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 achieve a specific task has become very important.

Dataset Introduction:

In the project, we use the EMNIST (Extended MNIST) dataset (https://www.kaggle.com/datasets/crawford/emnist

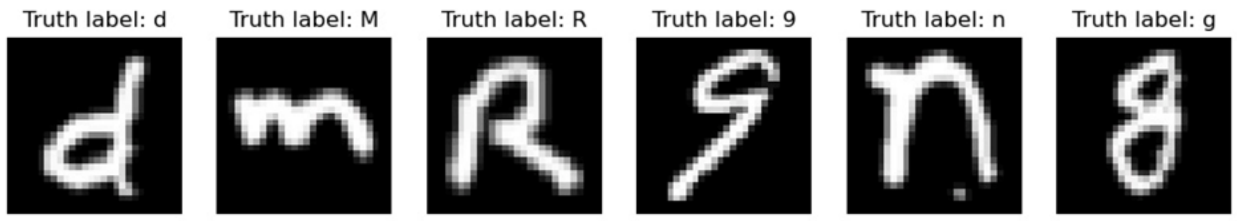
Links to an external site.

), 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.

Due to the EMNIST dataset including 6 different splits, we selected the “Balanced” dataset, which addressed the balance issues in the “ByClass” and “ByMerge” datasets. It is derived from the “ByMerge” dataset to reduce misclassification 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 as follows:



Project Introduction:

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

In this project, you need to implement two types of neural networks; one is the multilayer perceptron (MLP) networks with at least three hidden layers (i.e., neural networks with only fully-connected layers); the other is the Convolutional Neural Networks (CNNs) with at least two convolutional layers. For each network model, you need to 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:

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

During your training MLP and CNNs, you also need to decide the below hyperparameters that are most suitable for your MLP and CNN 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
   I. Print out the number of training/testing samples in the dataset.
   II. Plot some figures to visualize some samples in the dataset (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).

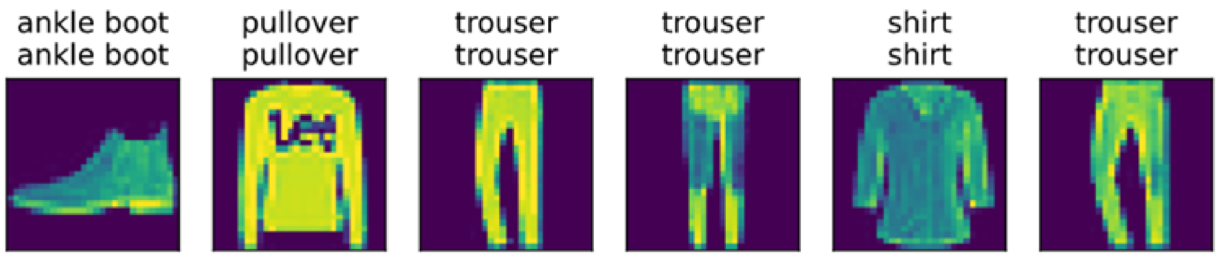
5.For the two models - MLP and CNNs, please complete the below steps, respectively, on the training dataset

III. 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.

IV. After you obtain the best version of the model (i.e., the model with the best techniques/hyperparameters combination), please:
   - Plot the loss function graph with respect to the iteration/epoch
   - Plot the accuracy graph with respect to the iteration/epoch
   - You can use either CPU or GPU to train your model, and please print the training time

6.After you obtain the best version of MLP and CNNs, please test both models on the testing dataset

- **I.** Compare the performance of MLP and CNNs
- **II.** Load your model and print the prediction of the top six samples in the testing dataset, and compare them with the true labels, which will be similar to the below (example from Fashionmnist).



- **III.** Plot the confusion matrix of two models – MLP and CNNs
- **IV.** Summarize the performance of two models using accuracy; please also report the precision, recall and F1 score of MLP and CNNs