J.Natn.Sci.Foundation Sri Lanka 2021 49 (2): 195 - 208
DOI: http://dx.doi.org/10.4038/jnsfsr.v49i2.9466

RESEARCH ARTICLE

An effective feature set for enhancing printed Tamil character recognition

MS Shafana[1,2*], RG Ragel[3] and TN Kumara[4]

[1] *Department of Information and Communication Technology, Faculty of Technology, South Eastern University of Sri Lanka, University Park, Oluvil.*
[2] *Post Graduate Institute of Science, University of Peradeniya, Peradeniya.*
[3] *Department of Computer Engineering, Faculty of Engineering, University of Peradeniya, Peradeniya.*
[4] *The MARCS Institute for Brain, Behaviour and Development, Western Sydney University, Australia.*

Submitted: 23 September 2019; Revised: 25 June 2020; Accepted: 22 January 2021

Abstract: Selection of features for extraction and classification are the essential factors in achieving high performance in character recognition. Feature extraction process produces feature vectors that define the shape and characteristics of the pattern to identify them uniquely. Many feature extraction and classification approaches are available for Tamil and other languages, but there is still room to identify a better set of features for extraction to obtain higher recognition rate of Optical Character Recognition (OCR) for Tamil printed text. This research aims at producing an efficient set of features for extraction, which is capable of increasing the accuracy and reducing the runtime to improve the performance of the best OCR system to classify isolated Tamil printed characters. The proposed set of features is experimented on a large dataset using One-versus-All (OVA) Support Vector Machine (SVM). Two types of the pool of different feature vectors are created with features used in this study such as basic, density, histogram oriented gradients (HOG), and transition. In comparison with the current best approach, the testing results of Pool 1 gives better recognition accuracy of 94.87 % for OVA SVM and 97.07 % for the Unbalanced Decision Tree (UDT) SVM algorithms, but could not reach an improved recognition speed. Likewise, the results of Pool 2 improves the performance of the system by giving not only better recognition accuracy of 94.30 % for OVA SVM and 96.35% for the UDT SVM algorithms but also reached an improved recognition speed than the selected best OCR approach. The proposed set of features improves the recognition rate by 2.57–3.14% on OVA SVM and 3.22–3.94% on UDT SVM.

Keywords: Basic features, feature extraction, OCR, OVA SVM, Tamil character recognition, UDT SVM.

INTRODUCTION

Optical Character Recognition (OCR)

Information sharing is necessary to increase the knowledge level of people. This sharing can be carried out using any kind of medium such as books, documents, or manuscripts. Millions of books, documents and manuscripts have been produced in this world from the ancient era. Most of the old documents are not available today, and some are scarce. We are unable to make use of them because they are not readily available.

Most of the documents are hard copies produced as printed documents or handwritten ones. Converting these documents into digital format leads to their preservation, making editing possible and producing many copies. Digital copies of materials can be stored in digital libraries and shared readily over the internet. They are also easy to reproduce. Electronic documents might be prepared in two ways. One way is by scanning the printed documents using the scanner or a digital camera. The other way of digitisation is creating the document by

* Corresponding author (zainashareef@gmail.com; https://orcid.org/0000-0003-3951-0326)

retyping from the start. Most of the organisations, which are maintaining digital libraries such as the World Digital Library (World Digital Library Home, n.d.), Hathi Trust Digital Library (HathiTrust Digital Library, n.d.) and Noolaham (நூலகம், n.d.), converts their collection into digital form by scanning. This type of digitisation produces un-editable images of documents which require a significant amount of storage space. Retyping a whole document needs more time and human resource to complete it. If a material could be prepared as editable, searchable, and portable with less storage space and preparing time, then it could be made entirely accessible to many people at the same time efficiently.

OCR is used to accomplish this purpose. It uses pattern recognition techniques to identify the patterns in images and converts them into an editable text format. These converted documents can be utilised easily by copying, searching or dividing into sub-documents.

