RESEARCH ARTICLE OPEN ACCESS

Structural Run Based Feature Vector to Classify Printed Tamil Characters Using Neural Network

Karthigaiselvi. M[1], Kathirvalavakumar. T[2],
Research Centre in Computer Science, V. H. N. Senthikumara Nadar College, Virudhunagar-626 001, Tamil Nadu, India

ABSTRACT
Feature Extraction plays most crucial and important role in character recognition. The selection of stable and representative set of features is the main problem in pattern recognition. Because of font characteristics and style variation of machine printed Tamil characters, feature extraction remains a problem. Feature extraction involves reducing the amount of resources required to describe a set of data. In this paper, new method has been proposed to extract structural features from Machine printed Tamil characters using horizontal and vertical projections. Based on the structural properties of upper and lower modifiers, characters are divided into various categories and features are extracted accordingly. The extracted features from the real life degraded documents are classified to identify the characters. The system has been tested with printed Tamil characters and achieves 99.67% character recognition accuracy on average. Experimental results show that structure and category of the characters are identified by the proposed method for the regular characters of various sizes.

Keywords**:** *Feature extraction, Tamil characters, Structural features, Horizontal projection, Vertical projection, Neural network, Backpropagation*

I. INTRODUCTION

A pattern recognition system is normally decomposed into two sub-systems of feature extraction and classification. Feature extraction is a procedure of removing superfluous information from data, or equivalently, reducing the dimension of data. Once a feature extractor is derived, effective features can be extracted from the data, with which the efficiency and processing speed of the classification step can be improved (Min et al., 2016). Data representation is the heart of pattern recognition system design. Data are represented by a number of features which can be binary, categorical or continuous. Finding a good data representation is very domain specific and related to available measurements (Guyon and Elisseeff, 2006).

Features can be broadly classified into three different categories (Raj and Abirami, 2012) namely structural, statistical and hybrid features. Structural features are physically a part of the structure of the character such as straight lines, arcs, circles, intersections etc. Statistical features derived from the statistical distribution of points like zoning, moments, n-tuples etc. In hybrid approach, the combination of the above two approaches provide an efficient and complete description of the characters. Kahan et al., (1987) have described a system which recognizes printed text of various fonts and sizes of the Roman alphabet. Thinning and shape extraction are performed directly on a graph of the run-length encoding of a binary image. The resulting strokes and other shapes are mapped, using a shape-clustering approach, into binary features which are then fed into a statistical Bayesian classifier. Certain confusion classes are disambiguated through contour analysis, and characters suspected of being merged are broken and reclassified. Heutte et al., (1998) have expressed a description of handwritten characters based on the combination of seven different families of features such as intersections with straight lines, holes and concave arcs, extrema, end points and junctions.

Bag et al., (2014) have proposed a recognition method for Bangla compound characters. The Bangla compound characters are recognized by extracting the convex shape primitives and using template matching scheme. Singh et al., (2013) have discussed few feature extraction techniques namely distance profile, projection histogram, zoning density, zernike moments and hybrid feature extraction for Gurmukhi script and numerals. Seethalakshmi et al., (2005) have presented OCR system for printed Tamil text using Unicode. The features such as character height, width, number of horizontal and vertical lines, etc. are considered for classification. Back propagation and SVM based classifiers are used for subsequent classification purpose. Aparna et al., (2002) and (2003) have presented OCR system for Tamil newsprint and Tamil magazines. Connected component extraction method is used for character segmentation and RBF classifier is used for character recognition. Aradhya et al., (2008) have proposed multilingual character recognition system for printed South Indian scripts (Kanada, Telugu, Tamil and Malayalam) and English

documents based on Fourier transform and principal component analysis techniques. Aparna and Ramakrishnan (2002) have used Discrete Cosine Transform (DCT) coefficients as feature vector to recognize the Tamil characters taken from magazines and novels. Nearest neighbour classifier was used for the classification of the symbol. Arora et al., (2008) have used four feature extraction techniques namely, intersection, shadow, chain code histogram and straight line fitting for handwritten Devnagari character recognition. Printed Urdu script is recognized using a combination of topological, contour and water reservoir concept (Pal and Sarkar, 2003).

Oriya script characters are recognized (Chaudhuri et al., 2002) using a combination of stroke and run-number based features, along with features obtained from the concept of water overflow from a reservoir. Rocha and Pavlidis (1994) have proposed a method for recognizing multi-font printed characters using the structural features namely convex arcs, strokes, singular points and their relationships. In (Trier et al., 1996), a survey on feature extraction methods for offline isolated character recognition is reviewed. The feature extraction methods include template matching, deformable templates, unitary image transforms, graph description, projection histograms, contour profiles, zoning, geometric moment invariants, zernike moments, spline curve approximation, fourier descriptors, gradient and gabor feature extraction. Kuo and Agazzi (1994) have created four component observation vectors for each pixel to recognize keywords embedded in a poorly printed document. The first component is the pixel value that can be 0 or 1. Second component is derived from three adjacent pixels along the same row, centeredaround the current pixel. The third and forth components are relative position of the pixel in its row and column.

Dedgaonkar et al., (2012) have discussed about feature extraction techniques such as projection method, border transition technique, and zoning method for character recognition. Bataineh et al., (2012) have proposed a global feature extraction method based on the statistical analysis of behaviour of edge pixels in binary image. Abubacker and Raman (2011) have proposed an approach for structural feature extraction for distorted Tamil character recognition. Kunte and Samuel (2007) have used Hu's invariant moments and Zernike moments to extract features of printed Kannada characters. Saba et al., (2011) have presented an improved fused feature extraction technique. They have used projection profile and transition feature extraction techniques to extract adaptation of the 2-Dimensional Principal Componant Analysis (2DPCA) algorithm for extracting features of online Tamil characters in a subspace. Karthikeyan (2013) has explored Tamil character recognition through the Hilditch algorithm. Chaudhuri and Pal (1998) have used stroke features to recognize printed Bangla OCR sytem. Das et al., (2015) have proposed soft computing paradigm embedded within the framework of existing statistical features from handwritten cursive characters. Sundaram and Ramakrishnan (2009) have used an two pass approach for recognition of handwritten Bangla characters.

A new feature extraction method is proposed using horizontal and vertical projection techniques based on structural properties of the Tamil character which involves simple procedure that provide a good representation of the character at different level of granularity even though it involves different fonts and styles. This paper is organized as follows: section 2 describes characteristics of Tamil script, section 3 describes the proposed Feature extraction method, section 4 describes Neural based feature classification and section 5 describes experimental results.

II. CHARACTERISTICS OF TAMIL SCRIPT

Tamil is a South Indian language spoken widely in Tamil Nadu, India. The Tamil script has 12 vowels, 18 consonants, 1 special character and 216 consonant based vowels. Hence a set of 247 symbols exists in the Tamil script. The basic characters of Tamil script are shown in Fig. 1. The modifier symbols specified in Fig. 2 occupy specific positions around the base character. While the modifiers that get added on the left or right side of the base character, it remain disjoint from the base character but the modifiers that are added either at the top or bottom of the base character is connected with that and spread to the upper and lower zones respectively

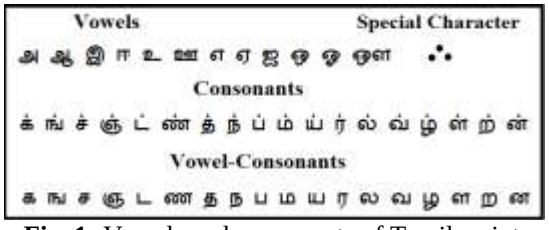


Fig. 1: Vowels and consonants of Tamil script

Writing style of the Tamil character is from left to right. The concept of upper/lower case character is absent in Tamil language.

