CISC 867: Project 1

Part I: Data Preparation

1. In this project, you will use the Leaf Classification dataset using a neural network architecture.
   You can use keras/tensorflow or write your own routines.

1) First, download the *data* file, load it, and
    1. Describe the data
    2. Clean the data
    3. Check the data for missing values or duplicates and carry out proper correction methods
    4. Visualize the data using proper visualization methods.
    5. Draw some of the images
    6. Carry out required correlation analysis
2) divide the data into a training and test set using approximately 80% for training.
3) Decide if you need to standardize the data, by computing the mean and standard deviation for each feature dimension using the training set only, then subtracting the mean and dividing by the stdev for each feature and each sample.
4) Encode the labels

Part II: Training a neural network

In this project, you need to implement a 3-layer MLP model (one input layer, one hidden layer with tanh activation and one output layer) which will be used to classify the data in Part I.
You can use the built-in modules in *keras* to build your model.
You also need to write the training function (training), and should explore the following hyperparameter settings:

- Batch size: Number of examples per training iteration.
- Hidden size: Try using different number of hidden nodes in your model and compare the performances.
- Dropout is an effective strategy to defend against overfitting. Adding a dropout layer after the hidden layer, and try using different dropout rate to compare the performances.
- Optimizer: Try using different optimizers such as SGD, Adam, RMSProp.
- Regularization (weight decay): L2 regularization can be specified by setting the *weight_decay* parameter in optimizer. Try using different regularization factor and check the performance.

- Learning rate, Learning rate scheduler: Learning rate is key hyperparameter in model training, and you can gradually decreasing the learning rate to further improve your model. Try using different learning rate and different learning rate scheduler to compare the performance.

  To get full credit, you should explore at least 4 different type of hyperparameters (from listed above), and choose at least 3 different values for each hyperparameters. For simplicity, you could analyze one hyperparameter at a time (i.e. fixing all others to some reasonable value), rather than perfoming grid search.
  If you use TensorBoard to monitor your training, you can directly attach the screenshots of the training curves (accuracy) in your report.
  To evaluate the performance of trained model, you also need to write a function (evaluation) which loads the trained model and evaluate its performance on train/test set.
  In your report, please clearly state what hyperparameters you explored, and what accuracy the model achieved on train/test set.

  Submission: Please include the following files (project_1.zip):

- project_1.pdf/docx
- project_1.py/ipynb