Handwritten Signature Verifier

A Project Synopsis

Submitted by

Tejas Jadhav

Under The Guidance Of

Prof. Abhay Kolhe

in partial fulfillment for the award of the degree of

M. TECH.
in
COMPUTER ENGINEERING
at

SVKM's NMIMS,
Mukesh Patel School of Technology Management & Engineering,
Mumbai

2018 - 19

CERTIFICATE

This is to certify that the project synopsis entitled "Handwritten Signature Verifier" is the proposed work by Tejas Jadhav of M. Tech. (Computer Engineering), MPSTME (NMIMS), Mumbai, during the III/IV semester of the academic year 2018 - 2019 is verified by me.

The presentation for the same is also verified by me.

Prof. Abhay Kolhe	Tejas Jadhav
Internal Mentor	

Table of Contents

1. Introduction

Biometrics is currently perceived as a vital technology for setting up secure access control. It utilizes physiological qualities of humans for recognizing individual, and signature is one of the characteristics usable for biometrics. Singular recognizable proof technology utilizing human countenances, more often than not called confront acknowledgment technology, has been concentrated primarily in western nations since 1990s. It has been the most widespread tool for paper documents verification for many decades. In real life the human-expert can verify handwritten signatures easily, but it is a complicated task for doing by machines today.

1.1 Real life applications

This application of Image processing has number of uses in the real world and are seen to be very critical in many industries. Some of them are as follows:

1. Finance: IT-Processing firms of German savings banks are offering their customers solutions to embed dynamic signatures securely into electronic documents in an Adobe Live Cycle environment
2. Insurance: Signing an insurance contract and documenting the consulting process that is required by EU legislation have caused several insurance companies to go paperless with either signature capturing tablets connected to a notebook or a tablet PC.
3. Real Estate: Increasingly popular among real estate agents in USA are options of paperless contracting through signing on Tablet PCs.
4. Health: The hospital of Ingolstadt is capturing and verifying the signatures of their doctors that fill electronic patient records on tablet PCs. The "National Health Service" organization in the UK has started such an implementation.
5. Telecom: Signing phone and DSL contracts in the telecom shops is another emerging market.

1.2 Types & methods of Signature Verification

There are two types of signature verification methods:

1. Offline Signature Verification Methods:

In this method, system makes use of a simple scanner or a camera that can take in image having signature & process the image further with whatever features it gets. This method can be seen useful in many applications such as banking cheques, medical certificates & prescriptions etc.

2. Online Signature Verification Methods:

Here the system makes use of special acquisition devices like stylus pens and tablets that can take in signature features in real time and may give better accuracy

1.3 Performance evaluation parameters

1. FAR – False Acceptance Rate :

The false acceptance rate in simple words is rate of falsely accepted signatures. This is given by the following formula:

$$FAR = \frac{\text{Total Number Imposter Signatures Accepted as Genuine}}{\text{Total Number of Forgery Tests Performed}}$$

2. FRR – False Recognition Rate : Rate of falsely rejected signatures

The false acceptance rate in simple words is rate of falsely rejected signatures. This is given by the following formula:

$$FRR = \frac{\text{Total Number Genuine Signatures Rejected as Imposter}}{\text{Total Number of Genuine Matching Tests Performed}}$$

3. EER – Equal Error Rate

The Equal Error Rate (EER) corresponds to the error value for which false acceptance rate is equal to false rejection rate. These rates determine the quality of an authentication system, but the acceptable values depend on the level of security desired for a specific application.

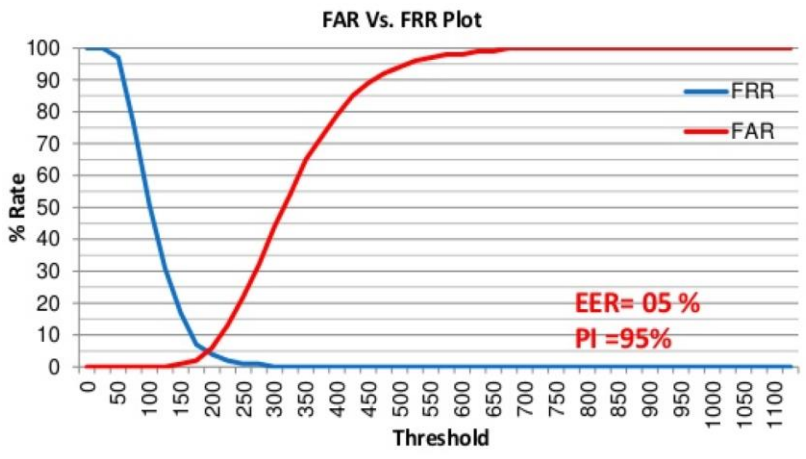


Figure 1: Performance evaluation graph

4. PI – Performance Index

Performance index is nothing but the opposite of the Equal error rate. This is calculated easily by the following formula:

PI = 100 – EER

2. Motivation

Today the human signature of a person is used as an identification of person because we are all know that the each person has distinct signature and every signature has its own physiology or behavioral characteristics. So the human signature used as an identification of person in various work like bank checks etc. The fraud person can easily generated the signature instead of unique signer in fraud way so we need a signature identification system

Nowadays, person identification is very important in security and resource access control. A reliable signature recognition system can be seen as an important part of law enforcement, finance & banking and many business processes. In such applications, an accurate recognition of person through his handwritten signature is critical.

Thus the motivation behind this project is the growing need for a full proof signature verification scheme which can guarantee maximum possible security from fake signatures. The idea behind the project is also to ensure that the proposed scheme can provide comparable and if possible better performance than already established offline signature verification schemes. The prospect of minimizing the memory space for storing signature image by the preprocessing extracted feature and the training is completed in acquiring less time and provide better accuracy. The need to make sure that only the right people are authorized to access high-security systems has paved the way for the development of systems for automatic personal authentication

3. Problem definition

Signature Recognition is the procedure of determining to whom a particular signature belongs to, as said earlier. This problem is a huge research topic under the Image Processing domain. But this problem is also sometimes crossed with Machine Learning domain along with the Image Processing domain. This is not as simple as said it seems. It is just easier said than done.

Thus the system to be designed must have an algorithm that will extract features and recognize and verify the signer's authentication. System would take as input signature images and tell us following things:

1. To whom the signature belongs to (Author Identification)
2. If the signature is forged or genuine (Signature Verification)

4. Literature review

This section lists down the various algorithms that we are to study. Most of them work in 3 stages, which are pre-processing, main processing i.e. feature extraction and then post-processing stage which includes decision making or classification. There are a high number of algorithms that can be implemented for recognition of handwritten signatures these days, but following are the algorithms which give other newly formulated algorithms a backbone.

In paper [1] the researchers are considering some simple parameters like speed, acceleration, pen down time, distance, etc. and based on the literature studies they discuss how these features can be used in the Image processing environment to improve the performance of handwritten signature. The approach provides a PI of 97.5 %.

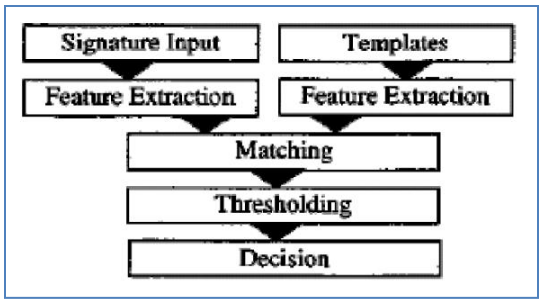


Figure 2: Flowchart of paper [1]

Features extracted and used for comparison include total Euclidian distance D of the pen travelled, Speed Vx and Vy express the functions of time, Acceleration Ax and Ay, Total time taken Tk, Length-to-width ratio, Amount of zero speed in direction *x* and *y* directions *Nvx* and *Nvy* , Amount of zero acceleration in direction *x* and *y* directions *Nax* and *Nay*

The paper [2] presents a crowdsourcing experiment to establish the human baseline performance for signature recognition tasks and a novel attribute-based semi-automatic signature verification system inspired in FDE analysis. It combines the DTW algorithm with Attribute based algorithm to obtain better accuracy and increase the recognition rate. With EER of 5%, we can say that recognition rate is 95%. Attribute based recognition features include Shape, Proportionality, Text-loops, Order, Punctuation, Flourish-characteristics, Hesitation, Alignment to the baseline, Slant of the strokes, Strokes-length, Character spacing

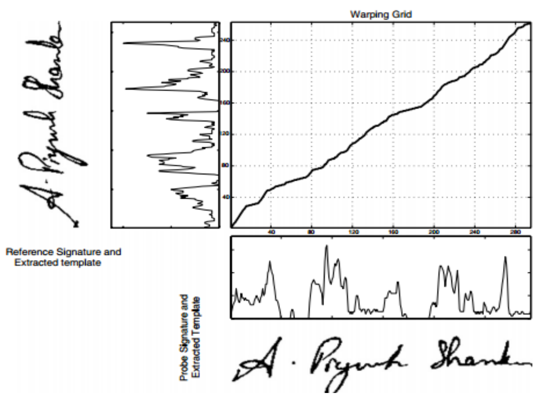


Figure 3: Dynamic Time warping

The dynamic time warping algorithm finds an optimal match between two sequences of feature vectors which allows for stretched and compressed sections of the sequence.

The proposed system in paper [3] functions in three stages. Pre-processing stage; this consists of grey scale conversion, binarisation and fitting boundary box. Feature extraction stage where total 16 radon transform based projection features are extracted which are used to distinguish the different signatures. Finally in the classification stage; an efficient BPNN, Back Propagation Neural Network is prepared and trained with 16 extracted features. The average recognition accuracy of this system ranges from 87% - 97% with the training set of 10–40 persons. Global features include_Height of the signature, Width of the signature, and Centroid along both X axis, Centroid along both Y axis

The radon transform derives projections of the image matrix along predefined directions. A projection of a two dimensional function f(x,y) is a group of line integrals. The radon transform calculates the line integrals from multiple sources along parallel paths, or beams, in a predefined direction. The beams are spaced one pixel unit apart. To represent an image, the radon transform takes multiple, parallel-beam projections of the image from different angles by rotating the source around the centre of the image. Radon transform of a signature image I for the angles specified in the vector 'theta' is computed using the Matlab function radon().

It returns a vector 'r' for each angle in theta containing the Radon Transform. Mean and standard deviation of all the vectors 'r' are calculated and taken as a local features.

The number of neurons in the first layer is n (n=16 in this work) which is equal to the dimensionality of the input pattern vectors (Number of input nodes equals number of input features used). The number of neurons in the output layer is 5 which are equal to the number of pattern classes.

Table 1: Local radon transform features

1	Height	9	Mean90
2	Width	10	Std90
3	Centroid of X axis	11	Mean120
4	Centroid of Y axis	12	Std120
5	Mean30	13	Mean150
6	Std30	14	Std150
7	Mean60	15	Mean180
8	Std60	16	Std180

The developed BPNN is trained with signature from different persons. Large numbers of images are required to ensure proper training of the NN.

In paper [4], researchers propose a new on-line writer authentication system using the pen altitude, pen azimuth, shape of signature, and writing pressure in real time. It is well expected that altitude and the azimuth of gripped pen under signing depends on the shape of writer's hand and the habit of writing. After individuality in the altitude and the azimuth of gripped pen under signing is explained, the experimental result with writing information by 24 writers is shown. It is found that the authentication rate of 98.2% is obtained.

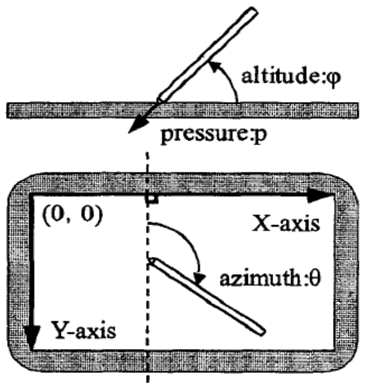


Figure 4: Data from tablet and pen.

The data set prepared is based on the following Features to be calculated dynamically at every instance of time: X-coordinate: *x(t)*, Y-coordinate: *y(t)*, Pressure: *p(t)*, Azimuth: $\theta(t)$, Altitude: $\phi(t)$

In paper [5], researchers propose a new on-line writer authentication system that uses the pen azimuth, point positions of signature, and writing pressure in real time for creating 7 fuzzy characteristics. The stability of the altitude and the azimuth under signing is also compared with the bit-mapped data and the pen pressure along time for one writer.

Researchers used collection of signatures for testing this system. Signature verification experiment has been conducted with 100 users across 25 original and 25 fake signatures for each user. The recognition rate shown was 99.8%. A 28-dimensional feature vector *for* the signature is formed.

In this paper [6] researchers present an off-line signature verification and recognition system using the global, directional and grid features of signatures. Support Vector Machine (SVM) was used to classify & verify the signatures and a classification ratio of 0.95 was obtained. Thus 95% accuracy is obtained as the recognition of signatures represents a multiclass problem SVM's one-against-all method was used.

The system researchers introduced is divided into two major sections: (i) Training signatures, (ii) Verification or recognition of given signature.
Features used were Signature area, Signature height-to-width ratio, Maximum horizontal histogram and maximum horizontal histogram, Maximum vertical histogram and maximum vertical histogram, Horizontal and vertical center of the signature are calculated using the formulas

In this system, a multi class system is constructed by combining two class SVMs, radial basis function is used which gave the best results. In the training phase, for each person 8 positive (genuine) and 82 (39 x 2 + 4) negative (forgery) examples are taken.

