Intro Deep Learning Homework 1

Jaskin Kabir
Student Id: 801186717
GitHub:
`https://github.com/jaskinkabir/Intro_Deep_Learning/tree/master/HM1`

January 2025

1 Problem 1: Multilayer Perceptrons For Image Classification

1a. *Three Hidden Layers*

Using three hidden layers, consisting of 64, 32, and 16 neurons respectively, the model was trained for 20 epochs. As seen in the curves graphed in Figure 1, the training loss curve has not yet begun to converge to the optimal solution after just 20 epochs. To fully train the model, more epochs would be required. There is also significant overfit, as indicated by the gap between the training and validation accuracy curves.

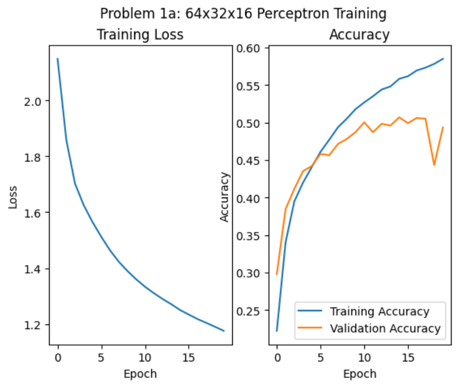

Figure 1: Loss and Accuracy Curves for Three Hidden Layers

The confusion matrix for this model can be seen in Figure 2.

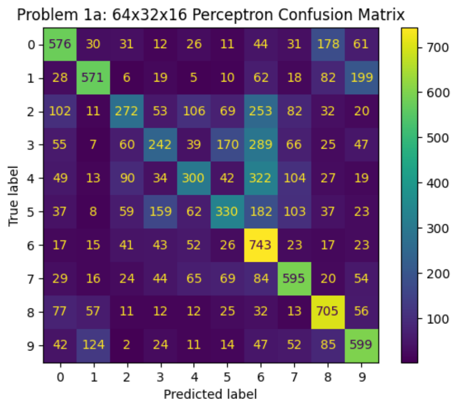

Figure 2: Confusion Matrix for Three Hidden Layers

1b. *Five Hidden Layers*

Using five hidden layers, consisting of 256, 128, 64, 32, and 16 neurons respectively, the model was trained for 20 epochs. As seen in the curves graphed in Figure 3, the model has not yet converged to the optimal solution after 20 epochs. There is also significant overfit, but not as pronounced as it is in the three hidden layer model. The confusion matrix

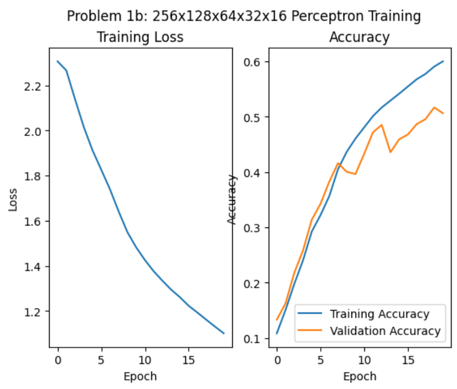

Figure 3: Loss and Accuracy Curves for Five Hidden Layers

for this model can be seen in Figure 4.

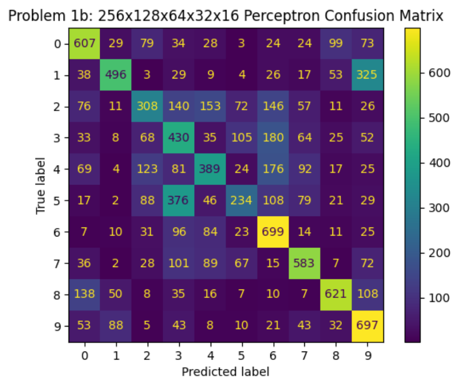

Figure 4: Confusion Matrix for Five Hidden Layers

Comparison

Table 1 shows that the more complex model performed better in every metric. However, even the 5-layer model was barely more accurate than a coin toss. The models are likely underfit, as they have not yet converged to the optimal solution. Additionally, the multilayer perceptron is not the ideal paradigm for image classification. A convolutional neural network would likely outperform both of these models.