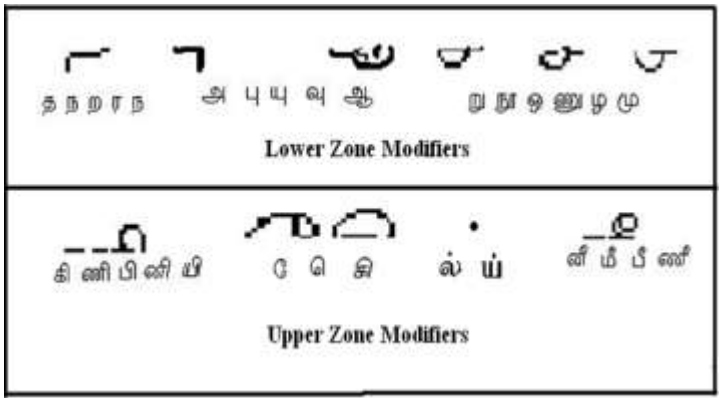


Fig. 2: Modifiers of Tamil script

Different zones are exemplified in Fig. 3 using few characters. Each character can be partitioned into three horizontal zones namely upper, middle and lower zones. The upper zone denotes the portion above the mean line, the middle zone covers the region below the mean line and above the base line, and the lower zone is the portion below the baseline. The upper zone is separated from the middle zone of a text line by mean line, and the middle zone is separated from the lower zone by base line. The horizontal line that passes through maximum number of uppermost points is the mean line and the horizontal line that passes through maximum number of lowermost points is the base line. The upper and lower zones may contain parts of modifiers. The characters can be categorized as follows:

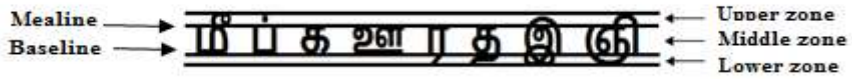


Fig. 3: Characters span over different zones

a) Upper zone characters -Characters spanned into upper and middle zone
An arc like convexity sub-symbol is seen almost at the upper part (upper zone) of the character. All characters of these types are called as "Kuril" letters. Totally 18 characters (*கி, ஙி, சி, ஞி, ணி, தி, நி, பி, மி, ரி, யி, லி, வி, ழி, ளி, றி, னி,*) containing this feature. The characters *கி, ஙி, சி, ணி, பி, மி, யி, லி, வி, ளி,* **and** *னி* of this type occupy upper and middle zone. Loop with the convexity sub-symbol " " present only at the upper part of characters are called as "Nedil" letters. Totally 18 Nedil characters are containing this feature, but the characters *கீ, ஙீ, சீ, ணீ, பீ, மீ, யீ, லீ, வீ, ளீ,* and *னீ* of this type occupy upper and middle zone.

b) Middle zone characters - Characters present in middle zone only
In Tamil Script, consonant vowels *க, ங, ச, ட, ண, ன, ப, ம, ய, ல, ள, வ* and the vowels *ஈ,ஊ, ஊ, எ,* , , and are occupying middle zone only.

c) Lower zone characters - Characters spanned into middle zone and lower zone
The vowel character *அ, ஆ, ஏ* and the compound characters *ஞ, த, ந, ர, ற, ழ, டி, கு, ஞு, ணு, து, நு, பு, மு, யு, ரு, லு, வு, ழு, ஞு, று, நூ, சூ, ஞூ, ணூ, தூ, நூ, பூ, மூ, யூ, ரூ, லூ, வூ, ழூ, ஞூ, றூ, னூ* occupy middle and lower zone.

d) Three zone characters - Characters spanned into upper, middle and lower zones
The Nedil Characters *ஞீ, தீ, நீ, ரீ, ழீ, றீ, டு,* Kurill characters *இ, ஞி, தி, நி, ரி, ழி, றி, டு* and the vowel characters *இ, ஐ, ஒ, ஓ* occupy all the three zones.

III. PROPOSED METHODOLOGY

3.1 Preprocessing

Gray scale image is essential in order to eliminate noise areas, smooth the background texture and to highlight the contrast between background and text areas. So the input RGB image is converted into gray scale image. Smoothing is performed on that grayscale image to reduce the amount of high frequency noise using Wiener filter. The well-known adaptive thresholding Niblack's approach (1986) is used for binarization. In this method

threshold value is based on local mean and standard deviation of all pixels in the window.

3.2 Skew detection and correction

The gray scale image pixel values are used in detecting skew angle. The upper left corner point is compared with upper right corner point to find out whether the document is left skewed or right skewed. A reference line is drawn from upper left corner point to lower left corner point. The skew of the document angle (θ°) is found by finding the orientation of the reference line. The document is rotated (θ°) in the anti-clockwise direction to correct the skew. Here point refers (row, column) of particular position. The proposed system can handle documents with skew angle between +45º and -45º.

3.3 Slant removal

Slant removal technique proposed by Parvez and Mahmoud (2013) is used to remove the slant on the segmented line when the characters are slanted to the right or left depending on the font style. In this technique, the contour of the threshold image is found, the edges (chain of connected pixels) of stroke are obtained and orientation of those edges close to the vertical is used for overall slant estimate. Slant is estimated by finding the average angle of 'near-vertical strokes'. An affine transformation with the estimated slant angle corrects the slant in the input image.

3.4 Projection

Projection techniques are extensively used in document image analysis and computed tomography. Basically horizontal and vertical projections are used. In case of binary image, horizontal and vertical projection profile is the sum of ON pixels along every row and every column of the image respectively. The horizontal and vertical projection P_h (r) and P_v (c) of the image f (r, c) are defined respectively as

$$P_h(r) = \sum_c f(r, c) \text{ , for all r,}$$
$$P_v(c) = \sum_r f(r, c), \text{ for all c,}$$

where r and c represent position of row and column respectively.

3.5 Tetra Bit Generation

The method proposed by Abirami and Murugappan (2011) is used for generating tetra bit. Two horizontal and two vertical bisections over the character produces tetra zones. Spaces on both sides of the characters are trimmed to extract the exact black pixel density. Character has been divided into nine zones and black tone & white tones of every zone have been grabbed. Bits for every zone have been generated based on its black pixel or white pixel density in that particular zone. When there is a domination of white pixels over black pixels corresponding zone value is set to 1 otherwise the zone value is set to 0. Sample character 'கி' with the grid of size 3 X 3and tetra bits generated for that character are illustrated in Fig. 4.

1	0	0
0	0	0
0	0	0

(a) **(b)**

Fig. 4: (a) Grid formation for a sample character (b) Tetra bits

3.6 Line Segmentation

To segment the text lines from the document image, the horizontal projection profile of the document image is calculated. White space between text lines is used to segment the text lines. In printed Tamil script, sometimes lower zone characters of a line touches the upper zone characters of next line, thus producing multiple horizontally overlapping lines. The horizontally overlapping lines make the line segmentation more difficult. It becomes difficult to estimate the exact position of a row which segments a line from the next line because of horizontal overlapping. To segment this kind of overlapped lines Projection Based Lines Segmentation (Kathirvalavakumar and Karthigaiselvi, 2013) is used. Observations reveals that rows with modifiers are with minimum projection values and there is a possibility to have two or more overlapped lines when the strip has greater projection value than the one-third of average projection values. Hence rows with modifiers are not considered for segmentation but a row with minimum horizontal projection is identified from the remaining rows and then the overlapped lines are separated into individual lines.

3.7 Word segmentation

Vertical projection profile is used for word segmentation. In the first step the distance between adjacent characters in the text line image are computed. In the second step the computed distances are classified as either inter-word distances or inter-character distances using threshold value. The distances between words are always larger than

distances between characters. Words can be segmented by comparing the distances with a suitable threshold. The threshold is defined as

$$Threshold = \frac{Sum\ of\ the\ distances\ of\ adjacent\ characters}{Number\ of\ distances}$$

When the distance value is greater than the threshold it is treated as a word gap otherwise it is a character gap. Words are segmented using the word gap.

3.8 Character Segmentation

Character segmentation is done after the individual words are identified. Vertical whitespaces serve to separate successive characters. Character segmentation algorithm namely SBCS (Structure Based Character Segmentation) is developed on the basis of structure of the Tamil characters and using the vertical projection method to split the words (Image of sequential characters) into sub images of individual characters. In printed Tamil script sometimes body of one character touches the body of another character thus producing touching characters. By experimental analysis, to identify the touching characters char_threshold (CT) has been defined as 175% of minimum width of the separated character. If the width of the separated character is greater than CT then the character can have two or more connected components. Touching characters are treated as a single character and thus leads to failure in character recognition phase. A column with minimum vertical projection is used to segment the touching characters. It has been identified from the experiments that minimum vertical projection values are appeared on the left most columns, right most columns and also appeared on the touching places. It makes difficult to estimate the exact position of a column which segments a character from the next character when the characters are touching. To estimate the correct segmentation column the leftmost and rightmost columns of the characters are not considered for processing and a column with minimum vertical projection is identified from the remaining columns. Based on this information and the structural properties of the leftmost character of the touched characters, the leftmost touching character is separated into individual character. After segmentation if the remaining right portion of the touched characters are still having more than one character the same procedure is to be applied until all the characters are separated.

Zone Identification

Upper zone characters have nonzero projection for all rows above the meanline. Lower zone characters have nonzero projection for all rows below the baseline. If the character is not belonging to upper zone or lower zone then it is treated as middle zone characters.

3.9 Feature Extraction

Feature extraction is a transformation of input data into a reduced set of features. It can be defined as the process of extracting distinctive information from digitized characters. The extracted features contain relevant information from the input data and are represented as a feature vector, which becomes its identity. Feature extraction is done after the individual characters are segmented using SBCS procedure. Selection of a feature extraction method is probably the single most important factor in achieving high recognition performance in character recognition systems (Raj and Abirami, 2012). The feature extraction stage analyses a character and selects a set of features that can be used to uniquely identify the character.

Tamil characters have many lines and curves which cross and intersect each other to form points of intersections or branching points. The positions of these branching points are considerably distinct for each character. It becomes difficult to extract feature vector from the Tamil character. Various feature extraction methods are used for extracting features of printed characters. In this work, structural run based feature vector technique (SRBFV) is proposed to extract structural features from printed Tamil characters. Structural features are less sensitive to character size.

