COMP5318 - Machine Learning and Data Mining: Assignment 1

March 26, 2019

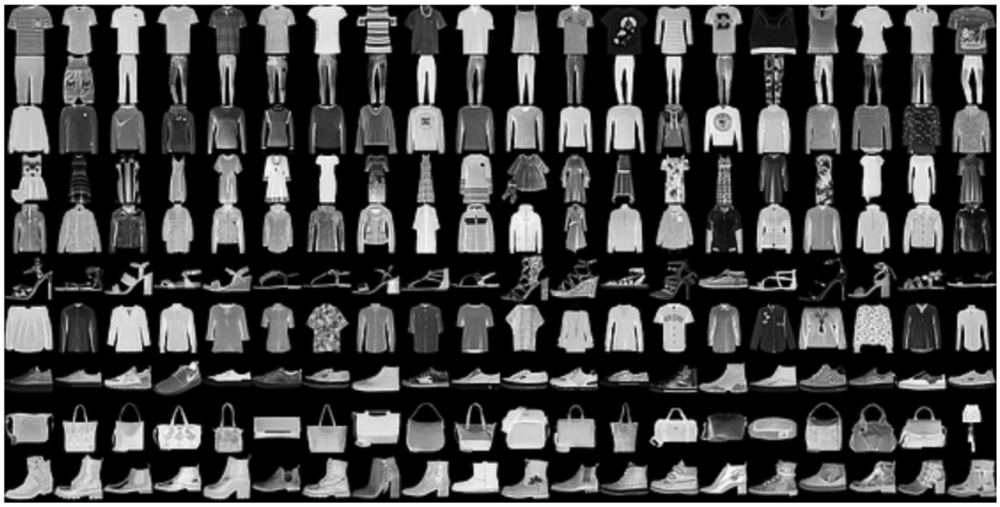
Due: 19 April 2019 5:00PM

1 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.

1.1 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 are preserved for the marking purpose. Here are examples illustrating the data examples (each class takes one row):



Dataset example

1.1.1 How to load the data

There are 4 main files (which can be download from link:
https://www.dropbox.com/sh/d442jv1j4f3rf40/AADoZK5vu58PaueHVCS2FWr6a?dl=0

1. images_training.h5
2. labels_training.h5
3. images_testing.h5
4. labels_testing_2000.h5

To read the hdf5 file and load the data into a numpy array, assuming the **data files are in the same folder as the .ipynb file**. Use the following code:

```
In [ ]: import h5py
        import numpy as np
        with h5py.File('images_training.h5','r') as H:
           data = np.copy(H['data'])
        with h5py.File('labels_training.h5','r') as H:
           label = np.copy(H['label'])
```

Then data would be a numpy array of the shape (30000, 28, 28), and label would be a numpy array of the shape (30000,). The file images_testing.h5 can be loaded in a similar way.

1.1.2 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:

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

We will load the output file using the code for loading data above. It's 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.

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

1.2 Task description

Each group consists of up to 3 students (at least 2 students). 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 20 points
- Report: max 80 points

Please see section 4 for the 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 required). If you have any ambiguity whether you can use a particular library or a function, please post on canvas under the "Assignment 1" thread.

1.3 Instructions to hand in the assignment

1.3.1 Go to Canvas and upload the following files/folders compressed together as a zip file

a) Report (a pdf file): The report should include each member's details (student IDs and names)
b) Code (a folder):

- Algorithm (a sub-folder): Your .ipynb file containing the Python codes.
- Input (a sub-folder): Empty. Please do NOT include the dataset in the zip file as they are too large. We will copy the dataset to the input folder when we test the code.
- Output (a sub-folder): **"predicted_labels.h5"** This file contains the predicted labels of test examples and must be in the output folder. We will use this file for grading. If you work as a group, only one student needs to submit the zip file which must be named as student ID numbers of all group members separated by underscores. E.g. **"xxxxxxxx_xxxxxxxx_xxxxxxxx .zip"**.

1.3.2 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 of the report.

1.3.3 The report must clearly show :

1. Details of your classifier

2. The predicted results from your classifier on test examples
3. Run-time
4. Hardware and software specifications of the computer that you used for performance evaluations.

1.3.4 A template for writing the report can be downloaded from

https://www.dropbox.com/sh/4ou0rn8l89vfkt1/AACZjf2f5AyWlczEm1EqS_7Va?dl=0.
Note that you have to strictly follow the format of the template. The maximum length of the report is 20 (including references).

1.3.5 A penalty of MINUS 20 persent points (-20%) per each day after the due date.

Maximum delay is 5 (five) days, after that assignments will not be accepted.

1.3.6 Rubric can be viewed here

https://www.dropbox.com/sh/uaj16zjxjev465l/AAD7uTYRZD_pnkabAgiftXa4a?dl=0
The due date to submit them on Canvas is 19 April 2019, 5:00PM.