Cairo University
Faculty of Engineering
Computer Engineering Department

Pattern Recognition and Neural Networks

Project Document

Writer Identification System

Fall 2020

Contents

1 Objectives **2**

2 Project Description **2**

3 Team Formation **3**

4 Final Deliverables **3**

5 Rules and Instructions **4**

6 Grading Criteria **4**

7 FAQ **5**

8 Delivery Instructions **6**

1 Objectives

This project aims to put your understanding of machine learning algorithms into practice with a real world problem. **By the end of this project you should be able to:**

1. analyze the problem, extract features, and choose the most appropriate preprocessing method (if necessary).
2. assess performance of different machine learning techniques.
3. design and implement your own ML pipeline.

2 Project Description

In this project, you are required to implement a **writer identification system** that will be able to identify a given writer from their handwritten script or text. This writer identification system proves very useful in a wide range of applications, such as the fields of security, biometrics, and forensics. Not limited to that, but it also raised interest in the analysis of historical texts, historical musical scores with unknown composers, forged signatures in official documents, and many other applications. You should implement a complete machine learning pipeline, *i.e.* the project should include (but not limited to) the following modules:

- Preprocessing Module.
- Feature Extraction/Selection Module.
- Model Selection and Training Module.

- Performance Analysis Module.

You will need to research about the topic, read research papers that tackle this application and similar applications, and do a literature survey that can help you identify the best approaches/ techniques that you can start with in order to improve the accuracy of your results.

You will work with the **IAM Handwriting Database** containing a large number of handwritten English scripts which you will divide it into:

- **Training set** to train your model.
- **Validation set** to tune the parameters of your model.
- **Test set** to report the results of your models.

Note that the dataset contains many forms of handwritten text including complete form text, extracted sentences and extracted words. You will work with **complete form text only.** The dataset can be found here: `http://www.fki.inf.unibe.ch/databases/iam-handwriting-database`

3 Team Formation

A team should be formed of 3 to 4 members. No team should have more than 4 members under any condition. Please register your team number in the provided spreadsheets. All team members should attend the final project discussion. If one team member failed to show up in the project discussion, then they are regarded as zero contributors in this project.

4 Final Deliverables

By the end of this project, each team should submit the following:

1. A **PDF report** that includes the following sections fully and clearly described, while describing all the work done and approaches adopted. You should include also the unsuccessful trials.
   (a) Project Pipeline.
   (b) Preprocessing Module.
   (c) Feature Extraction/Selection Module.
   (d) Model Selection/Training Module.
   (e) Performance Analysis Module.
   (f) (Optional) Any other developed modules.

(g) Enhancements and Future work.

The report should include also a workload distribution between team members.

2. **Code** as a zipped folder with the format code [team number].zip containing all code developed with a *readme* file including all packages or libraries needed to run this code and how to run the code. It is highly recommended to provide an executable file beside the source code.

5 Rules and Instructions

All students must follow the following rules and instructions:

1. All projects are submitted on the classroom.
2. Don't print any document or report. All submissions are electronic.
3. There will be a late penalty for the final project submission.
4. Any sign of cheating or plagiarism will not be tolerated. If one team got caught cheating or plagiarizing from another team, both teams will receive a ZERO for the project grade, in addition to a penalty up to 50% of the project grade.
5. Workload should be distributed fairly and equally. Team members who did not contribute effectively in the project will be penalized. 6. All team members should attend the final discussion. A team member who fails to show up will get a ZERO in the discussion grade.

6 Grading Criteria

The grading criteria of this project is divided as follows:

1. **Report and Discussion: (20%):** This point includes the report submission, the quality of the report and the discussion with all team members. It also evaluates how all team members fully understand the details of the project and can elaborate on the examiner questions.
2. **Project Performance (80%):** Both results accuracy (65%) and running time (15%) will be taken into consideration by the given weights. However, it's not a linear formula but the ranking procedure will be based only on these two factors. Take care that your language choice might affect the running time. This project is a competition-based one.

7 FAQ

1. **What programming language(s) can I use?**
   You may use any programming language of your choice!