Every character can be viewed as a collection of vertical columns. Collection of consecutive vertical columns containing at least one ON pixel is strip. Removal of blank rows and columns at the beginning and ending of the character is trimming. Number of transitions from foreground pixels to background pixels in horizontal direction from left to right symbolizes Z-RUN feature. Based on the observation, it has been identified that every Tamil character consists of maximum of four loops. Background connected component labelling technique existing in matlab is used for the detection of loops. Every character image is split into one or two strips to find the upper and lower part of the image. The upper and lower zone may contain some portion of modifiers. It has been identified that upper zone modifiers are fallen into four categories as shown in Table 1 and lower zone modifiers are fallen into fourteen categories as shown in Table 2. Identifying or recognizing Tamil characters needed

to identify modifier features existing on lower and upper part of the characters. The categories of upper and lower modifiers are checked with upper and lower modifiers of separated characters. If the feature value is one then the corresponding category is present otherwise the category is not present in the specified character.The features namely number of loops, number of objects, number of runs at first row, number of runs at last row, number of vertical lines and number of horizontal lines, tetra bit features, height of the run at last column and width of the last row of the trimmed image are extracted to identify and distinguish every middle zone portion of the character. The generated features are formed as vector for further processing. An overview of SRBFV algorithm is given in the Fig.5.

Table 1: Categories of Upper Zone Modifiers

Categories	Modifier Symbols	Sample Character images with the modifier
Type 1	◦	க், ங், ன், ப், ம், ய்
Type 2	◦	வி, ீ, னீ, யீ, மீ
Type 3	◦	கி, ஙி, சி, ணி
Type 4	◦	டு

Table 2: Categories of Lower Zone Modifiers

Categories	Modifier symbols	Sample character images with the modifier
Type 1	◦	கு, கூ, ஙு, ஙூ
Type 2	◦	ெ, ே
Type 3	◦	கை, ஙை, ை
Type 4	◦	ஔ
Type 5	◦	டி, டீ, டு, டூ
Type 6	◦	தெ, ஜெ, து, நு, ஔ, று
Type 7	◦	தா, ஜா, தீ, நீ, ஔ, றீ, ஹா
Type 8	◦	ஔ
Type 9	◦	நு, பூ, யூ, ஹ
Type 10	◦	ழு
Type 11	◦	மு
Type 12	◦	ஐ
Type 13	◦	அ
Type 14	◦	ஆ

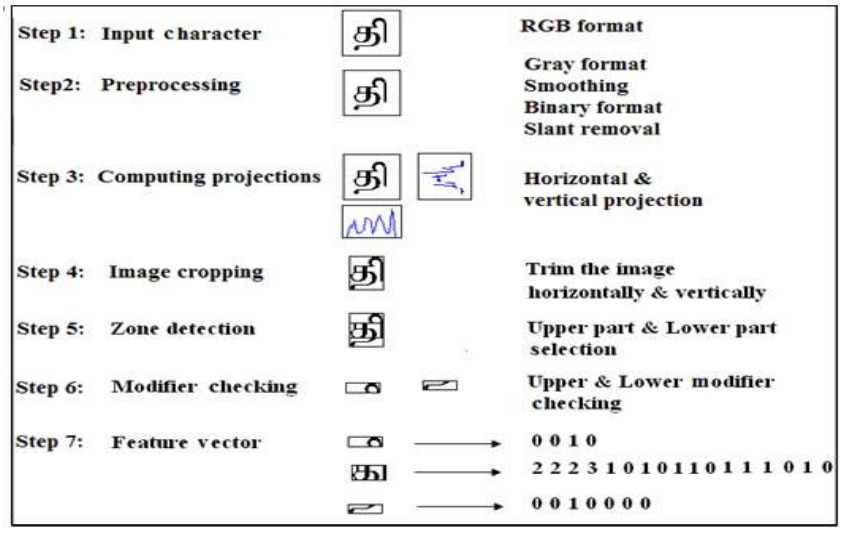


Fig. 5: Processing stages of SRBFV technique

3.9.1 Procedure for upper modifier checking

1. Split the upper part from the trimmed image (Portion above the meanline).
2. Find number of ***Z-RUN*** in each row and find starting and ending positions of each ***Z-RUN.***
3. Trim the upper part on all sides by finding horizontal projections for the upper part.
4. **Category 1- Dot modifier**
If each row has single ***Z-RUN*** then the upper part has small circle modifier.
5. **Category 2- Loop modifier**
If there is a loop then loop modifier occurs.
6. **Category 3 – Convex arc modifier**
Upper part has convexity arc if it satisfies the following conditions,
i. Not more than two ***Z-RUN.***
ii. White pixels counts in each row are in ascending order.
iii. No row has horizontal projection as width of the upper part.
iv. Only one object.
v. The ending position of ***Z-RUN*** of first row is lesser than the ending position of last ***Z-RUN*** of last row.
7. **Category 4 – Arc with centre vertical line modifier**
If number of ***Z-RUN*** of each row is in non decreasing order and also if it is as 1,1,1…, 2,2,2…, and 3,3,3… then the modifier is arc with centre vertical line. Here the occurrence of number of 1, 2, and 3 may differ.
8. If any one of the above category occurs then it shows occurrence of upper zone.

3.9.2 Procedure for lower modifier checking

1. Split the lower part from the image (Portion below the baseline).
2. Trim the lower part using horizontal and vertical projection.
3. Find the number of ***Z-RUN*** in each row and also find starting and ending positions of each ***Z-RUN.***
4. **Type 1**
i. Follow the procedure specified in Category 3 of 3.9.1, but here white pixel counts are in decreasing order and ending position of last run in the first row is greater than the ending position of last ***Z-RUN*** of last row.
ii. The number of ***Z-RUN*** in the last row to be 1 for this concave arc.
5. **Type 2**
Each row with single ***Z-RUN*** proceeds with RUN of white pixels. The run length of white pixels of each row is in decreasing order
6. **Type 3**
i. Each row with single ***Z-RUN***.
ii. Starting and ending position of Z-***RUN*** in each row is same.
7. **Type 4**
Modifier with two loops.
8. **Type 5**
i. Each row with single ***Z-RUN***.
ii. Run length of the ***Z-RUN*** of last row not equal to the width of the modifier.
iii. Run length of middle row ***Z-RUN*** is equal to width of the modifier or three fourth of the width of the modifier.
9. **Type 6**
i. Single loop.
ii. Run length of any ***Z-RUN*** does not equal to the width of the modifier.
10. **Type 7 and Type 8**
i. Single loop.
ii. Single ***Z-RUN*** in the last row and run length of the ***Z-RUN*** not equal to the width of the modifier.
iii. Run length of ***Z-RUN*** above the middle row is equal to the width of the modifier or greater than the three fourth of the width of the modifier and ending position ***(L)*** of that ***Z-RUN*** is connected with a vertical line.
iv. If ***L*** is equal to width of the modifier then it is of type 7 else type 8.
11. **Type 9 , Type 10, Type 11 and Type 12**
i Let ***k*** represent the maximum number of ***Z-RUN*** in a row, ***m*** represent number of rows having ***k Z-RUN***s and ***n*** represent number of rows having lesser than ***k Z-RUN***.
ii If ***m>=n*** then consider first and k^{th} ***Z-RUN*** of each row for processing, else consider first and $(k\text{-}1)^{th}$ ***Z-RUN*** of each row for processing.
iii Compare the first ***Z-RUN*** of each consecutive rows having different values and ***k*** or $(k\text{-}1)^{th}$ ***Z-RUN*** of the consecutive rows having different values.
iv If both ***Z-RUN***s are in descending order then it is of Type 11 else if both ***Z-RUN***s are in increasing order then the character is of Type10 else if first ***Z-RUN*** is in decreasing order, ***k*** or $(k\text{-}1)^{th}$ ***Z-RUN*** is in increasing order and only one ***Z-RUN*** in the last row then the character is of Type 9 else if first ***Z-RUN*** is in decreasing order, ***k*** or $(k\text{-}1)^{th}$ ***Z-RUN*** is in increasing order and two ***Z-RUN*** in the last row then the character is of Type 12.
12. **Type13**
i Modifier with no loop.
ii Two objects.
13. **Type 14**
If lower part is not belonging to any type specified in lower zone modifiers but have projection values then it is treated as Type 14.

3.9.3 Procedure for Tetra bit generation

1. For every zone of a character, traverse along x direction (x=0…w) for every y (y=0…h), where h=height and w= width.
2. For every occurrence of black pixel along the traversal, increment the black pixel counter (bpc) by 1.
3. For every occurrence of white pixel along the traversal, increment the white pixel counter (wpc) by 1.
4. Repeat steps 1 to 3 for every zone of the character.
5. Set the value of zone n to 1 when the ratio of wpc over the total number of black and white pixels in zone n exceeds 1.
6. Set the value of zone n to 0 when the ratio of bpc over the total number of black and white pixels in zone n exceeds 1.