	Accuracy	**Precision**	**Recall**	**F1 Score**
1a: 3-Layer Model	49.3	49.8	49.3	48.2
1b: 5-Layer Model	50.6	51.6	50.6	50.1
$\Delta_{a \to b}$	2.59	3.37	2.59	3.69

Table 1: Comparison of Evaluation Metrics

2 Problem 2: Housing Price Regression

1a. *ML Perceptron Regressor*

The housing dataset has several categorical features. In order to use this data to train the model, the categorical values like 'yes' or 'no' were converted to 1 and 0 respectively. Using 3 layers, with each , the model was trained for 750 epochs. As seen in the curves graphed in Figure 5, the model converged to its optimal solution. However, the gap between the training and validation losses indicates significant overfit.

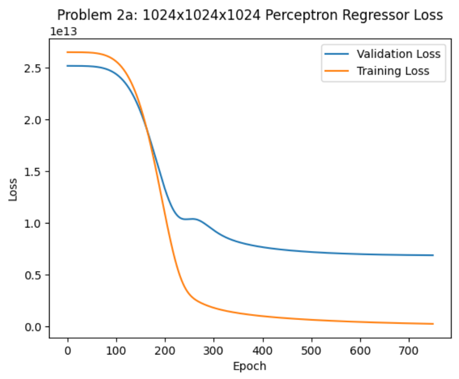

Figure 5: Loss and Accuracy Curves for Perceptron Regressor

1b. *One-Hot Encoding*

Instead of converting the categorical data to numerical values, one-hot encoding was used to convert each categorical feature into a number of binary features that correspond to each discrete value the feature could take on. This prevents the model from interpreting the categorical data as numerical and instead forces it to treat each category as a separate feature.

The same model architecture was used as in the previous section and the model was trained for the same number of epochs. The curves plotted in Figure 6 show that the model suffered from overfitting, but it was not as significant as the previous model.

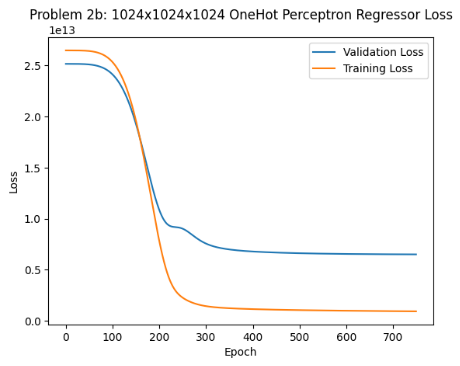

Figure 6: Loss and Accuracy Curves for One-Hot Encoding

1c. *One-Hot Encoding With Increased Complexity* Using the same one-hot encoding method, a more complex model was trained on the dataset. This model used 512 neurons per layer, but had 8 hidden layers. After training the model for 750 epochs, the validation loss curve began to increase due to overfit after about 150 epochs. Thus, the number of training epochs for this model was reduced to 150. As the curves plotted in Figure 7 show, the overfit was much less pronounced n this model, and the number of epochs was sufficient to reach an optimal solution.

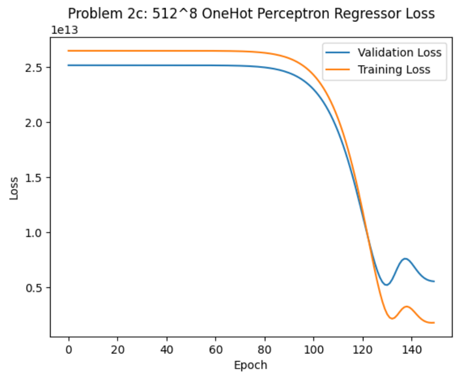

Figure 7: Loss and Accuracy Curves for One-Hot Encoding With Increased Complexity

Comparison

Table 2 shows the root mean square and mean absolute losses for each of the three regressor models discussed thus far. Additionally, the table includes the percent difference between each pair of metrics. From the difference in mean absolute error, it is clear that the one-hot encoding only slightly improved the overall performance of the model when compared to the integer encoding. However, the higher improvement in RMS error indicates the model tended to make fewer large errors. The increased complexity of the third model greatly improved its performance compared to both of the previously explored models, however. The increased complexity allowed the model to learn more about the training data, and it was by far the best performing model.

	RMS Error	**MAE**
2a: Integer Encoding	2.62×10^6	1.98×10^6
2b: One-Hot Encoding	2.56×10^6	1.96×10^6
2c: Complex One-Hot Model	2.35×10^6	1.80×10^6
$\Delta_{a\to b}\%$	2.63	0.65
$\Delta_{a\to c}\%$	10.58	8.48
$\Delta_{b\to c}\%$	8.17	7.88

Table 2: Comparison of Evaluation Metrics