2. **Is there any restriction on the techniques or approaches to use in any phase of the project?**
   You are free to use any approach or technique you find appropriate for the problem in hand. It's actually your task to find out the best combination of techniques that will yield the best results. However, you are **limited only to classical machine learning methods** such as *Bayesian Classifiers, KNN, Linear/Logistic Regression, Neural Networks (with two hidden layers as a maximum), Support Vector Machines, Principal Component Analysis, etc.*
   You are NOT ALLOWED to use **deep learning** techniques *e.g. Convolutional Neural Networks, Recurrent Neural Networks, etc.*

3. **How to find research papers for this topic?**
   There are many resources on the web tackling this problem. You can start by creating an account on EKB (Egyptian Knowledge Bank) with your university provided student email address, which will give you wide access to a huge number of research papers and journals. You can also use the Academic search engines such as Microsoft Academic or Google Scholar) a lot of articles are free for public.

4. **How will our project be tested?**
   We will test with our own test set (you have not seen before) in order to standardize the way all teams are graded and ranked. However, you should divide your dataset into Training, Validating and Testing set. The test set is identical to the IAM database but it is not directly extracted from it.

5. **How will our project be ranked?**
   Please refer to the Grading Criteria section for the grading criteria. The teams will be ranked according to some formula comprising the results accuracy as well as the time taken to generate the results, with the larger weight for the results accuracy. For example, Team X had results with accuracy 92.0%, with running time = 5 seconds, will be ranked above team with accuracy 80.0%, with running time = 2 seconds.

8 Delivery Instructions

8.1 Input Format

The project will be evaluated using our own test set. The test set is identical to the full-form images in the **IAM Handwriting database**. You will receive a folder named data. Under this directory, you will find N sub-directories where N is the number of test cases. These folders will be named **01, 02, 03, ..., 10, 11, 12, ..., 100, 101, 102,**. . . .etc.

Under each folder of the **N** folders, you will find three other sub-directories named **1, 2** and **3**, together with a test file. Each of the three folders contain two **png** images for writers 1, 2, and 3 respectively. The test file is named **test.png**

The hierarchy can be visualized as follows (for the first two test cases):

- **\data**
  - **\01**
    - **\1**
      - 1.png
      - 2.png
    - **\2**
      - 1.png
      - 2.png
    - **\3**
      - 1.png
      - 2.png
    - test.png
  - **\02**
    - **\1**
      - 1.png
      - 2.png
    - **\2**
      - 1.png
      - 2.png
    - **\3**
      - 1.png
      - 2.png
    - test.png

Make sure that you process the test cases in an **increasingly ordered sequence**. You will output the results in the same order as the input.

Each folder of the N folders represents an **iteration**, where in each iteration you will train your model with two sheets of paper for three different writers. Finally, you will apply your model on the test image to identify whether its writer 1, writer 2, or writer 3.

8.2 Output Format

You should generate **TWO** text files:

1. **results.txt**
2. **time.txt**

8.2.1 Results

In the first file **results.txt**, you should output the prediction of the model for every test case of the N test cases in a single line. The prediction of the model is either 1, 2, or 3.

The format of this file should be as follows:

```
2
3
1
2
1
1
3
⋮
3
1
```

This means that: In the first iteration, your model predicted the test image to belong to **writer 2**, whereas in the second iteration, your model predicted the test image to belong to **writer 3**, and so on until the Nth iteration where your model predicted the test image to belong to **writer 1**.

Important Notes:

1. Make sure that you output the result for the test cases in the same order as the input. This mistake is very common as you might think you are ingesting the test cases in order, however, you find you are reading the test cases in an incorrect way that you are processing folder 10 before 01 and so on.

2. This file should not include anything other than the model predictions as indicated.
3. Do not include the test case number.
4. Do not include any extra information.
5. Do not print all the test cases in one single line.

8.2.2 Time

In the second file **time.txt**, You should output the time taken (in seconds) by every iteration of the N iterations in a single line.

The format of this file should be as follows:

```
30.00
35.25
32.32
40.15
35.12
38.98
26.23
⋮
20.65
```

This means that: In the first iteration, your code ran in **30.00 seconds**, in the second iteration, your code ran in **35.25 seconds** and so on until the Nth iteration where your code ran in **20.65 seconds**.

Important Notes:

1. For every iteration, place a timer **after** reading the training images and the test image, and **after** generating the prediction for the test image. You should output the difference between the two times (i.e. the time taken for a single iteration to be processed in the complete pipeline without any I/O overhead).
2. The numbers in the above example are just random numbers and have nothing to do with the actual running times. These numbers are **NOT** a reference and they are just for illustration.
3. The running times should be rounded to **two decimal places**.
4. This file should not include anything other than the model running times as indicated.
5. Do not include the test case number.

6. Do not include any extra information.
7. Do not print all the test cases in one single line.
8. Do not print the results in the console. You have to generate the two files results.txt and time.txt