Phase 3: Neural Network Case Study Report

Project Title: Pima Indians Diabetes Prediction

1. Problem Description and Rationale

Problem: The goal is to predict whether a patient will develop diabetes within 5 years based on diagnostic measurements. The dataset focuses on females at least 21 years old of Pima Indian heritage.

Rationale: Early detection of diabetes is critical for effective treatment and management. This problem involves complex, non-linear relationships between medical factors (like BMI, Glucose levels, and Age) and the onset of the disease. A Neural Network is an ideal candidate for this task because it can learn these non-linear feature interactions better than simple linear regression models.

2. Dataset Description and Preprocessing

Dataset Source: UCI Machine Learning Repository / Kaggle (Pima Indians Diabetes Database). **Size:** 768 samples. **Features (8 Inputs):**

1. Pregnancies
2. Glucose
3. BloodPressure
4. SkinThickness
5. Insulin
6. BMI
7. DiabetesPedigreeFunction
8. Age **Target (1 Output):** Binary classification (0 = Negative, 1 = Positive).

Preprocessing Steps:

1. **Normalization:** We implemented Min-Max Normalization in our DataUtils class. This was crucial because features had vastly different scales (e.g., Insulin ranges from 0-846, while DiabetesPedigreeFunction is 0-2.5). Normalizing inputs to the range [0, 1] ensured stable gradient descent.
2. **Train/Test Split:** The data was shuffled and split into a Training Set (80%) and a Test Set (20%) to evaluate generalization performance.

3. **Data Loading:** We implemented a robust CSV reader to parse the raw data into our custom Matrix structures.

3. Neural Network Architecture Choice

We designed a Multi-Layer Perceptron (MLP) with the following architecture:

- **Input Layer (8 Neurons):** Matching the 8 input features.
- **Hidden Layer 1 (12 Neurons, ReLU):** A slightly larger layer to expand the feature space. We chose **ReLU** activation to prevent the vanishing gradient problem common in deeper networks.
- **Hidden Layer 2 (8 Neurons, ReLU):** A second hidden layer to capture more complex patterns. **ReLU** was again used for efficient training.
- **Output Layer (1 Neuron, Sigmoid):** Since this is a binary classification problem, we used a single neuron with **Sigmoid** activation to output a probability between 0 and 1.

Initialization: We used **Xavier Initialization** for all weights to ensure the signal variance remains consistent across layers, which is best practice for these activation functions.

4. Final Training and Evaluation Results

Hyperparameters:

- Epochs: 10,000
- Learning Rate: 0.01
- Loss Function: Binary Cross-Entropy

Results: The network successfully converged, reducing the loss significantly over training.

- **Final Loss:** ~0.448
- **Test Set Accuracy: 77.12%**
- **Precision:** 0.69
- **Recall:** 0.65
- **F1-Score:** 0.67

