CS5300 Final Project Report - Phase 2

Scott Sanchez

December 2, 2024

Introduction

Phase 2 of this project focused on developing a neural network capable of overfitting the dataset. The goal was to experiment with increasing the capacity of the model—by adding more neurons and layers—until the network memorized the data, achieving near-perfect accuracy. This was to teach us about the size and complexity of the model needed for the dataset, and to establish a baseline for future generalization tasks.

Model Architecture and Training

Dataset Overview

The dataset prepared in Phase 1 was used as input. It included 5,000 rows and multiple numerical and categorical features after cleaning and preprocessing. The target variable, *Severity*, was one-hot encoded into four classes corresponding to the severity levels (1 through 4). Features were normalized to ensure consistent input ranges across all columns.

Neural Network Architecture

The final model used in this phase had the following architecture:

- **Input Layer:** Accepts 43 input features (after preprocessing).
- **Hidden Layers:**
  - First layer: 256 neurons, ReLU activation.
  - Second layer: 128 neurons, ReLU activation.
  - Third layer: 64 neurons, ReLU activation.
  - Fourth layer: 64 neurons, ReLU activation.
- **Output Layer:** 4 neurons (one for each class) with softmax activation.
- **Optimizer:** Adam
- **Loss Function:** Categorical Crossentropy
- **Metrics:** Accuracy
- **Epochs:** 1,500

This architecture was chosen to provide sufficient capacity for memorizing the dataset.

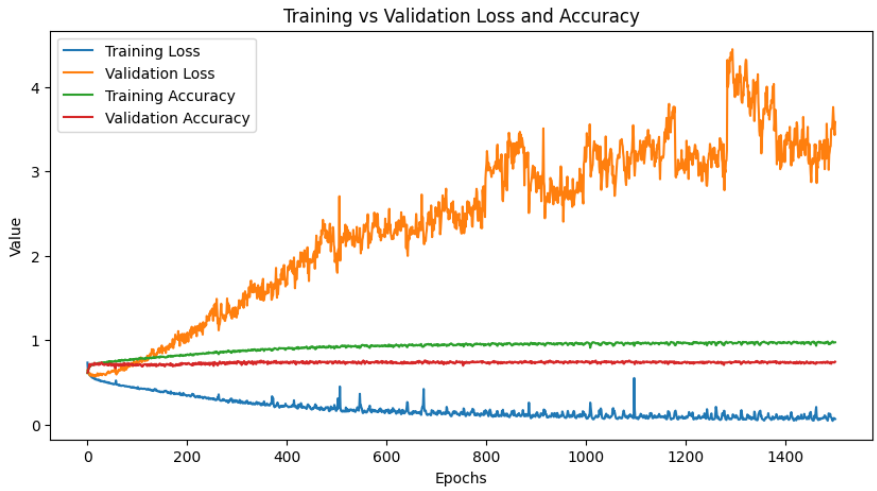

Figure 1: Progression of the Model Overfitting Progress

Results and Observations

Model Performance

By the end of the training, the model achieved an accuracy of nearly 100%. Meanwhile, the training loss (a measure of the model's *inaccurcy* on the training data) approached zero. This is shown in the plot (Figure 1) as the green line very closely approaches 1, while the blue line approaches zero.

By the end of training, the model achieved an accuracy of approximately **97%**. In short, the model became very good at predicting outcomes for elements of the training data set.

As mentioned above, the idea of this phase was to *over*-fit the model. And this is exactly what happened. While the model became very good at predicting outcomes for the *training* data set, it failed to generalize that accuracy to the validation set.

As the model became over-fit, the gap widened between the model's accuracy on the training data and its accuracy on the validation data. By the end, there was a nearly-23-percentage-point difference between the model's training accuracy and its validation accuracy. This may be seen as the gap between the green line and the red line in Figure 1.

Even more interesting is the validation loss curve, shown in orange on the plot. Initially, as the model trained, the validation loss decreased, as expected. But as the model focused on memorizing the training data, the validation loss sudenly skyrocketed, reaching values of over 4.

Overall, the results show that the model has sufficient capacity to over-fit (fully learn) the training data.

Challenges and Adjustments

Discovering Model Complexity

Experimenting with model architectures was a challenge. At first, smaller models (e.g., 128 neurons in the first layer) plateaued around 92% accuracy. Those smaller models simply did not have the capacity to fully learn the dataset. Gradually increasing the number of neurons and layers resolved this issue.

Conclusion

Phase 2 successfully achieved the goal of overfitting the dataset, providing a solid foundation for building a generalizable model in future phases. By experimenting with increasingly complex architectures, the model achieved 97% accuracy, showing it was able to memorize the dataset.