CHAPTER 3

3. PROBLEM MODELING

Writer identification is an important research area undergone for past three decades. Large variety of scientific and business applications are involved in finding the solution to process large volume of data through handwritten recognition and writer identification [72]. Recent advances in computational engineering, artificial intelligence, data mining, image processing, pattern recognition and machine learning have proved that it is possible to automate writer identification. Writer identification has been studied extensively for several languages like Chinese, English, Arabic, Persian, Kurdi, Jawi and Urdu [73] and also for several Indian languages like Hindi, Oriya, Bangla, Kannada, and Malayalam. Various approaches like pattern matching, statistical, machine learning have been adopted for processing the documents and developing the writer identification models for these languages.

This work proposes a new model for learning the writer’s identity constructed on Tamil handwriting. Handwritten documents written by the writers are scanned and segmented into words. Words are further segmented into characters for character level writer identification. The character and paragraph writings in Tamil are analyzed and their descriptive features are defined. Finally the Writer identification problem is formulated as classification task and a pattern classification technique namely Support Vector Machine (SVM) with kernels has been employed to construct the model. The processes that have been incorporated during modeling automatic writer identification include corpus preparation, preprocessing, feature extraction, training and writer prediction. The architecture of the proposed system is shown in Fig. 3.1

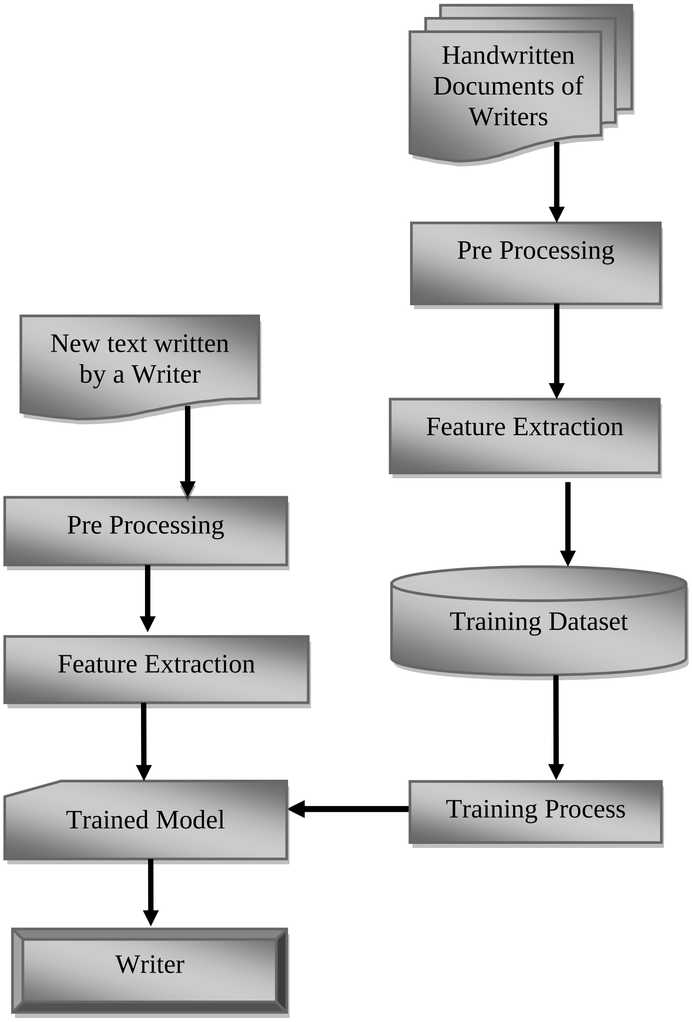


Fig. 3.1 Proposed Writer Identification Model

The research work proposed here for modeling writer recognition is based on offline text dependent approach. The primary focus of this research is to provide an efficient machine learning based solution for the problem of individual identification by converting into pattern classification problem. The above processes are described in the following sections.

3.1. CORPUS PREPARATION

Individuals tend to make mistakes while writing since many of the consonants and vowels has comparable articulate. As Tamil handwritten text is not available online, own corpus is developed to carry out this research. Text written in Tamil is the key data for this research. Tamil letters has 12 vowels called soul letters (உயிரெழுத்து uyireluttu), 18 consonants called body letters (மெய்யெழுத்து meyyeluttu) and one character called aytam (ஆய்தம்). The complete Tamil letters has thirty one independent form and additional 216 combinant letters combined to have a total 247 combinations (உயிர்மெய்யெழுத்து uyirmeyyeluttu) of a consonant and a vowel, a mute consonant, or a vowel alone. Some Tamil letters namely க்ஷ ksa and ஸ்ரீ sri which is borrowed from Sanskrit language. The unique characters shown in Fig. 3.2 are more multifaceted in nature. This will make the writers like greenhorns and illiterate to consign more mistakes while earshot and writing the scripts.

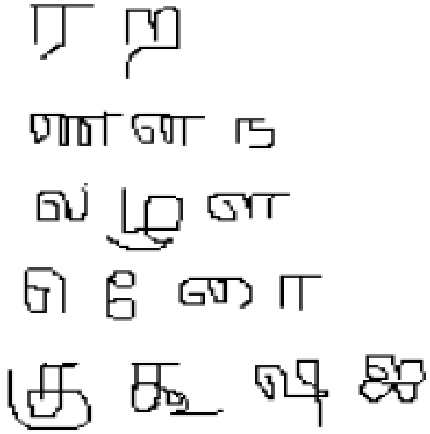
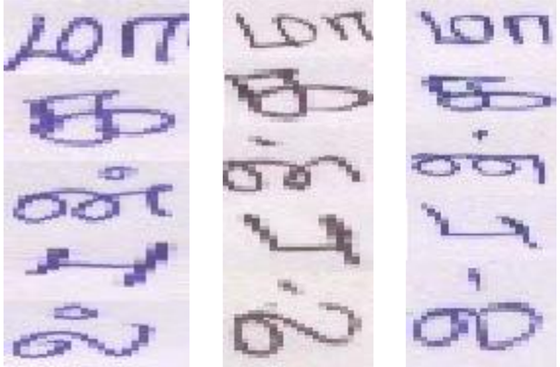


Fig. 3.2 Multifaceted Calligraphies in Tamil

These are some of the issues that need to be addressed during identification of individual based on their handwriting. In applications like forensic document analysis, handwriting with limited words will not be sufficient to predict the writer accurately. So the handwritten text with more sentences is required for training the classifier. In the proposed system these challenges are managed by considering the appropriate features and attempted to produce promising results. Accordingly, Tamil handwriting identification has more complexities than any other language.

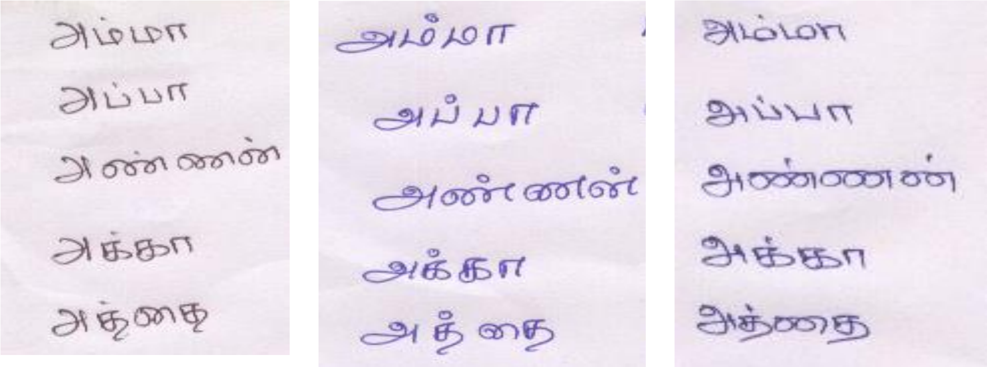
A text document with 100 paragraphs is designed and prepared by considering the intricacies involved in Tamil alphabets/characters. This document was circulated to 300 different writers and text dependent handwritten documents have been collected. These handwritten documents are scanned using scanner of resolution 300 dpi (enclosed in Appendix A) and they are segmented into word and character images. In applications like signature verification, biometric authentication, only single word handwriting is used to recognize the individual. Hence from each paragraph image 30000 JPEG text independent word images and 30000 JPEG text independent character images are selected and three separate corpuses are prepared.

The writers of different age groups are considered here. The individuals have been identified based on various factors like education level, age, gender and locality. Tamil handwritings have been collected from people working in government and private organizations, teachers working in school and colleges, school dropout girls and women, blue-collar workers (skilled and unskilled workers), senior citizens and home makers. Sample of text dependent characters, words and paragraph images are shown in Fig. 3.3, Fig. 3.4, and Fig. 3.5,