Middle zone and three zone features

The features namely number of loops, number of objects, number of runs at first row, number of runs at last row, number of vertical lines and number of horizontal lines, tetra bit features, height of the run at last column and width of the last row of the trimmed image are extracted to identify and distinguish every middle zone character. The features obtained from upper zone, middle and lower zones provide the features for three zone characters. The feature vector is generated after all features are extracted by SRBFV.

4. Neural based classification and recognition

Single hidden layer feedforward neural networks are considered for character classification and recognition. Characters belonging to each zone are using pair of neural networks, one for classifying them as Tamil character or not and another for recognition of the classified characters. In the classification phase punctuation marks (period, comma, question mark, exclamation mark, colon, semicolon, hyphen, dash, parentheses, brackets, ellipses, apostrophe, quotation marks, slash), numerals, Aayuthaezhuthu and grantha consonants used in Tamil scripts are treated as other characters. After classification only Tamil characters are considered for recognition and other characters are ignored by the system. The features of upper, middle, lower and three zone characters are used to train the networks. Number of features of the corresponding network decides number of input neurons of the network. In the classification neural network one neuron is used in the output layer but 'k' number of neurons are used in the output layer of the recognition neural network, k derives from n, where n ($<= 2^k$) represents number of characters belonging to the particular zone. Fig. 6 illustrates various components of feature classification and character recognition steps. Fig. 7 illustrates procedure for character classification and recognition for a sample character.

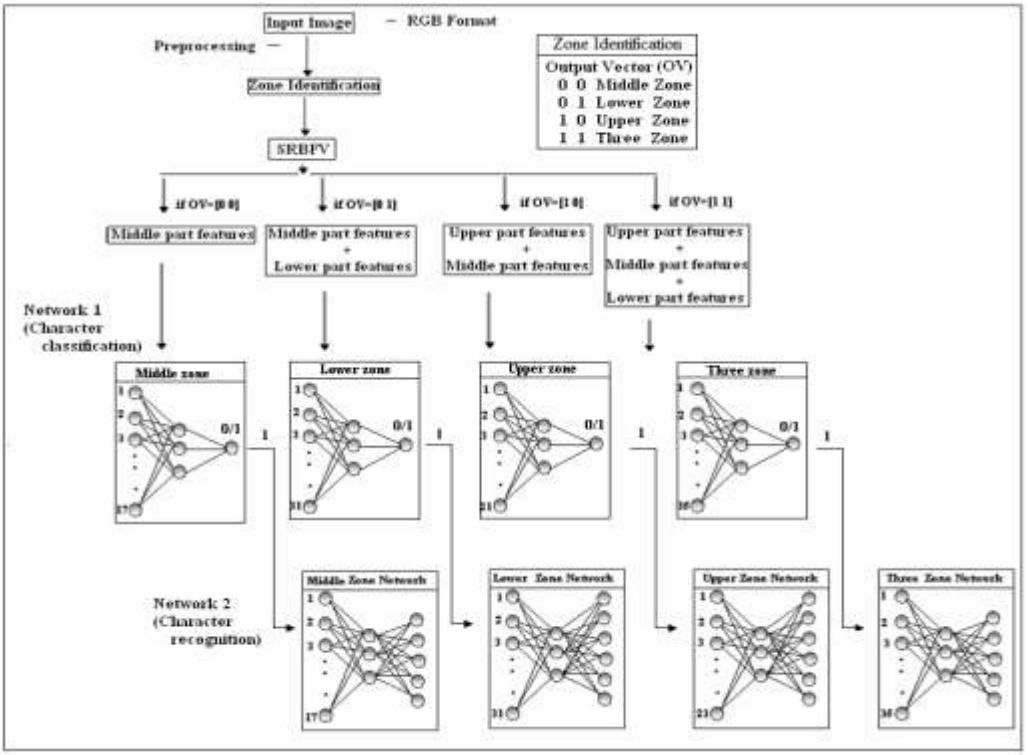


Fig. 6: Components of character classification and recognition phase

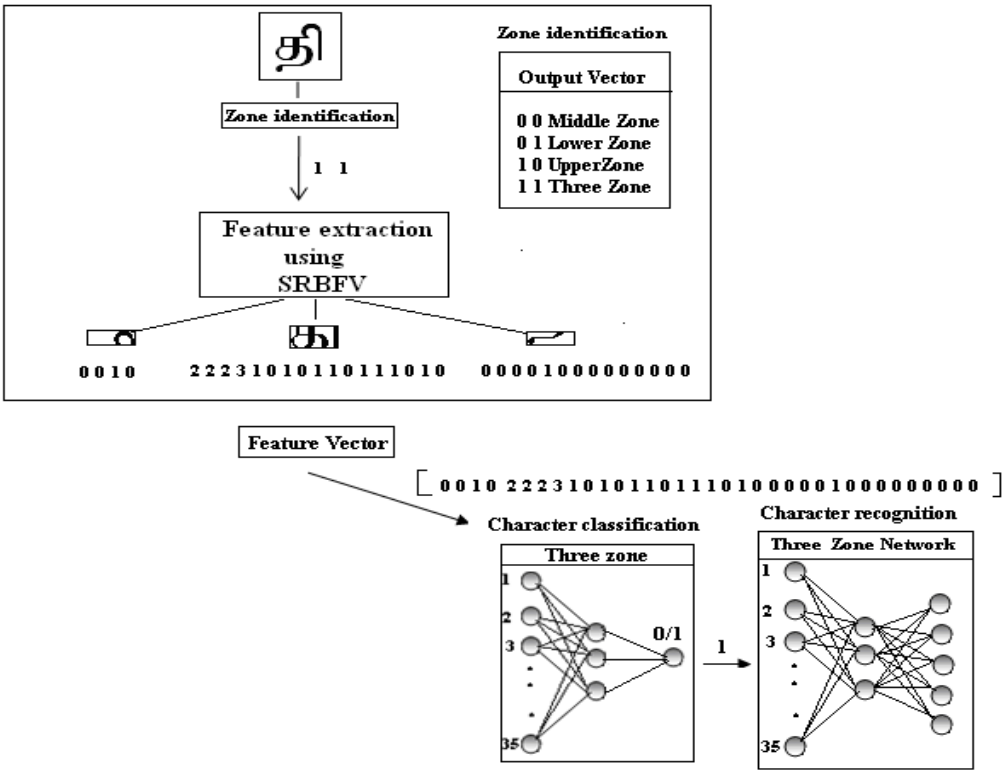
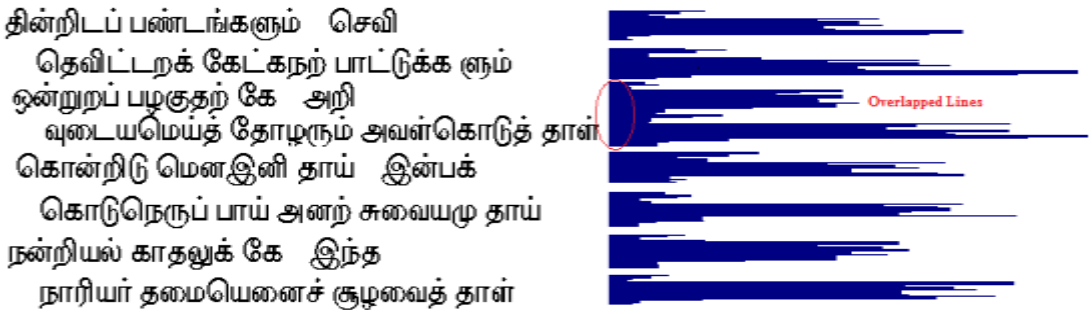


Fig. 7: Character recognition phase

IV. . EXPERIMENTAL RESULTS

Experiments are carried out with images of printed Tamil documents obtained from different Tamil literary periodicals. Local database is created and collected from (32-37) with different fonts and styles. Documents are with single column text regions and with pdf format. These documents are initially converted into jpeg image format. Some of the documents contain good quality printing; some others are of inferior printing and paper quality etc. The documents are with many overlapped lines and touched characters with different fonts and styles. 1000 different documents have been taken with different level of skew, slant as well as size. 700 documents are taken for training and all 1000 documents are used for testing. The proposed method is implemented in Matlab 13. The neural networks are trained for 25 different times after fixing the learning parameters. The average results are tabulated here.

Fig. 8a shows one of the documents considered for experiment. It contains even number of overlapping lines. Fig. 8b shows the histogram of horizontal projection of the document. Overlapped lines are marked with a circle. The document has 8 lines but only 7 line breaks are recognized by the histogram and is shown in Fig. 9a. It shows that the document is with overlapped lines. The segmented lines obtained from the proposed procedure are shown in Fig. 9b.



(a) (b)

Fig. 8: (a) Original image of document1 (b) Histogram of document 1

(a) (b)

Fig. 9: (a) Strips obtained using horizontal projection (b) Identified Lines using proposed method

A segmented line in document1 and its histogram of vertical projection profile are shown in Fig. 10(a) and 10(b) respectively. The space between the words is greater than the space between the characters is shown in the histogram of vertical projection profile. Segmented four words are shown in Fig. 11. Words are separated successfully from all the thousand documents.