The proposed system of paper [7] functions in three stages. Pre-processing stage consists of four steps namely gray scale conversion, binarization, thinning and fitting boundary box process in order to make signatures ready for the important feature extraction. Feature extraction stage consists total 59 global and local wavelet based energy features to be extracted which are used to classify the different signatures. Finally in classification, a simple Euclidean distance classifier is used as decision tool. The average recognition accuracy obtained using this model ranges from 90% to 100% with the training set of images of 10 to 30 persons.

The proposed method in paper [8] is based on the hypothesis; reducing the variability of signatures leads to boost up the recognition rate. Therefore, the variance reduction technique is applied to normalize offline handwritten signatures by means of an adaptive dilation operator. Then the variability of signatures is analyzed in terms of coefficient of variation (CV). The optimal CV is obtained and used as a threshold limit value to be acceptable variance reduction. The average recognition rate after experimental analysis is found to be 94.87%.

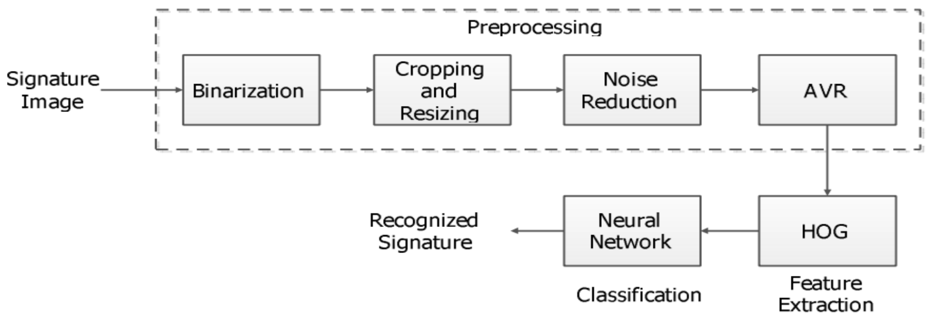


Figure 5: Flowchart for paper [8]

The paper [9] proposed a SIFT and a SURF algorithm which is used for enhanced offline signature recognition. This process, Bag-of-word features, was operated by making vector quantization technique, which outlined the key points for each training image inside a unified dimensional histogram. They put features of bag-of-word inside multiclass Support Vector Machine (SVM) classifier established upon the radial basis function (RBF) for a training and testing. They used Open CV C++ as an image processing tool and tool for feature extraction. In this paper, we compare the performance of SIFT on SVM based RBF kernel with SURF on SVM based RBF kernel. It was found out that the use of SIFT with SVM-RBF kernel system, it has an accuracy of 98.75% compared that of SURF with SVM-RBF kernel it has an accuracy of 96.25%. The SURF algorithm (speeded up robust transform) is composed mainly two parts. In the first part, locate the interest point in the image. The surf features are calculated and points are matched between the input and out signatures. The surf features are computed and points are matched across the input and out signatures. SURF detectors are looked in the significant points in the image, and descriptors are used to get the feature from vectors at each interest point just as in the SIFT. Hessian-matrix approximation consists of SURF to put through and locate the key points rather than variant aspects of the Gaussians (DOG) filter prepared in SIFT. This algorithm is very much similar to SIFT algorithm. But, it is actually three times faster than SIFT. About 64 dimensions can be used in SURF to save the time consumption for the two traits which are the matching and the computation.
Scale Invariant Feature Transform is one of several computer vision algorithms which is clearly aimed at extracting distinctive and invariant features from images. Features are extracted by making the SIFT algorithm invariant to image scale, rotation, and partially robust to modifying viewpoints and changes in illumination.

In paper [10], we proposed and implemented an innovative approach based on upper and lower envelope and Eigen values techniques. Envelope represents the shape of the signature. The feature set consists of features such as large and small Eigen values computed from upper envelope and lower envelope and its union values. Both the envelopes are combined by performing union operation and their covariance is calculated. The ratios and the differences of high and low points of both these envelopes are calculated. Lastly average values of both the envelopes are obtained. These features set are coupled with support vector machine classifier that lead to 98.5% of accuracy.

In the paper [11] researchers, use this daily based biometric characteristic for identification and classification of students' papers and various exam documents used at University of Mostar. Paper uses a number of Global features, Grid Features, SIFT features. For classification, Support Vector Machine is used and the accuracy obtained is of 88.97%. Global features includes Aspect ratio, number of pixels that belong to the signature, Baseline shift, Global Slant Angle, Number of Edge Points, Number of Cross-Points and Spatial Symbols, Horizontal and vertical Center of gravity, Length name, Length surname, Maximal horizontal and vertical histogram, the length and ratio of Adjacency Columns, Heaviness of signature, Ratio of first vertical pixel to height, Number of Closed Loops.

In paper [12], a bank cheque is taken and the signature is collected from the bank cheque by taking it out by cropping the area of interest. The image is then trained and stored into the trained database using an efficient Feed forward artificial neural network. Then signatures to be tested are compared with the signatures that are stored into the test database. The accuracy of the system is tested out to be 85.00%. In pre-processing, Otsu's thresholding or binarization algorithm is used which is one of the finest one, allows image object to be separated from its background and later Gaussian low pass filter is applied to remove unwanted noise.

Fuzzy Logic and Artificial Neural Network Based Off-line Signature Verification and Forgery Detection System is presented in paper [13]. As there are distinct and necessary variations in the feature set of each signature, so in order to match a particular signature image with the database image, the structural features of the signatures and also the local feature variations in the signature characteristics are used. The paper suggests that, variation in personality of signatures, because of age, sickness, geographic location and emotional state of the person actuates the problem

A new approach to document image retrieval based on signature is described in paper [14]. The database consists of document images with English text combined with headlines, logo, ruling lines, trade mark and signature. In searching a particular repository of business documents, the actual work to be done is using a query signature image to retrieve from a database. DT-RCWF and DT-CWT are used for extraction features and recognition rate of images is of 79.32%.

This paper [15] focuses on the pre-processing phase, which is an alternative way to improve the accuracy and to make such factors stable. This research is basically based on the hypothesis that, a table signature size is able to boost up the efficiency which means the recognition rate. Polar Scale Normalization, Adaptive Variance Reduction, Histogram of Oriented Gradients are the techniques used in the system and 98.39% of accuracy is shown.

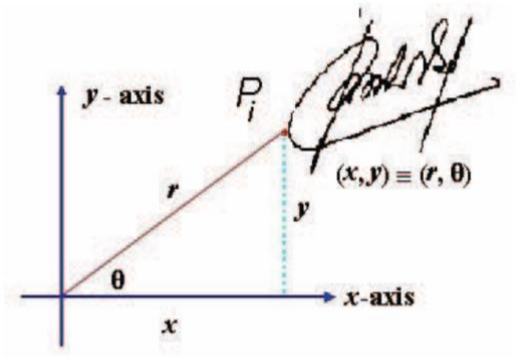


Figure 6: Polar scale normalization

The pre-processing techniques include Binarization, Complementation, Cropping, and Resizing. The classification of the test data is done using an efficient artificial neural network.

A signature verification system based on Dynamic Time Warping (DTW) is proposed in paper [16]. Pre-processing techniques have Maximum Length Vertical Projection (MLVP) method, Minimum Length Horizontal Projection (MLHP) method. The technique basically works by extracting the vertical projection based features from signature images and by comparing probe and reference feature templates using elastic matching classification. A 98% accuracy is gained and show promising results.

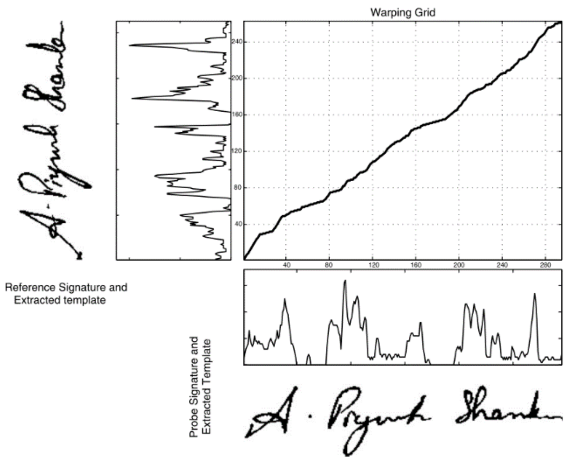


Figure 7: DTW algorithm of paper [16]

Present paper [17] focuses on different steps including browsing a bank cheque, pre-processing, feature extraction, recognition. The paper extracts and uses features such as Contrast, Homogeneity, Energy and Entropy for comparing two different signature images.

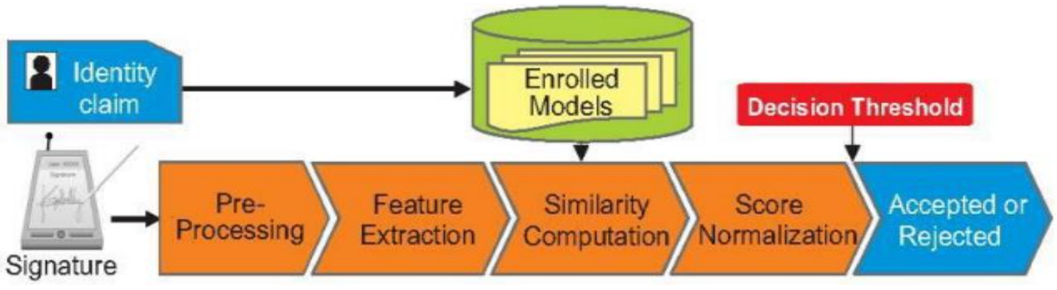


Figure 8: Flowchart of paper [17]

SIFT and a SURF algorithm which is used for enhanced offline signature recognition is used in paper [18]. This process, Bag-of-word features, was operated by making vector quantization technique, which outlined the key points for each training image inside a unified dimensional histogram. Accuracy obtained is 98.75%. SVM classification has limitations in speed and size while both phase of try and test of the algorithm and the chosen of the kernel function parameters.

A writer independent offline handwritten signature verification model, also known as global model, for signature verification is proposed in the paper [19]. Otsu's thresholding is used for pre-processing the image. System uses Local Binary Pattern based feature vector extraction along with its variants and classifies the images using SVM. An accuracy of 95.75% is received and the results shown are promising.

An offline signature verification using neural network is projected in paper number [20], where the signature is written on a paper are obtained using a scanner or a camera captured and presented in an image format. In pre-processing, color to grayscale, and finally to black and white, Resizing the image, thinning. Features extracted include Eccentricity, Skewness, Kurtois, Orientation Entropy, Euler number, Solidity, Mean, Standard deviation. The classification is done using a Cascaded feed-forward back-propagation networks and recognition rate of 92.5% is obtained.

System converts a scanned signature to a shape form and Eigen-signature construction is proposed for extracting the feature vector from a shape formed signature. The test feature vector is said to belong to i^{th} class, if it possess minimum distance with i^{th} class sample when compared to other class samples Thus paper [21] shows accuracy of 91.40%. The pre-processing techniques include binarizing or thresholding. Noise is eliminated using a simple morphological filter, thinned. The test feature vector is said to belong to i^{th} class, if it possess minimum distance with i^{th} class sample when compared to other class samples.

The purpose in paper number [22] is to select relevant features from those features set. For doing that, researchers compute the importance score of each features using two methods: Information Gain Ratio and Correlation. Over 440 Histogram features, 550 Fresh features, 220 DCT features were extracted an accuracy obtained was 95.5%. Limitations seen are rotation normalization and time normalization. Once all the global features are extracted well, the next trick is ranking these features. Ranking score is obtained from Information Gain Ratio and Correlation. So the result obtained are two list of 1210 features set ranked in order of their Gain. From each list, the features set are then divided into sub features set having 10 first ranked features, 20 first ranked features, 30 first ranked features, and up to 1210 features. So each ranked list will produce sub features set of 121 features.

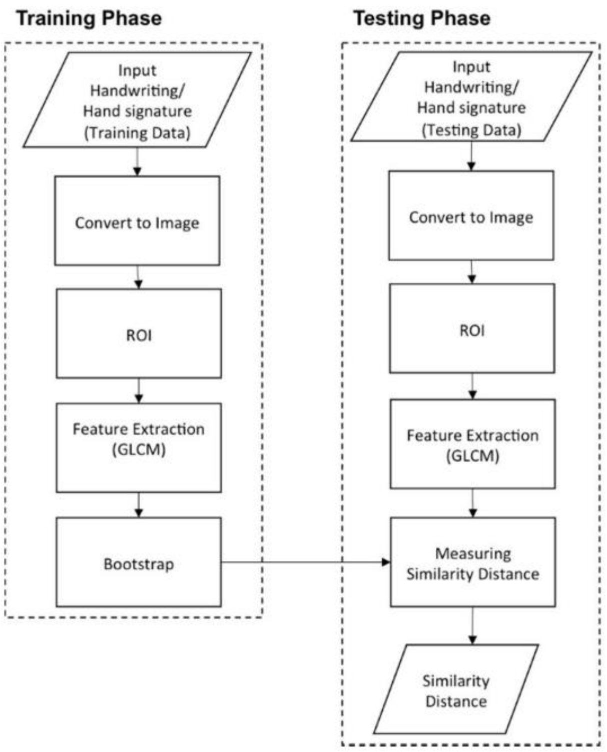

Training Phase
Testing Phase
Input Handwriting/ Hand signature (Training Data)
Convert to Image
ROI
Feature Extraction (GLCM)
Bootstrap
Input Handwriting/ Hand signature (Testing Data)
Convert to Image
ROI
Feature Extraction (GLCM)
Measuring Similarity Distance
Similarity Distance

Figure 9: Flow Diagram for Paper [23]

Signature and handwriting recognition on a mobile device using the Gray Level Co-occurrence Matrix (GLCM) for texture-based feature extraction and the bootstrap for performing single classifier model is proposed in the paper [23]. The accuracy obtained is 88.46%.