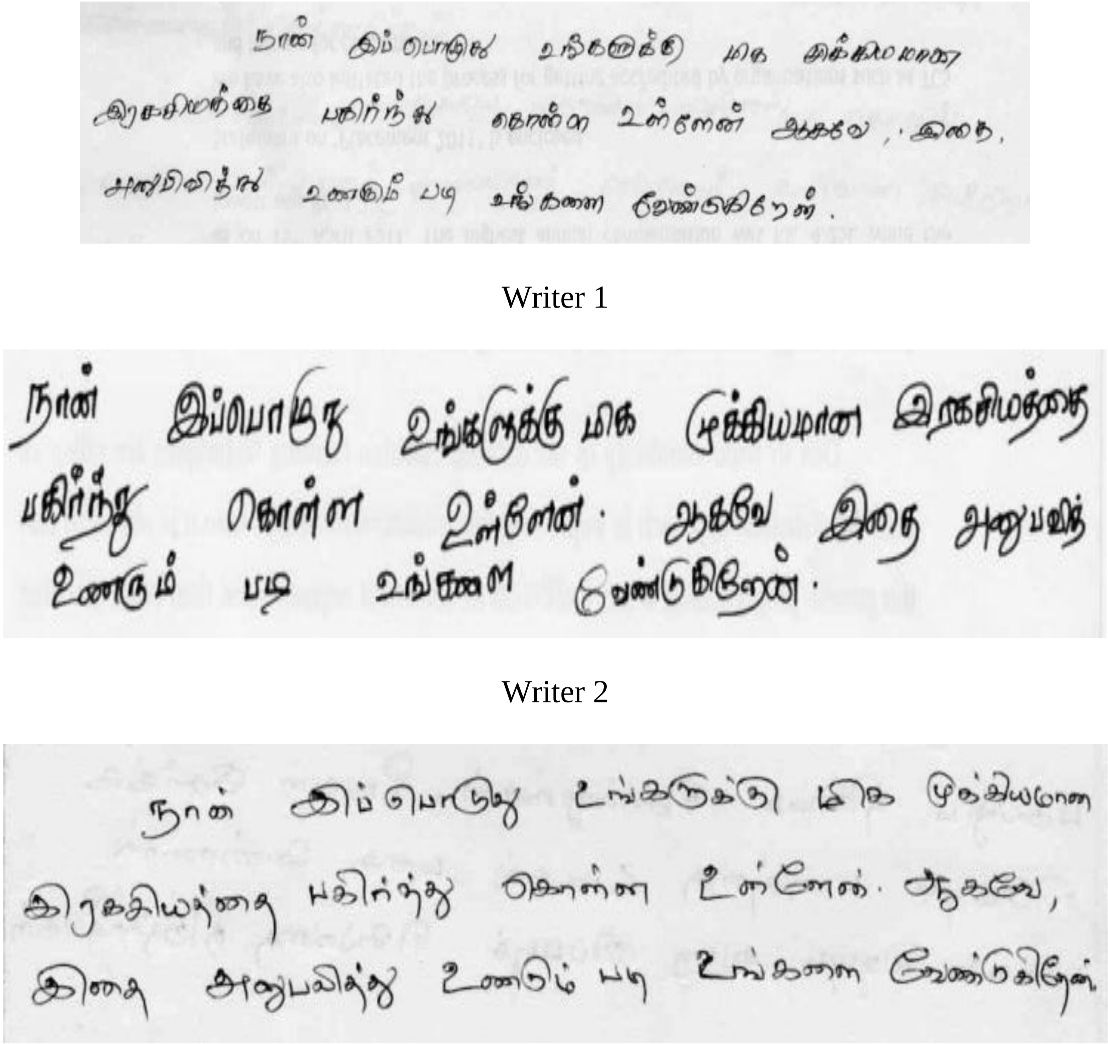
Writer 1 Writer 2 Writer 3

Fig. 3.3 Text Dependent Character Images – Sample Data



Writer 1 Writer 2 Writer 3

Fig. 3.4 Text Dependent Word Images – Sample Data



நான் இப்பொழுது உங்களுக்கு மிக முக்கியமான
இரகசியத்தை பகிர்ந்து கொள்ள உள்ளேன் ஆகவே, இதை.
அனுபவித்து உணரும் படி உங்களை வேண்டுகிறேன்.

Writer 1

நான் இப்பொழுது உங்களுக்கு மிக முக்கியமான இரகசியத்தை
பகிர்ந்து கொள்ள உள்ளேன். ஆகவே இதை அனுபவித்து
உணரும் படி உங்களை வேண்டுகிறேன்.

Writer 2

நான் இப்பொழுது உங்களுக்கு மிக முக்கியமான
இரகசியத்தை பகிர்ந்து கொள்ள உள்ளேன். ஆகவே,
இதை அனுபவித்து உணரும் படி உங்களை வேண்டுகிறேன்.

Writer 3

Fig. 3.5 Text Dependent Paragraph Images – Sample Data

3.2. PREPROCESSING

Preprocessing is an important task in any mining activity. The aim of preprocessing is to clear undesired distortions which will help to enhance further processing. There are substantially positive effects based on image pre-processing will improve quality of feature extraction and the results obtained from image analysis.

There are no many features discriminating character handwriting and word handwriting, since much variation in writing patterns of characters and words cannot be recognized due to the formation of words with few characters. Hence same preprocessing and feature extraction process have been adopted in both character and word handwriting images.

This section explains image preprocessing tasks such as noise removal, binarization [22] [74], edge detection and thinning which have been adopted in this research.

3.2.1. Character and Word Text Image Preprocessing

The best preprocessing approach on Tamil character and word is to normalize all character and word into a uniform size of pixels and to remove the background noise of the character in order to smooth the process of identification. The preprocessing tasks carried out here on the character images and word images are noise removal, binarization, edge detection and thinning.

Noise Removal: Noise in an image is removed using median filtering. In this method, the neighboring pixels are ranked according to brightness (intensity) and the median value becomes the new value for the central pixel. Median filters facilitate by rejecting few kinds of noise, specifically, “shot” or impulse noise wherein the individual pixels are characterized by extreme values. During median filtering operation, neighborhood window pixel values are marked and ranked in lieu of intensity, hence with regards to pixel that are being evaluated the middle value (the median) is the output value. Median filters offer three main features that are advantageous:

- Contrast across steps is not reduced, as output values that exist comprise of those that are apparent in the neighborhood (no averages).
- Median filtering here shows no shift boundaries, which usually occurs with conventional smoothing filters (an inherent contrast dependent problem).
- Since the median is less sensitive than the mean to extreme values (outliers), those extreme values are more effectively removed.

The results of noise removal are shown in Fig. 3.6.a and Fig. 3.6.b

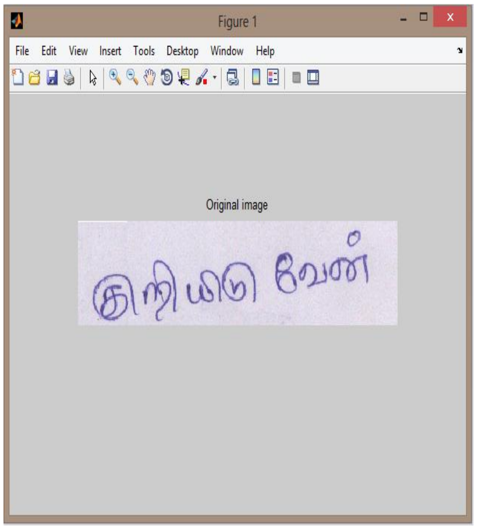


Fig. 3.6. a Before Noise Removal

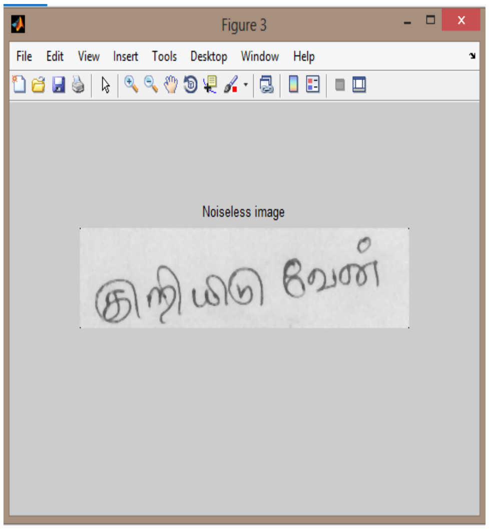


Fig. 3.6. b After Noise Removal

Binarization: This converts gray scale image into binary image using Otu's method. Otsu's method computes a global threshold (level) that can be used to convert an intensity image to a binary image with im2bw (a MATLAB function). Where level is a normalized intensity value that lies in the range [0, 1]. Graythresh function deploys the Otsu's technique wherein threshold is chosen to reduce black and while pixels intraclass variance. The results of binarization are shown in Fig. 3.7.a and Fig. 3.7.b

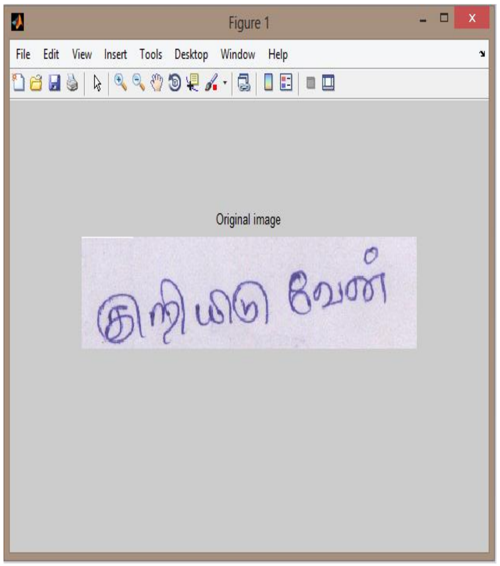


Fig. 3.7.a Non- Binarized Image

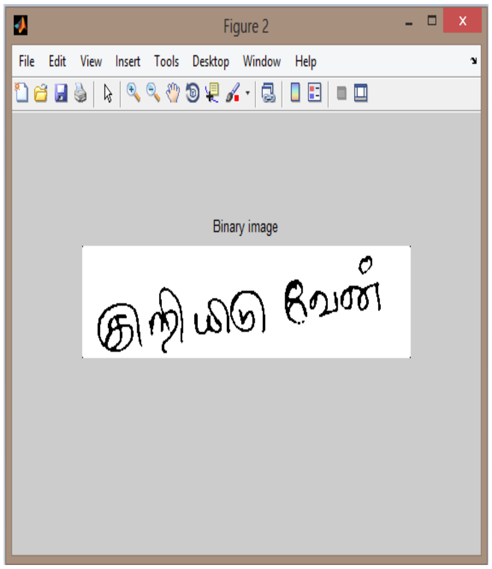


Fig. 3.7.b Binarized Image

Edge Detection: Edges in the binary image are detected using sobel method. The Sobel method finds edges using the Sobel approximation to the derivative. Where gradient of I is maximum it returns edges specifically at those points. Edge ignores all edges that are not stronger than thresh. If thresh do not specified, or if thresh is empty [23], edge chooses the value automatically. Edges are detected for both word images and character images and the results are shown in Fig. 3.8.a and Fig. 3.8.b

