Homework 1: Neural Network Classifier

***GPUs are <u>not</u> necessary for speeding up the neural network training process in this homework.**

Description

In this homework you will practice to write a Neural Network classifier in Python using the ***PyTorch*** framework. You need to understand how a Neural Network classifier works, including back propagation and gradient descent in order to implement this homework successfully. The goal of this homework is:

- To implement and understand a Neural Network classifier.
- Get familiar with using pytorch.

Instruction

- The dataset used in this homework is **CIFAR-10**. You may need these packages: pytorch, torchvision, (for the CIFAR-10 dataset), NumPy, and OpenCV (for reading images). The commonly used classifier is Softmax.
- You are strongly recommended to use ***<u>PyTorch</u>*** framework.
- ***<u>You can add as many layers as you want, however, NO convolutional layers are allowed</u>*****.** Optimization techniques such as mini-batch, batch normalization, dropout and regularization might be used.
- Note that in PyTorch, the softmax function is integrated into the Cross-Entropy loss function, which means you don't need to add softmax layer at the end of your neural network if you choose CE loss as your loss function.
- Requirements:
  1. Contain a **training function** that will be called to train a model with the command "**python classify.py train**".
  2. Save the model in a folder named "**model"** after finishing the training process.
  3. Show the **testing accuracy** in each iteration of the training function. The **test accuracy** should be greater than or equal to **<u>45%</u>** in the end using the CIFAR-10 dataset. **Note that the testing accuracy is not the training accuracy!**

Loop	Train Loss	Train Acc %	Test Loss	Test Acc %
1/10	0.3489	24.8214	0.2590	32.1796
2/10	0.2504	36.0774	0.2455	37.4011
3/10	0.2390	40.4464	0.2386	40.2591
4/10	0.2317	43.2693	0.2340	41.9007
5/10	0.2259	45.4767	0.2309	43.2852
6/10	0.2213	47.2195	0.2281	43.7994
7/10	0.2174	48.8059	0.2261	45.4015
8/10	0.2137	50.2170	0.2240	45.7278
9/10	0.2105	51.3459	0.2228	46.5783
10/10	0.2073	52.5711	0.2211	46.9244

Model saved in file: ./model/model.ckpt

4. Implement a **testing function** that accepts the command "**python classify.py test xxx.png"** to test your model by loading it from the folder "model" created in the training step. The function should read "xxx.png" and predict the output. The output might not match the true image type because this type of classifiers cannot achieve high accuracy.

```
prediction result: car
```

- **Some hints** to improve the accuracy:
  1. add more linear classifier layers (pay attention to the dimension of weights and biases);
  2. apply an activation function like ReLU between the layers;
  3. use a powerful optimizer like Adam;
  4. use drop out technique;
  5. apply dropout and regularization.

Submission

- You need to submit a **zip** file including:
  1. a python file named "**classify.py**";
  2. a generated model folder named "**model**";
  3. A report that includes your code and two screenshots of training and testing results.
- The "**classify.py**" file should be able to run with the following commands:
  1. **python classify.py train**
     to train your neural network classifier and generate a model in the model folder;
  2. **python classify.py test xxx.png**
     to predict the class of an image and display the prediction result.
- The **zip** file should be named using the following convention:
  <Last-Name>_<First-Name>_HW1.zip
  Ex: Bourne_Jason_HW1.zip
- Note:
  - Do **NOT** put any print function (for debugging) other than showing the results.
  - **Do NOT submit/zip dataset (CIFAR10) with your code. Delete the dataset folder from your zip file!!!**
  - Comment your code.

Grading criteria

- Your model will be tested by running "**python classify.py predict xxx.png"** with additional testing images. Please make sure your functions work correctly.
- The testing accuracy should be greater than or equal to **45%** in the end. There will be 1-point deduction for every 1% of accuracy degradation based on 45%.
- Upload the zip file to the eLC system before 11:59PM (EST Time) **02/24/2026.**
- The penalty for late submission is a 10% grade deduction for each day.