The paper [24] examines authentication systems based on handwritten signature and the main informative parameters of signature such as size, shape, velocity, pressure, etc. along with DCT, DFT. System using K-Nearest neighbors' algorithm and Random forest algorithm for classification and the accuracy of 95% is shown.

An offline signature recognition system which uses histogram of oriented gradients is presented. Pre-processing techniques used are color normalization, median filtering, angle normalization and exact bounding box, resized into 256×512 pixels. Feature extraction is done with Gradient Computation, Gradient Vote, and Normalization Computation. A three layered feedforward backpropagation neural network is used for classification and the recognition rate of paper [25] is 96.87%

In this paper [26], the trace transform based affine invariant features are applied for signature verification. The diametric and trace functions are appropriately chosen to compute a set of circus functions from each signature image. Recognition rate of 76% is obtained. Low accuracy, more invariant functional will be designed for usefulness in signature verification.

This paper [27], deals with the analysis of discriminative powers of the features that can be extracted from an on-line signature, how it's possible to increase those discriminative powers by Dynamic Time Warping as a step in the pre-processing of the signal coming from the tablet. The accuracy obtained is 99.7%. Pre-processing is done using Filtering, equispacing by Linear Interpolation, Normalization, DTW Alignment, Derived Signals: Speed and Acceleration The main processing includes init minus min End minus min Average Root Square Average Time over 0 Crossing 0 Mean over 0 Standard deviation Number of traces, Time of signature, Writing Time of signature / Time of Signature, Height / Width, Area, Length.

Paper [28] uses a texture base approach called the Local binary Pattern algorithm along with SVM, The signature models are trained with genuine signatures on white background and tested with other genuine and forgeries mixed with different backgrounds. Results depict that a basic version of local binary patterns (LBP) or local directional and derivative patterns are more robust than others like GLCM features to the grey level distortion with SVM with histogram oriented kernels as a classifier or rotation invariant uniform LBP. The proposed configurations are checked and evaluated under different situations and conditions: changing the number of training signature images, database with different inks, multiple signing sessions, increasing the number of signers and combining different features Results have also been provided when looking for the signature in the check and segmenting it automatically. In all these cases, the best results were obtained with the LDerivP feature set, which improve the results obtained in the predefined baseline, showing quite significant improvements with fictitious signature images.

Paper [29] uses Conic section function neural network (CSFNN) circuitry was designed for offline signature recognition. CSFNN is a unified framework for multilayer perceptron (MLP) and radial basis function (RBF) networks and the size of feature vectors was not suitable for designed CSFNN chip structure owing to the input limitations of the circuit. Pre-processing is done using noise reduction algorithm, skeletonization. The few main disadvantages of analog systems include its sensitivity to ambient noise and to temperature. Accuracy shown is 95.5%.

This paper [30] explores the usefulness of local binary pattern (LBP) and local directional pattern (LDP) texture measures to discriminate off-line signatures. Comparison between these several texture normalizations is generated in order to look for reducing pen dependence. The recognition rate is 91.92%. The pre-processing techniques include binarized, cropped, or-exclusive operation, Texture histogram normalization. Least Squares Support Vector Machine (LS-SVM) is applied for classification.

The goal of this study [31] is to investigate the effect of combining transform features to authenticate signatures. Due to genuine human error and lack of consistency, comparing signatures requires pre-processing to assist with standardization. An EER of

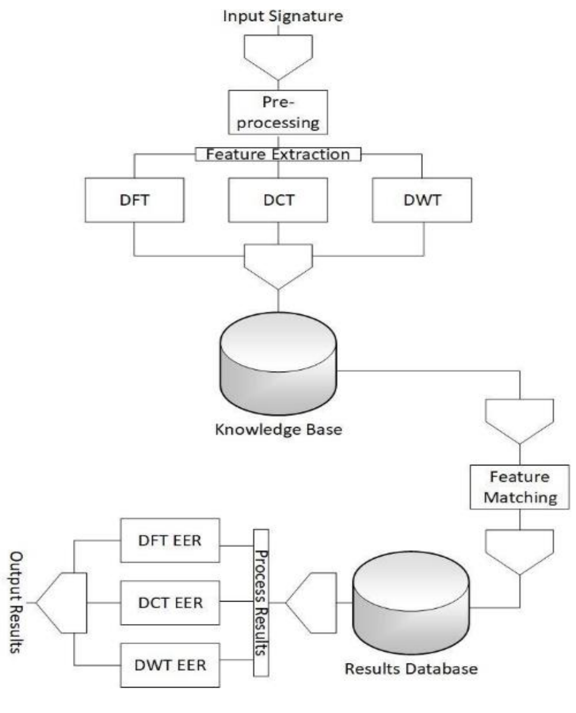


Figure 10: Flowchart of system in paper [31]

2.46% which means a recognition rate of 99.40% is shown by the system. The features extracted include x coordinate, y coordinate, pen pressure, azimuth, altitude, DFT, DCT and DWT. Classification is done with Fast Dynamic Time Warping (FastDTW) algorithm

In this paper [32], gender discrimination has been proposed by feature extraction method. The proposed framework considers handwritten Hindi signature of each individuals as an input for gender detection. Recognition rate obtained is 92.90%.

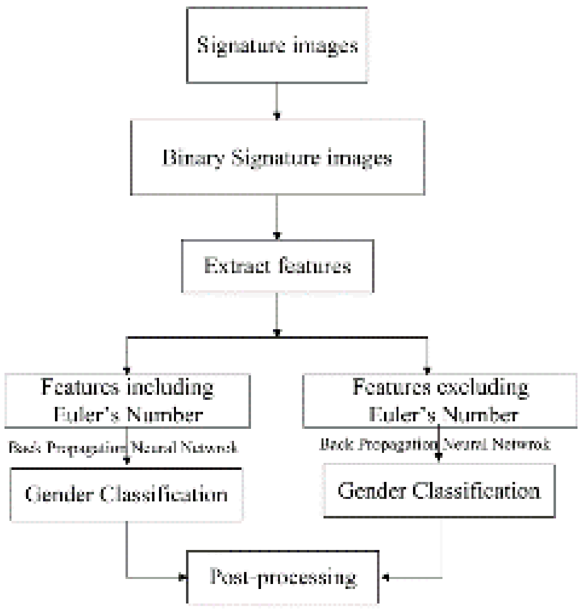

Signature images
Binary Signature images
Extract features
Features including Euler's Number
Features excluding Euler's Number
Back Propagation Neural Network
Back Propagation Neural Network
Gender Classification
Gender Classification
Post-processing

Figure 11: Flow chart of paper [32]

In order to solve the shortcomings of manual identification in technical accuracy and subjectivity, this paper [33] proposed an off-line signature identification method based on Support Vector Machine (SVM). Accuracy shown here is 85.00%. Dataset are stained by useless border. Other pre-processing techniques used in the system include Image binarization, denoising, removing blank margins. Features include global feature like height, width, area and slant angle and sum of black pixels. Identification rates within and between writing systems is prone to swing in some degree under different partitioning strategy.

This paper [34] titled presents a set of geometric signature features for offline automatic Signature verification based on the description of the signature envelope and the interior stroke distribution in polar and Cartesian coordinates. Feature extraction method includes Outline Detection and Representation, Feature Vector Based on Polar Coordinates, Feature Vector Based on Cartesian Coordinates. Classification is performed using The HMM Signature Model, Support Vector Machine Signature Model and Euclidean Distance-Based Signature Model.

The proposed system [35] segments each signature curve based on pen's velocity value. The signature curve, would be decomposed in low or high partition according to velocity's value. For each partition, hand movement direction between two consequent point Extracted. 95.10%. Pre-processing technique here are removing noise, translation invariance, rotation 7 scale invariance. Signature features extracted and used shape of signature would be partitioned based on dynamic feature such as velocity or pressure. Classification is done using Hidden Markov Model.

Optical flow is used to define a stability model of the genuine signatures for each signer in paper [36]. Stability between the unknown signature and reference signatures is estimated and consistency with the stability model of the signer is evaluated. Accuracy obtained is 96.00%. In pre-processing, signature image size was adjusted to a fixed area. Optical flow vectors were used for the analysis of the local stability of a signer to detect the stable regions in the signature. Optical flow vectors are for the analysis of the local stability of a signer to detect the stable regions in the signature.

In this paper [37] an efficient off-line signature verification method based on an interval symbolic representation and a fuzzy similarity measure is proposed. 88.26% Local Binary Pattern, interval-valued symbolic model. A histogram-based threshold technique, mean filter for noise removal, minimum bounding boxes. Classification of the signature image is done based on the similarity value, an adaptive writer-dependent acceptance threshold

A new approach of showing online signatures by interval-valued symbolic features. The online signature also gives global features that are to be extracted and used to form an interval-valued feature vectors. Methods for signature verification and recognition based on the symbolic representation are also proposed in paper [38] Recognition rate is 95.40%.

This paper [39] explores the utility of information derived from the dynamic time warping (DTW) cost matrix for the problem of online signature verification. The prior research and experimentations in literature primarily utilize only the DTW scores to authenticate a test signature. Accuracy measured is 97.24%. As the paper itself suggests local features & histogram representation shall be an interesting extension.

This paper [40] presents a new approach for online signature verification that exploits the potential of local stability information in handwritten signatures. Different from previous models, this approach classifies a signature using a multi domain strategy. Accuracy obtained is 97.90%. Both variable and stable regions of a signature image object could be considered for supporting the advanced personalized techniques in signature verification and recognition, since probably, from a behavioural point of view, a variability model of a signer/author could also be very complementary and informative to a stability model.

5. Proposed work (algorithm/ techniques/ experimental setup)

5.1 Algorithm/technique to be used

The proposed work intends to stretch and extend one of the papers discussed in the above section of literature review which includes the features extraction of Local Binary Pattern from the signature images and to generate an efficient offline signature recognition and verification system. This algorithm is also to be combined with some simple features of the signature image that can have local features that define features of a part or parts of the signature as well as the global features which requires signature as a whole.

At architectural level the basic structure and flow of the system is very much similar to how most of the research finds out. The flowchart block diagram would start with image acquisition where the image would be taken in as input. Then the images would go through series of pre-processing stages such as Grayscale, Binarization, BoundaryBox and noise removal filters. In feature extraction stage the main focus would be to use LBP variant algorithm along with some simple image features. The generated feature set of all the images along with their respective classes would be given to a classification algorithm. This may include Support vector machine, Artificial Neural network, K-nearest neighbors or Euclidian distance based approach or the combination of them. Finally this stage would give out the class to which the input test image belongs to as output decision.