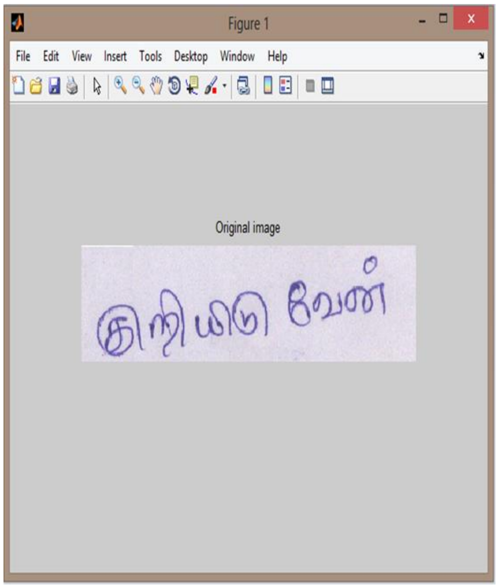


Fig. 3.8. a Original Image

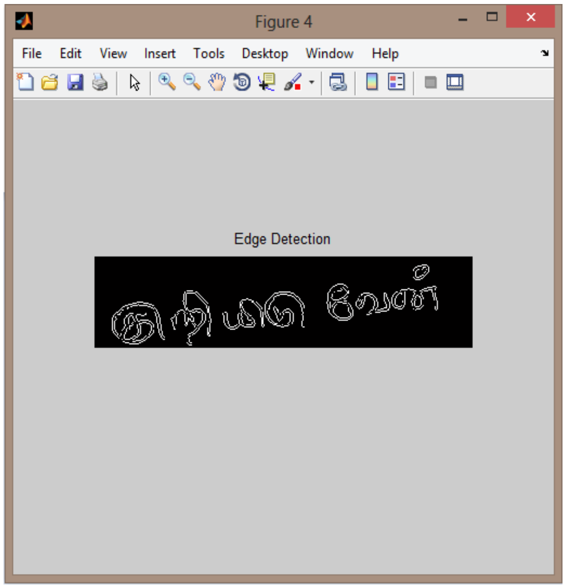
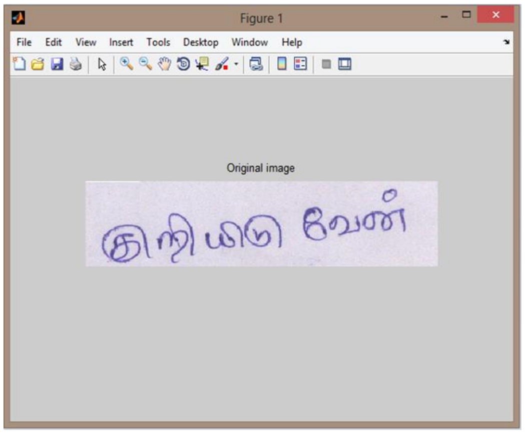


Fig. 3.8. b Edge Detected Image

Thinning: Morphological operations are used for thinning the binary image. Thinning operation thins objects to lines. Once the pixels are eliminated object without holes shrinks to a minimally connected stroke, subsequently object with holes shrinks to a connected ring, which is halfway between each hole and outer boundary. The results of thinning operation for both word images and character images are shown in Fig. 3.9.a and Fig. 3.9.b.



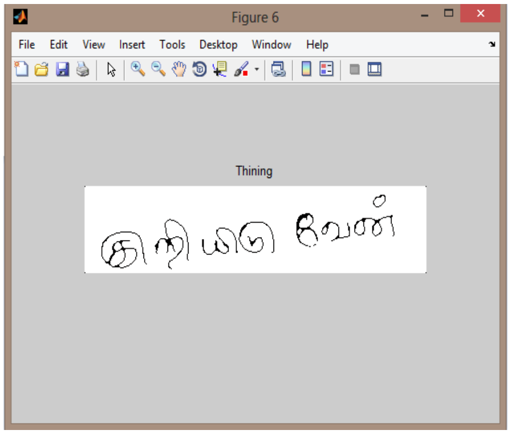


Fig. 3.9.a Original Image **Fig. 3.9.b Thinned Image**

3.2.2. Paragraph Text Image Preprocessing

Here normalization technique is performed prior to preprocessing in order to correct the skewed words in the handwriting image. Space that is present between the vertical and horizontal lines is then normalized for running a texture analysis while generating a well-defined pattern. Scanned images are thereafter preprocessed to remove the noise and converted into grayscale images to carry further preprocessing tasks. The preprocessing tasks carried out here are edge detection, image dilation and box bounding.

Edge detection: Edges in the binary image are detected using sobel method. Sobel approximation to the derivative is deployed while using the Sobel method to determine the edges. It returns edges at those points where the gradient of I is maximum. Edges that are comparatively less strong to the threshold are ignored by the Edge. Assuming that thresh is not specified, or if thresh is empty, edge chooses the value automatically. Edges are detected for all document images and the result of edge detection for a sample image is shown in Fig. 3.10.

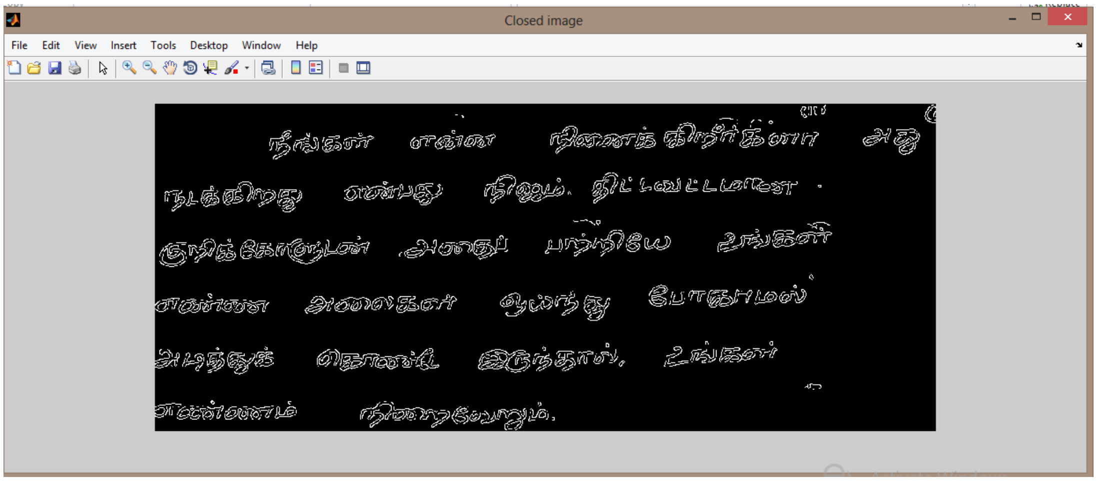

Closed image
File Edit View Insert Tools Desktop Window Help
நீங்கள் என்ன நினைக்கிறீர்களோ அது
நடக்கிறது என்பது நிஜம். திட்டவட்டமான
குறிக்கோளுடன் அதைப் பற்றியே உங்கள்
எண்ண அலைகள் ஓய்ந்து போகாமல்
அடித்துக் கொண்டு இருந்தால், உங்கள்
எண்ணம் நிறைவேறும்.

Fig. 3.10 Edge Detection

Dilation: The dilation is a fundamental morphological operation. In this the dilation adds the pixels to the boundaries of the images. Based on the size and shape of the structuring element the pixels are added. Here, after the image is converted into grayscale image, the dilation operation is performed. The result of dilation is shown in Fig. 3.11.

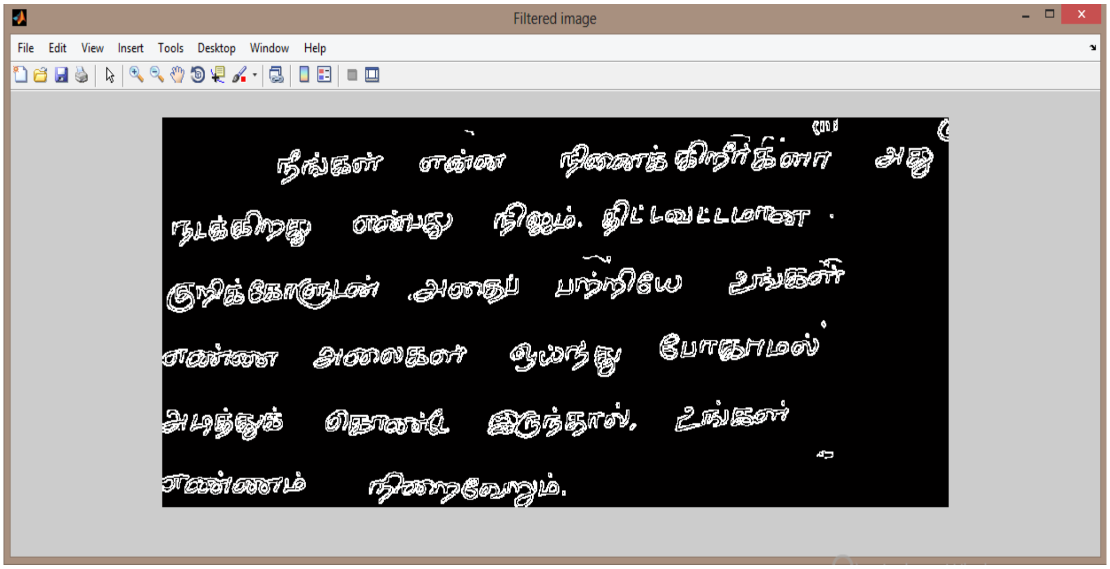

Filtered image
File Edit View Insert Tools Desktop Window Help
நீங்கள் என்ன நினைக்கிறீர்களோ அது
நடக்கிறது என்பது நிஜம். திட்டவட்டமான
குறிக்கோளுடன் அதைப் பற்றியே உங்கள்
எண்ண அலைகள் ஓய்ந்து போகாமல்
அடித்துக் கொண்டு இருந்தால், உங்கள்
எண்ணம் நிறைவேறும்.