Fig. 10: (a) Original image of line1 (b) Histogram of line1

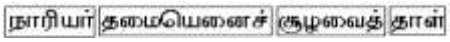


Fig. 11: Identified words using the proposed method

Vertical projection segments the word தமையெனைச் into 7 partitions த(14), ம(20), மெ (31), ய(15) , ன(20), ன (21) and ச் (13) with CT(27) is shown in Fig. 12 (width of the characters within the parentheses specifies the width of the character). The third partition has greater width value than CT. It implies that the third partition மெ is with touched characters and it requires segmentation. As a result of splitting procedure the touched characters are split into individual characters ம and ெ. Table 3 summarizes the results obtained by line, word and character segmentation procedures.

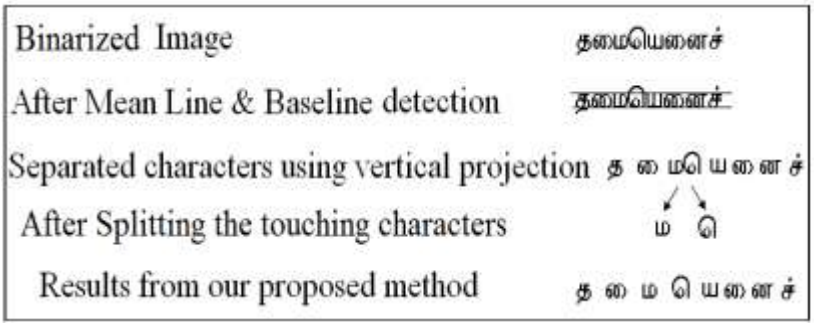


Fig. 12: Splitting of touching characters

Table 3: Summary of segmentation results

Segmentation procedure	**Number of documents : 1000**		**Accuracy**
Line (PBLS)	# of lines	32507	99.91%
	#of segmented lines	32480	
Word	# of words	174892	99.94%
	# of segmented words	174800	
Character (PSBCS)	# of characters	1072015	99.98%
	# of segmented characters	1071865	
	# of touched characters	2000	92.5%
	# of segmented touching characters	1850	

After the separating the characters each character image is trimmed; vertical and horizontal projections are applied on the trimmed characters to identify feature vectors. During this process, Upper portion of the image is passed to upper modifier checking procedure. In Table 4, the first row shows the result of the character . All rows of the upper part are with zero projections it confirms that the input image doesn't have upper modifier. The white pixel (background) count in between two transitions in all rows of the lower part of the character are 15, 12, 8, 0, 0 and 0; also they are in descending order; ending position 22 of last Z-RUN in the first row is greater than the ending position 15 of Z-RUN of last row and the number of Z-RUN in the last row is 1, these confirm the existence of concave arc in lower zone. From these results the character '**கு**' is categorized as Type 1 of lower zone character. The character '*பூ*' and '*ழ*' shown in second and third rows of Table 4 are not having upper zone. The features corresponding to loop along with concave like arc of Type 9 and lower arc of Type 11 are found in the lower part of the characters *பூ* and *ழ* respectively. The rows 4, 5 and 6 in Table 4 show the experimental results of the upper zone characters *க்*, *ஹி* and *சீ* respectively. All rows in the upper part of the character '*க்*' has a continuous ON pixels namely Z-RUN, which confirms the existence of dot modifier present at the upper part. The following upper zone characters *க், ங், ச், ட், ண், ப், ம், ய், ல், வ், ள், ன்,* and three zone characters *ஞ், த், ந், ர், ழ், ற்* have single Z-RUN in all rows of the upper part. It confirms the existence of upper zone part in these characters.

During the experiments, the following characteristics are observed from the character *ஹி* specified in fifth row of Table 4. It has 5 rows in its upper part. Number of transitions found in row1 to row 5 are 1, 1, 1, 2 and 2 respectively and white pixel (background) counts in between two transitions on each row are 0, 0, 0, 9, and 10; those are in ascending order. The width of the upper part is 15 and the horizontal projection values of upper part are 6, 10, 12, 9 and 7. Here no row has horizontal projection value equivalent to the width of the upper part. The upper part is identified with single object. The ending position of first row is 10 which less than the ending position of last row that is 15. Similarly during execution, the same conditions are found in the characters *கி, ஙி, சி, ஞி, பி, மி, யி, லி, வி, ளி,னி* which confirm that these are all of category 2 that is modifier convex arc in the upper zone. The character in sixth row of Table 4 and the other characters *கீ, ஙீ, சீ, ஞீ, பீ, மீ, யீ, லீ, வீ, ளீ, னீ* are found to have a loop in their upper part structure during experiment. It shows that these characters are with upper zone. From the lower part selection procedure, the characters *க், ஹி, சீ* specified in rows 4, 5 and 6 of Table 4 are identified as characters without lower part as there are zero projection. All rows in the upper part of the character in seventh row of Table 4 have Z-RUNs in non decreasing order 1, 1, 1, 2, 2, 2, 3 and 3 from top portion and are not greater than 3. These confirm the existence of an arc with centre vertical line modifier. It has been found that first three rows of the lower part of the character have two transitions and the remaining three rows have single transition. The white pixel (background) count in between two transitions in all rows of lower part of the character are 12, 9, 6, 0, 0 and 0 those are in descending order; ending position 18 of last Z-RUN in the first row is greater than the ending position 13 of Z-RUN of last row and the number of Z-RUN in the last row is 1, which confirms the existence of convex arc modifier present at the lower part. The existence of upper and lower zones mark that it is of three zone character.

The character shown in eighth row of Table 4 belongs to Alankaram font. The character *ர* is not having upper zone because the features of the modifiers loop, convex, dot and arc with centre vertical line are not resulted in the experiment. The character has single Z-RUN in each row of the lower part; this satisfies the properties Type 3 and Type 5 of lower part. In Type 3, ending positions of each Z-RUN in each row must be same but here the ending positions are 10, 3, 4, and 3. Number of pixels of Z-RUN of last row is 3 which is not equal to the width of the modifier hence not of Type 5. It shows that the lower part does not satisfy the property of any type. Number of pixels of Z-RUN at middle row in the lower part is 4 which is not equal to the width 10 of the modifier. These confirm the absence of lower and upper part in the character. So this character is identified as middle zone character, but actually this character is of lower zone. Hence the experimental result contradicts the actual feature.

The character specified in ninth row of Table 4 is of Agni font. During experiment, the upper part haszero projection and hence it is not upper zone character. The lower part below the base line is separated from the character. The white pixel count in-between two transition in all rows of lower part of the character are 12, 8 and 0 which are in descending order; ending position 18 of last Z-RUN in the first row is greater than the ending position 15 of single Z-RUN of last row. It confirms that the concave arc modifier is found on the lower part and hence it is of lower zone character. It has been observed that the character '*கு*' of font Agni and Latha are satisfying the conditions of lower zone property even though one font is regular and another is informal.

The character '*ழ*' shown in tenth row of Table 4 is Latha font. Upper zone is not found from the experiment because the upper part has zero

projection. In the separated lower part the ending positions of first Z-RUN of each row of lower part are 12, 8, 8, 8, 8, 5, 5, 6, 15, 14, 13, 10 and ending positions of last Z-RUN of each row are 0, 25, 25, 25, 25, 19, 17, 16, 0, 0, 0, 0. These two sets of position values satisfy the properties of Type 11 and hence the existence of lower zone.

The character ங் in eleventh row of Table 4 is identified as a character without lower part as there is no projection in that part. It has 4 rows in its upper part. Number of transitions on row 1 is 1, row 2 is 2, row 3 is 1, and row 4 is 1. Here row 2 has more than 1 transition; it contradicts the rule that each row should be with 1 transition. Loop is also not found. The horizontal projections of upper portion are 3, 5, 2 and 2 and none of the projection is equivalent with the width 9 of the character. White pixels count in lower part are 0, 6, 0 and 0 which are not in ascending order. The ending position 4 of Z-RUN in the first row is greater than the ending position of the last row. The number of transitions is not in the order of 1, 2, and 3. The above results conclude that the features corresponding to the modifiers loop, dot, convex and arc with centre vertical li ங் re not found in the upper part of the character . The experimental results categorize the character wrongly as middle zone instead of upper zone.

