First Assignment

Building a neural network model for classification problem

8 April, 2025

Project Goal:

This project aims to build a neural network model to solve the EMNIST classification problem. As deep neural networks are widely used in various domains, such as computer vision, natural language processing and medical image analysis, how to create a proper neural network to implement the specific task has become very important.

Dataset Introduction:

In the project, we use the EMNIST[1] (Extended MNIST) dataset, which is a set of handwritten character digits derived from the NIST Special Database 19 and converted to a 28x28 pixel image format and dataset structure that directly matches the MNIST dataset.

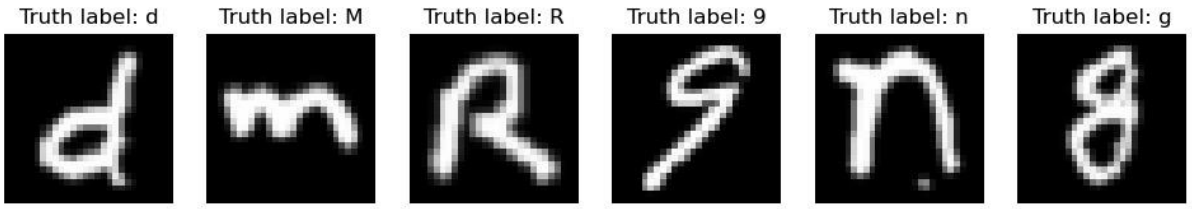
```
- EMNIST ByClass:  814,255 characters, 62 unbalanced classes.
- EMNIST ByMerge:  814,255 characters, 47 unbalanced classes.
- EMNIST Balanced: 131,600 characters, 47 balanced classes.
- EMNIST Letters:  145,600 characters, 26 balanced classes.
- EMNIST Digits:   280,000 characters, 10 balanced classes.
- EMNIST MNIST:    70,000 characters,  10 balanced classes.
```

Due to the EMNIST dataset includes 6 different splits, we select “Balanced” dataset, which had addressed the balance issues in the “ByClass” and “ByMerge” datasets. It is derived from the “ByMerge” dataset to reduce mis-classification errors due to capital and lower-case letters and also has an equal number of samples per class. The “Balanced” dataset information is as follows:

- Train: 112,800
- Test: 18,800
- Total: 131,600
- Classes: 47 (balanced)

If we visualize the EMNIST images, they look like as follows:

[1] https://www.kaggle.com/datasets/crawford/emnist



Project Introduction:

Write the python code to build various neural networks and compare the performance of different network models.

In this project, you need implement three types of neural networks

1) A Multilayer Perceptron (MLP) with at least three hidden layers (i.e., neural networks with only fully-connected layers);
2) A Convolutional Neural Networks (CNNs) with at least two convolutional layers.
3) A ResNet network (Extend the CNN you designed by incorporating residual connections)

For each network model, you need consider to multiple parameters to obtain the best performance. You can use various techniques (you have learnt) to overcome overfitting, underfitting, unstable gradient, etc. You need to utilize and explore the following techniques to train your neural network models:

1) Adaptive Learning Rate (e.g., learning rate schedulers) (explore at least two learning rate scheduling methods)
2) Activation function (ReLU, Leaky ReLU, ELU, etc.) (explore at least three activation functions)
3) Optimizers (SGD, ADAM, RMSprop, ASGD, AdaGrad, etc.) (exlore at least three optimizers)
4) Batch Normalization (explore two options: with Batch normalization, without Batch normalization)
5) L1 & L2 regularization (explore three options: without L1&L2 regularization, with L1 regularization, with L2 regularization)
6) Dropout (explore two options: with or without Dropout)

***Note**: Below is one strategy you can try to effectively explore the combinations of the above techniques:*

- *Initially, establish a baseline model configuration utilizing your existing knowledge of deep learning. For instance, you might select ReLU, ADAM, Batch Normalization, L2 Regularization, and Dropout, etc., as the initial setup of your model.*
- *Following this, investigate each optimization technique separately, while preserving the most effective settings from previous explorations. For exmaple,*
    - *Explore learning rate scheduling methods and identify the most suitable one; Incorporate this method into your model*
    - *Explore activation functions and select the most appropriate one; Use this activation function in your model*
    - *Explore various optimizers and choose the optimal one for your model*
    - *Continue this process to explore additional techniques*
- *Upon completing these explorations, your resulting model should demonstrate superior performance compared to the initial baseline configuration.*

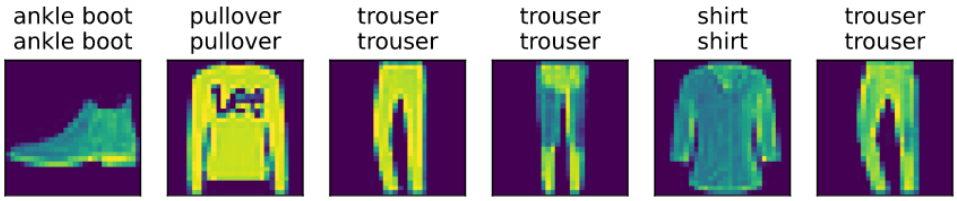
This is one possible strategy, of course, you can use other feasible strategies, as long as you can demonstrate your ability to evaluate and investigate various techniques in order to determine the most appropriate deep learning model for a specific task. The assessment of this critical skill is one of the key goals of this assignment.

