National University of Sciences and Technology (NUST)
School of Electrical Engineering and Computer Science

**Department of Computing**

**CS 471: Machine Learning (3+1)**

**Class: BSCS12-A**

# Lab 5: Feature Extraction Offline Signature Verification

**Date: 15-10-2024**

**Time: 2:00 pm – 5:00 pm**

**Instructor: Prof. Dr. Muhammad Moazzam Fraz**

**Lab Engineer: Ms. Iram Tariq Bhatti**

**Course Learning Outcomes (CLOs)**

| No. | Course Learning Outcomes | Graduate Attributes (CS) | BT Level | Teaching & Learning Methods | Assessment Methods |
|---|---|---|---|---|---|
| CLO-1 | Explain theoretical concepts behind various machine learning algorithms and techniques | GA-2 | C-2 | Active/Blended | Quizzes, Assignments, ESE, MSE |
| CLO-2 | Analyse a given problem and select the most suitable supervise / unsupervised machine learning algorithm for the task. | GA-3 | C-4 | Active/Blended | Quizzes, Assignments, ESE, MSE |
| CLO-3 | Implement various machine learning algorithms using modern software libraries for a range of applications. | GA-5 | P-3 | Cooperative Learning | Labs, Project |
| CLO-4 | Contribute effectively in a team to complete a given task | GA-6 | A-2 | Cooperative Learning | Labs, Project |

## Introduction

The purpose of this lab is to get familiar with offline signature verification and perform initial verification routines by extracting and engineering features.

## Objectives

After completing this lab, students will be able to understand how to:

- Extract and engineer features from signature images

## Software Tools/Requirements

- Solutions should be made in Python
- Use PIL or OpenCV for image processing
- numpy

## Lab Task

In this lab, you will implement the following tasks.

1. Develop a bounding box around the signature content.
2. Find out the centroid of the signature.
3. Segment signature from centroid vertically and horizontally (the signature will be divided into four pieces)
4. Repeat steps 2-3 until you have segmented the image into 64 cells.
5. Calculate black to white transitions for each of the 64 cells.
6. Compare the features (black to white transitions from each cell/piece) from one signature to the next signature (corresponding cells only) and mark the cells which look stable on the basis of number of black to white transitions. Repeat this for all the 25 ref signatures and populate a feature set as explained in the lab.
7. Calculate aspect ratio of each cell.
8. Compute the (i). skew and (ii) slant angles from each cell, and analyze the cells with stable slant and skew throughout all the 25 reference signatures. Note: For understanding of slant and skew, please refer to the provided research paper.

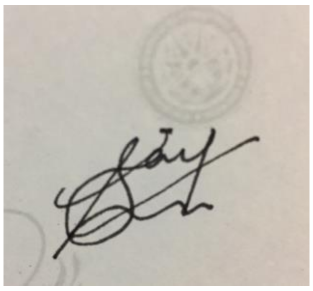
You should perform the aforementioned tasks for each of the images in Reference folder in the dataset provided to you. Save extracted features in text files using numpy.

Some notes that might be helpful for completing this lab:

PS: It’s a group lab of at max 2 members with only one submission against each group.

## Description

In this section, we will walk through the lab tasks and see how to complete them. This walkthrough is to provide you a starting point for your own implementations, and hence is abstract and leaves out many implementation details. You are supposed to figure them out yourself. We will use the following signature as a reference.



### Preprocessing

Convert the signature to a binary (black-and-white) image before extracting features. With PIL, this can be done by using the `Image.convert()` method. For further information on how to use PIL, refer to PIL handbook at https://pillow.readthedocs.io/en/stable/handbook/index.html At this point, your signature image should look like this:

### Task 1: Developing a bounding box

In this task, you have to locate the signature in the image and develop a bounding box around the signature content only, ignoring the white space around it.

```
Algorithm
left := width
right := 0
top := height
bottom := 0
for x in (0, width) and y in (0, height)
      color := image.getpixel(x, y)
      if color is 0
            if x > right
                  right := x
            if x < left
```

```
                left := x
          if y > bottom
                bottom := y
          if y < top
                top := y
```

After completing this task, your signature image should look like this:

## Task 2: Locating the centroid

Centroid is the point where center of mass of the signature image is located. It can be computed using the algorithm given below.

```
Algorithm
cx := 0
cy := 0
n := 0
for x in (0, width) and y in (0, height):
      if image.color(x, y) is 0:
            cx := cx + x
            cy := cy + y
            n := n + 1
cx := cx / n
cy := cy / n
```