S.No	Input Image (Binarized)	After Trimming	Meanline & Baseline detection	Uppermodifier	Lower modifier
1.				–	Concave arc (Type 1)
2.				–	Lower Arc along with Loop (Type 9)
3.				–	Lower Arc (Type 11)
4.				a Dot Modifier	–
5.				Convex Arc	–
6.				Loop Modifier	–
7.				Arc with center vertical line	Concave Arc Modifier
8.				–	Not a defined Modifier
9.				–	Concave arc (Type 1)
10.				–	Lower Arc (Type 11)
11.				Not a defined Modifier	–

Table 4: Upper & lower modifiers categorization

Seventeen features are identified from middle zone characters such as number of loops, number of objects, number of runs at first row, number of runs at last row, number of vertical lines, and number of horizontal lines, tetra zone, vertical line at last column of trimmed image, and horizontal line at last row of trimmed image and are tabulated in Table 6. The experiment returns 3 loops for the character ண and zero loop for the characters ட, ப. Number of objects represent number of connected components present in the middle zone. From the experiment, 3 objects are identified from the character ஈ and 2 objects from the character உ. The character ச is resulted with single Z-RUN at first row and two Z-RUNs at last row. To identify the number of vertical lines appear in a character, we treat continuous vertical projections with a difference of maximum 2 as single vertical line. Similarly a continuous horizontal projection with a difference of maximum 2 is treated as a single horizontal line. The experiment identifies 1 vertical line and 1 horizontal line for the character �டை. Features extracted from upper, middle, lower and three zone characters are illustrated in Table 5, 6, 7 and 8 with few sample characters.

Table 5: Different features extracted from upper zone characters using proposed method

S.No	Input Image	Dot	Loop	Curve Arc	Arc with centre vertical line	No. of loops	No. of objects	No. of Runs at First row	No. of Runs at Last row	No. of Vertical lines	No. of Horizontal lines	Tetra Bits																				Vertical line at last column of trimmed image	Horizontal line of last row of trimmed image
1	ಕ	1	0	0	0	3	1	1	1	3	2	0	0	0	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
2	ಕಿ	0	0	1	0	2	1	2	2	2	3	0	0	0	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
3	ಬ	0	0	1	0	0	2	3	1	1	3	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
4	ಲ್	0	1	0	0	1	1	2	1	1	4	0	0	0	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
5	ಶ	0	0	1	0	1	2	2	2	2	2	0	1	0	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 6: Different features extracted from middle zone characters using proposed method

S.No	Input Image	No. of loops	No. of objects	No. of Runs at First row	No. of Runs at Last row	No. of Vertical lines	No. of Horizontal lines	Tetra Bits																								Vertical line at last column of trimmed image	Horizontal line of last row of trimmed image
1	ಕ	2	1	1	2	1	0	1	0	1	1	0	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0
2	ಕಿ	0	1	2	3	3	0	0	0	0	1	1	0	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
3	ಜ್ಞ	3	1	3	4	1	0	0	0	1	1	0	1	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
4	ಲ್	0	1	1	1	1	1	0	1	0	0	1	1	0	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0
5	ಶ	1	2	2	2	0	0	1	0	0	0	0	1	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 7: Different features extracted from lower zone characters using proposed method

S.No	Input Image	No. of loops	No. of objects	No. of Runs at first row	No. of Runs at Last row	No. of Vertical lines	No. of Horizontal lines	Tetra Bits																								Vertical line at last column of trimmed image	Horizontal line of last row of trimmed image	Lower zone modifiers Types (1-14)														
1	ரு	1	1	1	1	1	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0
2	ரூ	0	1	1	1	1	2	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0
3	ரி	0	1	3	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0
4	ரீ	0	1	2	1	1	1	0	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0
5	ரெ	1	1	2	3	1	1	1	1	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0

From our experiment, it has been observed that when the characters are belonging to the font Latha, Vijaya and Bamini, the proposed procedure extract the features correctly for all sizes but for the type Agni, Alankaram and Ananthabiravi the proposed procedure is not able to extract the features correctly for most of the characters. By analyzing the results, it has been observed that the correctly identified characters are of regular fonts and unidentified characters are of informal stylish.

The experiment is executed with different learning parameters λ for different zone characters

Table 8: Different features extracted from three zone characters using proposed method

S.No	Input Image	Del	Loop	Curve Arc	Arc with centre vertical line	No. of loops	No. of objects	No. of Runs at first row	No. of Runs at Last row	No. of Vertical lines	No. of Horizontal lines	Tetra Bits																						Vertical line at last column of trimmed image	Horizontal line of last row of trimmed image	Lower zone modifiers Types (1-14)														
1	ரு	1	0	0	0	0	1	1	2	1	2	0	1	0	1	0	1	0	1	1	1	1	1	1	1	1	1	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0
2	ரூ	0	0	1	0	0	2	3	3	1	3	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0
3	ரி	0	0	1	0	2	2	2	3	1	3	1	1	1	1	0	1	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0
4	ரீ	1	1	0	0	0	1	1	1	2	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	1	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0
5	ரெ	0	0	1	0	1	1	3	4	3	3	0	1	0	1	0	1	0	1	1	1	1	1	1	1	1	1	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0

and is identified that λ =0.05, 0.05, 0.05 and 0.07 give fast convergence during the training of upper, middle, lower and three zone characters respectively. The networks structure for classification and recognition differ only in output layer. Network architecture considered for classifying upper, middle, lower and three zone characters are 32-39-1, 17-32-1, 31-53-1 and 35-35-1 respectively. Neural network architecture considered for recognizing upper, middle, lower and three zone characters are 32-39-6, 17-32-5, 31-53-6 and 35-35-6 respectively. The above architectures are fixed by trial and error. Different data sets are used for testing and training. The classification results are listed in Table 9.

Table 9: Results of character classification

Types of characters	Class	Precision	Recall	F1-score	Support	Accuracy
Upper zone	0	1.00	0.98	0.99	184	99.662%
	1	1.00	1.00	1.00	1000	
	avg / total	1.00	1.00	1.00	1184	
Lower zone	0	1.00	1.00	1.00	25	100%
	1	1.00	1.00	1.00	1788	
	avg / total	1.00	1.00	1.00	1813	
Middle zone	0	1.00	1.00	1.00	87	100%
	1	1.00	1.00	1.00	1026	
	avg / total	1.00	1.00	1.00	1113	
Three zone	0	1.00	0.99	1.00	180	99.94%
	1	1.00	1.00	1.00	1496	
	avg / total	1.00	1.00	1.00	1676	

In this table, the class 0 indicates special characters and the class 1 indicates Tamil characters. Receiver operating characteristic **(**ROC) curves and confusion matrices of classification are shown in Fig. 13 and Fig. 14 respectively. The obtained recognition rate of upper, middle, lower and three zone characters are listed in Table 10. Learning curves of the networks for upper, middle lower and three zone characters are shown in Figure 15.

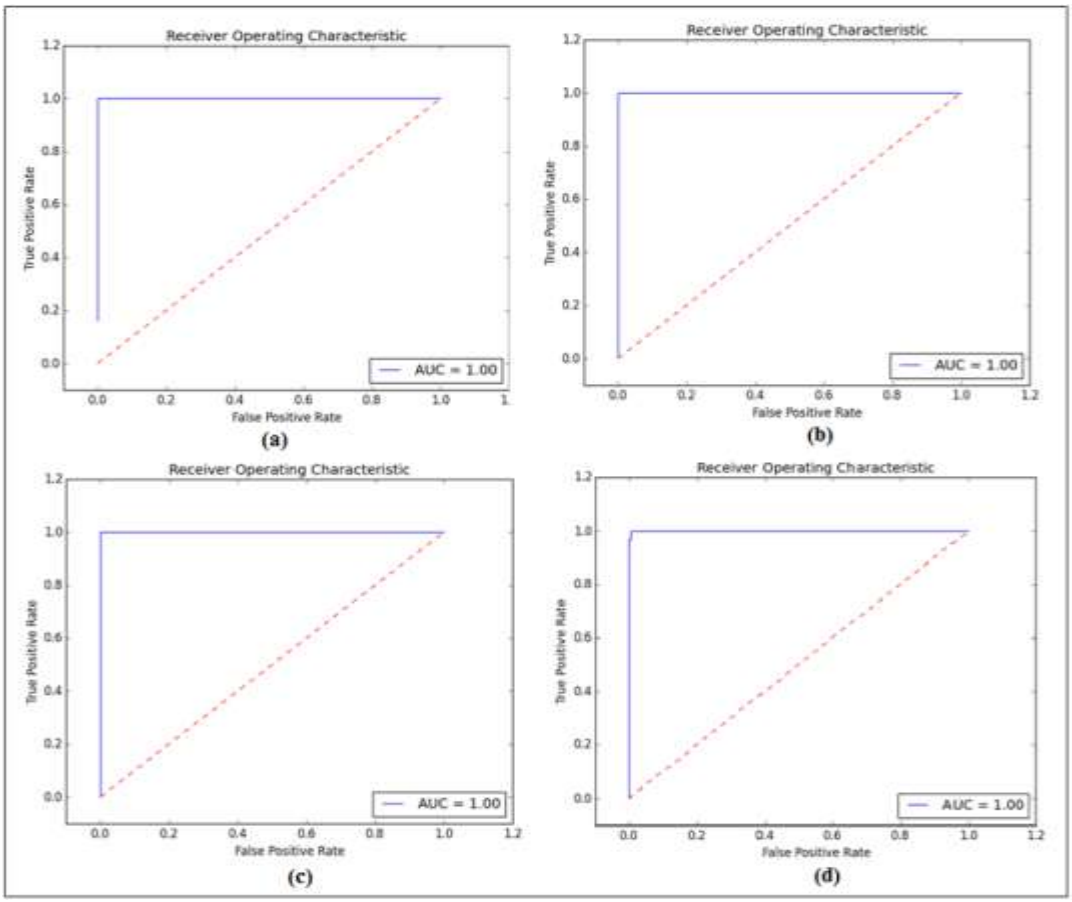


Fig. 13: ROC curves of 1) Upper zone b) Lower zone c) Middle zone d) Three zone characters

