**University of Central Missouri**

**Department of Computer Science & Cybersecurity**

**CS5720 Neural Networks and Deep Learning**

**Summer 2025**

**Home Assignment 1.**

**Student name: Nithish Reddy Thakkiti**

**Submission Requirements:**

* Once finished your assignment push your source code to your repo (GitHub) and explain the work through the ReadMe file properly. Make sure you add your student info in the ReadMe file.
* Submit your GitHub link and video on BrightSpace.
* Comment your code appropriately ***IMPORTANT.***
* Make a simple video about 2 to 3 minutes which includes demonstration of your home assignment and explanation of code snippets.
* Any submission after provided deadline is considered as a late submission.

1. **Tensor Manipulations & Reshaping**

**Task: Tensor Reshaping & Operations**

1. Create a random tensor of shape (4, 6).
2. Find its rank and shape using TensorFlow functions.
3. Reshape it into (2, 3, 4) and transpose it to (3, 2, 4).
4. Broadcast a smaller tensor (1, 4) to match the larger tensor and add them.
5. Explain how broadcasting works in TensorFlow.

**Expected Output:**

* Print rank and shape of the tensor before and after reshaping/transposing.

1. **Loss Functions & Hyperparameter Tuning**

**Task: Implement and Compare Loss Functions**

1. Define true values (y\_true) and model predictions (y\_pred).
2. Compute Mean Squared Error (MSE) and Categorical Cross-Entropy (CCE) losses.
3. Modify predictions slightly and check how loss values change.
4. Plot loss function values using Matplotlib.

**Expected Output:**

* Loss values printed for different predictions.
* Bar chart comparing MSE and Cross-Entropy Loss.

1. **Train a Neural Network and Log to TensorBoard**

**Task Description**

1. Load the MNIST dataset and preprocess it.
2. Train a simple neural network model and enable TensorBoard logging.
3. Launch TensorBoard and analyze loss and accuracy trends.

**Expected Outcome:**

* The model should train **for 5 epochs** and store logs in the "logs/fit/" directory.
* You should be able to visualize **training vs. validation accuracy and loss** in TensorBoard.

**4.1 Questions to Answer:**

1. What patterns do you observe in the **training and validation accuracy curves**?

**Ans.**

Early Epochs: Both training and validation accuracy rise steadily.

Later Epochs: Training accuracy often continues improving.

Validation accuracy may plateau or decline, indicating possible overfitting.

Loss curves typically mirror accuracy: training loss drops steadily; validation loss may bottom out and then rise.

1. How can you use **TensorBoard to detect overfitting**?

**Ans.**

Use TensorBoard to identify overfitting by observing:

* Divergence between training and validation accuracy curves — training improves while validation stagnates or worsens.
* Validation loss increases after reaching a minimum, even as training loss decreases.

These signs suggest the model is memorizing training data rather than generalizing well.

1. What happens when you increase the number of epochs?

**Ans**.

Initially: More epochs help the model learn better representations, improving accuracy.

Eventually:Training accuracy keeps rising.

Validation accuracy may decline due to overfitting.

Model becomes less generalizable.