Assignment 1_2020_Sem1

October 19, 2020

1 COMP5318 - Machine Learning and Data Mining: Assignment 1

Due: Wed 21 Oct 2020 11:59PM

2 Summary

The goal of this assignment is to build a classifier to classify some grayscale images of the size 28x28 into a set of categories. The dimension of the original data is large, so you need to be smart on which method you gonna use and perhaps perform a pre-processing step to reduce the amount of computation. Part of your marks will be a function of the performance of your classifier on the test set.

2.1 Dataset description

The dataset can be downloaded from Canvas. The dataset consists of a training set of 30,000 examples and a test set of 5,000 examples. They belong to 10 different categories. The validation set is not provided, but you can randomly pick a subset of the training set for validation. The labels of the first 2,000 test examples are given, you will analyse the performance of your proposed method by exploiting the 2,000 test examples. It is NOT allowed to use any examples from the test set for training; or it will be considered as cheating. The rest 3,000 labels of the test set are reserved for marking purpose. Here are examples illustrating sample of the dataset (each class takes one row):

There are 10 classes in total: 0 T-shirt/Top 1 Trouser 2 Pullover 3 Dress 4 Coat 5 Sandal 6 Shirt 7 Sneaker 8 Bag 9 Ankle boot

2.1.1 How to load the data

There is a Input folder including 4 main files (which can be downloaded from Canvas): 1. images_training.h5 (30000 samples for training) 2. labels_training.h5 3. images_testing.h5 (5000 samples for testing) 4. labels_testing_2000.h5

To read the hdf5 file and load the data into a numpy array, assuming the **training data files are in the ./Input/train** and **testing data file are in ./Input/test**. Use the following code:

Then data would be a numpy array of the shape (30000, 784), and label would be a numpy array of the shape (30000,). It is noted that the labels_testing_2000 only contain 2000 samples for your testing and fine-tuning parameters. We will evaluate your model on full 5000 samples which is not

provided.
The file images_testing.h5 can be loaded in a similar way.

```
[ ]: import h5py
     import numpy as np
     import os
     print(os.listdir("./Input/train"))
```

```
[ ]: with h5py.File('./Input/train/images_training.h5','r') as H:
         data_train = np.copy(H['datatrain'])
     with h5py.File('./Input/train/labels_training.h5','r') as H:
         label_train = np.copy(H['labeltrain'])

     # using H['datatest'], H['labeltest'] for test dataset.
     print(data_train.shape,label_train.shape)
```

Showing a sample data. The first example belongs to class 0: T-Shirt/Top

```
[ ]: import matplotlib.pyplot as plt
     data_train = data_train.reshape((data_train.shape[0], 28, 28))
     plt.imshow(data_train[0], cmap=plt.get_cmap('gray'))
     plt.title("class " + str(label_train[0]) + ": T-shirt/Top" )
     plt.show()
```

2.1.2 How to output the prediction

Output a file "predicted_labels.h5" that can be loaded in the same way as above. You may use the following code to generate an output file that meets the requirement:

```
[ ]: import numpy as np
     # assume output is the predicted labels
     # (5000,) with h5py.File('predicted_labels.h5','w') as H:
     H.create_dataset('Output',data=output)
```

We will load the output file using the code for loading data above. It is your responsibility to make sure the output file can be correctly loaded using this code. The performance of your classifier will be evaluated in terms of the top-1 accuracy metric, i.e.

$$\text{Accuracy} = \frac{\text{Number of correct classifications}}{\text{Total number of test examples used}} * 100\%$$

2.2 Task description

Your task is to determine / build a classifier for the given data set to classify images into categories and write a report. The score allocation is as follows: * Classifier (code): max 60 points * Report: max 35 points * Others: max 5 points Please refer to the rubric in Canvas for detailed marking scheme. The report and the code are to be submitted in Canvas by the due date. This assignment must be submitted in Python3. Although you are allowed to use external libraries for optimisation and linear algebraic calculations, you are NOT allowed to use external libraries for basic pre-processing or classification. For instance, you are allowed to use scipy.optimize for gradient descent or scipy.linalg.svd for matrix decomposition. However, you are NOT allowed to use sklearn.svm for classification (i.e. you have to implement the classifier yourself). If you have any ambiguity whether you can use a particular library or a function, please refer to Canvas -> Modules -> ”Assignment 1 FAQs” for clarification.

2.3 Instructions to hand in the assignment

2.3.1 Go to Canvas -> Assignments -> “Assignment 1” and submit 3 files only: the report and the code file.

1) Report (a .pdf file): The report should include each member’s details (student IDs and names)
2) Code (2 files include: a .ipynb file and a PDF file): The code must be able to be run with the following folder structure:
    - Algorithm (the root folder): Your .ipynb file containing Python code will be placed on this folder when we test and run your code. The PDF file is generated from .ipynb file (File => Save as PDF file)
    - Input (a sub-folder under Algorithm): We will copy the test dataset into this Input folder when we test and run your code. Please make sure your code is able to read the test dataset from this Input folder.
    - Output (a sub-folder under Algorithm): Your code must be able to generate a prediction file named “predicted_labels.h5” to be saved in this Output folder. The prediction file should contain predicted labels of the test dataset. We will use your prediction output file for grading purpose.

Since this is a individual work, each student needs to submit all the files which must be named with student ID numbers following format e.g. “SIDxxxx_report.pdf”, “SIDxxxx_code.ipynb”, “SIDxxxx_code.ipynb.pdf”.

2.3.2 Your submission should include the report and the code.

A plagiarism checker will be used. Clearly provide instructions on how to run your code in the Appendix section of your report.

2.3.3 The code must clearly show :

```
1. Details of your implementation for each algorithm
2. Fine-tune hyper-parameters for each algorithm and running time
3. The comparison result between algorithms
4. Hardware and software specifications of the computer that you used for performance evaluati
```

2.3.4 The report must clearly show :

```
1. Details of your classifier
2. The predicted results from your classifier on test examples
3. Results comparison and discussion
4. Following the format in rubric : Introduction -> Methods -> Experiments result and discussi
5. The maximum length of the report is 20 (including references)
```

2.3.5 A penalty of MINUS 20 percent (-20%) for each day after the due date.

The maximum delay for assignment submission is 5 (five) days, after which assignment will not be accepted.

You should upload your assignment at least half a day or one day prior to the submission deadline to avoid network congestion.

Canvas may not be able to handle a large number of submission happening at the same time. If you submit your assignment at a time close to the deadline, a submission error may occur causing your submission to be considered late. Penalty will be applied to late submission regardless of issues.

2.3.6 All files required for assignment 1 can be downloaded from Canvas -> Assignments -> Assignment 1