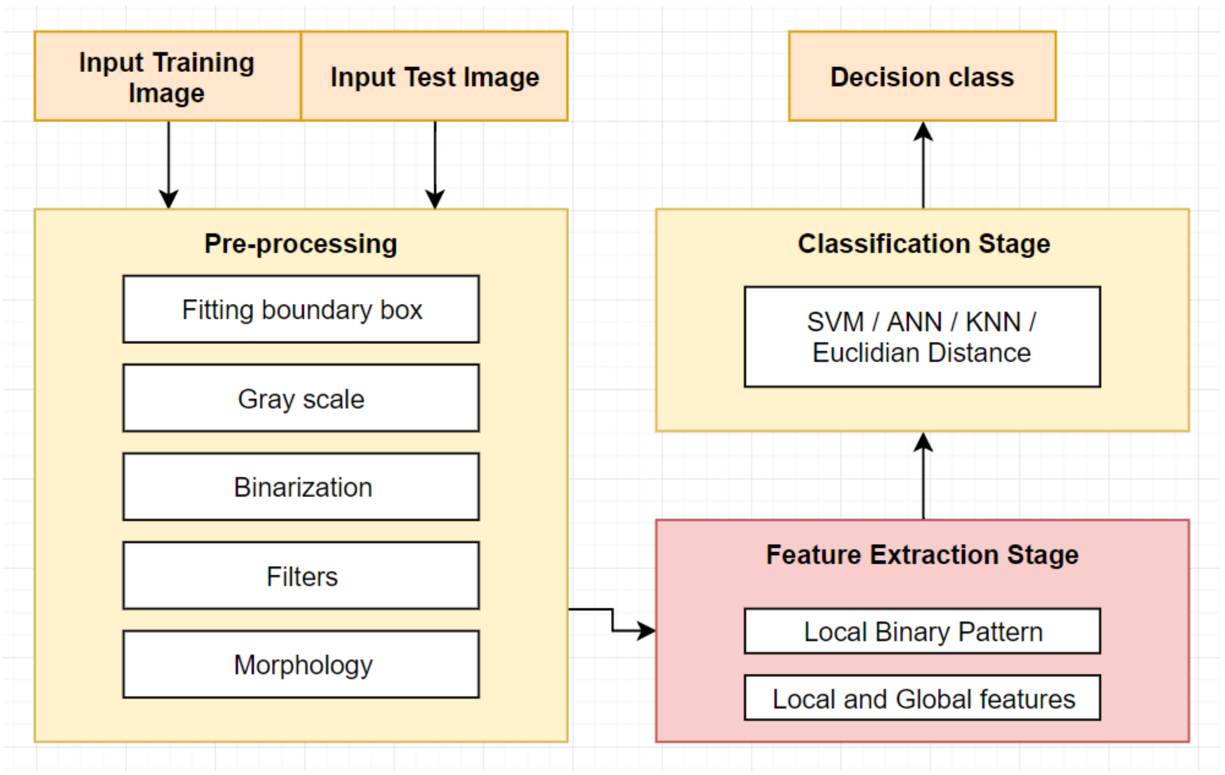


Figure 12: Block diagram of the proposed approach

5.2 Implementation tools end setup

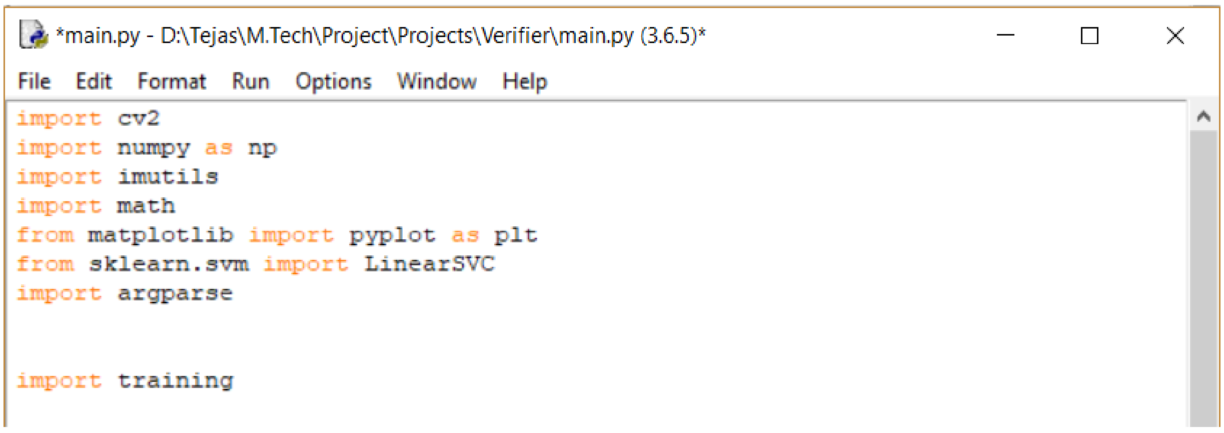
Implementation programming tool to be used here is Python. Python is powerful... and fast; plays well with others, runs everywhere, is friendly & easy to learn and is Open. It is a popular programming language used in web development (server-side), software development, mathematics, system scripting.

- Python can be easy to pick up whether you're a first time programmer or you're experienced with other languages. The following pages are a useful first step to get on your way writing programs with Python!
- The community hosts conferences and meetups, collaborates on code, and much more. Python's documentation will help you along the way, and the mailing lists will keep you in touch.
- Python is developed under an OSI-approved open source license, making it freely usable and distributable, even for commercial use. Python's license is administered by the Python Software Foundation.
- The Python Package Index (PyPI) hosts thousands of third-party modules for Python. Both Python's standard library and the community-contributed modules allow for endless possibilities.

Contains number of libraries, which are easy to install & import...

- openCV : *Computer vision and machine learning software library*.
- numPy : *Scientific computing & array-processing*
- Imutils : *Functions to make basic image processing functions easier*
- Math : *Provides access to the mathematical functions*
- MatplotLib : *Python 2D plotting library*
- sklearn.svm.linearSVC : *Linear Support Vector Classification*.
- Argparse : *Python command-line parsing library*

Since using this programming tool with along with these libraries it is essential to download and install them. So following figure shows the libraries included in the Python's editor tool called IDLE.


main.py - D:\Tejas\M.Tech\Project\Projects\Verifier\main.py (3.6.5)
File Edit Format Run Options Window Help

```
import cv2
import numpy as np
import imutils
import math
from matplotlib import pyplot as plt
from sklearn.svm import LinearSVC
import argparse


import training
```

Figure 13: Necessary packages

6. Work done after topic approval

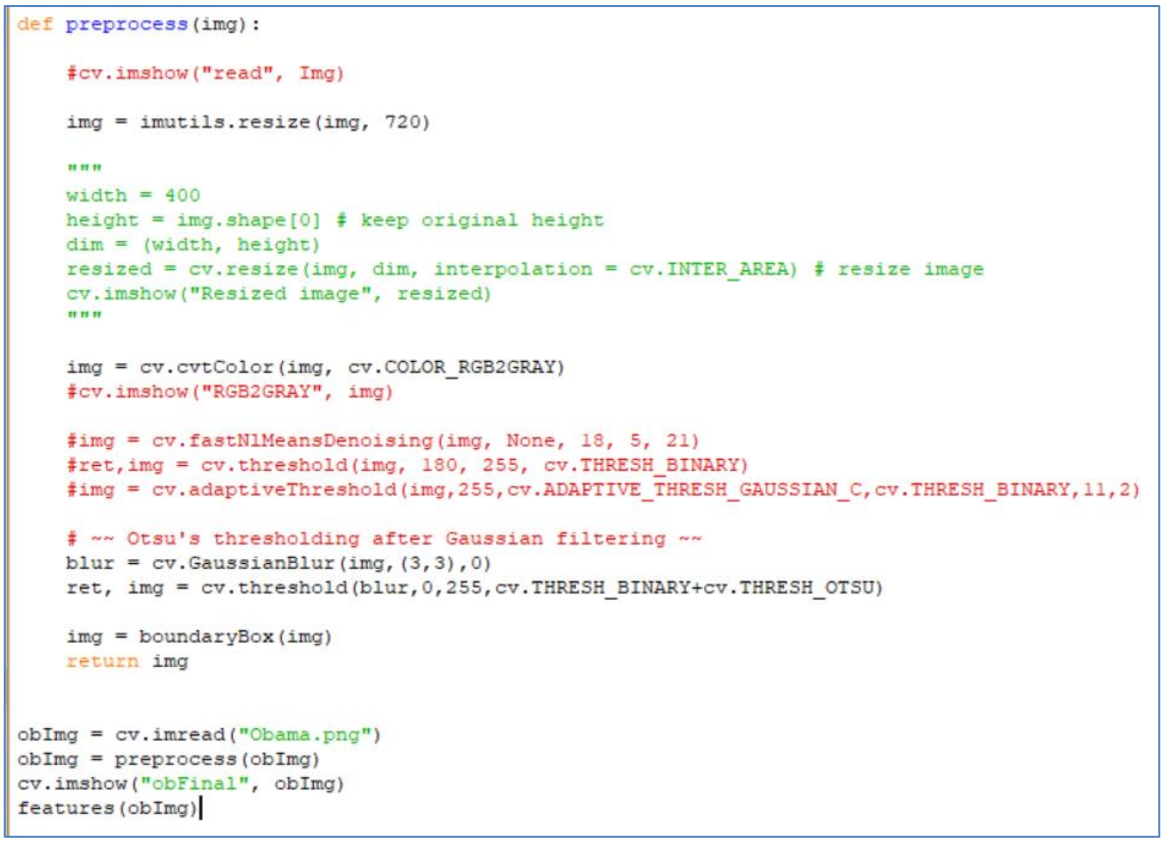
6.1 Research

The work done after topic approval includes research of the related research papers to explore more and more about in depth knowledge of the existing algorithms that can be used for signature recognition and verification. The research allowed me to finalize one of the algorithm that can be used during my research which is the Local Binary Pattern based approach. Along with finalization of the algorithm, I have also studied over the implementation and programming tools that can be used for working with images on pixel level. Thus I found many powerful tools like MATLAB. But the one that caught my eye is Python and thus I have finalized the implementation tool as well. About the language and its features are given in detail in link provided in the references [41].

6.2 Implementation

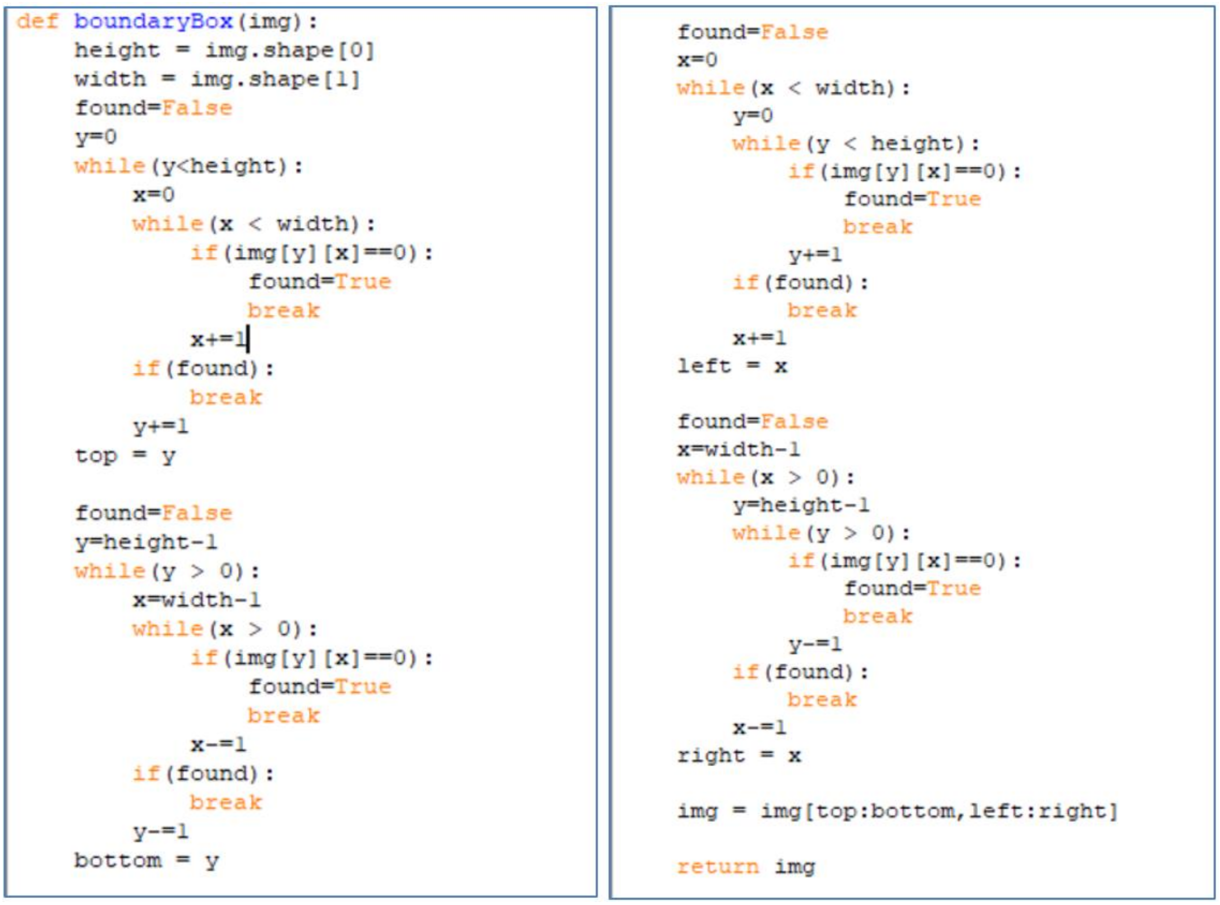
Implementing with python include some basic preprocessing functions to be used and applied over a sample image, to get use to with the process and to learn the few of many pre-defined function that Python and its library packages allow us to do.

Preprocessing stage that have been implemented include reading the image resizing, grayscale, denoising, threshold based segmentation, boundary box generation displaying the image



```
def preprocess(img):

    #cv.imshow("read", Img)

    img = imutils.resize(img, 720)

    """
    width = 400
    height = img.shape[0] # keep original height
    dim = (width, height)
    resized = cv.resize(img, dim, interpolation = cv.INTER_AREA) # resize image
    cv.imshow("Resized image", resized)
    """

    img = cv.cvtColor(img, cv.COLOR_RGB2GRAY)
    #cv.imshow("RGB2GRAY", img)

    #img = cv.fastNlMeansDenoising(img, None, 18, 5, 21)
    #ret,img = cv.threshold(img, 180, 255, cv.THRESH_BINARY)
    #img = cv.adaptiveThreshold(img,255,cv.ADAPTIVE_THRESH_GAUSSIAN_C,cv.THRESH_BINARY,11,2)

    # ~~ Otsu's thresholding after Gaussian filtering ~~
    blur = cv.GaussianBlur(img,(3,3),0)
    ret, img = cv.threshold(blur,0,255,cv.THRESH_BINARY+cv.THRESH_OTSU)

    img = boundaryBox(img)
    return img


obImg = cv.imread("Obama.png")
obImg = preprocess(obImg)
cv.imshow("obFinal", obImg)
features(obImg)
```

Figure 14: Preprocessing stages on python



```python
def boundaryBox(img):
    height = img.shape[0]
    width = img.shape[1]
    found=False
    y=0
    while(y<height):
        x=0
        while(x < width):
            if(img[y][x]==0):
                found=True
                break
            x+=1
        if(found):
            break
        y+=1
    top = y

    found=False
    y=height-1
    while(y > 0):
        x=width-1
        while(x > 0):
            if(img[y][x]==0):
                found=True
                break
            x-=1
        if(found):
            break
        y-=1
    bottom = y
```

```python
    found=False
    x=0
    while(x < width):
        y=0
        while(y < height):
            if(img[y][x]==0):
                found=True
                break
            y+=1
        if(found):
            break
        x+=1
    left = x

    found=False
    x=width-1
    while(x > 0):
        y=height-1
        while(y > 0):
            if(img[y][x]==0):
                found=True
                break
            y-=1
        if(found):
            break
        x-=1
    right = x

    img = img[top:bottom,left:right]

    return img
```

Figure 15: Boundary box cropping

The boundary box cropping is one of the preprocessing stages that wasn't a predefined function so I got the liberty to create it myself. Its general idea is to remove the white spaces from all four sides of the image and make the problem a little smaller. The idea here was to simply keep cropping out all the rows until we find the first black pixel from top and bottom and do the same for columns from both the sides.