Character recognition can be divided into two broad categories: optical (printed) and handwritten character recognition. Moreover, handwritten character recognition is divided into two types based on the input devices. One is online character recognition that uses digitizer tablets or PDA screens to get the input and identifies the characters in real time. The other is offline character recognition that uses scanners and cameras to get the image inputs and determines the characters by applying some image processing algorithms. In case of optical character recognition, the input may be a good quality document or a degraded one. This research relates to printed character recognition. A sophisticated, well integrated optical character recognition system should carry the main parts as pre-processing and segmentation, character recognition, and post-processing. Pre-processing and segmentation step contains binarization, noise removal, skew detection and correction, text or non-text classification and segmentation procedures. Character recognition step comprises feature extraction and classification procedures. Finally, post-processing is to identify and make the recognised text free from linguistic misspellings due to OCR.

The implementation methods of OCR differ from language to language. Up to now, several approaches have been proposed for many languages including Tamil. But those are with some shortcomings and lack accuracy. In this study, those available approaches of Tamil OCR have been intensely reviewed and found out the state-of-the-art among them. The overall aim is to suggest a best set of features to improve the accuracy with an increased recognition speed of the state-of-the-art of OCR for Tamil languages. The main focus was on the feature selection. Based on the analysis results of the literature review, it was found that the approach proposed in Ramanan (2015) has the best of ideas and highest accuracy among the available strategies. Therefore, the method explained in Ramanan (2015) was chosen as the state-of-the-art for printed Tamil character recognition. Through this work, it is intended to suggest a better approach for feature selection which is capable of increasing the accuracy and reducing the runtime of the system.

Tamil language

Tamil is a Dravidian language which is a descendant of Proto-Dravidian. Tamil language has a history of more than 2000 years starting from 300 BC. It is categorised into three periods such as Old Tamil (300 BC-AD 700), Middle Tamil (700 AD-1600 AD) and Modern Tamil (1600 AD-present). It is an official language of two countries, Sri Lanka and Singapore. It also impacts the educational career in many countries. Tamil has the official stage in the Indian State of Tamil Nadu (Tamil Language, n.d.).

Characters in Tamil Script

There are 247 characters in Tamil Script. 12 vowels [short vowels (kuril); 5, long vowels (nedil); 7], 18 consonants, 216 consonant vowels and one special character (aaydham) (A Brief Introduction to Tamil Language & Culture, n.d.).

1. Vowels
   - Short Vowels (kuril) : அ, இ, உ, எ, ஒ
   - Long Vowels (nedil) : ஆ, ஈ, ஊ, ஏ, ஐ, ஓ, ஔ
2. Consonants
   - vallinam (வல்லினம்) : க், ச், ட், த், ப், ற
   - mellinam (மல்லினம்) : ங், ஞ், ண், ந், ம், ன
   - idayinam (இடயினம்) : ய், ர், ல், வ், ழ், ள
3. aaydham
   ஃ
4. Consonant vowels (Composite characters) :
   Examples: கு, கூ, சு, சூ, ஞு, ஞூ, டி, டீ, டு, டூ, ணு, ணூ, து, தூ