Fig. 3.11 Dilated Image

Box bounding: After the image is converted into dilated image, the pixels in the image are converted into white pixels. To box bound a word; the white pixels are taken for each character with a pixel space of 30 pixels. Once it exceeds the pixel space of 30 then a word is box bounded and it starts from the next word to box bound until it exceeds the space of 30 pixels. The words in the images are box bounded and the result of box bounding is shown in Fig. 3.12.

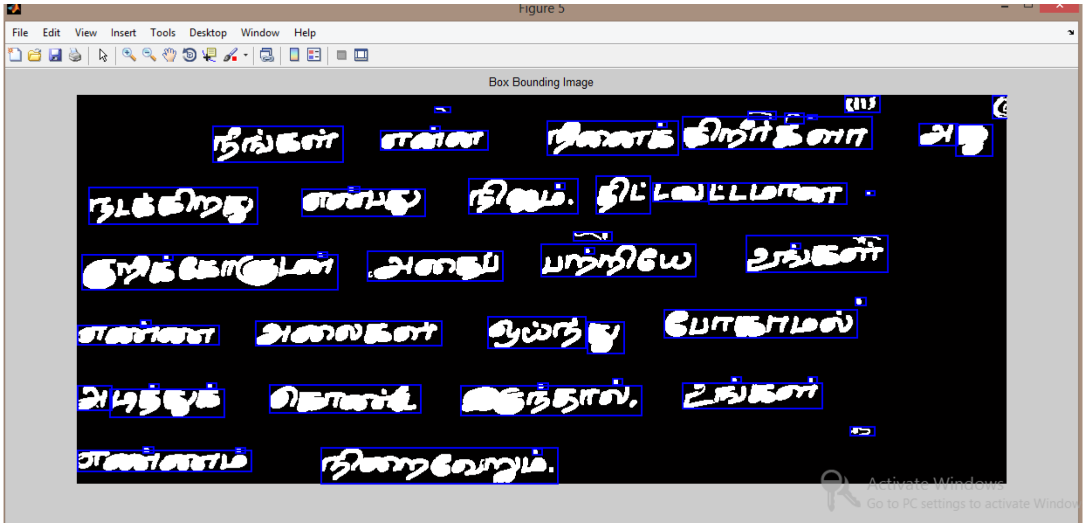


Fig. 3.12 Box Bounded Image

After box bounding, individual words and characters in handwriting images are guaranteed to be upright and changed to line fit to make the text normalized.

1. Normalization of Handwriting Image

Normalization requires two different stages. First the detection and correction of the skewed words with the handwriting images has been performed [29]. Then, the space between horizontal and vertical lines of a text has been normalized.

2. Skewed Words Normalization

Handwriting images are not assured to be exact during scanning as it is in the document. Handwriting images may distress the writer identification if there is disposition of different characters, words and lines. This issue is overwhelmed by skew normalization procedure is implemented. This procedure can be performed using the following steps,

Step 1: In a handwriting image Horizontal Projection Profile (HPP) method is applied to detect text lines and empty spaces.

Step 2: Closing procedure is applied to the image using a 3×3 structuring elements. (The middle row of the element is set so as to close the image in the horizontal direction to avoid joining text lines is called closing procedure).

Step 3: Select the connected components.

Step 4: In that the minimum, maximum and mean connected component heights is calculated.

Step 5: Filter out the smallest 5% of height in the text to eliminate punctuation, quotes and so on.

Step 6: Formerly eliminate components with a height > 2 mean height to remove components which are even now connected with more than one text line.

Step 7: Then perform the following, in the remaining connected component.

Step 8: Copy the component into a blank image, in which the image has the component bounding box size.

Step 9: In that the line fit on the connected component is performed.

Step 10: Finally, base line fitting in which the base lines are computed from the HPP of the deskewed image.

3.3. FEATURE EXTRACTION

Feature extraction plays a vital role in improving the classification effectiveness and computational efficiency. A set of distinctive features describing the writing style and writer's invariance is extracted to form a feature vector. In the proposed writer identification model the complexities in Tamil alphabet system are handled by considering the appropriate features of the handwriting to produce promising results. Various features extracted from character, word and paragraph text images are described in the following sections.

3.3.1. Feature Extraction from Character / Word Text Images

Global Features and Local Features are the two types of features that can extracted from handwritten text image. Global features are features taken by considering the text image as image rather than handwriting [75]. Local features are features taken by considering the text image as handwriting. Many structural properties of the handwriting can be derived as local features. In this work more importance is given to local features. The local features are grouped into word measurement features, morphological features, fractal features, GSC features comprising gradient structural, concavity attributes. The feature extraction process of the above local features is described below.

Word Measurements Features

Word measurement feature is the quantitative evaluation to a specific handwriting, which can be compared with other handwriting images. Word measurements features are area, length, height, Length from baseline to upper edge, Length from baseline to lower edge, ascender line, descender line, slope angle, junctions and loops of characters/words.

Length

Length of the word is found by successively penetrating each column in the binary image to find the first and last pixels in the image and store their column numbers. The length of the image is calculated by subtracting the column number of last pixel to the column number of first pixel. The length of the word is shown in Fig 3.13.

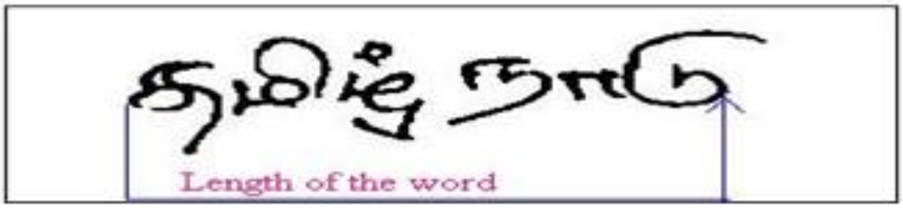


Fig. 3.13 Length of the Word

Height

Height of the word is found by consecutively probing each row in the binary image. The first and last pixels of the image are found and the corresponding row numbers are stored. The height of the image is computed by subtracting from the row number of last pixel to the row number of first pixel. Fig 3.14 illustrates height of the word.

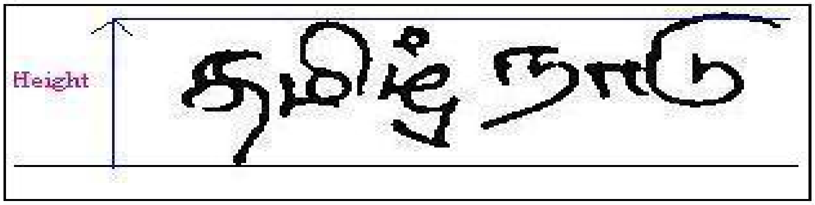


Fig. 3.14 Height of the word

Area

Area of the word is calculated as the product of height and length.

Length from Baseline to Upper Edge

The length of the text from baseline to upper edge is calculated by first determining the baseline position of the image. This is functioned by casting an array where the index is row number in the image. Then, the number of black pixels in each row is calculated and the results are stored in an array. If the process is completed for whole image, the maximum value of the array is identified and the corresponding row number is stored as the baseline. The length of the image from the baseline to the upper edge is computed by subtracting the row number of first pixel in the image from its row number of the baseline. The length from baseline to upper edge is shown in Fig 3.15.

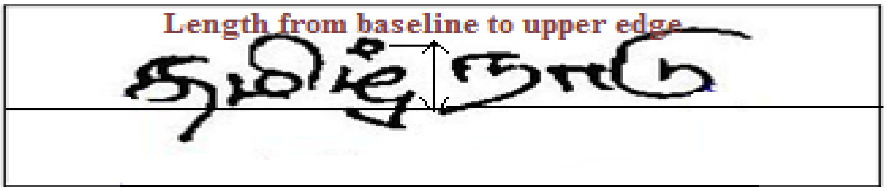


Fig. 3.15 Length from Baseline to Upper Edge

Length from Baseline to Lower Edge

The length of the binary image from the baseline to the lower edge is determined by calculating the baseline row number, as above. Then, the row number of the last pixel of the image is considered. The length of the image from the baseline to the lower edge is calculated by subtracting the last pixel row number to the baseline row number. Length from baseline to lower edge is shown in Fig 3.16.

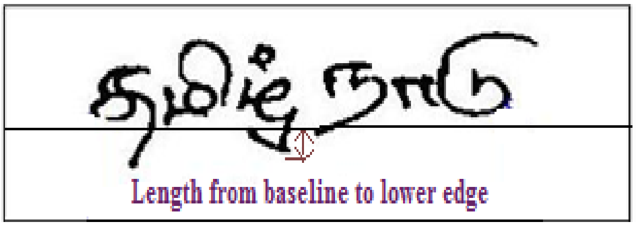


Fig. 3.16 Length from Baseline to Lower Edge

Ascender and Descender Baseline

Ascender baseline is the first non-zero value of column and the descender baseline is the last non-zero value of column of the vertical histogram of the line. Ascender and descender baseline is shown in Fig 3.17.