After few of the preprocessing stages the signature image would become a lot cleaner without noise and ready to be used for further processing. Following image shows the same.

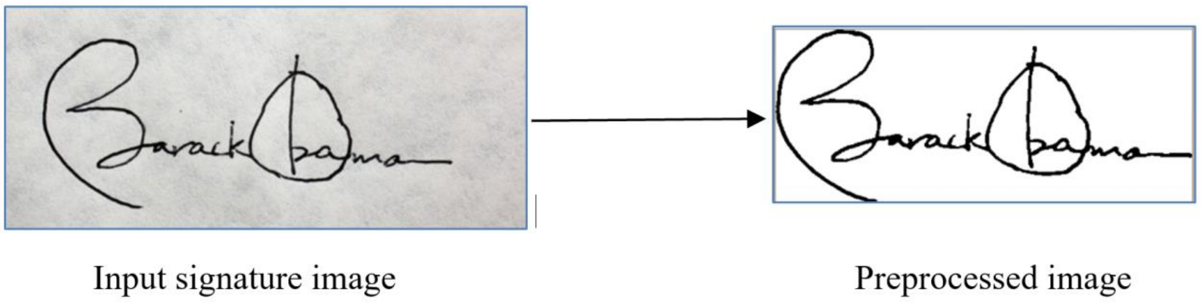


Figure 16: Preprocessing signature image

6.3 Dataset preparation

The dataset preparation stage could include preparing the data myself by taking signature of people around me or it could be much easier by simply downloading the online available datasets that people around the world generate and upload for others to use. I have thus chose to go for the second option and download the dataset from the Web.

The data set I have taken from one of the websites with the uploaded by researchers online on the website mentioned in [15]. Since the dataset downloaded is vast and contains a lot of images I have prepared a subset of the vast dataset and prepared the following set of images:

Author A

1. 24 genuine = 16 training + 8 test
2. 24 forged = 16 training + 8 test

Author B

3. 24 genuine = 16 training + 8 test
4. 24 forged = 16 training + 8 test

Author C

5. 24 genuine = 16 training + 8 test
6. 24 forged = 16 training + 8 test

Total 96 training images & 48 testing images are prepared for three different authors/writers of 6 different classes. These classes correspond to the same classes that are classification algorithm will give us as output.

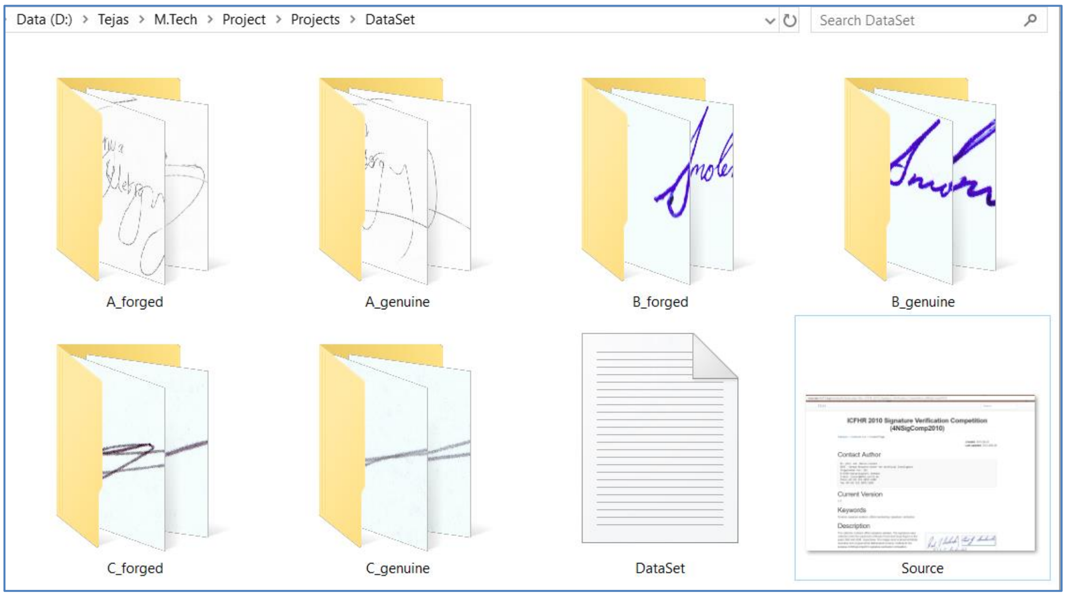


Figure 17 : Prepared dataset

7. Work done after Test-1 (M-1)

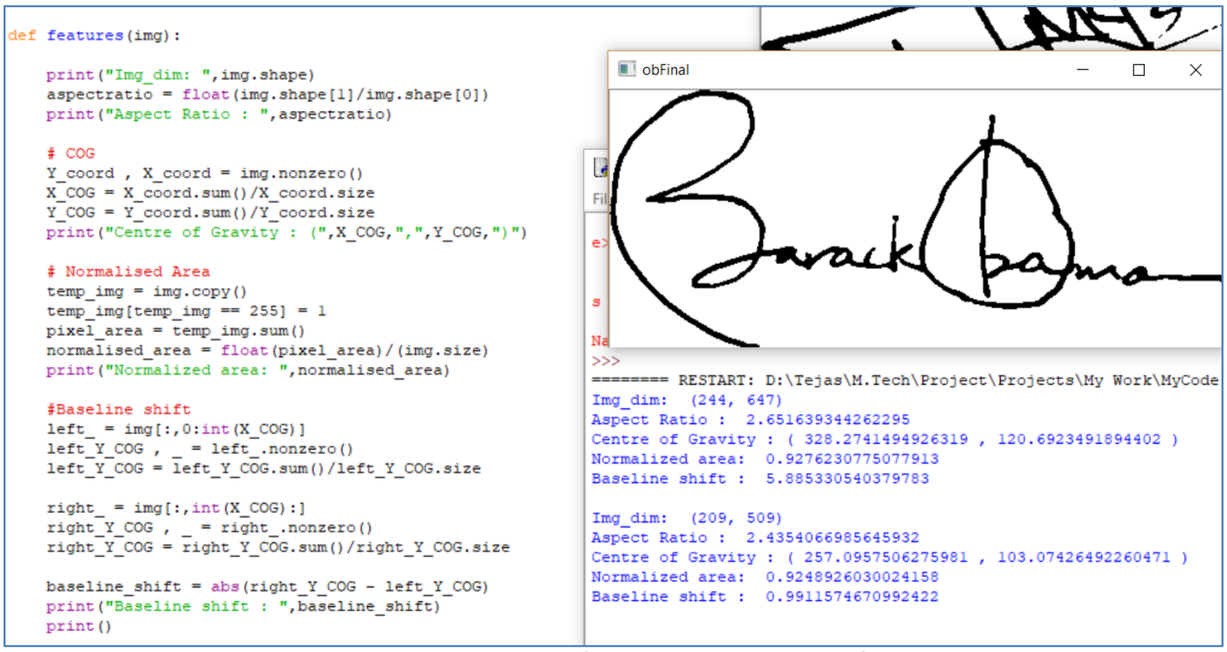
7.1 Research

The work done after Test-1 (M-1) includes further research of the related research papers to explore more and more about in depth knowledge of the existing algorithms that can be used for signature recognition and verification. The research allowed me to read more research papers on one of the algorithm that I have finalized to use, Local Binary Pattern based approach. Along with finalization of the algorithm, I have read and also studied more number of research papers based on Signature Recognition and Verification and an excel sheet of all the Literature Review has been prepared The literature review contains total of 40 research papers based on the topic Signature Recognition. Most of the papers make use of 3 stages: Preprocessing stage, Feature extraction stage, Classification stage

7.2 Implementation

Implementing part done after test-1 (M1) with python includes basic functions to be used and applied over the set of signature images, to get use to with the process and to learn the few of many pre-defined function that Python and its library packages allow us to do.

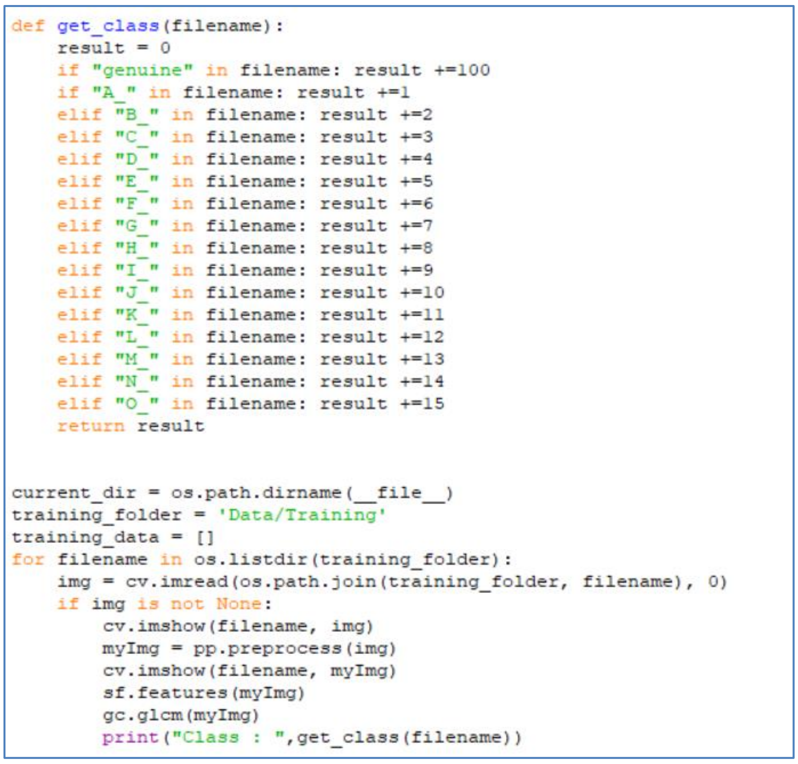
Implementation of Feature extraction stage is also done in this system. A separate file namely ‘simpleFeat.py’ has the code shown in figure 18. The code extracts simple shape based features like Aspect ratio, Centre of Gravity, Normalized area, Baseline shift. The output for the same is also shown in the same figure 18. The features extracted are different for different images and therefore will allow us or rather will allow the system to differentiate among different images.



```
def features(img):

    print("Img_dim: ",img.shape)
    aspectratio = float(img.shape[1]/img.shape[0])
    print("Aspect Ratio : ",aspectratio)

    # COG
    Y_coord , X_coord = img.nonzero()
    X_COG = X_coord.sum()/X_coord.size
    Y_COG = Y_coord.sum()/Y_coord.size
    print("Centre of Gravity : (",X_COG,",",Y_COG,")")

    # Normalised Area
    temp_img = img.copy()
    temp_img[temp_img == 255] = 1
    pixel_area = temp_img.sum()
    normalised_area = float(pixel_area)/(img.size)
    print("Normalized area: ",normalised_area)

    #Baseline shift
    left_ = img[:,0:int(X_COG)]
    left_Y_COG , _ = left_.nonzero()
    left_Y_COG = left_Y_COG.sum()/left_Y_COG.size

    right_ = img[:,int(X_COG):]
    right_Y_COG , _ = right_.nonzero()
    right_Y_COG = right_Y_COG.sum()/right_Y_COG.size

    baseline_shift = abs(right_Y_COG - left_Y_COG)
    print("Baseline shift : ",baseline_shift)
    print()
```

```
>>>
======== RESTART: D:\Tejas\M.Tech\Project\Projects\My Work\MyCode
Img_dim:  (244, 647)
Aspect Ratio :  2.651639344262295
Centre of Gravity : ( 328.2741494926319 , 120.6923491894402 )
Normalized area:  0.9276230775077913
Baseline shift :  5.885330540379783

Img_dim:  (209, 509)
Aspect Ratio :  2.4354066985645932
Centre of Gravity : ( 257.0957506275981 , 103.07426492260471 )
Normalized area:  0.9248926030024158
Baseline shift :  0.9911574670992422
```