Table 1: Year-wise analysis of existing works

Reference	Year Published	Extracted Features	Coverage	Accuracy
Siromoney *et al*., 1978	1978	Binary matrix representation	125 Symbols	not specified
Ramakrishnan & Mahata, 2000	2000	Height, upward/downward extension, geometric moment feature with nearest neighbour classifier	154 Symbols	99.1 %
Dhamayanthi *et al*., 2000	2000	Matrix representation with BPNN	30 Chars	90 %
Dhanya & Ramakrishnan, 2001	2001	DCT based feature with SVM classifier	Not specified	96–97.5 %
Aparna & Ramakrishnan, 2002	2002	Height, upward/downward extension, DCT based feature with truncated DCT coefficient with Nearest neighbour classifier	154 Symbols	98 %
Hewavitharana & Fernando, 2002	2002	Height, zone-based pixel density with statistical classifier	26 Chars	79.7 %
Aparna & Chakravarthy, 2003	2003	Radial basis function neural network	157 Symbols	94 %
Seethalakshmi *et al*., 2005	2005	Ten basic features with SVM classifier	247 Chars	66 %
Shivsubramani *et al*., 2007	2007	Moment invariant feature extraction, using multiclass hierarchical SVM.	126 Symbols	97 %
Sutha & Ramaraj, 2007	2007	Closed boundary trace, Fourier descriptor, and back-propagation network for a classifier.	Not specified	97 %
Shanthi & Duraiswamy, 2007	2007	Zone based pixel density	106 Chars	87.4 %
Kannan *et al*., 2008	2008	horizontal information and vertical information using HMM	Not specified	93–96.45 %
Suresh, 2008	2008	feature vector consists of distances of the pattern from the frame in sixteen different direction	67 Char	76–94 %
Kannan & Prabhakar, 2008	2008	HMM and SVM with radial basis function neural network	Not specified	92.3 % -94.1 %
Apte & Gado, 2010	2010	Horizontal line, vertical line, branching and their position with Euclidean distance	Not specified	72 %
Subramanian & Kubendran, n.d.	2013	centroid of the character (X, Y), four non-linear combinations of central moments	31 chars	not specified
Sundar & John, 2013	2013	HOG features, topological features, and the histogram of orthogonal distances. BPNN, FLDA.	Not specified	48.7–99.45%
Raja & John, 2013	2013	Loops, curves, Horizontal line, vertical line and their positions with BPNN, SVM, DT classifier	30 chars	95 %
Karthieswari *et al*., 2014	2014	Random Fourier features using random kitchen sinks.	Not specified	98.70 %
Devi & Baboo, 2014	2014	Ten basic features and used Raspberry Pi device	Not specified	Not specified
Ramanan *et al*., 2015	2015	Features of basic, density, histogram of oriented gradients (HOG) and transition with DAG, UDT SVMs organised in a hybrid decision tree.	124 symbols	98.80 %
Pugazhenthi & Vallarasi, 2015	2015	Template matching	Not specified	96.29 %

The objective of this paper is to propose the best set of features to enhance the accuracy of Printed Tamil character recognition with an increased recognition speed. Moreover, this study finds out the best approach among the existing OCR approaches by having a thorough analysis on their performance for the purpose of proving that the feature sets proposed through this work have better results than the best available at present.

Pre-processing and segmentation

OCR system goes through three main phases; pre-processing and segmentation, character recognition and post-processing. In order to achieve higher recognition rates, it is mandatory to decrease the variation of the input image that causes a reduction in recognition rate and increases the complexities. Therefore, OCR systems use pre-processing and segmentation phases to overcome this problem. Moreover, the input image can be a scanned image, image captured by camera or merely a print-screen of a page. The input may contain not just the text but pictures, equations, tables, etc. Pre-processing and segmentation are required steps to extract the text and convert the input image of the text to segmented components. These processes transform the raw data into a format that will be more readily and efficiently processed.

Binarization process is used to convert the grey scale or colour images to binary images. Thresholding is the technique used in binarisation. It separates the foreground of an image from its background. Median filtering is performed to reduce the noise as in Ramanan (2015). For the skew detection and correction process, the algorithm proposed in Shafii (2014) was applied. For the testing purpose of this approach, two types of datasets were used. One type (Tobacco-800 Complex Document Image Dataset and Groundtruth, n.d.) is used in Shafii (2014), and the other one is *UJTDdocF* (Datasets of printed Tamil characters and printed documents - the University of Jaffna, n.d.). For the page layout analysis process, the approach proposed by Shafii (2014) was used. He has proposed an effective algorithm for segmentation of Persian Scripts and the same algorithm was applied to the segmentation process.

The State-of-the-art of OCR for Tamil Languages

The state-of-the-art was found through an analysis on existing works. This study used a list of factors in finding the state of the art such accuracy, coverage of characters, extracted features, used classifiers, and amount of dataset used for training and testing phases. A summary of this review is listed in Table 1.

There are a total of 247 characters in Tamil script. These full set of characters can be formed using a total of 124 unique symbols. Among the existing approaches, some authors prepared their system to recognise the characters, and others have prepared the system to recognise the unique symbols. In some approaches, more than 124 symbols are covered as their scope (Ramakrishnan & Mahata, 2000; Aparna & Ramakrishnan, 2002; Aparna & Chakravarthy, 2003). Those symbol sets include isolated Tamil characters, English numerals, and punctuation marks. Among the existing approaches, four studies achieved an accuracy above 98 %. Ramakrishnan and Mahata (2000) reached 99.1 % of efficiency, and they covered the full set of character symbols. They just used around 4000 samples only to test and train their system. Aparna and Ramakrishnan (2002) also reached 98 % of accuracy with the scope of recognising the full set of character symbols. They also used just 4000 samples to test and train their system. Karthieswari *et al*. (2014) came up with an approach to reach an accuracy of 98.70 %, but they did not specify the number of characters they covered and the amount of data used to test and train. The approach proposed by Ramanan *et al*. (2016) reached a better accuracy of 98.80 % with the scope of a 124 symbol set (full set of Tamil character symbols). They have used a vast dataset with 12,400 data samples and this is the highest amount used in a Tamil character recognition system. However, they did not specify the recognition speed of their system.