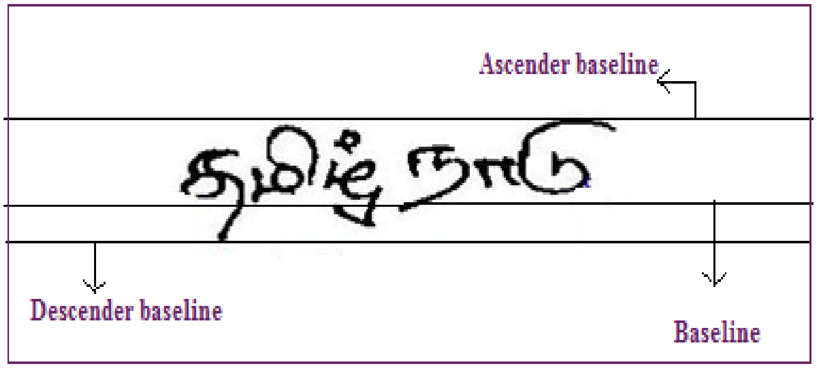


Fig. 3.17 Ascender and Descender Baseline

Junctions

Junctions occur where two strokes meet or cross and are found in the skeleton as points with more than two neighbors. It produces number of junctions, positions of each junction, angle and distance between the junctions of the thinned image.

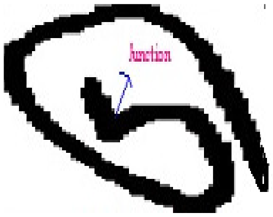


Fig. 3.18 Junction of the Character

Loops

The loops of a character are the major distinguishing feature for many writers. The loop function gives loop length, angle of loop, position of the loop, area and average radius of the loop of the edge image [76].

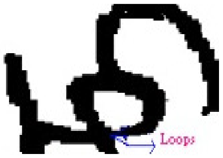


Fig. 3.19 Loops of the Character

Edge based Directional Features

Edge-based directional features helps to improve the performance of writer identification in comparison with number of non-angular feature. It is based on the joint probability distribution of the angle combination of two hinged edge fragments. Edge based directional features are edge direction distribution, edge hinge distribution and run length distribution.

Edge Direction Distribution

In edge direction distribution, first the edge of the binary image is detected using Sobel detection method. The edge detected images are labelled using 8- connected pixel neighborhood. Then the number of rows and columns in a binary image is found using size function. Next, the first black pixel in an image is found and this pixel is considered as center pixel of the square neighborhood. Then the black edge is checked using logical AND operator in all direction starting from the center pixel and ending in any one of the edge in the square. In order to avoid redundancy the upper two quadrants in the neighborhood is checked because without on-line information, it is difficult to identify the way the writer travelled along the edge fragment. This will gives us ‘n’ possible angles. Thereafter, each pixel’s verified angle is counted as an n-bin histogram. This is then further normalized as a probability distribution and at an angle that has been measured at the horizontal renders probability of an edge fragment oriented in the image. ‘n’ here is considered in 4, 8, 12, and 16.

Edge Hinge Distribution

To capture the curvature of ink trace, which is very distinctive for different writer, edge hinge distribution is needed, which is calculated with the help of local angles along the edges. Edge hinge feature takes in to consideration the two edge fragments that stem from the center pixel and, thereafter, calculation for the two fragments of a ‘hinge’ and joint probability distribution of the orientations is performed. Finally, normalized histogram gives the joint probability distribution for “hinged” edge fragments oriented at the angles 1 and 2. The orientation is counted in 16 directions for a single angle. From the total number of combinations of two angles only non- redundant values are considered and the common ending pixels are eliminated.

Run Length Distribution

Run lengths are estimated based on the binarized image assuming that the black pixels are consistent with ink trace or the white pixels correspondingly match the respective background. Scanning procedures are of two types: horizontal along the rows of the image and vertical along the column of the image. Next, the probability distribution is interpreted by using the normalized histogram of run lengths. Orthogonal information to the directional features is obtained by using the run lengths.

Moment Invariants

Moment invariants or geometric moment invariants are commonly used in pattern recognition. These geometric features are used to recognize object when the object is changed in its transformations such as size or scaling, rotation, translation and orientation. Moment invariants are invariants that are derived from moments. In an object, invariants are classified into two different forms such as continuous and discrete domain. Moment invariants are used to invariants in a continuous domain. But in discrete domain scaling or rotations are used to define the invariants. A distinctive set of features are calculated for an object in order to identify the same object with different size and orientation. In this work the following seven moments are computed.

$$\begin{aligned}
&M1 = \eta 20 + \eta 02, \\
&M2 = (\eta 20 - \eta 02)^2 + (2\eta 11)^2, \\
&M3 = (\eta 30 - 3\eta 12)^2 + (3\eta 21 - \eta 03)^2, \\
&M4 = (\eta 30 + \eta 12)^2 + (\eta 21 + \eta 03)^2, \\
&M5 = (\eta 30 - 3\eta 12)(\eta 30 + \eta 12)\left[(\eta 30 + \eta 12)^2 - 3(\eta 21 + \eta 03)^2\right] \\
&+ (3\eta 21 - \eta 03)(\eta 21 + \eta 03)\left[3(\eta 30 + \eta 12)^2 - (\eta 21 + \eta 03)^2\right], \\
&M6 = (\eta 20 - \eta 02)\left[(\eta 30 + \eta 12)^2 - (\eta 21 + \eta 03)^2\right] \\
&+ 4\eta 11(\eta 30 + \eta 12)(\eta 21 + \eta 03), \\
&M7 = (3\eta 21 - \eta 03)(\eta 30 + \eta 12)\left[(\eta 30 + \eta 12)^2 - 3(\eta 21 + \eta 03)^2\right] \\
&- (\eta 30 + 3\eta 12)(\eta 21 + \eta 03)\left[3(\eta 30 + \eta 12)^2 - (\eta 21 + \eta 03)^2\right].
\end{aligned} \tag{3.1}$$

Morphological Features

Morphological features are used to extract outward appearance of handwriting image such as shape, size, structure, colour, and pattern. Morphological features are directional opening, directional closing, directional erosion, k-curvature, Skew angle, slant angle, auto-correlation, entropy, aspect radio, end points, and height of 3 main handwriting zones such as upper zone, lower zone and middle zone and average width of writing. These features are defined below.

Slope Angle

The slope of angle **A** is the ratio of height to length. In geometry, it is also referred to as the tangent of the angle **A** and denoted by tan (**A**), which gives us the slope angle.

Slant Angle

It is the angle of the word forms against the baseline. It is estimated on structural features by maxima and minima of the word are detected and targets uniform slant angle estimation.

Auto-Correlation

Auto-correlation function identifies the presence of predictability in writing. By giving the offset value, every row of the image is shifted onto itself. Then the normalized dot product is found between the original row and the shifted row. Auto-correlation function is computed for all rows and the sum is normalized to obtain a zero-lag correlation of 1.

Entropy

Entropy provides the average information of an image such as luminance, contrast and pixel value [77]. It is calculated using the equation (3.2):

$$E = H[p(g)] - \sum_{j-1}^{j} p(j) H[p_j(g)] \tag{3.2}$$

Aspect Ratio

Aspect ratio is considered as one of the global features in writer identification. It is calculated as ratio of width to height.

End Points

End-points contain only one pixel in their 8-pixel neighborhood. It is computed using end point function which gives the number of end points in the thinned image.

Thus a total of 26 features are extracted from a single character / word image which creates a feature vectors. A sample feature vector for text image given in Fig. 3.13 is shown in Table. 3.1.

Table 3.1 Feature Vector of a Single Word Image

Features	Feature Values
Length of the word	9
Height of the word	19
Area of the word	171
End Points	32668
Loops	49
Junctions	11310
Aspect Ratio	4.2667
Length from baseline to lower edge	135
Length from baseline to upper edge	-1
Ascender baseline	1
Descender baseline	133
Slope Angle	-6.4186
Slant Angle	88.1882
Entropy	0.30622
Moment Invariants 1	5.5511
Moment Invariants 2	29.3608
Moment Invariants 3	4.9229
Moment Invariants 4	4.6377
Moment Invariants 5	15.783
Moment Invariants 6	24.8243
Moment Invariants 7	-19.4515
Edge Direction Distribution	35
Edge Hinge Distribution	65
Horizontal Run Length Distribution	47
Vertical Run Length Distribution	72
Auto-Correlation	30

3.3.2. Feature Extraction from Paragraph Text Images

In case of paragraph text images, global features are Gabor filter and Gray Level Co-occurrence Matrix whereas the local features identified are Generalized Gaussian Density (GGD), Contourlet GGD, Directional features. These five categories of features are aggregated and partial spam optimization method is used to select the contributive features.

Gabor Filter

The Gabor filter is a bandpass filter used in the image processing for feature extraction. Here the Gabor filter [78] requires an input image with N*N pixel image along with the frequency (f) and an angle θ. Here the θ and f specifies the location for the Gabor filter. As the size of the image is N*N, the frequency used here is 4, 8, 16 and 32 cycles/degree. The parameters used in this filter are bandwidth, phase shift and lambda.