Figure 18: Extraction of Simple shape based features

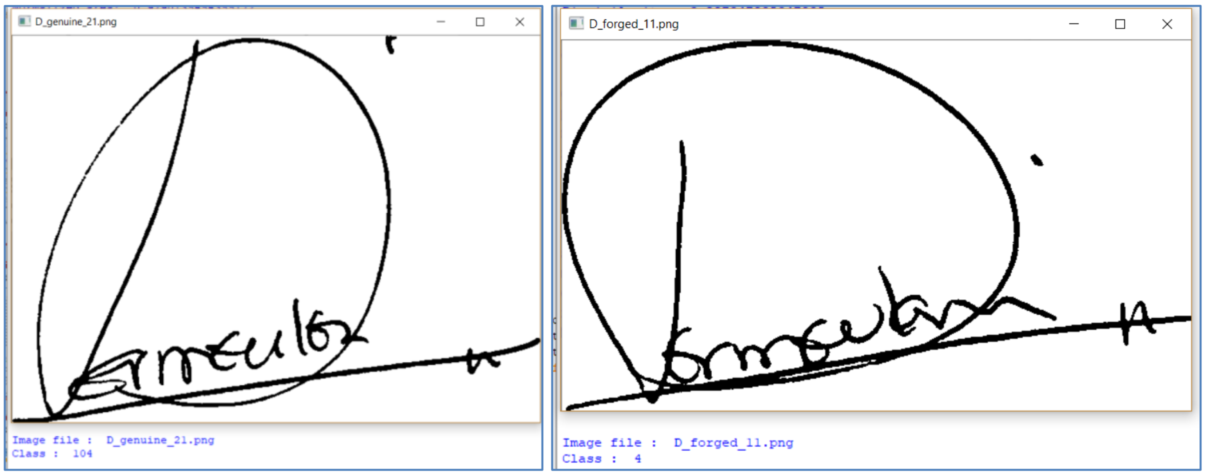


```python
def get_class(filename):
    result = 0
    if "genuine" in filename: result +=100
    if "A_" in filename: result +=1
    elif "B_" in filename: result +=2
    elif "C_" in filename: result +=3
    elif "D_" in filename: result +=4
    elif "E_" in filename: result +=5
    elif "F_" in filename: result +=6
    elif "G_" in filename: result +=7
    elif "H_" in filename: result +=8
    elif "I_" in filename: result +=9
    elif "J_" in filename: result +=10
    elif "K_" in filename: result +=11
    elif "L_" in filename: result +=12
    elif "M_" in filename: result +=13
    elif "N_" in filename: result +=14
    elif "O_" in filename: result +=15
    return result


current_dir = os.path.dirname(__file__)
training_folder = 'Data/Training'
training_data = []
for filename in os.listdir(training_folder):
    img = cv.imread(os.path.join(training_folder, filename), 0)
    if img is not None:
        cv.imshow(filename, img)
        myImg = pp.preprocess(img)
        cv.imshow(filename, myImg)
        sf.features(myImg)
        gc.glcm(myImg)
        print("Class : ",get_class(filename))
```

Figure 19: Classifying training data set code

The learning of the system would be dependent heavily on a supervised base learning, which means that the training set images must have the class information in some form. Here I have renamed the training set images in such a way that the name will define the class it belongs to. The python code within the main .py file will first read all the images in the given directory and get the class of each based on its name. Since we have 15 authors each having forged and genuine signatures, we currently have a total of 30 classes.

For example, in the below two images have two different names since they belong to two different classes. If the class calculated is a 3 digit number above 100 it means the signature is genuine and the other two digits define the author of the training image. We could also later on binarize the classification number to reduce complexity.



```
Image file :  D_genuine_21.png
Class :  104
```

```
Image file :  D_forged_11.png
Class :  4
```

Figure 18: Classifying training data set output

7.3 Dataset preparation

The work done after Test-1 (M-1) also includes dataset preparation include downloading more and more of the online available datasets that people around the world generate and upload for others to use. I have thus downloaded the dataset from the Web.

The data set I have taken from one of the websites with the uploaded by researchers online on the website mentioned in References *[42]*. Since the dataset downloaded is vast and contains a lot of images I have prepared a subset of the vast dataset and kept aside directory of folders of images:

Total 15 Authors, all having genuine as well as forged signatures. The dataset is divided in an approximate ratio of 2:1 for preparing training and testing data respectively. Therefore 376 training images and 190 testing images now combine to a total of 566 images

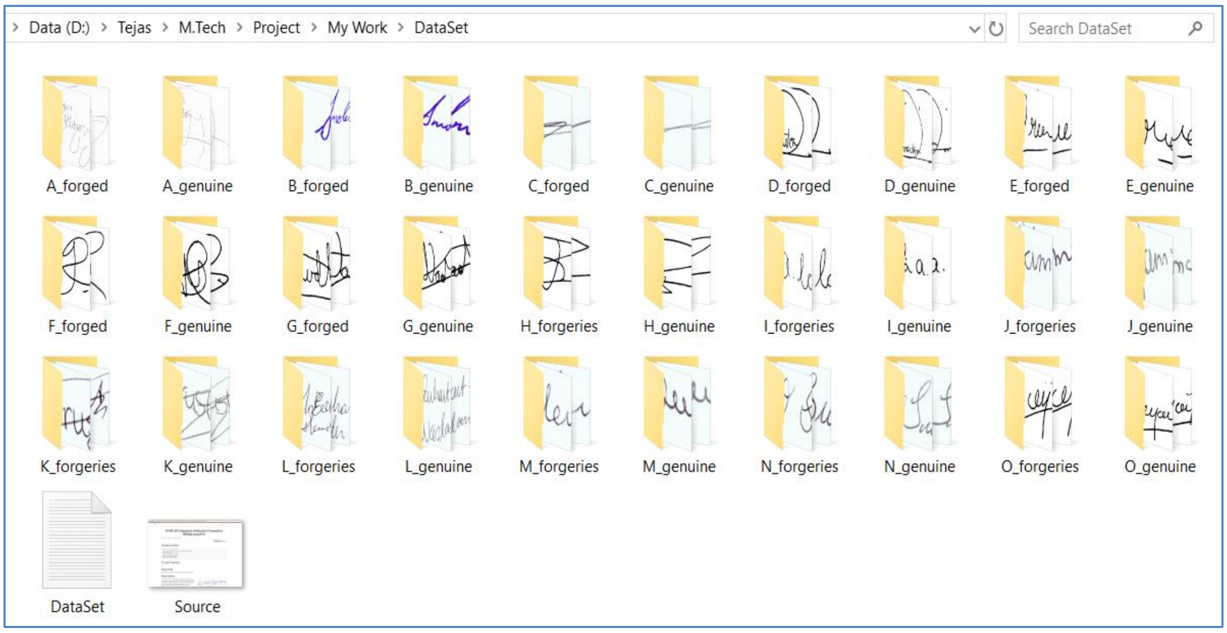
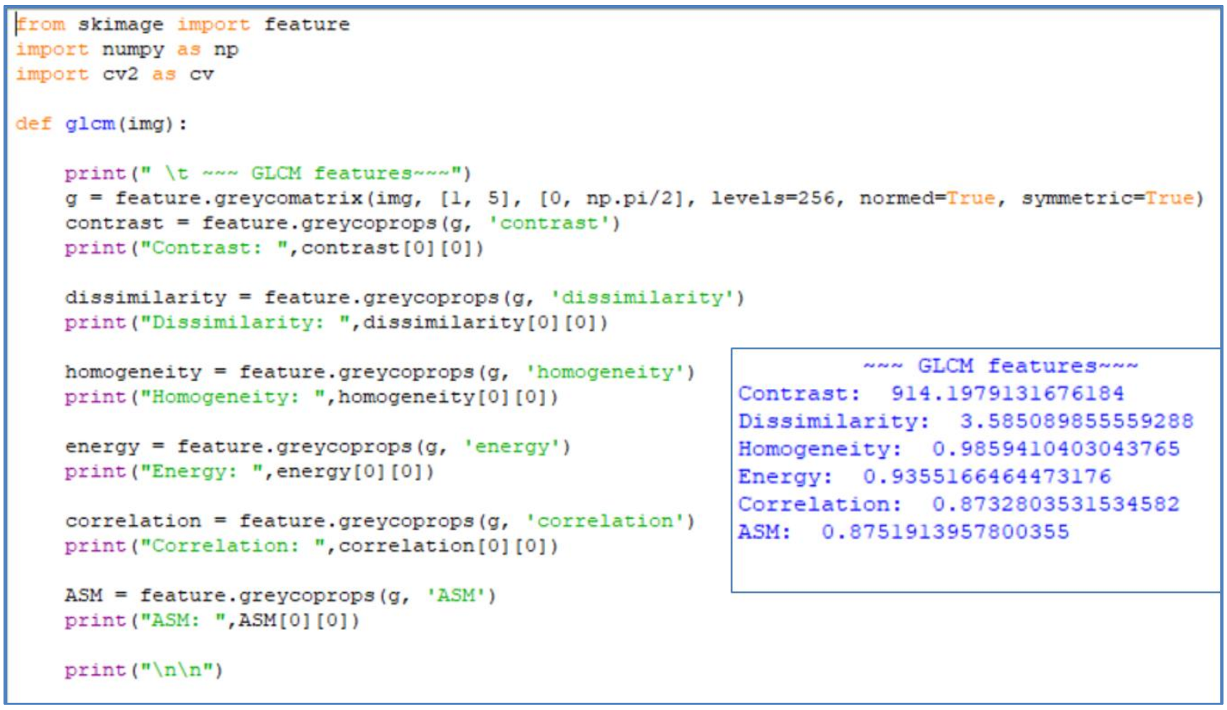


Figure 21: Prepared dataset

8. Work done after Test-2 (M-2)

8.1 Implementation

A statistical method of examining texture that considers the spatial relationship of pixels is the gray-level co-occurrence matrix (**GLCM**), also known as the gray-level spatial dependence matrix. The skimage package of Python allows us to extract gray level co-occurrence matrix based features using the feature class methods and makes our task simpler. The above code shown in figure 12 has GLCM feature extraction which includes contrast, Dissimilarity, Homogeneity, Energy, Correlation and ASM. The output is also shown in the same figure which has numerical values for each image passed to the function.



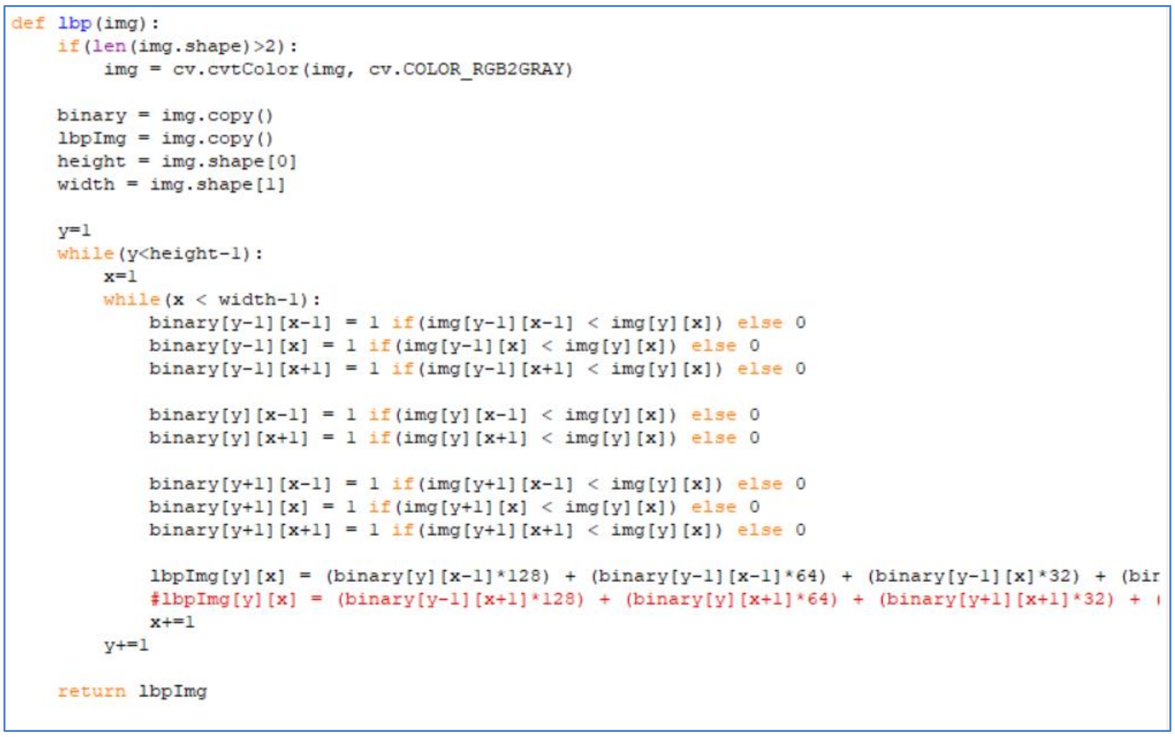
```
from skimage import feature
import numpy as np
import cv2 as cv

def glcm(img):

    print(" \t ~~~ GLCM features~~~")
    g = feature.greycomatrix(img, [1, 5], [0, np.pi/2], levels=256, normed=True, symmetric=True)
    contrast = feature.greycoprops(g, 'contrast')
    print("Contrast: ",contrast[0][0])

    dissimilarity = feature.greycoprops(g, 'dissimilarity')
    print("Dissimilarity: ",dissimilarity[0][0])

    homogeneity = feature.greycoprops(g, 'homogeneity')
    print("Homogeneity: ",homogeneity[0][0])

    energy = feature.greycoprops(g, 'energy')
    print("Energy: ",energy[0][0])

    correlation = feature.greycoprops(g, 'correlation')
    print("Correlation: ",correlation[0][0])

    ASM = feature.greycoprops(g, 'ASM')
    print("ASM: ",ASM[0][0])

    print("\n\n")
```

```
         ~~~ GLCM features~~~
Contrast:  914.1979131676184
Dissimilarity:  3.585089855559288
Homogeneity:  0.9859410403043765
Energy:  0.9355166464473176
Correlation:  0.8732803531534582
ASM:  0.8751913957800355
```

Figure 22: Feature extraction using GLCM

The main texture based features are extracted from the Local Binary Pattern image. The LBP image generation code is designed and ready to be used for texture based feature extraction. The LBP image generation function coded in python is shown in figure 13. The LBP image generation algorithm works in iterative manner running a mask of odd number of pixels over the image. The proposed system is currently using 3x3 mask. In the mask the center pixel is compared with all the surrounding pixels in the mask. The higher value is replaced with 0 and the lower value is replaced with 1. Then the pixels are read in clockwise manner to form a binary number. This binary number is converted to decimal number to be then replaced by the center pixel. The mask is now moved forward in x direction and when the row is completed next row is taken for processing by increasing moving the mask in y direction.

