CSC 487: Deep Learning

Homework 1: Classification with Tabular Data

In this homework you will develop different models to classify tree species based on data extracted from hyperspectral imagery. Hyperspectral imagery is captured using a special imager that densely samples the light spectrum; whereas a typical camera records three channels centered on red, green, and blue wavelengths, a hyperspectral imager splits the light spectrum into hundreds of channels, including visible and non-visible wavelengths! Different objects have unique "signatures" in the hyperspectral data which can be detected using machine learning.

Our data comes from this paper:

Fricker, G. A., Ventura, J. D., Wolf, J. A., North, M. P., Davis, F. W., & Franklin, J. (2019). A convolutional neural network classifier identifies tree species in mixed-conifer forest from hyperspectral imagery. *Remote Sensing*, *11*(19), 2326.

The hyperspectral imaging data was collected as part of the NSF NEON project and covers a strip of land in the San Joaquin Experimental range north of Fresno. The team walked the forest with a GPS unit and marked the locations and species of trees in the study area. We then extracted the hyperspectral data around each location in the imagery to form the dataset.

The provided notebook downloads and unpacks the data into features labels in a train/test split.

Instructions

1. **Inspect the data.** See the first discussion prompt below.
2. **Pre-process the data.** You will pre-process the features using Principal Components Analysis (PCA) to reduce the number of dimensions, making the learning process easier. Fit the PCA model to the training features and apply it to both the train and test features. Use 32 components and make sure to set whiten=True. You will use the result of PCA as the input to your classifiers, not the original raw hyperspectral data.
3. **Classifiers using scikit-learn.** Make a linear classifier and a neural network (NN) classifier using scikit-learn and calculate accuracy on the test set for each classifier. The N should have three layers and a hidden layer size of 100.

4. **Classifiers using PyTorch.** Now implement the linear classifier and NN using PyTorch. You will need to implement the following steps:
    a. Create a TensorDataset and DataLoader for the train and test splits. Use a batch size of 32. The train loader should use random shuffling but the test loader should not.
    b. Create a function to calculate model accuracy, given the model and data loader. The function should iterate through the batches in the loader, add up the number of correct predictions in each batch, divide the sum by the total number of predictions, and return the result.
    c. Create a function to train a model. Use the SGD optimizer with learning rate 1e-2 and weight decay of 0.001. Train for 100 epochs and use the cross entropy loss function. In each epoch, loop through the training data loader and do the following: zero out the gradients; calculate model outputs and loss; run the backward step to calculate gradients; and run the optimizer step. At the end of each epoch, calculate train and test accuracy and print out the current accuracy numbers.
    d. Use your functions to train both a linear classifier and a NN model.

Report

Your report should include the following:

- **Code explanation:**
    - Briefly explain your solution and any design choices you made that weren't specified in the instructions.
    - Clearly describe any external sources that were used (e.g. websites or AI tools) and how they were used.
- **Discussion:**
    - What is the shape and data type of each provided matrix? What are the rows and columns of the matrices? What are the ranges? How many classes are there and what are the classes (the answer is in the paper linked above). How many examples are provided of each class in the train and test splits?
    - Compare the two PyTorch models in terms of test performance and overfitting. Also compare your PyTorch results to your scikit-learn results.

Deliverables

Python notebook and report document due Sunday, Jan. 26, 11:59 pm.