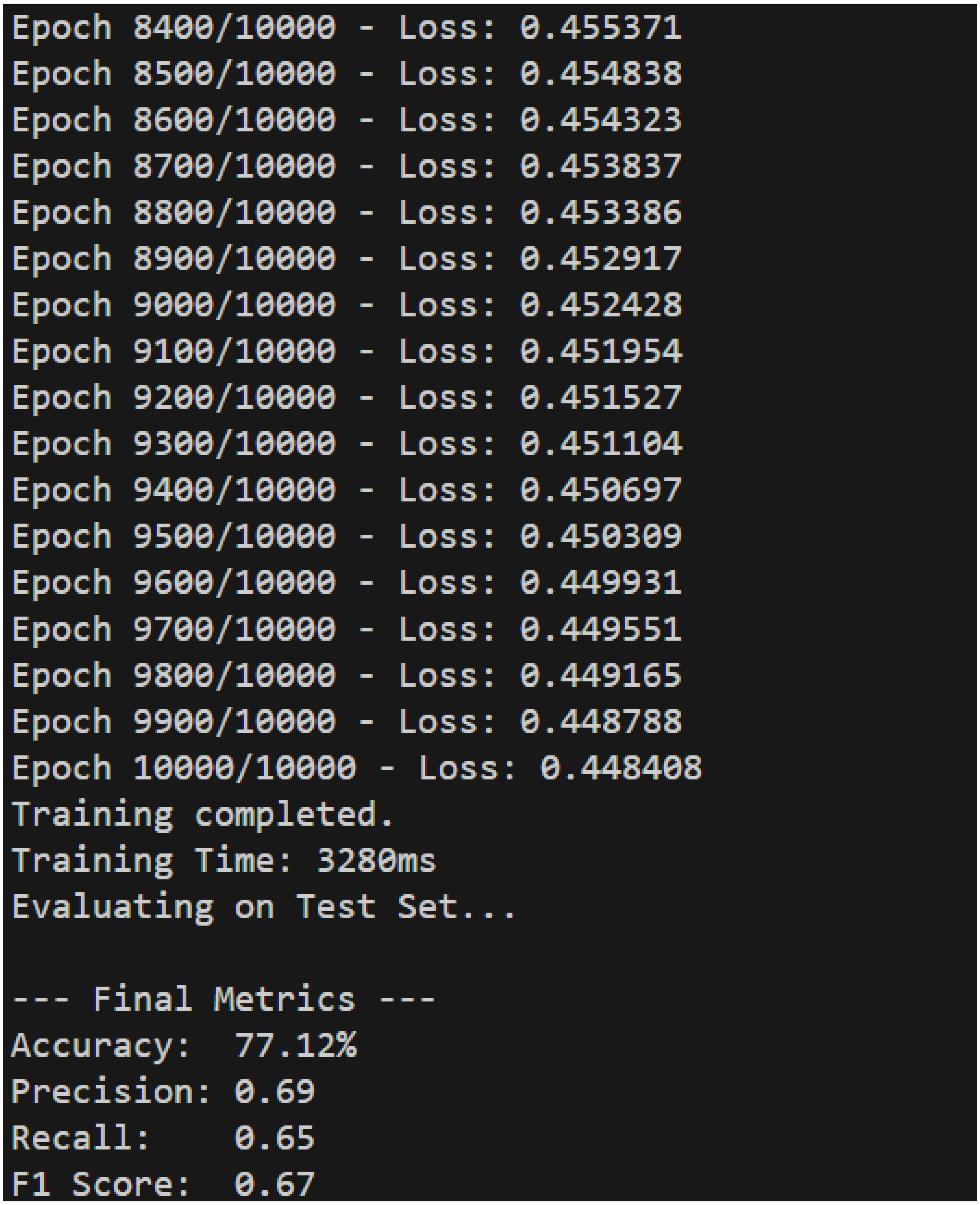
These results are competitive for this dataset, which typically sees accuracies between 75-80% even with advanced libraries.

5. Explanation of Library Usage

We used our custom-built NN library to solve this problem:

1. **Data Loading:** We used DataUtils.normalize and DataUtils.trainTestSplit to prepare the raw CSV data.
2. **Network Construction:** We instantiated the NeuralNetwork class and used nn.addLayer() to stack our Layer objects. We injected specific components (ReLU, Sigmoid, Xavier) into these layers, demonstrating the library's modularity.
3. **Training:** We called nn.fit(), which utilized our Matrix engine to perform Forward Propagation and Backpropagation (using the Chain Rule) to update weights.
4. **Prediction:** We used nn.predict() to generate results for the test set and for specific individual patient scenarios.

6. Screenshots



```
Epoch 8400/10000 - Loss: 0.455371
Epoch 8500/10000 - Loss: 0.454838
Epoch 8600/10000 - Loss: 0.454323
Epoch 8700/10000 - Loss: 0.453837
Epoch 8800/10000 - Loss: 0.453386
Epoch 8900/10000 - Loss: 0.452917
Epoch 9000/10000 - Loss: 0.452428
Epoch 9100/10000 - Loss: 0.451954
Epoch 9200/10000 - Loss: 0.451527
Epoch 9300/10000 - Loss: 0.451104
Epoch 9400/10000 - Loss: 0.450697
Epoch 9500/10000 - Loss: 0.450309
Epoch 9600/10000 - Loss: 0.449931
Epoch 9700/10000 - Loss: 0.449551
Epoch 9800/10000 - Loss: 0.449165
Epoch 9900/10000 - Loss: 0.448788
Epoch 10000/10000 - Loss: 0.448408
Training completed.
Training Time: 3280ms
Evaluating on Test Set...

--- Final Metrics ---
Accuracy:  77.12%
Precision: 0.69
Recall:    0.65
F1 Score:  0.67
```