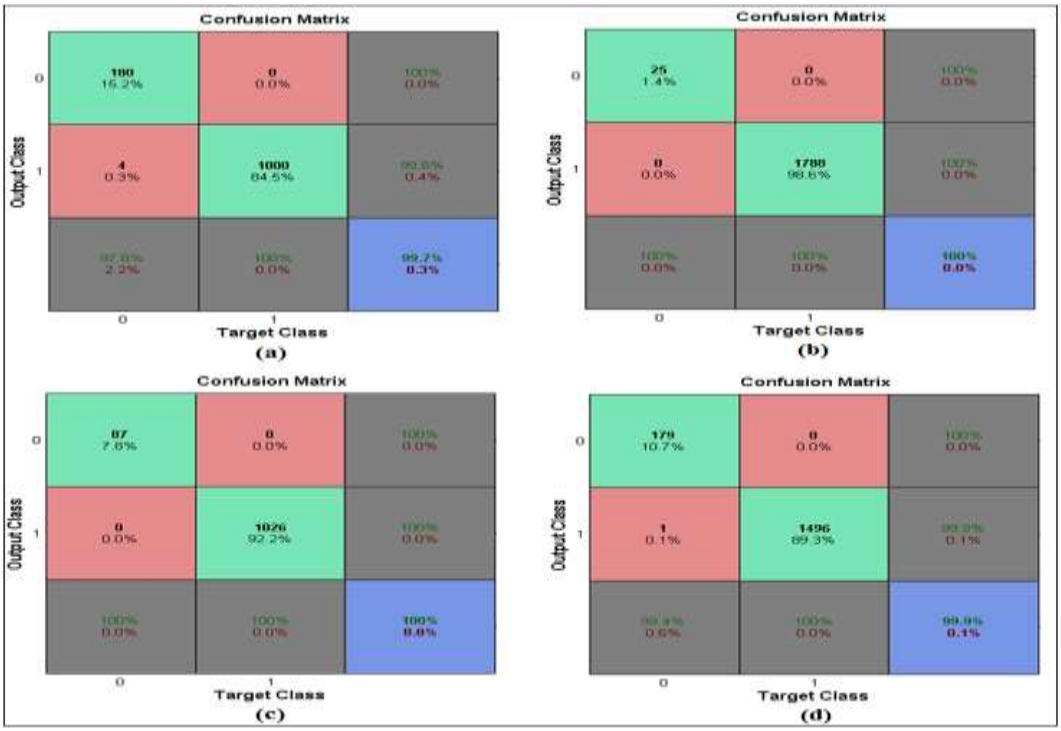


Fig.14: Confusion matrix of a) Upper zone b) Lower zone c) Middle zone d) Three zone for character classification network

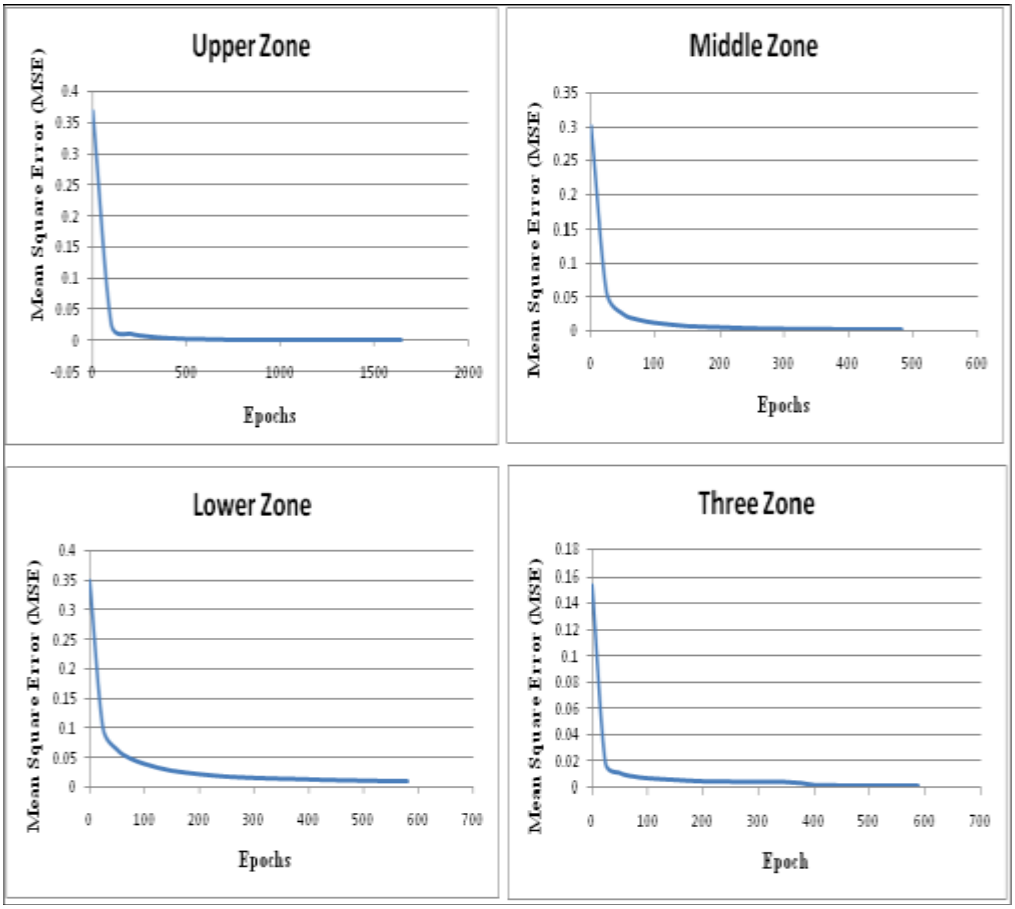


Fig. 15: Learning curves for (a) Upper zone (b) Middle zone (c) Lower zone (d) Threezone characters

Table 10: Results of character recognition

S.No	Zone	Network Structure	Learning parameter	Error	Epoch	Training	Testing	Matched patterns	Average Precision	Average Recall	Average F1-score	Accuracy
1	Upper	21-39-6	0.05	0.001	1645	2706	1004	1000	0.99	1.00	0.99	99.601%
2	Middle	17-32-5	0.05	0.000998	483	2289	1026	1024	1.00	1.00	1.00	99.805%
3	Lower	31-53-6	0.05	0.009989	580	1924	1788	1776	0.99	0.99	0.99	99.328%
4	Three	24-35-6	0.07	0.000998	589	1916	1497	1496	1.00	1.00	1.00	99.933%

Aradhya *et al.* (2008) have considered 50,000 samples with clear and degraded characters and achieved accuracy of 99.01% with F-PCA, and 96.9% with PCA. 30,000 samples are considered as noisy dataset and achieved the accuracy 94% with F-PCA and 92.56% with PCA for south Indian scripts. Seethalakshmi *et al.* (2005) have considered five fonts like Arial unicode MS, Anjal (Nalinam), Amudham, Elango and Shree_tam fonts for recognition and obtained recognition rates 66.66%, 60.00%, 57.14%, 55.55% and 50.00% respectively for those fonts. The recognition system proposed by Aparna *et al.* (2002) for Tamil magazine documents obtained the character recognition accuracy from 94% to 97% and for newsprint documents in (2003) 94% recognition rate is obtained when the text block having few touching characters but the recognition rate varied from 85% to 90% when the touching characters present in the text part is increased. The system proposed by Aparna and Ramakrishnan (2002) recognizes characters in an average of 98% when 4000 samples were used as training data.

The result of the proposed system implies that the system is capable of handling regular fonts of different sizes. At the best, recognition rates 99.601%, 99.805%, 99.328% and 99.933% are achieved for upper, middle, lower and three zone characters respectively and an average of 99.67% recognition rate is obtained by the proposed method. The comparative results are listed in Table 11.

V. CONCLUSION

In this paper new structural feature extraction method is proposed for printed Tamil characters using projection method and classifies the characters using neural network. Based on the modifiers characters are categorized as upper, middle, lower and three zone. In Tamil, few characters are with two or three symbols without being connected. These types of characters are separated into two or three different symbols accordingly and each symbol is treated as individual character for extracting its features separately. Experimental results show that proposed procedure is size independent but can be applied only for regular fonts. This procedure is simple to follow as it involves just counting on and off pixels and transition of foreground to background pixels. The proposed work classifies characters in the Tamil documents into Tamil characters and other characters. Also it recognizes the Tamil characters after ignoring the other characters in the script. Artificial neural network is used for classification and recognition. The proposed method gives good accuracy in classification as well as recognition. The overall performance of this proposed network is 99.67%. The result seems favourable when compared with the results in the literature. The performance of classification and recognition model is mainly based on the structural properties of printed Tamil characters. As a whole, this model offers a satisfactory success rate.

