CSCI-567 Assignment 1

In this assignment, you'll gain familiarity with PyTorch as you implement a few image classifiers. You can find the code for the in-class MNIST demonstration here. You can use it as much or as little as you'd like. You'll get the best practice if you avoid relying on it and spend time with the PyTorch documentation. But the demo code is here to help you get unstuck.

Dataset Loading

We will use CIFAR-10 as our dataset. Start by loading the dataset (see the PyTorch documentation for guidance). Next, split the data into train, validation, and test splits. Normalize data to have mean 0 and standard deviation 1. This will improve training in the next step.

Training Your Classifiers

MLP

Just as demonstrated in the discussion, you can Implement a classification model for CIFAR-10 using MLP layers. You should be able to achieve 50% accuracy with three linear layers with ReLU as the activation function.

CNN

Try to use CNN layers in your classifier. You should be able to achieve 60% accuracy with three CNN layers.

ResNet

There is a widely used and powerful model architecture, ResNet. Fortunately, PyTorch has a built-in implementation of it. Check its document and try to use ResNet to train your classification model. You should be able to achieve 70% accuracy easily with ResNet-18.

ResNet is based on CNN and batch normalization layers. Batch normalization is implemented in PyTorch too (`BatchNorm2D`). Try to read the ResNet paper and implement ResNet by yourself. You can check the correctness of your implementation by comparing the performance of your implementation with the performance of the built-in one.

Data Augmentation

Data augmentation is a technique people usually use when training a model. The main idea is to perturb the input and use the perturbed data along with the original data to train the model. This is as if we have more data to train the model. The library torchvision implemented many commonly used perturbations that can be used for data augmentation. Check the document and try to use some of them when training your classifier. Using RandomHorizontalFlip and RandomCrop, you should be able to achieve 80% accuracy easily with ResNet-18.

Model Analysis

Take a look at some examples of model predictions. When the model is incorrect, is it incorrect in reasonable ways? Look into how the model misclassifies images using e.g., a confusion matrix.

Questions/Observations:

1. Do you observe that the accuracy of the training set is usually greater than the accuracy of the testing set?
2. Do you observe that the accuracy of the testing set is usually close to the accuracy of the validation set?
3. In your experiment, do you observe your model overfit? Do you observe that ResNet-18 does not overfit as easily even though ResNet-18 has many parameters?