Assignment 2: Implementing Neural Networks

for AIDI1009-24F-10827: Neural Networks

—

Due Date: **19th July 2025 @11:55 PM**

For this assignment, students will implement a neural network to classify handwritten digits using Python and Pytroch. The task is a **multiclass classification problem** where students need to train a model to distinguish between digits (0-9) using the **MNIST dataset**.

Objective

The goal is to achieve high classification accuracy by implementing, training, and tuning a neural network. This assignment will test the student's understanding of key neural network concepts, including layers, activation functions, loss functions, and model evaluation.

Dataset

The MNIST dataset contains 60,000 training images and 10,000 test images of handwritten digits from 0 to 9. Each image is 28x28 pixels in grayscale. Use the following code to load the dataset:

1 Tasks

1.1 Dataset preprocessing

1. **Normalziation:** Scale pixel values from [0, 255] to [0, 1].
2. **Reshape:** Convert each image from a 2D array of 28x28 pixels to a 1D array of 784 pixels.

```
# Normalize and reshape data
 train_images = torch.tensor(train_images.reshape(60000, 28 * 28), dtype=torch.
 float32) / 255
 test_images = torch.tensor(test_images.reshape(10000, 28 * 28), dtype=torch.float32)
  / 255
```

1.2 Model Architecture

Build a **feedforward neural network** with the following structure:

1. **Input Layer:** Accepts 784 features (28x28).
2. **Hidden Layers:** Experiment with 1-2 hidden layers, each with 128 or 256 neurons.
3. **Activation Function:** Use ReLU activation for the hidden layers.
4. **Output Layer:** Use a softmax layer with 10 units (one for each digit class).

```
import torch
import torch.nn as nn

class SimpleNN(nn.Module):
    def __init__(self):
        super(SimpleNN, self).__init__()
        self.model = nn.Sequential(
            nn.Linear(784, 128),
            nn.ReLU(),
            nn.Linear(128, 64),
```

```
            nn.ReLU(),
            nn.Linear(64, 10),
            nn.Softmax(dim=1)
        )

    def forward(self, x):
        return self.model(x)

model = SimpleNN()
```

1.3 Model Compilation

1. **Loss function:** Use **sparse categorical cross-entropy** as the loss function since this is a multiclass classification problem with integer labels.
2. **Optimizer:** Use **Adam optimizer** for training.
3. **accuracy:** Monitor the model's **accuracy** metric.

1.4 Model Training

1. Train the model on the training data.
2. Set epochs to 10-15 and use a batch size of 32.
3. Use 20% of the training data as a validation set.

1.5 Evaluation

1. • Evaluate the model on the test dataset and report the accuracy.

1.6 Analysis and Report

1. Plot the training and validation accuracy and loss over epochs to analyze any **overfitting** or **underfitting**.
2. Discuss how the number of neurons and hidden layers affects performance.
3. Describe any improvements attempted, such as adding dropout layers or batch normalization.

2 Report Format

Name your report AIDI1009-24F-10827: Assignment#2 LastNameFirstName.pdf. Below is the general format of the report required:

1. The front page (i.e. title page) should contain only the following:
   - Course #, Course Name and Date
   - Your Name and ID
   - Assignment # and title of the assignment.
2. Introduce the problem to be solved:
   - Problem Statement
     (a) Briefly describe the problem solved in the assignment.
     (b) Assumptions and Constraints
     (c) Constraints could be for example using certain libraries, datasets, or a specific programming language.
3. Answer all questions posed in Section 3. Append the following to your answers:

   (a) Plots and Graphs.
   (b) Tables.
   (c) Figures.

4. Python Code
   (a) Python code for the problem
   (b) Document your code.