Acknowledgement

We thank the University Grants Commission, Government of India for giving financial support (MRP: F.No. 42-144/2013(SR)) for doing this project.

Table 11: Comparative results of recognition of printed Tamil characters

Method proposed by	Documents	Dataset	Classifiers	Features	Accuracy
Aradhya*et al.* (2008)	Clear and degraded characters	50,000 samples, two fonts with non touching characters	Nearest neighbour classifier	F-PCA	99.01%
				PCA	96.9%
	Noisy dataset	30,000 samples, two fonts with non touching characters	Nearest neighbour classifier	F-PCA	94%
				PCA	92.56%
Seethalakshmi*et al.* (2005)	Printed Tamil text documents	5 fonts with non touching characters	Backpropagation feedforward network	Structural features	100% for Single font
					50% for 2 or more fonts
Aparna K.H *et al.* (2002)	Tamil newsprint	Single font	RBF	Contour chain code	94% for few touched characters
					85-90% for more touched characters
Aparna K.H *et al.* (2003)	Tamil magazine	Single font	RBF	Structural features	94% for few touched characters
					90-97% for more touched characters
Aparna and Ramakrishnan (2002)	Magazines, novels, etc.	4,000 samples with non touching characters	Nearest neighbour classifier	Geometric moments (discrete cosine transform)	98%
Proposed	Degraded& noisy printed Tamil literary periodicals	1071865 samples of regular fonts with more touched characters	Backpropagation feedforward network	Structural, runlength and projection	99.67%

REFERENCES

[1] Min, H. K., Hou, Y., Park, S., Song, I., 2016, "A computationally efficient scheme for feature extraction with kernel discriminant analysis", Pattern Recognition, Vol. 50, pp. 45-55.

[2] Guyon, I., Elisseeff, A., 2006, "Introduction to Feature Extraction", StudFuzz 207, Springer-Verlag Berlin Heidelberg, pp. 1-2.

[3] Raj, M. A. R., and Abirami, S.,2012, "A Survey on Tamil Handwritten Character Recognition using OCR Techniques", In: Computer Science Conference Proceedings (CSCP'12), CS&IT 05, pp. 115-127.

[4] Kahan, S., Pavlidis, T., and Baird, H.S., 1987, "On the Recognition of Printed Characters of Any Font and Size", IEEE Trans. on Pattern Anal. Machine Intell., 9(2), pp. 274 – 288.

[5] Heutte, L., Paquet, T., Moreau, J.V., Lecourtier, Y., and Olivier, C., 1998, "A structural statistical feature based vector for handwritten character recognition", Pattern Recognition Letters, 19(7) , pp. 629-641.

[6] Bag, S., Harit, G., and Bhowmick, P., 2014, "Recognition of Bangla compound characters

using structural decomposition", Pattern Recognition, 47 (3), pp. 1187-1201.

[7] Singh, G., Kumar, C.J., Rani, R., Dhir, D.R., "Feature extraction of gurmukhi script and numerals: a review of offline techniques", International Journal of Advanced Research in Computer Science and Software Engineering, 3(1), pp. 257-263.

[8] Seethalakshmi, R., Sreeranjani, T. R., and Balachandar, T., 2005, "Optical Character Recognition for printed Tamil text using Unicode", Journal of Zhejiang University Science, 6A (11), pp. 1297-1305.

[9] Aparna, K. H., and Chakravarthy, V. S., 2003, "A Complete OCR System Development of Tamil Magazine Documents", Tamil Internet.

[10] Aparna, K. H., JaganathanSumanth, Krishnan, P., and Chakravarthy, V. S., 2002, "An Optical Character Recognition System for Tamil Newsprint", International Conference on Universal Knowledge and Language, pp. 881-886.

[11] Aradhya, V.M., N., Kumar, G.H., and Noushath, S.,2008,"Multilingual OCR system for South Indian scripts and English documents: An approach based on Fourier transform and principal component analysis", Engineering Applications of Artificial Intelligence, 21 (4), pp. 656-668.

[12] Aparna, K. G., and Ramakrishan, A. G., 2002, "A Complete Tamil Optical Character Recognition System", Springer LNCS 2423, pp. 53-57.

[13] Arora, S., Bhattacharjee, D., Nasipuri, M., Basu, D. K., and Kundu, M.,2008"Combining Multiple Feature Extraction Techniques for Handwritten Devnagari Character Recognition", IEEE Region 10 Colloquium and the Third ICIIS, Kharagpur, India, pp. 8-10.

[14] Pal, U. and Anirban Sarkar, 2003, "Recognition of Printed Urdu Script", Proceedings of the Seventh International Conference on Document Analysis and Recognition.

[15] Chaudhuri, B.B., Pal, U. and Mitra, M., 2002,"Automatic recognition of printed Oriya script", Sadhana, 27(1), pp. 23–34.

[16] Rocha, J., and Pavlidis, T., 1994, "A shape analysis model with applications to a character recognition system", IEEE Transaction on PAMI, 16 (4), pp. 393-404.

[17] Trier, O. D., Jain, Anil K., and Taxt, T., 1996, "Feature Extraction Methods for Character Recognition- A Survey", Pattern Recognition, 29 (4), pp. 641-662.

[18] Kuo, S. S. and Agazzi, O. E. (1994) 'Keyword Spotting in Poorly Printed Documents Using Pseudo 2-D Hidden Markov Models', IEEE Transaction on Pattern Analysis and Machine Intelligence, 16(8), pp. 842-848.

[19] Dedgaonkar, S. G., Chandavale, A. A., Sapkal, A. M., 2012, "Survey of Methods for Character Recognition", International Journal of Engineering and Innovative Technology (IJEIT), 1(5), pp. 180-189.

[20] Bataineh, B., Abdullah, S.N.H.S., and Omar, K., 2012,"A novel statistical feature extraction method for textual images: Optical font recognition", Expert Systems with Applications, 39(5), pp. 5470–5477.

[21] Abubacker, N.F., and Raman, I., 2011,"An Approach for Structural Feature Extraction for Distorted Tamil Character Recognition", International Journal of Computer Applications, 22 (4), pp. 24-28.

[22] Kunte, R. S., and Sudhaker Samuel, R. D., 2007, "A simple and efficient optical character recognition system for basic symbols in printed Kannada text", Sadhana, Vol. 32, pp. 521–533.

[23] Saba, T, Rehman, A., and Sulong, G., 2011,"Improved Statistical Features For Cursive Character Recognition", International Journal of Innovative Computing, Information and Control, 7 (9), pp. 5211-5224.

[24] Sundaram, S., and Ramakrishnan, A. G., 2009, "An Improved Online Tamil Character Recognition Engine using Post-Processing Methods", 10th International Conference on Document Analysis and Recognition, pp. 1216-1220.

[25] Karthikeyan, V.,2013,"Hilditch's Algorithm Based Tamil Character Recognition", International Journal of Computer Science & Engineering Technology (IJCSET), 4(3), pp. 268-273

[26] Chaudhuri, B. B., and Pal, U., 1998, "A Complete Printed Bangla OCR System", Pattern Recognition, 31 (5), pp. 531-549.

[27] Das, N., Sarkar, R., Basu, S., Saha, P.K., Kundu, M., and Nasipuri, M., 2015 "Handwritten Bangla character recognition using a soft computing paradigm embedded in two pass approach", Pattern Recognition, 48 (6), pp. 2054-2071.

[28] Niblack, W., 1986,"An Introduction to Digital Image Processing", Prentice-Hall, Englewood Cliffs, pp.115-116.

[29] Parvez, M.T., and Mahmoud, S. A.,2013, "Arabic handwriting recognition using structural and syntactic pattern attributes", Pattern Recognition, 46 (1), pp. 141–154.

[30] Abirami, S., and Murugappan, S., 2011, 'Scripts and Numerals Identification from Printed Multilingual Document Images", In: Computer Science Conference Proceedings AIAA CS&IT 03, pp. 129–146.
[31] Kathirvalavakumar, T., and Karthigai Selvi, M., 2013,"Efficient Touching Text Line Segmentation in Tamil Script Using Horizontal Projection", International conference on Mining Intelligence and Knowledge Exploration, LNCS 8284, pp. 279-288.
[32] http://www.tamilagaasiriyar.com/p/tamil-e-books.html
[33] http://books.tamilcube.com/tamil/
[34] http://knowingyourself1.blogspot.in/2011/04/free-tamil-books-tamil-pdf-books.html
[35] http://www.projectmadurai.org/pmworks.html
[36] http://www.dinamalar.com
[37] http://kalvimalar.dinamalar.com/tamil/