CSE 569: Project 2 -Report

Rohan Chhibba

Introduction

The subsequent segment of the project comprises dual assignments. The first task necessitates the development and execution of a three-layer Multi-Layer Perceptron (MLP) de novo, subsequently subjecting it to assessment using supplied data. The second task entails the exploration of deep learning methodologies by employing a Convolutional Neural Network (CNN) on the MNIST dataset.

Task 1: Three-Layer MLP Implementation

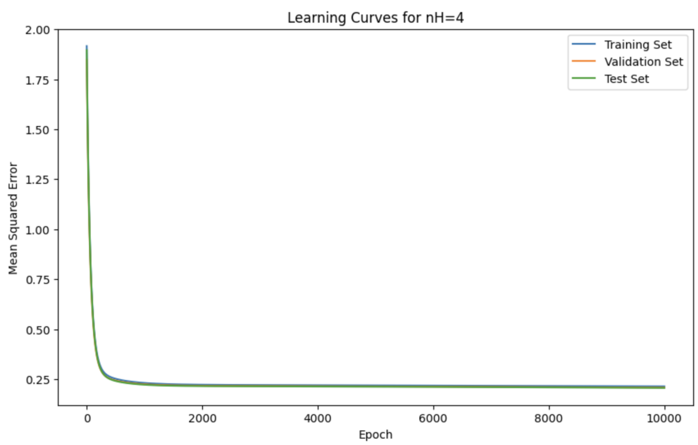
Method

1. Executed the creation of a 2-nH-1 Multi-Layer Perceptron (MLP) without reliance on pre-existing libraries or frameworks.
2. Performed feature normalization, to avoid features with larger values to dominate training, as done for the 1st project.
3. Then, I compared the 2 most popular activation functions, sigmoid and ReLU, and found ReLU to be better in terms of vanishing gradient and faster learning rate. And thus I went forward with the latter.
4. Used Mean Squared Error (MSE) as the loss function(as mentioned in the project).
5. Performed Batch-gradient descent, over stochastic or mini batch, considering advantages of the former like stable convergence, less involvement of randomness in weight updates, etc,.

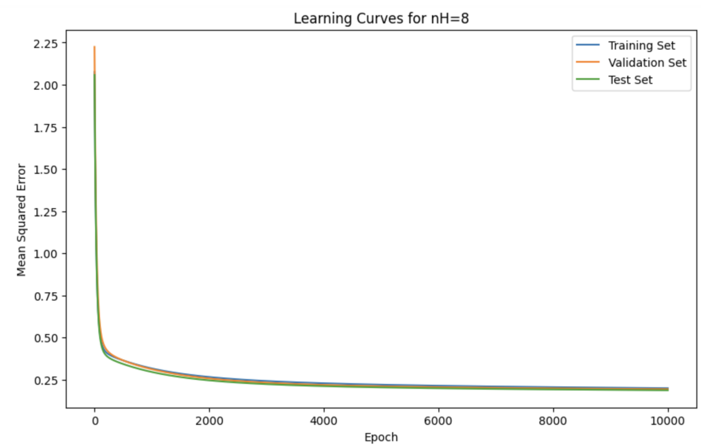
6. Conducted experiments with different numbers of hidden nodes (nH) values: 4, 8, 10, 12, 14.
7. Performed training until validation loss didn't decrease below a threshold. This method ensures prevention from overfitting, optimal model selection, and a more generalized performance.
8. Reported the nH value that resulted in the best classification accuracy for the testing set.

Observations

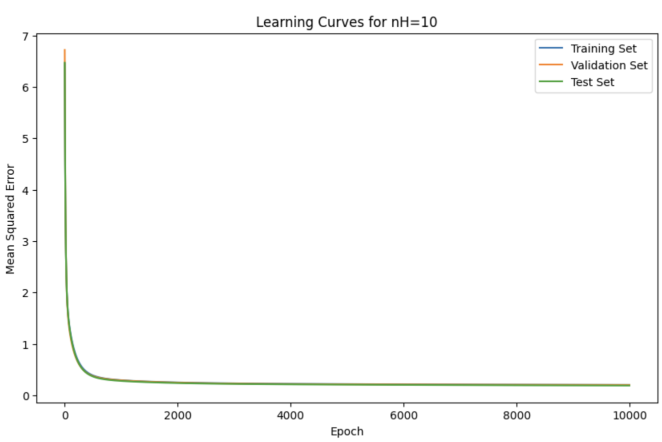
Below are the plots as required in the assignment:-