The core of Gabor filter based feature extraction is the 2D Gabor filter function and is defined as below,

$$x' = x \cos \theta + y \sin \theta \qquad (3.3)$$

$$y' = -x \sin \theta + y \cos \theta \qquad (3.4)$$

For each central frequency f, filtering is performed at 0, 45, 90 and 135 degrees. This gives a total of 16 output images for each frequency, from which the writer’s features are extracted. These features are the mean and the standard deviation of each output image. Thus 32 features are obtained for each and every pixel in a given input image. The same process is repeated for all the pixels in the input image and finally 256 features obtained for a given input image. The feature values resulted for a paragraph text image shown in Fig .10 are given below

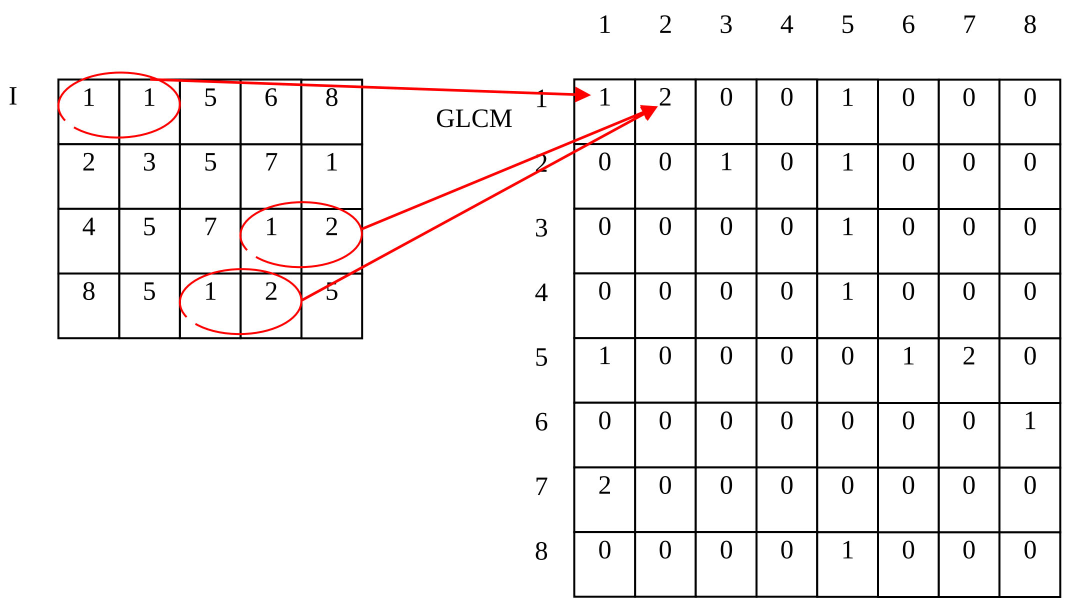
```
0.009042815   0.009042815   -9.23E-05       -9.23E-05       -9.23E-05       -
9.23E-05 0.442770934 0.442770934 -0.063264664   -0.063264664
0.001291831   0.001291831   0.999630957   0.999630957   0.999838521
0.999838521   0.999819144   0.999819144   0.999815453   0.999815453
63.74445109   63.74445109   15.99870817   15.99870817   255.9110337
255.9110337   0.001771195   0.001771195   0.009042815   0.009042815
0.001771195   0.001771195   -8.97E-06       -8.97E-06       0.000130501
0.000130501   0.999838521   0.999838521   0.999913878   0.999913878
0.999919975   0.999919975   63.98966535   63.98966535   0.15173968
0.157846651   0.164345005   0.168516334   0.170875857   0.172504489
0.174273737   0.177063328   0.180418256   0.18207516     0.18074514
0.177471331   0.174557694   0.172653233   0.169828849   0.164876119
0.159711226   0.159085154   0.166384682   0.179753875   0.193852951
0.204331964   0.209509647   0.210607756   0.210618832   0.211480047
0.212978998   0.213066459   0.210290803   0.205085384   0.197774904
0.187306602   0.172732717   0.154541271   0.135525313   0.120385056
0.115501502   0.123911278   0.138242685   0.152435293   0.166621277
0.182761863   0.200161429   0.215837931   0.227007892   0.233503065
0.238602567   0.246252716   0.257914259   0.271014693   0.280524537
0.283196999   0.279813637   0.274122676   0.269343341   0.265131994
```

Gray Level Co-occurrence Matrix

A GLCM is a matrix where number of rows and columns is equal to number grey levels G in an image [24]. It is defined over an image to be the distribution of co-occurring values in the given offset. It is a way of extracting second order statistical features. It is used to measure the spatial relationships between pixels. This method is based on the belief that texture information is contained in such relationships. The GLCMs are constructed by mapping the grey level co-occurrence probabilities based on spatial relation of pixels in different angular direction. Greycomatrix function formulates GLCM through the process of calculation of the frequency of pixels with grey-level value I that appear horizontally adjacent to the pixels having value j. Each element (i, j) in GLCM clearly states the frequency of the pixels with values I and their occurrence that appear horizontally adjacent to a pixels with value j.

Based on a scaled image version the Grey co matrix calculates the GLCM. Assuming that the image is binary in that case the greycomatrix function scales the images to two grey-level. However if these are intensity images, Greycomatrix function scales the image to eight grey-level. It can specify the number of Grey level Greycomatrix function to scale the image by using the ‘NumLevel’ parameter, and the grey co matrix scales the values using the ‘GreyLimits’ parameter.

For example consider the following GLCM of 4 by 5 images I (Fig.3.20 GLCM here shows Element (1, 1) with values 1 as only one instance is apparent in the image wherein two, horizontally adjacent pixels possess values 1 and 1. GLCM also shows Element (1, 2) with value 2 as there exists two instances in the images where two, horizontally adjacent pixels possess values 1 and 2. Greycomatrix function continues this process to fill in all the values in the GLCM.



1	1	5	6	8
2	3	5	7	1
4	5	7	1	2
8	5	1	2	5

	1	2	3	4	5	6	7	8
1	1	2	0	0	1	0	0	0
2	0	0	1	0	1	0	0	0
3	0	0	0	0	1	0	0	0
4	0	0	0	0	1	0	0	0
5	1	0	0	0	0	1	2	0
6	0	0	0	0	0	0	0	1
7	2	0	0	0	0	0	0	0
8	0	0	0	0	1	0	0	0

Fig. 3.20 Calculation of GLCM

There are 22 texture features associated with GLCM and are illustrated below.

Energy

This is also called uniformity or angular second moment. It measures the textural uniformity that is pixel pair repetitions. It detects disorders in textures and grasps a maximum value equal to one.

$$energy\ (ene) = \sum_i \sum_j g_{ij}\ ^2 \tag{3.5}$$

Entropy

This statistic measures the disorder or complexity of an image. The image is not texturally uniform then the value of entropy is large and many GLCM elements have very small values. Complex textures tend to have high entropy.

$$entropy\ (ent) = -\sum_i \sum_j g_{ij} \log_2 g_{ij} \tag{3.6}$$

Where g_{ij} is the gray level values of the element (i, j)

Contrast

It measures the spatial frequency of an image and difference moment of GLCM. In a contiguous set of pixels the difference between the highest and the lowest values will be determined. It measures the amount of local variation present in the image.

$$contrast\ (con) = \sum_i \sum_j (i-j)^2 g_{ij} \tag{3.7}$$

Variance

It is a measure of heterogeneity and is strongly correlated to first order statistical variable such as standard deviation. Variance increases when the gray level values differ from their mean.

$$variance\ (var) = \sum_i \sum_j (i-\mu)^2 g_{ij}\ where\ \mu\ is\ the\ mean\ of\ g_{ij} \tag{3.8}$$

Where μ is the mean value.

Homogeneity

If weights gradually lessen from the diagonal, resultant will be comparatively bigger for windows and have comparatively little contrast. The inverse of the contrast weight, with weights decreasing exponentially away from the diagonal is called the weight of Homogeneity.

$$homogeneity\ (hom) = \sum_i \sum_j \frac{1}{1+(i-j)^2} g_{ij} \qquad (3.9)$$

Correlation

Here correlation feature is nothing but the image gray tone linear dependencies that are in inherent. GLCM correlation is quite a different calculation from the other texture measures. It also has a more intuitive meaning to the actual calculated values: 0 is uncorrelated, 1 is perfectly correlated.

$$correlation\ (cor) = \frac{\sum_j \sum_j (ij) g_{ij} - \mu_x \mu_y}{\sigma_x \sigma_y} g_{ij} \qquad (3.10)$$

Autocorrelation

An autocorrelation function can be evaluated that measures the coarseness. The linear spatial relationships between primitives can be evaluated based on this function. Assuming that the primitives are large, there is a gradual decrease in the function once the distance increases however it decreases relatively quickly if texture contains small primitives. Though when primitives are periodic, with distance there is evident autocorrelation increase and decrease periodically.

$$p(x,y) = \left. \frac{1}{(L_x - |x|)(L_y - |y|)} \iint_{-\infty}^{\infty} I(u,v) I(u+x, u+v)\, du\, dv \middle/ \frac{1}{L_x L_y} \iint_{-\infty}^{\infty} I^2(u,v)\, du\, dv \right. \qquad (3.11)$$

Sum Average

$$sum\ average\ (sa) = \sum_{i=2}^{2N_g} ig_{x+y}(i) \tag{3.12}$$

Sum Entropy

$$sum\ entropy\ (se) = -\sum_{i=2}^{2N_g} ig_{x+y}(i) \log\{g_{x+y}(i)\} \tag{3.13}$$

Sum Variance

$$sum\ variance\ (sv) = \sum_{i=2}^{2N_g} (i - sa)^2 g_{x+y}(i) \tag{3.14}$$

Difference variance

$$difference\ variance = variance\ of\ g_{x-y} \tag{3.15}$$

Difference Entropy

$$difference\ entropy\ (se) = -\sum_{i=0}^{N_g-1} g_{x-y}(i) \log\{g_{x-y}(i) \tag{3.16}$$

Information Measures of Correlation

i) Information Measures of Correlation 1 (IMC1)

$$IMC1 = \frac{HXY - HXY1}{\max\{HX, HY\}} \tag{3.17}$$

ii) Information Measures of Correlation 2 (IMC2)

$$IMC2 = \sqrt{(1 - \exp[-2.0(HXY2 - HXY)]} \tag{3.18}$$

where,

$$HXY = -\sum_{i}\sum_{j} g_{ij}\log_2 g_{ij} \quad where\ HX\ and\ HY\ are\ entopies\ of\ g_x\ and\ g_y \tag{3.19}$$

$$HXY1 = -\sum_{i}\sum_{j} g_{ij}\log_2\{g_x(i)g_y(j)\} \tag{3.20}$$

Cluster shade

$$\mathrm{Shade} = \sum_{i=0}^{G-1}\sum_{j=0}^{G-1}\{i + j - \mu_x - \mu_y\}^3 * P(i,j) \tag{3.21}$$