## Task 3: Dividing the image at centroid to create four segments

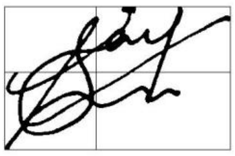
You should have the following at this point:

1. Bounding box = (left, right, top, bottom)
2. Centroid = (cx, cy)

You now have to divide the image into four segments. The four segments can be computed by locating the boundaries of each of the four segments:

1. (left, cx, top, cy) are the boundaries of top-left segment
2. (cx, right, top, cy) are the boundaries of top-right segment
3. (left, cx, cy, bottom) are the boundaries of bottom-left segment
4. (cx, right, cy, bottom) are the boundaries of bottom-right segment

After completing this task, your signature image should look like this:



## Task 4: Finding black to white transitions

In this lab task, you have to find black to white transitions for each of the four segments. That is, you have to calculate the number of white pixels in neighborhood of black pixels in the image.

```
Algorithm
prev := image.color(0,0)
n := 0
for x in (1, width) and y in (1, height):
      curr = image.color(x,y)
      if curr is 255 and prev is 0
            n := n + 1
      prev := curr
```

## Completion check

On successful completion of the lab tasks, you should have the following:

1. Bounding box of the signature in image
    a. B = (left, right, top, bottom)
2. Coordinates of centroid of the signature
    a. C = (cx, cy)
3. Four values of black to white transitions, for each of the four segments
    a. T = (TL, TR, BL, BR)

## Description

In this section, we will walk through the lab tasks and see how to complete them. This walkthrough is to provide you a starting point for your own implementations, and hence is abstract and leaves out many implementation details. You are supposed to figure them out yourself.

At the end of last task, we had the following signature:

Now, we will extend our solution to divide the signature into 64 cells instead of 4, for each cell, extract the following features:

1. Centroid
2. Black to white transitions
3. Aspect ratio (= width/height)

After extracting these features from each cell, dump them to text files. You may use numpy's dump function for this, or any method of your choice. Repeat this process for each of the reference signatures. This means that, for each reference signature, there would be three files, each with 64 entries, one for each cell.

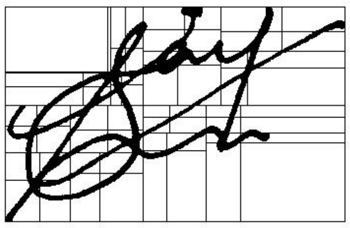
### Task 1: Dividing the signature into 64 cells

Extend your solution from previous task where you divided the image into four segments after calculating bounding box and centroid, to divide it into 64 cells. This can be done either iteratively or recursively.

```
Algorithm
split(image, left, right, top, bottom, depth=0):
      cx, cy = findCentroid(image, left, right, top, bottom)
      if depth < 3:
            split(image, left, cx, top, cy, depth + 1)
            split(image, cx, right, top, cy, depth + 1)
            split(image, left, cx, cy, bottom, depth + 1)
            split(image, cx, right, cy, bottom, depth + 1)
      else:
            t = findTransitions(image, left, right, top, bottom)
            r = findRatio(left, right, top, bottom)

            # save centroid, transitions and ratio to file
```

After completing this task, your signature image should look like this:

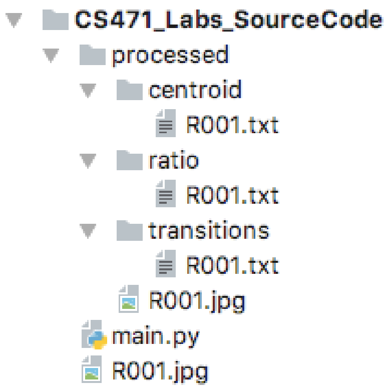


## Completion check

On successful completion of the lab tasks, you should have the following:

1. For each of the Ref signature images
   a. For each of the 64 cells
      i. Coordinates of centroid, C = (cx, cy)
      ii. Black to white transitions, T
      iii. Aspect ratio, R = width of cell / height of cell

The feature sets calculated above should be dumped in text files, using numpy or otherwise. Your project files would be something like:



```
▼ CS471_Labs_SourceCode
    ▼ processed
        ▼ centroid
            R001.txt
        ▼ ratio
            R001.txt
        ▼ transitions
            R001.txt
          R001.jpg
      main.py
      R001.jpg
```

Each of the .txt files contains values of a particular feature for each of the 64 cells. The screenshot shows the output after processing one signature, R001.jpg. You have to repeat the tasks for all Ref signatures in the dataset.

## Deliverable:

Submit the folder that contains source code, processed text files for each aspect of the each reference image before deadline on LMS.