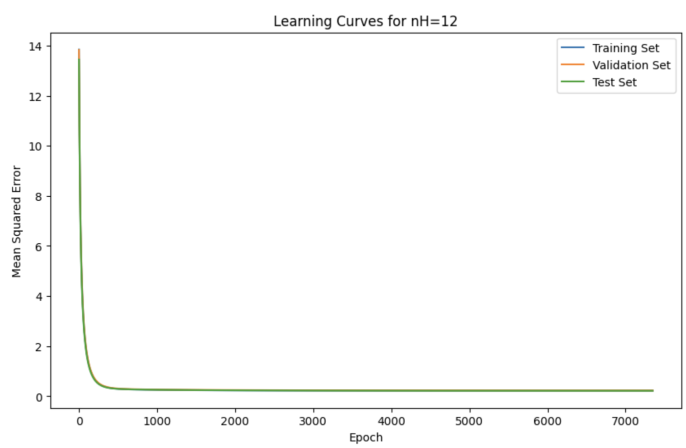


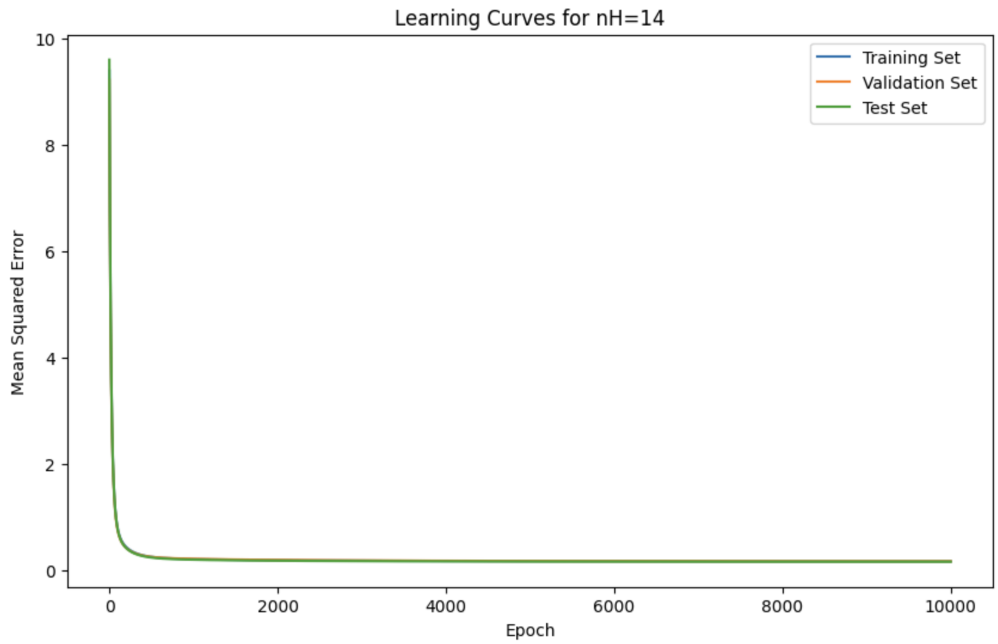

```
Classification Accuracy for nH=4: 63.00%
==============================
```




```
Classification Accuracy for nH=8: 68.45%
==============================
```







Results

- As discussed in the assignment, we for different initializations, ran for nH, and took the average accuracy values to compare the values:-

  Average of 10 runs for each case of nH(neurons in hidden layer)

Number of hidden nodes	Classification Accuracy(%)
4	65.91
8	67.84
10	72.10
12	70.38
14	69.15

Conclusion

- As it can be clearly seen, the accuracy increases from nH = 4 to 10, peaks, and then hovers around in that region. Thus, we could **conclude that nH ~ 10 is a good number of nodes** in the hidden layer for training on this data.

Task 2: CNN Implementation on MNIST Dataset

Method

- Utilized the Keras deep learning library to implement a CNN for Handwritten Digits Recognition on the MNIST dataset.
- Established the Convolutional Neural Network (CNN) with predefined parameters, encompassing convolutional strata, max pooling, densely connected strata, and a softmax output layer.
- Conducted exploratory analyses involving hyperparameter manipulation encompassing variables such as kernel size, quantity of feature maps, neuron count within fully connected layers, and learning rate.
- Reported the test accuracy for at least five different parameter settings, providing explanations for parameter choices based on hypotheses.
- Note:-The initial query evinced a proclivity for high accuracy by default, thereby attenuating the consequential impact of subsequent modifications. To foster a more nuanced understanding, I employed experiment 3 as a referential baseline, juxtaposing the relative performances of experiments 1 and 2 therewith. Furthermore, experiments 4 and 5 were executed in strict adherence to the default parameters enunciated within the posed question.
- Following are the five experiments we performed in this task:-
    - ***Kernel Size in convolutional layer:-*** It was expected that using larger kernel sizes in convolutional layers increases the receptive field, captures more complex features, and may reduce spatial dimensions.
    - ***Reduction in the first layer in the count of feature maps:-*** A reduction in the number of feature maps in a CNN limits the network's capacity to

capture diverse and distinctive features, resulting in lower accuracy due to a constrained ability to represent complex patterns in the data.

- ***Reducing neurons in layers completely connected:-*** Reduces the model's capacity to capture complex relationships and may lead to underfitting, resulting in lower accuracy.
- ***Extremely high learning rate:-*** As we've studied in the class as well, unusually higher learning rates could cause the model to diverge instead of converging.
- ***Change in stride in Max. Pooling Layers:-*** Altering the stride in Max Pooling Layers influences spatial down-sampling: a larger stride increases computational efficiency but reduces spatial resolution more aggressively, while a smaller stride preserves more spatial information at the cost of higher computation.

Results and Observations

- CNN achieved baseline performance on the MNIST dataset, with a test accuracy of **98.54**%.
- Varied hyperparameters to observe their impact on test accuracy.
- Results and observations were summarized for each parameter setting.
- Below are the results for the 5 experiments conducted above:-
    - The accuracy increases as the kernel size increases. - **86.71%**
    - Reduction in the number of feature maps leads to reduction in accuracy of the model as well. - **68.82%**
    - Only after the number of neurons are decreased significantly do we see reduction in classification accuracy. - **86.98%**
    - As we expected, the loss increases with every passing epoch, because the gradient is diverging. - **11.35%**
    - As expected, we observe that the accuracy decreases. - **88.91%**

Conclusion

The project provided insights into the performance of a three-layer MLP and a CNN on specific tasks. Task 1 showcased the influence of hidden layer nodes on classification accuracy, while Task 2 explored the impact of various CNN hyperparameters on Handwritten Digits Recognition. Understanding the behavior of neural networks through experimentation is crucial for optimizing model performance. This is very important while coming up with our own models for different problems, where we aren't given initially the number of layers, or nodes, or other hyperparameters as above. Thus, this task helps us get a glimpse from the end of the researcher.