# Homework 2: Convolutional Neural Network

**Deadline: 9-30-2023**

***GPUs may be needed for speeding up this training process. You are not allowed to use pre-defined ConvNet and ResNet18 (You will build the model architecture layer by layer).**

## Description

In this homework you will practice how to write Convolutional Neural Network (CNN) in Python with TensorFlow or PyTorch. You need to understand how CNN works, including backpropagation and gradient descent in order to implement this homework successfully. The goals of this homework are:

- To understand the steps to train/test the classifier for image classification.
- Understand architecture of CNN and how to connect each layer together by using TensorFlow or PyTorch.

## Data preview:

The goal of this project/challenge is to predict handwrite digital numbers between 0-9**.**

The dataset description can be found: https://paperswithcode.com/dataset/mnist

In the past, the best model achieved **99% accuracy**. Your best model should achieve at least 85% accuracy.

## Step 1: Data

Before starting to train a model, please get familiar with the dataset. When you look at the dataset, please answer the following questions: 1) How many data samples are included in the dataset? 2) Which problem will this dataset try to address? 3) What is the minimum value and the maximum value in the dataset? 4) What is the dimension of each data sample? 5) Does the dataset have any missing information? E.g., missing features. 6) What is the label of this dataset? 7) How many percent of data will you use for training, validation and testing? 8) What kind of data pre-processing will you use for your training dataset?

*Hint: You should use the same test dataset to compare the models' performance.*

## Step 2: Model

Here I selected DNN, ConvNet, and ResNet as model. However, you will experience different hyperparameters. Please change the following hyperparameters and report the model performance in the testing dataset.

| Model | Accuracy |
|---|---|
| DNN | |
| ConvNet | |
| ResNet | |

| Any Other? | |
|---|---|
| | |
| | |
| | |

**Step 3: Objective**

Cross-entropy is the loss function you will use to train your models.

$$CE = -\sum_{i=i}^{n} y_i \log(\hat{y}_i)$$

where i is i'th class.

**Step 4: Optimization**

You are going to select your optimization function to train the models. Report the optimization you selected in this section and explain the reason for using the optimization.

**Step 5: Model selection**

Based on your training experience, which model gives the best performance (accuracy). Have you experienced different learning rates?

| Model | LR: 0.1 | LR: 0.01 | LR: 0.001 | LR: 0.0001 |
|---|---|---|---|---|
| DNN | | | | |
| ConvNet | | | | |
| ResNet | | | | |
| Any Other? | | | | |
| | | | | |
| | | | | |

Any other learning rate you would like to try? Have your tried other learning rate technique? How do you avoid overfit your model and underfit your model?

**Step 6: Model performance**

In this step you should report your model performance, which you did in the previous steps.

A. Report the accuracy and F1 for models you tried.
B. The model training plots (training and validation loss, accuracy, and F1).
C. Please add the AUC plot of test dataset in this step.

Hint: ROC AUC documentation:
https://scikit-learn.org/stable/auto_examples/model_selection/plot_roc.html

F1 for multi-class:
https://www.baeldung.com/cs/multi-class-f1-score

**What should you submit?**

You should submit a zip file (named as firstname_lastname_hw2.zip – e.g. jun_bai_hw2.zip) containing:

1. Your homework report from step 1- 6. You will answer all the questions in each step and fill the tables in step 2, step 5 and performance plot in step 6. Missing any part will lose some points. Please double check you have addressed all the questions.

2. Your code of all models. Each model should include a README file explaining how to run the model. Your code should be well commented. In your code, you should have a function called *test_model* and *feature_visualization* functions. The *test_model* function will load the trained model and load test dataset to predict. The *feature_visualization* will plot the first layer and second layer features for CNN based models.

3. Your highest performed DNN ConvNet, and ResNet model weights.

4. A folder contains screenshots of iteration of models' training and testing.