Faculty of Engineering
Cairo University

CAIRO UNIVERSITY

Pattern Recognition and Neural Networks
Semester Project

Team No. 15

# Writer Identification System

(Final Report)

Submitted to:

Eng. Hussein Fadl

19th Jan. 21

## Project Pipeline:

Our pipeline typically is divided into 4 major modules:
1- Pre-processing
2- Feature Extraction
3- Model Training
4- Inference and Classification

**1- Pre-processing:**
The first processing stage on the IAM-like image, in which 2 typical processes are done:

- Paragraph Extraction: in which the header and footer of the image are cropped, and the handwriting paragraph is extracted.
- Lines Extraction: in which the paragraph is divided into a set of images, each correspond to a line of the paragraph

Its output is a list of small line images extracted from the input image.

**2- Feature Extraction:**

In which Local Binary Patterns are extracted from each of the images in the list, histogram of each of them is calculated as typical 256-length array, then compressed into around 10 bins.

Its output is a list of histogram arrays for each of the listed line images.

**3- Model Training:**

In which Random Forest Classifier, with depth=100 and number of estimators=100 is adopted from sklearn library, and is trained with the list of histograms, each labeled with its true writer number.

Its output is a trained model on the 6 training images.

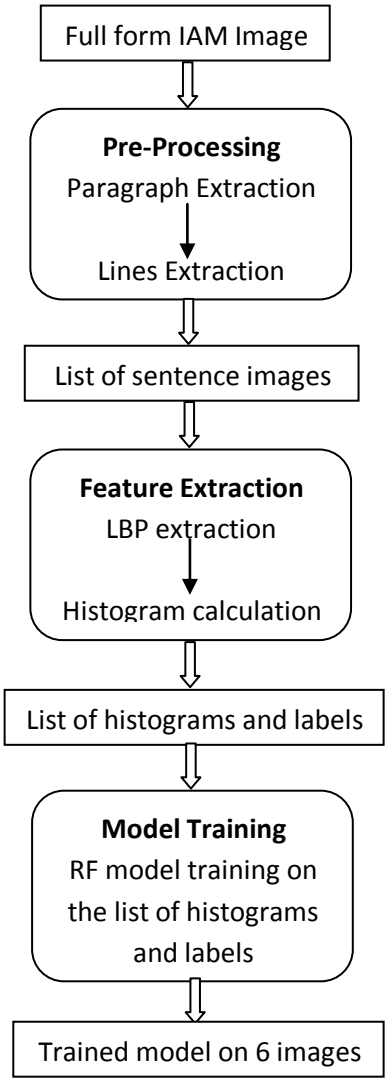
**4- Inference and Classification:**

In which the same processes of preprocessing and feature extraction are done on the test image, a histogram list for each line in the paragraph is obtained which is given to the trained model to predict which of the 3 writers the image most likely belongs to according to the majority vote of its lines histograms.

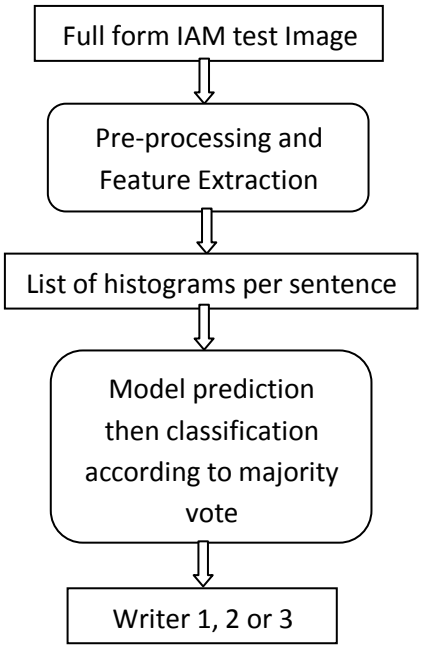
Its output is a final prediction, typically (1, 2 or 3) indicating the highest probable writer.

Following is an illustrating graph of the full pipeline:

Training:




Inference:



## Performance Analysis:

Results.txt, time.txt are 2 files that are printed after each successful test session, indicating the predicted labels for each of the test cases, alongside with the time taken by each of them during being processed through the full pipeline. These indicate how fast and faithfully-resulting our system is, in correspondence with each test session independently.

## Enhancements and Future Work:

Time can be enhanced by using libraries that work on speeding up python code through embedded C conversion and compilation. The whole code could be written in C++ that will decrease the time also. We may build the library of local binary pattern to deal with the black pixels only to decrease time. For Accuracy, we achieved 100% on our 101 training folders we have done. So, there's no further works to do on our side, but it depends on its behavior on your test folders that we expect to bring good results as us, but also, it could be enhanced by adopting deep neural networks approach to be more accurate on any new test cases.

## Workload Distribution:

| Member Name | Activities |
|---|---|
| Ahmed Hamdy | - Preprocessing: Paragraph Extraction<br>- Dataset Handling Script |
| Ayat Mostafa | - Model Training and Classification<br>- Performance Analysis |
| Mohammed Abdallah | - Feature Extraction |
| Nada Adel | - Preprocessing: Handwriting Lines Extraction |

## Unsuccessful Trials:

Unsuccessful (less accurate) trials include using KNN Classifier with different K values, failure is due to frequent ties that are handled by random picking, also SVM classifier yielded close but less accuracy. Less powerful features describing the handwriting style of letter size, orientation and white spacing also failed. We tried to work with the whole image but it gives bad accuracy. So, we started to think on increasing our dataset by taking each line alone and in the test image, each line will be classified to a writer corresponding to the highest probability of classification and then we will take the more classified writer by all these lines. We tried to take each word alone but it takes more time. So, we settled on working with lines.