DS5007 Deep Learning Lab 1 - Neural Network Implementation and Optimization

Date: 17/01/2025
Max Marks: 10
Deadline: 24/01/2025, 12:00 PM

Instructions

1. Provide well-commented, indented code with meaningful variable names.
2. Write the task description in separate text blocks before the corresponding code block.
3. Carefully follow the task requirements and use only the specified libraries or approaches.
4. Ensure all plots have appropriate axis labels, titles, and legends.
5. Submit a single Jupyter Notebook (.ipynb) file named YourName_YourRollNo_Assignment1.ipynb.

Tasks

Task 1: Backpropagation Implementation for Binary Classification (6 Marks)

1. **Dataset Creation (1 Marks)**
   - Generate a synthetic dataset for binary classification using the `make_blobs` function from the `scikit-learn` library.
   - The dataset should have 1000 samples, 2 features, and their respective binary labels.
   - Split the dataset into train and test sets.
   - Plot the training data using `matplotlib`.
2. **Implement Backpropagation (4 Mark)**
   - Create a small neural network from scratch with the following specifications:
     - Input layer with 2 neurons.
     - One hidden layer with 4 neurons using the sigmoid activation function.
     - Output layer with 1 neuron using the sigmoid activation function.
   - Implement forward and backward propagation manually to train the network.
   - Update the weights using gradient descent.
   - Evaluate the model on the test set and print its accuracy.
3. **Decision Boundary Visualization (1 Marks)**
   - Plot the decision boundary of the trained model using the training dataset.

Task 2: Multiclass Classification Using TensorFlow (4 Marks)

Dataset Preparation

- Use the MNIST dataset for this task (refer to TensorFlow documentation for loading the dataset).
- Split the dataset into train, validation, and test sets.

1. **Model Creation and Training (2 Marks)**
   - Create a neural network using TensorFlow for multiclass classification. The model should include:
     - Input layer matching the dataset features.
     - At least two hidden layers with a customizable number of neurons.
     - An output layer with softmax activation (the number of neurons should equal the number of classes).
   - Experiment with the following:
     - Regularization techniques: L1, L2, and Dropout.
     - Optimizers: SGD, Adam, and RMSprop.
     - With and without Batch Normalization.
2. **Model Evaluation and Hyperparameter Optimization (2 Mark)**
   - Train the model and evaluate it on the test set using accuracy, loss, and a confusion matrix.
   - Identify the optimal combination of:
     - Regularization technique.
     - Optimizer.
     - Batch Normalization usage.
   - Justify your findings based on the results obtained.

- Implement strategies for hyperparameter optimization using tools such as a custom implementation using a for loop or `Keras Tuner` or `Optuna`.