Cluster Prominence

$$\mathrm{Prom} = \sum_{i=0}^{G-1}\sum_{j=0}^{G-1}\{i + j - \mu_x - \mu_y\}^4 * P(i,j) \tag{3.22}$$

A GLCM is a matrix where the number of rows and columns is equal to the number of gray level G in the image, where

$$\mu_x = \sum_{i=0}^{G-1} iP_x(i) \tag{3.23}$$

$$\mu_y = \sum_{j=0}^{G-1} iP_y(j) \tag{3.24}$$

Dissimilarity

$$\mathrm{Diss} = \sum P_{i,j} * |i - j| \qquad (3.25)$$

A total of 22 features using GLCM have been extracted for the preprocessed paragraph text images. The GLCM features resulted for a paragraph text image shown in Fig. 10 are given below

0.25840629	0.246959849	0.230972377	0.213029194	0.198423463
0.193196132	0.198278035	0.209219231	0.221585767	0.23292406
0.242989416	0.252481322	0.261031029	0.265989608	0.263900922
0.253542737	0.236820916	0.217508248	0.198921594	0.182628767
0.168905098	0.158409477			

Generalized Gaussian Density

The basic idea of the wavelet-based GGD method is to establish corresponding wavelet-based GGD model for a handwriting image which are considered as mathematical functions segregate data into separate frequency components. These components are then studied with a resolution that corresponds to its scale. The parameters α, β are regarded as the features of the handwritten image [30]. For each handwritten image, the GGD model cut the image using different frequency components are cut into regions. Each region is called as a wavelet subband. For each wavelet subband and the estimated parameters α, β which are optimal then the probability is maximized for improving accuracy of writer identification. The probability is estimated as,

$$P\ (\{\alpha, \beta\}/X) \qquad (3.26)$$

Each wavelet subband has coefficients as X={x1, x2……}. After the error probability is estimated the likelihood function is defined as,

$$L(x|(\alpha,\beta)) = log\prod_{i=1}^{L} p(x_i|(\alpha,\beta)) \qquad (3.27)$$

According to the langrange optimization, the likelihood equation is obtained as follows,

$$\frac{\partial L(X|\{\alpha,\beta\})}{\partial\alpha} = -\frac{\mathrm{N}}{\alpha} + \sum_{i=1}^{N} \frac{\beta|x_i|^{\beta}\alpha^{-\beta}}{\alpha} \tag{3.28}$$

After the likelihood function is estimated then α, β values are obtained using the following equations,

$$\frac{\partial L(x|\{\alpha,\beta\})}{\partial\beta} = -\frac{N}{\beta} + \frac{N\varphi(1/\beta)}{\beta^2} - \sum_{\mathrm{i}=1}^{\mathrm{N}} \left(\frac{\mathrm{x_i}}{\alpha}\right) \log\left(\frac{\mathrm{x_i}}{\alpha}\right) \tag{3.29}$$

Where

$$\varphi(z) = \tau\ (z)/\tau(z) \tag{3.30}$$

$$\hat{\alpha} = \left(\frac{\beta}{N}\sum_{i=1}^{N}|x_i|^{\beta}\right)^{\frac{1}{\beta}} \tag{3.31}$$

By substituting the equation (3.31) in equation (3.29), the estimation of the β is be calculated by using the following equation (3.32),

$$1 + \frac{\varphi(1/\hat{\beta})}{\hat{\beta}} - \frac{\sum_{i=1}^{N}|x_i|^{\hat{\beta}}\, log|x_i|}{\sum_{i=1}^{N}|x_i|^{\hat{\beta}}} + \frac{\log((\hat{\beta}/N)\sum_{i=1}^{N}|x_i|^{\hat{\beta}})}{\hat{\beta}} = 0 \tag{3.32}$$

Once the value of the β is calculated then it is substituted in the equation 4 to find the value of α. The, β values of each wavelet subband are combined and the mean is calculated. By this process α, β values are obtained for each pixel and finally 2025 features have been obtained for paragraph text image shown in Fig. 10 which are given below.

0.151807307	0.147478061	0.143761382	0.14205421	0.143882702
0.146922164	0.149071887	0.150651811	0.152423285	0.15517236
0.159014253	0.162570934	0.165009497	0.16676901	0.168375673
0.170062729	0.172824629	0.176930282	0.181119297	0.18364764
0.183531355	0.181510877	0.178373955	0.174477915	0.169964132
0.16514127	0.16133806	0.160145177	0.161867701	0.164988394
0.167365512	0.167387556	0.165040936	0.162051041	0.160474354
0.161175846	0.162567155	0.161604933	0.156439307	0.147301842
0.13599506	0.125406232	0.118467815	0.116388924	0.118112414
0.122584401	0.129735266	0.13846368	0.146417683	0.151868302
0.15412681	0.153411997	0.151262905	0.150140189	0.151565134
0.154829129	0.157831047	0.158682679	0.156722589	0.15310862
0.151206929	0.155229674	0.166000176	0.17927577	0.189308203
0.192369309	0.188599746	0.18253551	0.181585829	0.190752739
0.206627377	0.221736191	0.230991611	0.233413803	0.231421986
0.228986935	0.229600401	0.23380216	0.238000882	0.238051934
0.233121776	0.225687176	0.22000156	0.220393944	0.227926219
0.237760723	0.245210586	0.249419668	0.25182132	0.253797162
0.254338033	0.250796803	0.241859923	0.229366251	0.216260827
0.204021697	0.193841115	0.186326694	0.181353451	0.179141996

Contourlet Generalized Gaussian

In GGD, wavelet transform is used to decompose the image into subbands in different frequency and orientation. But wavelet is able to capture only limited directional operation which is important issue in image analysis. To overcome this problem, multiscale and directional representations can be used to efficiently capture the image's geometrical structures such as edges or contours. Contourlet transform [79] that is based on the proficient two-dimensional multiscale as well as directional filter bank can successfully process images that own a smooth contour. Here the contourlets are implemented using the double filter bank named Pyramidal Directional Filter Bank (PDFB). In PDFB, the laplacian pyramid is used to decompose the images into multiscale using 9-7 filter bank. Then the directional filter bank is used to analyze the multiscale into a four directional subbands. Multiscale and directional decomposition stages in the contourlet transform are independent of each other, because while using PBFB filter it gets a cascade structure. The parameters α, β of the GGD model is taken and used to represent the contourlet subband. The GGD is given as below,

$$p(x;\alpha,\beta) = \frac{\beta}{2\alpha\tau(1/\beta)} exp^{-1(|x|/\alpha)^{\beta}} \qquad (3.33)$$

Where $\tau(.)$ is the Gamma function, i.e.,

$$\tau(.) = \int_0^{\propto} exp^{-t} t^{Z-1} dt, Z > 0 \qquad (3.34)$$

Various methods are available to estimate the parameters α and β. Here the maximum likelihood estimation is used to estimate the parameters α, β by converting the subband image into multi dimensional vector and it is defined as given below,

$$L(x;\alpha,\beta) = log \prod_{i=1}^{L} p(x_i;\alpha,\beta) \qquad (3.35)$$

By using the following equation the features are extracted for the subband images.

$$\frac{\partial L(x;\alpha,\beta)}{\partial \alpha} = -\frac{L}{\alpha} + \sum_{1}^{L} \frac{\beta |x_i|^{\beta} \alpha^{-\beta}}{\alpha} \qquad (3.36)$$

In this process the values of the GGD is taken as input and the filters mentioned above are applied to extract the features of a given input image. The 225 features obtained for a paragraph text image shown in Fig. 10 are given below

0.179958135	0.18301225	0.185557618	0.18481557	0.179675029
0.171629876	0.163953486	0.159845802	0.159987935	0.161498218
0.161521558	0.159617146	0.156883536	0.154358574	0.152644925
0.152655233	0.156012821	0.1642499	0.175654678	0.186217805
0.193702758	0.198444554	0.202692842	0.209597178	0.218889658
0.225417966	0.223939387	0.212768773	0.194446843	0.175624273
0.166121675	0.170196858	0.182121614	0.194576832	0.203390147
0.208647464	0.21261274	0.216523066	0.219372982	0.219263808
0.215471097	0.209037144	0.202451401	0.198360364	0.197218987
0.196991312	0.19604319	0.194375371	0.191873016	0.188762429
0.186477716	0.18550329	0.183502377	0.178255334	0.170131056
0.162411561	0.157828676	0.156340569	0.157395529	0.159998149
0.162450558	0.163603974	0.163600262	0.163906672	0.166052509
0.170643841	0.17764123	0.186161135	0.194547053	0.20130046
0.20601535	0.20890258	0.208992919	0.204312958	0.193953522
0.178645823	0.161264194	0.148304738	2.546875	2.33984375
2.42578125	2.609375	2.4609375	2.5390625	2.33203125
2.41015625	2.35546875	2.5	2.546875	2.59765625
2.56640625	2.34375	2.58984375	2.578125	2.40625
2.51171875	2.46875	2.51953125	2.609375	2.453125

Directional Features

The normalized feature vector for classification is obtained using the directional features here. Here first the input image is characterized into four types, such as vertical line, horizontal line, left diagonal and right diagonal [80]. The values are calculated from the four directions, as fraction of the distance traversed across the image. If the transition is computed from left to right, a transition found close to the left is assigned a high value compared to a transition computed further to the right. A maximum value (MAX) is defined as the largest number of transitions that is recorded in each direction. If there are less than MAX transitions recorded, then the remaining MAX transitions are assigned values of 0. The transition value is calculated for a particular direction. To calculate the directional transition, the transition value is divided by a predetermined number. Here the predetermined number is 10. Thus eight features are obtained for one transition. Then this process is repeated for the remaining transitions and 256 features are obtained, using the following equation (3.37),

nrFeatures * nrTransitions * nrVectors * resampledMatrixHeight (Width) (3.37)