The output of the LPB image generation function is a darker image which is shown in figure 14. The darker image formed show out the texture out of the image and allow us or shall rather allow the system to take out the texture based features more easily.



```
def lbp(img):
    if(len(img.shape)>2):
        img = cv.cvtColor(img, cv.COLOR_RGB2GRAY)

    binary = img.copy()
    lbpImg = img.copy()
    height = img.shape[0]
    width = img.shape[1]

    y=1
    while(y<height-1):
        x=1
        while(x < width-1):
            binary[y-1][x-1] = 1 if(img[y-1][x-1] < img[y][x]) else 0
            binary[y-1][x] = 1 if(img[y-1][x] < img[y][x]) else 0
            binary[y-1][x+1] = 1 if(img[y-1][x+1] < img[y][x]) else 0

            binary[y][x-1] = 1 if(img[y][x-1] < img[y][x]) else 0
            binary[y][x+1] = 1 if(img[y][x+1] < img[y][x]) else 0

            binary[y+1][x-1] = 1 if(img[y+1][x-1] < img[y][x]) else 0
            binary[y+1][x] = 1 if(img[y+1][x] < img[y][x]) else 0
            binary[y+1][x+1] = 1 if(img[y+1][x+1] < img[y][x]) else 0

            lbpImg[y][x] = (binary[y][x-1]*128) + (binary[y-1][x-1]*64) + (binary[y-1][x]*32) + (bir
            #lbpImg[y][x] = (binary[y-1][x+1]*128) + (binary[y][x+1]*64) + (binary[y+1][x+1]*32) +
            x+=1
        y+=1

    return lbpImg
```

Figure 23: Feature extraction using LBP

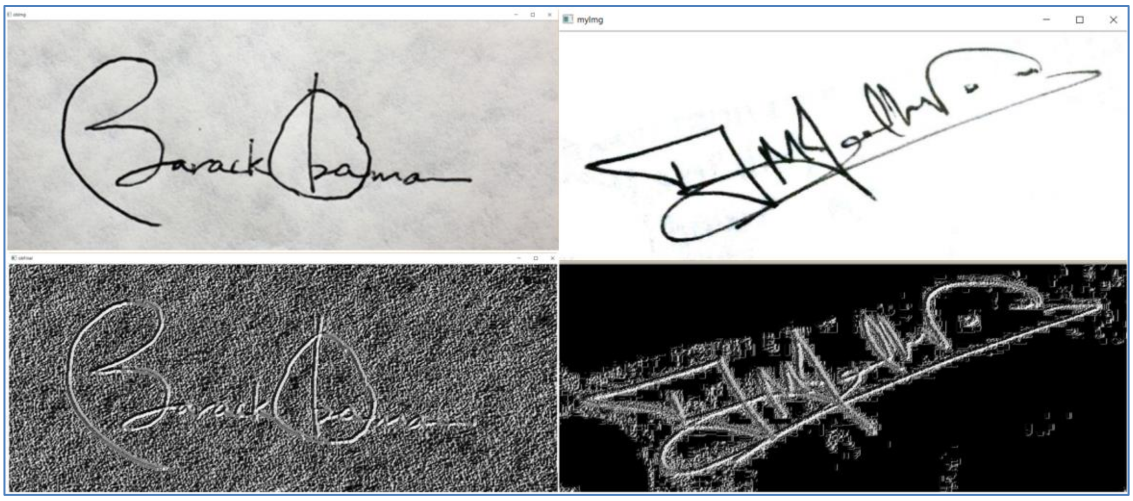
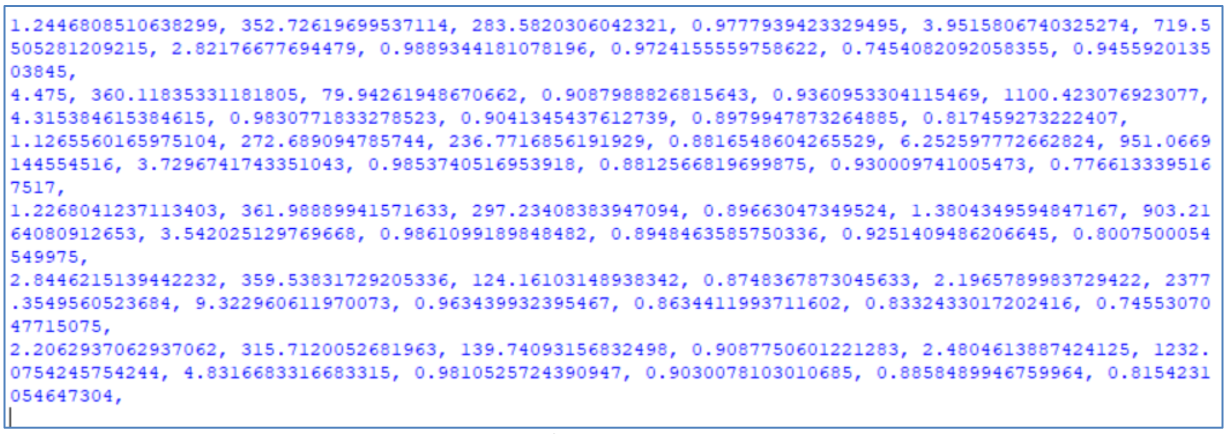


Figure 24: Output of Feature extraction using LBP

The features extracted are to be stored in 2 dimensional variable here in the system named as 'featureVector' where each row has features for each image. Now all the features are then collected in this 2 dimensional array to be later used during classification of the test images. The Feature Vector output is generated for some images and is shown in the figure 25.



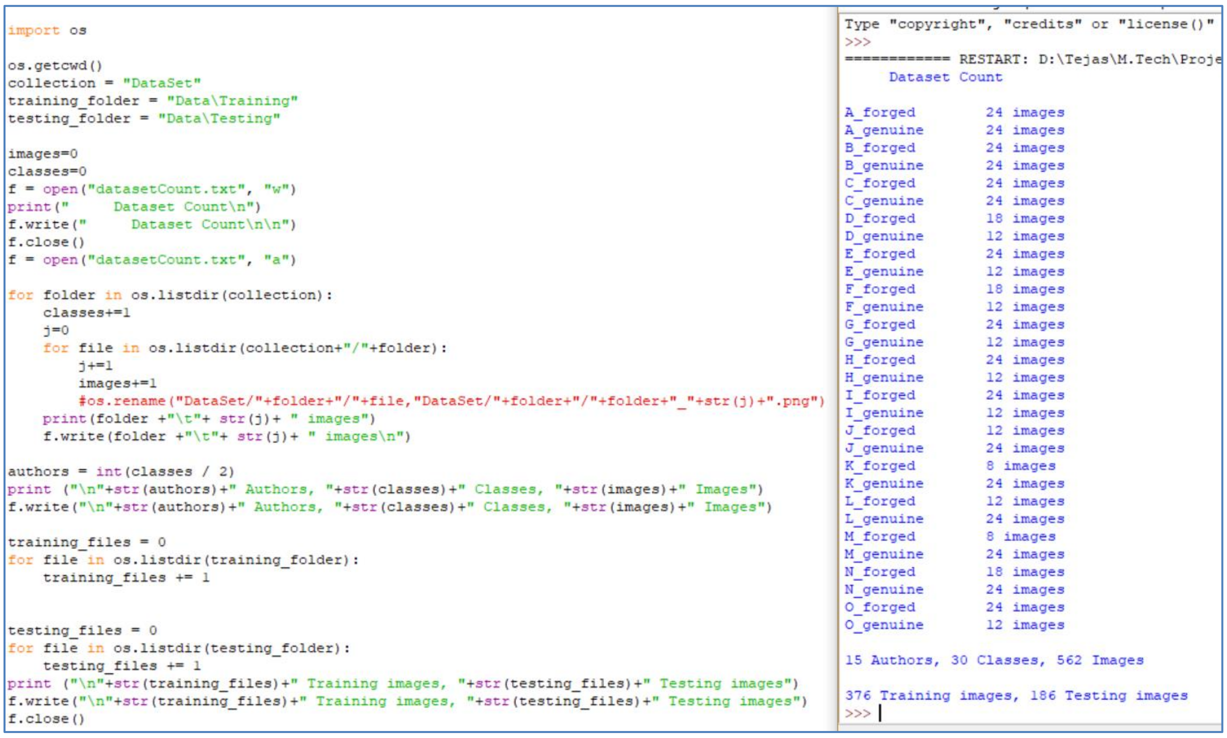
```
1.2446808510638299, 352.72619699537114, 283.5820306042321, 0.9777939423329495, 3.9515806740325274, 719.5
505281209215, 2.82176677694479, 0.9889344181078196, 0.9724155559758622, 0.7454082092058355, 0.9455920135
03845,
4.475, 360.11835331181805, 79.94261948670662, 0.9087988826815643, 0.9360953304115469, 1100.423076923077,
4.315384615384615, 0.9830771833278523, 0.9041345437612739, 0.8979947873264885, 0.817459273222407,
1.1265560165975104, 272.689094785744, 236.7716856191929, 0.8816548604265529, 6.252597772662824, 951.0669
144554516, 3.7296741743351043, 0.9853740516953918, 0.8812566819699875, 0.930009741005473, 0.776613339516
7517,
1.2268041237113403, 361.98889941571633, 297.23408383947094, 0.89663047349524, 1.380434959484716 7, 903.21
64080912653, 3.542025129769668, 0.9861099189848482, 0.8948463585750336, 0.9251409486206645, 0.8007500054
549975,
2.8446215139442232, 359.53831729205336, 124.16103148938342, 0.8748367873045633, 2.1965789983729422, 2377
.354956052368 4, 9.322960611970073, 0.963439932395467, 0.8634411993711602, 0.8332433017202416, 0.74553070
47715075,
2.2062937062937062, 315.7120052681963, 139.74093156832498, 0.9087750601221283, 2.4804613887424125, 1232.
0754245754244, 4.8316683316683315, 0.9810525724390947, 0.9030078103010685, 0.8858489946759964, 0.8154231
054647304,
```

Figure 25: Output of Feature extraction using LBP

8.2 Dataset analysis

The dataset prepared is kept in directory format in “dataset\” folder as shown in Figure 15. So whenever a new signer is added two folders are to be created in the “dataset” folder. Whenever a new signature image is to be added in any of the existing folders, image can be directly copied in the respective folder with any name but in ‘.png’ format.

The ‘dataset.py’ python code as shown in figure 15 finds all the image folders in the dataset folder and all the images in those respective folders. The code then renames the image files in the format, ‘C_forged_12.png’. This is the standard image file name format kept for the classification of the training images based on the image name. The code also counts the number of training images, testing images, number of signers and also number of classes. The output of the same is shown in in figure 15.



```python
import os

os.getcwd()
collection = "DataSet"
training_folder = "Data\Training"
testing_folder = "Data\Testing"

images=0
classes=0
f = open("datasetCount.txt", "w")
print("     Dataset Count\n")
f.write("     Dataset Count\n\n")
f.close()
f = open("datasetCount.txt", "a")

for folder in os.listdir(collection):
    classes+=1
    j=0
    for file in os.listdir(collection+"/"+folder):
        j+=1
        images+=1
        #os.rename("DataSet/"+folder+"/"+file,"DataSet/"+folder+"/"+folder+"_"+str(j)+".png")
    print(folder +"\t"+ str(j)+ " images")
    f.write(folder +"\t"+ str(j)+ " images\n")

authors = int(classes / 2)
print ("\n"+str(authors)+" Authors, "+str(classes)+" Classes, "+str(images)+" Images")
f.write("\n"+str(authors)+" Authors, "+str(classes)+" Classes, "+str(images)+" Images")

training_files = 0
for file in os.listdir(training_folder):
    training_files += 1


testing_files = 0
for file in os.listdir(testing_folder):
    testing_files += 1
print ("\n"+str(training_files)+" Training images, "+str(testing_files)+" Testing images")
f.write("\n"+str(training_files)+" Training images, "+str(testing_files)+" Testing images")
f.close()
```

```
Type "copyright", "credits" or "license()"
>>>
============ RESTART: D:\Tejas\M.Tech\Proje
     Dataset Count

A_forged        24 images
A_genuine       24 images
B_forged        24 images
B_genuine       24 images
C_forged        24 images
C_genuine       24 images
D_forged        18 images
D_genuine       12 images
E_forged        24 images
E_genuine       12 images
F_forged        18 images
F_genuine       12 images
G_forged        24 images
G_genuine       12 images
H_forged        24 images
H_genuine       12 images
I_forged        24 images
I_genuine       12 images
J_forged        12 images
J_genuine       24 images
K_forged        8 images
K_genuine       24 images
L_forged        12 images
L_genuine       24 images
M_forged        8 images
M_genuine       24 images
N_forged        18 images
N_genuine       24 images
O_forged        24 images
O_genuine       12 images

15 Authors, 30 Classes, 562 Images

376 Training images, 186 Testing images
>>>
```

Figure 26: Output of Feature extraction using LBP

9. Matching of implementation with proposed plan

Following Gantt chart shows the action plan for the project to be done in a monthly basis for different modules. For the amount of work done till the end of November, research work, planning and analysis for the project has been completed. Designing of the project's development architecture and flow is under way and can be said that it is in a way done as well as mentioned in the Gantt chart. As shown in the above section a part of implementation is also been done and we can say that the project work is around 50% done.

