CSE473s – Machine Learning Systems

Final Project Report

Milestone 1 & Milestone 2

Milestone 1 – Foundational Neural Network Library

Milestone 1 focused on designing and validating a custom neural network library implemented purely using NumPy.

The goal was to ensure mathematical correctness, modular design, and functional learning capability before

extending the system to deeper architectures.

Library Architecture

The library consists of modular components including Dense layers, activation functions (ReLU, Sigmoid, Tanh, Softmax),

a Mean Squared Error loss, a Stochastic Gradient Descent optimizer, and a Sequential model container.

All numerical operations rely exclusively on NumPy.

Backpropagation Design and Gradient Checking

Backpropagation was implemented using vectorized matrix calculus.

Dense layer gradients were derived for weights, biases, and inputs.

Numerical gradient checking using finite differences verified analytical gradients with relative errors on the order of 1e-11,

confirming correctness.

XOR Validation

A 2–4–1 multilayer perceptron was trained on the XOR problem using the custom library.

The model converged to a final MSE of approximately 6.56e-4 and correctly classified all four XOR input patterns,

demonstrating functional end-to-end learning.

Milestone 2 – Autoencoder and Latent Space Analysis

Milestone 2 extended the custom library to train an autoencoder on the MNIST dataset.

This milestone evaluated reconstruction quality, latent representations, and downstream classification performance.

Autoencoder Architecture and Training

The autoencoder consisted of a 784–128–32 encoder and a symmetric decoder.

Training used MSE loss and SGD optimization.

The reconstruction loss decreased steadily, producing recognizable digit reconstructions.

Latent Space Classification

Latent vectors extracted from the encoder were used to train an SVM classifier.

The classifier achieved approximately 97% accuracy on the MNIST test set,

indicating that the learned latent space was discriminative.

Comparison with TensorFlow/Keras

An equivalent autoencoder implemented in TensorFlow/Keras trained faster and achieved lower reconstruction loss.

However, the custom NumPy-based implementation produced competitive latent representations,

validating the correctness of the custom library.

Final Conclusion

Together, Milestones 1 and 2 demonstrate a complete, mathematically correct neural network framework built from scratch.

The system successfully supports both supervised and unsupervised learning and provides a strong foundation

for further experimentation and analysis.