The feature values resulted for a paragraph text image shown in Fig. 10 are given below

2.34375	2.41796875	2.578125	2.53125	2.2890625
2.4296875	2.5546875	2.51171875	2.421875	2.40625
2.71875	2.4453125	2.41015625	2.51953125	2.44921875
2.671875	2.484375	2.4921875	2.30078125	2.3828125
2.59375	2.4609375	2.46875	2.34765625	2.51171875
2.53125	2.515625	2.37109375	2.47265625	2.25390625
2.5703125	2.5390625	2.33203125	2.3125 2.5	2.54296875
2.515625	2.43359375	2.55078125	2.61328125	2.37890625
2.4140625	2.44921875	2.484375	2.4140625	2.6171875
2.5859375	2.54296875	2.34375	2.46484375	2.46484375
2.421875	2.41796875	2.5390625	2.52734375	2.48046875
2.39453125	2.54296875	2.62890625	2.4765625	2.3828125
2.46875	2.54296875	2.5	2.546875	2.390625
2.59375	2.6171875	2.4609375	2.33203125	2.24609375
2.421875	2.55078125	2.56640625	2.3671875	2.40234375
2.3359375	2.3515625	2.73828125	2.43359375	2.2890625
2.578125	2.3984375	2.4765625	2.67578125	2.5
2.37890625	2.21484375	2.59765625	2.578125	2.390625
2.44921875	2.48828125	2.66015625	2.38671875	2.37890625
2.54296875	2.58203125	2.57421875	2.39453125	

The above five categories of features are captured by developing MATLAB code. A total of about 2784 features, forming a feature vector have been generated for all preprocessed paragraph text images in the corpus.

3.4. FEATURE SELECTION

The feature selection method used here is particle swarm optimization method. The PSO method is used to reduce the number of features obtained during the feature extraction. The feature selection is done to in order to speed up the processing rate and predictive accuracy**.** Here, the features are extracted for each and every pixel for a given input image. So to reduce the dimension of feature vector, feature selection method is used. If there are n number of features, then the threshold value of the feature selection is n<10. The features are reduced based on the weight assigned to each feature. After the weight is assigned to each feature, the features are selected in the descending order based on the weight of the features. A total of about 100 features are selected for each category of feature.

The number of features obtained using gabor filter during feature extraction is 256. Using PSO feature selection only 100 features are selected. The 2025 GGD features are reduced to 100 using PSO. Similarly, the 225 contourlet GGD features are reduced to 100 features. The directional features extracted during feature extraction process for an image is about 256. The PSO method is used to reduce the number of features to 100. All the GLCM features have been taken to represent the textural information of a handwritten image. Finally the feature vectors for paragraph text images are formed by pooling all 422 features. The count of features before and after feature selection is shown in Table 3.2.

Table 3.2 Count of Features Before and After PSO Feature Selection

Features	Number of Features Before PSO	Number of Features After PSO
Gabor	256	100
GLCM	22	22
GGD	2025	100
Contourlet GGD	225	100
Directional	256	100

3.5. DATASETS

Three independent training datasets have been developed as given below to build writer identification models.

Dataset based on character text images: This training dataset named TWINC has been created by extracting 26 features described in section 3.3.1, from the 30000 character level handwritten images. The set of 26 features forms a feature vector for which the class label is assigned from 1 to 300 as there are 300 writers. The training dataset TWINC with 30000 feature vectors is developed for building writer identification model.

Dataset based on word text images: As mentioned in section 3.2, the same discriminators are considered in case of word level handwritings and captured from 30000 word images to create the second dataset. The set of 26 features forms a feature vector for which the class label is assigned from 1 to 300 and the dataset named TWINW with 30000 instances has been developed.

Dataset based on paragraph text images: The third dataset is prepared based on paragraph level handwritten text images. The features as described in section 3.3.2 have been computed and a set of 30000 feature vectors have been created. The dimension of each feature vector here is 422. Since the corpus consists of 30000 paragraph images of 300 writers, for each feature vector the class label is assigned from 1 to 300. The training dataset named TWINP with 30000 instances is developed for building writer identification model.

The above three datasets is normalized using min-max normalization. Normalization is used here to fit the data values in a specific range. Min Max Normalization transforms a value A to B which fits in the range [C, D] and is given by the Equation 3.38. In this research work the values are normalized to fall within the range 1-3.

$$B = \frac{(A - min_A)}{(max_A - min_A)} * (D - C) + C \qquad (3.38)$$

3.6. TRAINING AND TESTING

A training set is used to fit the models and the validation set or development test set used to estimate test error for model selection. The test set or evaluation test set used for assessment of the generalization error of the finally chosen model. During supervised learning evaluation of the induced function is carried out using a separate set of inputs and the function values obtained are referred to as the testing set. Any hypothesized function is said to generalize when it guesses well on the testing set.

Testing and Evaluating Classifier Accuracy

Accuracy estimate is used to measure, how accurately a given classifier will be able to identify the class label of the future data. The accuracy of a classifier on a given test set is the percentage of test instances that are correctly classified by the classifier. Error rate is the proportion of errors made over the number of testing instances. The error estimate is more accurate when the test dataset is larger. The common techniques for assessing the classifier accuracy are

- Hold-Out Method.
- Cross-Validation Method.
- K-fold cross validation Method.
- Leave-one-out cross validation Method.

Hold-Out Method

The holdout method is the simplest kind of cross validation. The data set comprises of two sets, namely training set and testing set. The function approximator is apt for the training set exclusively. The function 'approximator' is required to predict output values for testing set data and prior to this has not seen the output values. Accumulated errors prior to this so that mean absolute test set error is achieved, then this is deployed for model evaluation. One of the key benefits using this technique is that it comparatively preferable to the residual method and computation time too is almost the same. Though evaluation here can possibly have a high variance and can be substantially dependent where the data points are resultantly part of the training set and end up in the test set. Hence evaluation can differ considerable based on the way division is carried out.

Cross-Validation Method

Cross validation is a model evaluation method that is better than residuals. The problem with residual evaluations is that one does not give an indication of how well the learner will do when it is asked to make new predictions for data it has not already seen. Another alternative to overcome this inherent issue is not to completely deploy the entire data set when training a learner. Some of the data is removed before training begins. Then when training is done, the data that was removed can be used to test the performance of the learned model on ``new" data. Whole class of model can be evaluated based on this methods are called cross validation.

K-fold Cross-Validation Method

K-fold cross validation is one way to improve over the holdout method. Data set consists of k subsets and the holdout technique has a repetitive cycle equivalent to k times. Every time, one among the k subsets becomes the test set and the remainder k-1 subsets become part of the training set. Average error that exists among k trials is then calculated. Using this technique is apt as the division of data is not that relevant. All the data point becomes a part of the test set exactly once, wherein the training set is k-1 times. When k increases forthcoming estimate variance reduces. Major drawback from this technique is that training algorithm has to again commence from the beginning scratch k times, in other words it means that it takes k times computation to make a conclusive evaluation.

Leave-one-out Cross Validation Method

This method is used in the field of machine learning to determine how accurately a learning algorithm will be able to predict data that it was not trained on. When using the leave-one-out method, the learning algorithm is trained multiple times, using all but one of the training set data points. The form of the algorithm is as follows:

- For k = 1 to R where R is the number of training set points
- Temporarily remove the k^{th} data point from the training set.
- Train the learning algorithm on the remaining R - 1 point.
- Test the removed data point and note your error.
- Calculate the mean error over all R data points.

This method is useful because it does not waste data. When training, all but one of the points are used, so the resulting regression or classification rules are essentially the same as if they had been trained on all the data points. The main drawback to the leave-one-out method is that it is expensive - the computation must be repeated as many times as there are training set data points. K-fold cross validation that has been directed towards the logical extreme, K equals N, given the data points in the set. Which infers that N separate times, the function 'approximator' is equipped to work almost all the data excluding a single point as well another point for whom the prediction has been made.

In the classification problem, it is identified as positives the class and as negatives the class. Doing so, the following standard definitions are obtained: 1) True Positives (TP): predicts class as class. 2) True Negatives (TN): predicts non-class as class. 3) False Positives (FP): predicts non-class as non-class. 4) False Negatives (FN): predicts class as non-class. As a consequence, there are several measurements that can be used for comparison and in the proposed work Accuracy, Precision, Recall and F – score to build the model are focused. They are defined below as measurements for a specific class.

$$\text{Recall } = \frac{TP}{TP+FN} \tag{3.39}$$

$$\text{Precision} = \frac{TP}{TP+FP} \tag{3.40}$$

$$\text{F-Measure} = 2.\ P.R/\ (P+R) \tag{3.41}$$

$$\text{Accuracy} = \frac{TP+TN}{TP+TN+FN+FP} \tag{3.42}$$

In this research, the hold out method is used wherein the three datasets presented in section 3.5 are divided into training and test datasets. 80% of the instances are used for training sets and 20% for test sets. These training datasets are employed for building the model and test datasets are made use for evaluating the models.

3.7 SUMMARY

This chapter describes the framework for learning the writer's identity constructed on Tamil handwriting. Handwritten documents written by the writers are scanned and segmented into words. Words are further segmented into characters for character level writer identification. The character writings in Tamil are analyzed and their describing features are defined. The paragraph writings in

Tamil are analyzed and their describing features are defined. The essential tasks such as corpus preparation, preprocessing, feature extraction, are also described in this chapter. Finally the writer identification problem is formulated as classification task and pattern classification techniques have been employed to solve the problem. The implementation of the writer identification model using SVM is discussed in the subsequent chapter.