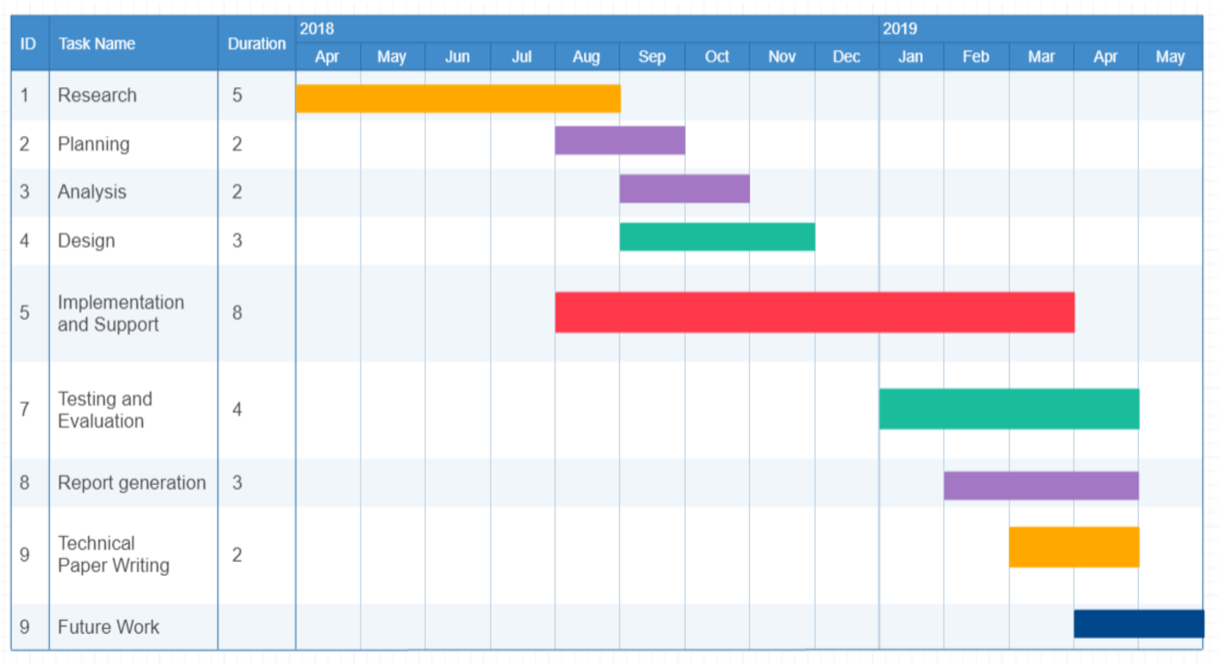


Figure 27: Gantt Chart

10. References

[1] S. F. A. Zaidi and S. Mohammed, “Biometric Handwritten Signature Recognition,” 2007.

[2] D. Morocho, A. Morales, J. Fierrez, and R. Vera-Rodriguez, “Towards human-assisted signature recognition: Improving biometric systems through attribute-based recognition,” *ISBA 2016 - IEEE Int. Conf. Identity, Secur. Behav. Anal.*, 2016.

[3] S. A. Angadi, S. Gour, and G. Bhajantri, “Offline Signature Recognition System Using Radon Transform,” *2014 Fifth Int. Conf. Signal Image Process.*, pp. 56–61, 2014.

[4] S. Hangai, S. Yamanaka, and T. Hammamoto, “ON-LINE SIGNATURE VERIFICATION BASED ON ALTITUDE AND DIRECTION OF PEN MOVEMENT,” *Proc. 15th Int. Conf. Pattern Recognition. ICPR-2000*, vol. 3, no. l, pp. 479–482, 2000.

[5] I. V Anikin and E. S. Anisimova, “Handwritten signature recognition method based on fuzzy logic,” *2016 Dyn. Syst. Mech. Mach.*, 2016.

[6] E. Ozgunduz, T. Senturk, and M. E. Karsligil, “Off-line signature verification and recognition by support vector machine,” *13th Eur. Signal Process. Conf.*, no. 90, pp. 1–4, 2005.

[7] S. A. Angadi and S. Gour, “Euclidean Distance Based Offline Signature Recognition System Using Global and Local Wavelet Features,” *2014 Fifth Int. Conf. Signal Image Process.*, pp. 87–91, 2014.

[8] Ruangroj Sa-Ardship and K. Woraratpanya, “Offline Handwritten Signature Recognition Using Adaptive Variance Reduction,” *7th Int. Conf. Inf. Technol. Electr. Eng. (ICITEE), Chiang Mai, Thail. Offline*, pp. 258–262, 2015.

[9] M. A. Djoudjai, Y. Chibani, and N. Abbas, “Offline signature identification using the histogram of symbolic representation,” *5th Int. Conf. Electr. Eng. - Boumerdes*, pp. 1–5, 2017.

[10] A. B. Jagtap and R. S. Hegadi, “Offline Handwritten Signature Recognition Based on Upper and Lower Envelope Using Eigen Values,” *Proc. - 2nd World Congr. Comput. Commun. Technol. WCCCT 2017*, pp. 223–226, 2017.

[11] T. Marušić, Ž. Marušić, and Ž. Šeremet, “Identification of authors of documents based on offline signature recognition,” *MIPRO*, no. May, pp. 25–29, 2015.

[12] S. L. Karanjkar and P. N. Vasambekar, “Signature Recognition on Bank cheques using ANN,” *IEEE Int. WIE Conf. Electr. Comput. Eng.*, no. December, 2016.

[13] G. S. Prakash and S. Sharma, “Computer Vision & Fuzzy Logic based Offline Signature Verification and Forgery Detection,” *IEEE Int. Conf. Comput. Intell. Comput. Res.*, 2014.

[14] M. S. Shirdhonkar and M. B. Kokare, “Document image retrieval using signature as query,” *2011 2nd Int. Conf. Comput. Commun. Technol. ICCCT-2011*, pp. 66–70, 2011.

[15] R. Sa-Ardship and K. Woraratpanya, “Offline Handwritten Signature Recognition Using Polar-Scale Normalization,” *8th Int. Conf. Inf. Technol. Electr. Eng. (ICITEE), Yogyakarta, Indones.*, pp. 3–7, 2016.

[16] A. Piyush Shanker and A. N. Rajagopalan, “Off-line signature verification using DTW,” *Pattern Recognit. Lett.*, vol. 28, no. 12, pp. 1407–1414, 2007.

[17] Nancy and P. G. Goyal, “Signature Processing in Handwritten Bank Cheque Images,” *Int. J. Recent Innov. Trends Comput. Commun.*, vol. 2, no. 5, pp. 1239–1243, 2014.

[18] A. T. Nasser and N. Dogru, “Signature recognition by using SIFT and SURF with SVM basic on RBF for voting online,” *Proc. 2017 Int. Conf. Eng. Technol. ICET 2017*, vol. 2018–Janua, pp. 1–5, 2017.

[19] A. Kumar and K. Bhatia, “A Robust Offline Handwritten Signature Verification System Using Writer Independent Approach,” *3rd Int. Conf. Adv. Comput. Autom.*, 2017.

[20] H. Anand, “Enhanced Signature Verification and RECOGNITION USING MATLAB,” *Int. J. Innov. Res. Adv. Eng.*, vol. 1, no. 4, pp. 88–94, 2014.

[21] B. H. Shekar and R. K. Bharathi, “Eigen-signature: A robust and an efficient offline signature verification algorithm,” *Int. Conf. Recent Trends Inf. Technol. ICRTIT 2011*, pp. 134–138, 2011.

[22] A. R. Rahardika, “Global Features Selection for Dynamic Signature Verification,” *Int. Conf. Inf. Commun. Technol. Glob.*, pp. 348–354, 2018.

[23] T. Handhayani, A. R. Yohannis, and Lely Hiryanto, “Hand Signature and Handwriting Recognition as Identification of the Writer using Gray Level Co- Occurrence Matrix and Bootstrap,” *Intell. Syst. Conf.*, no. September, pp. 1103–1110, 2017.

[24] A. Beresneva, A. Epishkina, and D. Shingalova, “Handwritten Signature Attributes for its Verification,” pp. 1477–1480, 2018.

[25] M. P. Patil, Bryan Almeida, Niketa Chettiar, and Joyal Babu, “Offline Signature Recognition System using Histogram of Oriented Gradients,” *Int. Conf. Adv. Comput. Commun. Control*, 2017.

[26] M. M. Kumar and N. B. Puhan, “Offline signature verification using the trace transform,” *2014 IEEE Int. Adv. Comput. Conf.*, pp. 1066–1070, 2014.

[27] O. Miguel-Hurtado, L. Mengibar-Pozo, M. G. Lorenz, and J. Liu-Jimenez, “On-Line Signature Verification by Dynamic Time Warping and Gaussian Mixture Models,” *2007 41st Annu. IEEE Int. Carnahan Conf. Secur. Technol.*, pp. 23–29, 2007.

[28] M. Ferrer and J. Vargas, “Robustness of offline signature verification based on gray level features,” *IEEE Trans. Inf. FORENSICS Secur.*, vol. 7, no. 3, pp. 966–977, 2012.

[29] B. Erkmen, N. Kahraman, R. A. Vural, and T. Yildirim, “Conic section function neural network circuitry for offline signature recognition,” *IEEE Trans. Neural Networks*, vol. 21, no. 4, pp. 667–672, 2010.

[30] M. A. Ferrer, A. Morales, and J. F. Vargas, “Off-line signature verification using local patterns,” *2nd Natl. Conf. Telecommun.*, pp. 1–6, 2011.

[31] M. Tahir and M. U. Akram, “Online Signature Verification using Hybrid Features,” *Conf. Inf. Commun. Technol. Soc. Online*, pp. 11–16, 2018.

[32] M. Pal, S. Bhattacharyya, and T. Sarkar, “Euler number based feature extraction technique for Gender Discrimination from offline Hindi signature using SVM & BPNN classifier,” *2018 Emerg. Trends Electron. Devices Comput. Tech.*, pp. 1–6, 2004.

[33] W. Pan and G. Chen, “A Method of Off-line Signature Verification for Digital Forensics,” *12th Int. Conf. Nat. Comput. Fuzzy Syst. Knowl. Discov.*, pp. 488–493, 2016.

[34] M. A. Ferrer, J. B. Alonso, and C. M. Travieso, “Offline geometric parameters for automatic signature verification using fixed-point arithmetic,” *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 27, no. 6, pp. 993–997, 2005.

[35] S. A. Farimani and M. V. Jahan, “An HMM for Online Signature Verification Based on Velocity and Hand Movement Directions,” *Iran. Jt. Congr. Fuzzy Intell. Syst.*, pp. 205–209, 2018.

[36] G. Pirlo and D. Impedovo, “Verification of static signatures by optical flow analysis,” *IEEE Trans. Human-Machine Syst.*, vol. 43, no. 5, pp. 499–505, 2013.

[37] A. Alaei, S. Pal, U. Pal, and M. Blumenstein, “An Efficient Signature Verification Method Based on an Interval Symbolic Representation and a Fuzzy Similarity Measure,” *IEEE Trans. Inf. Forensics Secur.*, vol. 12, no. 10, pp. 2360–2372, 2017.

[38] D. S. Guru and H. N. Prakash, “Online Signature Verification and Recognition: An Approach based on Symbolic Representation.,” *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 31, no. 6, pp. 1059–73, 2009.

[39] A. Sharma and S. Sundaram, “On the Exploration of Information from the DTW Cost Matrix for Online Signature Verification,” *IEEE Trans. Cybern.*, vol. 48, no. 2, pp. 611–624, 2018.

[40] G. Pirlo, V. Cuccovillo, M. Diaz-Cabrera, D. Impedovo, and P. Mignone, “Multidomain Verification of Dynamic Signatures Using Local Stability Analysis,” *IEEE Trans. Human-Machine Syst.*, vol. 45, no. 6, pp. 805–810, 2015.

[41] Python : https://www.python.org/

[42] Dataset source :- http://www.iapr-tc11.org/mediawiki/index.php?title=Datasets_List#Handwritten%20Documents