During your training MLP, **CNN** and **Resnet**, you also need to decide the below hyperparameters that are most suitable for your MLP, **CNN** and **Resnet** models, for example,

- Number of Hidden layers
- Number of hidden neurons for each hidden layer
- …

When building the neural networks, you can use any libraries (pandas, NumPy, matplotlib …) and frameworks (PyTorch, Tensorflow, …). However, your code must include the following steps:

1. Indicate the imported packages/libraries
2. Load the dataset (you can import PyTorch dataset library to download the dataset)
3. Split data into a training dataset and a testing dataset (e.g., using DataLoader in PyTorch)
4. Understand the dataset and visualize the dataset:
    a. *Print-out the number of training/testing samples in the dataset.*
    b. *Plot some figures to visualize some samples as above figures (we will provide the mapping .txt file for "Balanced" dataset, if you can implement mapping from the index to the classes by yourself, you can ignore the .txt file).*
5. For the three models - **MLP**, **CNN** and **Resnets**, please complete the below steps respectively on **the training dataset**
    a. *For the techniques/hyperparameters you choose to explore, you can use cross validation to find the most suitable combination (of those techniques/hyperparameters) under the* ***evaluation metric of accuracy****.*
    b. *You can use Data Augmentation techniques to improve model performance (e.g., random rotation or adversarial noise injection, etc.)*
    c. *After you obtain the best version of the model (i.e., the model with the best techniques/hyperparameters combination), please:*
        i. *Plot the loss function graph respect to the iteration/epoch*
        ii. *Plot the accuracy graph respect to the iteration/epoch*
        iii. *You can use either CPU or GPU to train your model and please print the training time and memory usage*
6. After you obtained the best version of MLP, **CNN** and **Resnets**, please test both models on the testing dataset
    a. *Compare the performance of* MLP, **CNN** and **Resnets**
        i. *Load your model and print the prediction of the top six samples in testing dataset, compare them with the true labels, which will be similar to the below (example from Fashionmnist).*



ii. *Plot the confusion matrix of three models – MLP, CNN and Resnet*
iii. *Summarize the performance of three models using accuracy, please also report the precision, recall and F1 score of MLP, CNN and ResNets*
iv. *Analyze the impact of residual structure on CNNs*

b. *Explain the impact of the model's feature maps on its output predictions (e.g., using Grad-CAM to visualize the attention mechanisms)*

Submission

- Deadline: **10 PM, 14 May, 2025**
- Submit onhttps://www.bb.ustc.edu.cn/webapps/assignment/uploadAssignment?content_id=_161725_1&course_id=_28457_1: Submit two files that include a report (in **.pdf** format) and the Python script (in **.ipynb** format); please name your files in the format of: "Group_组号" (e.g., Group_01.pdf and Group_01.ipynb)
- **Every student should submit two files, although the students from the same group can submit the same files.**

Assessment

The project assessment contains two parts:

1. Python code: ***(50%)***
   You need to provide your PYTHON CODE and running results in *.ipynb* script, which includes all the steps in **Section Project Introduction**, and add comments to describe/explain each line of your python code
   a. Step 1 - Step 4 (5%)
   b. Step 5 - MLP (10%)
   c. Step 5 - CNNs (10%)
   d. Step 5 - ResNet (10%)
   e. Step 6 (15%)

2. Report: ***(50%)***

   A short report of a **maximum of five pages** (not considering the cover page and references, in case you want to include them). This report should contain:

   a. A Section explaining the project **(10%),** including (but not limited to):

      i. Introduction of the dataset

ii. Describe the structure of your MLP, CNN and Resnets, and discuss how you build up them.

b. A Section explaining the rationale of your design on MLP, CNN and Resnets (**25%**), including (but not limited to):

i. What kind of hyperparameters or techniques you choose to tune and explore

ii. Why you choose those hyperparameters or techniques

iii. For each technique, please describe how it affects your model's performance and analyze the reasons why the technique can boost or decrease the performance of your models

iv. Do you ever have overfitting or underfitting issue while training the models? How do you deal with overfitting or underfitting issue?

c. A section describing and explaining your results **(10%),** including (but not limited to):

i. Describe and explain *the training loss, testing loss and testing accuracy* of MLP, CNN and Resnets.

ii. Describe and explain *the predicted results of your models.*

iii. Your analysis on the performance of MLP, CNN and Resnets based on your own AI knowledge. You need to discuss the pros and cons of MLP, CNN and Resnets and analyse why a certain model outperformances another model.

d. The final conclusions **(5%),** including (but not limited to):

i. Your own reflection on this project, for example, what you have learned via this project, which part you could do better next time, etc.