**Critical Thinking 2**

Viet Ngo

Colorado State University Global

CSC580: Applying Machine Learning and Neural Networks - Capstone

Isaac Gang

09/22/2024

**Critical Thinking #2**

After training with a batch size of 100 and an epoch size of 20, the accuracy of the TensorFlow1 model is 0.9487. Interestingly, using the same testing hyperparameters, the accuracy of the TensorFlow2 model is 1.00000. One possible explanation for this difference is that in the TensorFlow 2 code, the final layer uses softmax directly, which ensures that the model's output is a proper probability distribution. In the TensorFlow 1 code, the softmax\_cross\_entropy\_with\_logits\_v2 function applies the softmax internally, but this is not directly reflected in the model's output. This could result in slightly different numerical behavior during backpropagation.

Figure 1 – Training result of TensorFlow1 model.



Figure 2 – Training result of TensorFlow2 model.



Figure 3 – Misclassification results of TensorFlow1 model.

Text

Description automatically generated

If the size of the hidden layer is doubled, from 512 to 1024 neurons, the accuracy of the model is slightly better. With more neurons in the hidden layer, the model has more capacity to learn complex patterns from the data. However, if the model becomes too complex for the size of the dataset, there will be a plateau or decrease in test accuracy due to overfitting. On the other hand, if the size of the hidden layer is halved, the accuracy of the model is slightly worse.

Figure 4 – Training result of TensorFlow1 model with 1024 neurons in the hidden layer.

Text

Description automatically generated

Figure 5 – Training result of TensorFlow1 model with 256 neurons in the hidden layer.

Text

Description automatically generated

A learning rate of 1.0 yielded the best accuracy, which suggests the model benefited from faster convergence in this case, likely without too much instability. A learning rate of 0.1 also provided high accuracy, which shows that slightly lower rates could still converge well, offering a good trade-off between convergence speed and stability. The lower learning rate of 0.01 resulted in slower convergence and a tendency toward underfitting, while the mid-range rate of 0.5 worked fairly well but was slightly less optimal due to some potential overshooting. This demonstrates that in this model, faster learning rates tend to work better, possibly due to the simplicity of the task and model architecture.

Figure 6 – Table of model accuracy with different learning rates

|  |  |
| --- | --- |
| **Learning Rate** | **Model Accuracy** |
| 0.01 | 0.9274 |
| 0.1 | 0.9508 |
| 0.5 | 0.9487 |
| 1.0 | 0.9520 |

With two hidden layers and a learning rate of 1.0, there is a significant decline in the model accuracy, dropping to 0.098. The model is likely diverging or not learning properly. With a high learning rate, weight updates are large. In more complex networks, like the one with 2 hidden layers, these large updates can easily lead to overshooting the optimal weights, preventing the model from converging properly. The weights might keep oscillating or get stuck in a suboptimal region.

After implementing a grid search to test all the different combinations of 1 or 2 hidden layers, learning rates of 0.01, 0.1, 0.5, 1.0, and batch sizes of 50, 100, and 200, the best accuracy achieved with this multi-layer perceptron is 0.9597, with 1 hidden layer, a learning rate of 1.0, and a batch size of 50.

Figure 7 – Model accuracy results with grid search implementation.

Text

Description automatically generated