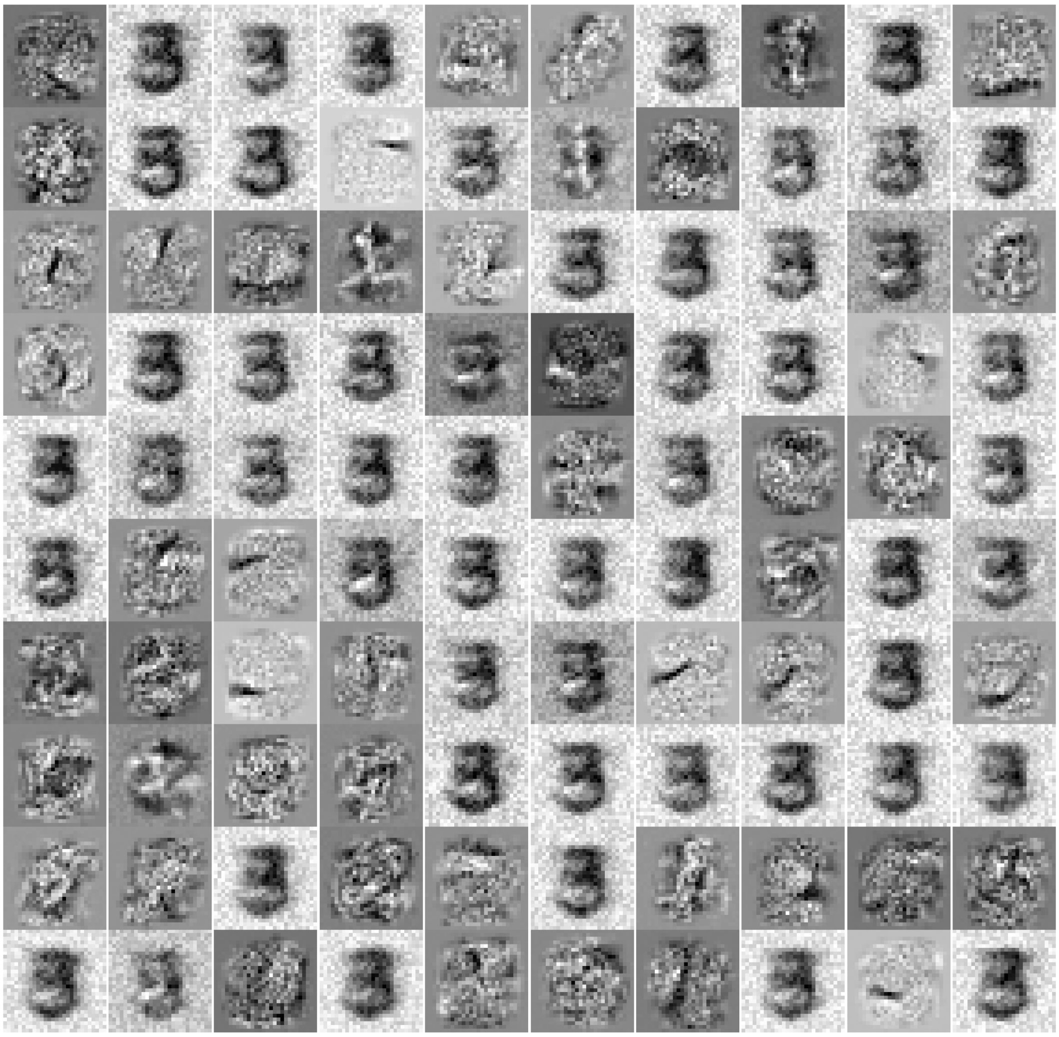
(c) Visualize your best results of the learned W as one hundred 28×28 images (plot all filters as one image, as we have seen in class). Do the learned features exhibit any structure?



	Cross_entropy_error_train	Misclassification_error_train	Cross_entropy_error_test	Misclassification_error_test	Accuracy_test
42	0.002937	0.045000	0.259231	0.0389	96.11
422	0.018808	0.451667	0.294355	0.0485	95.15
442	0.021964	0.583333	0.250466	0.0450	95.50
4222	0.156996	4.453333	0.258184	0.0653	93.47
4422	0.006148	0.131667	0.294285	0.0447	95.53
4442	0.011437	0.285000	0.289813	0.0447	95.53

The table of evaluation metrics is shown above.

We select the best model based on mean misclassification error on test set which can represent the test accuracy, and the cross entropy error on test set. Singel layer neural network with seed 42 has the lowest mean test misclassification error and relatively low test cross-entropy error, so we identify it as the best model.



This is the visualization of the parameters learned from the best model with seed 42. It's clear that the most frequent pattern is the feature like a shape of "3" with a shade like a shape of "8". There are also some chaotic features and some features with only part of the number or strokes shown in the restored features.

(d) Try different values of the learning rate. You should start with a learning rate of 0.1. You should then reduce it to .01, and increase it to 0.2 and 0.5. What happens to the convergence properties of the algorithm (looking at both average cross entropy and % incorrect)? Try momentum of 0.0, 0.5, 0.9. How does momentum affect convergence rate? How would you choose the best value of these parameters?

4. (5 points) Redo part 3(a) - 3(d) with a CNN i.e. with one 2-D convolutional layers → Relu activation → Maxpooling with appropriate hyperparameters. Compare the best result from the single layer neural network and the CNN, what could you conclude?

5. (5 points) Redo part 3(a) - 3(d) with your favorite deep learning architecture (e.g., introducing batch normalization, introducing dropout in training) to beat the performance of SVM with Gaussian Kernel, i.e., to have a test error rate lower than 1.4%.