Through the analysis it was clear that none of the work carried out yet meets the following requirement.

- To identify Tamil characters, which include all the set of symbols (equal to or above 124) and tested using a reasonable amount of datasets (to examine all the symbols), with speed and reasonable recognition accuracy.

However, among these existing works the best approach is proposed by Ramanan *et al*. (2016), which is a part of Ramanan (2015). The failure rate of this approach is 1.2 %. Typically, an A4 sized printed paper can contain an average of 2000 characters per page. This failure rate might cause 24 errors per page. Such an error is not acceptable for a working recognition system. Therefore, an efficient set of features is proposed to increase this recognition accuracy.

METHODOLOGY

Character recognition in a scanned image is carried out with the use of pattern recognition techniques. Furthermore, character recognition phase can be divided into two main steps namely Feature Extraction and Classification. Feature extraction is a process of generating the feature vectors, which can be used to

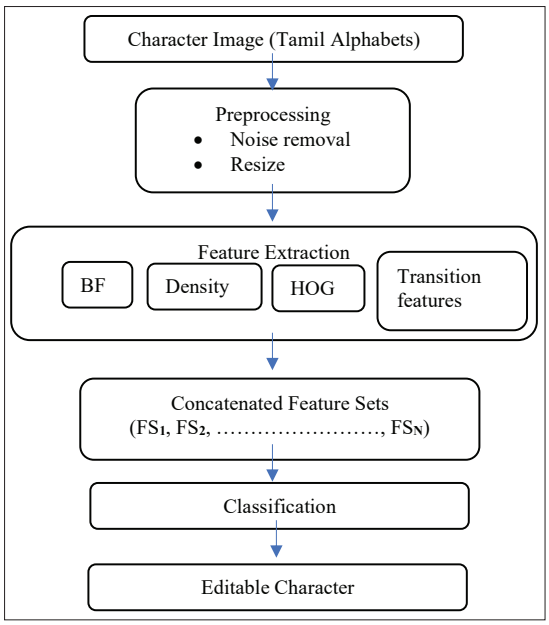

Figure 1: Framework of character recognition

represent the characters uniquely. Feature selection to extract, the vector plays a vital role in character recognition, and the feature extraction process has a primary role in improving recognition accuracy as well. Figure 1 depicts the steps of character recognition. It consists of pre-processing, feature extraction and classification steps.

Table 2: New features suitable to correctly classify the misclassified characters of (Ramanan, 2015)

Input	Error	Suitable Features
அ	1(ஆ)	As_Loop, Num_End, Dz_L
ஆ	1(அ)	Dz_L, As_Loop, Num_End
ஓ	2(ஒ)	Num_Loop, Num_End, Num_Int, Dz_L
த்	1(ந்)	Num_Loop, Num_End, Dz_L
ந்	1(த்)	Num_Loop, Num_End, Dz_L
ற்	1(ற)	Num_Obj
ூ	ு	Num_Loop, Num_End, Num_Int, Dz_L
நீ	1(நி)	Num_Loop, Num_End, Num_Int, Dz_L
றீ	1(றி)	Num_Loop, Num_End, Num_Int, Dz_L
நி	1(நீ)	Num_Loop, Num_End, Num_Int, Dz_L
றி	1(றீ)	Num_Loop, Num_End, Num_Int, Dz_L
ரி	1(ரீ)	Num_Loop, Num_End, Num_Int, Dz_L
ண	1(ன)	Num_Loop, As_Loop, Num_Int, Dz_L
ணூ	1(ண)	Num_Loop, Num_End, As_Loop, Num_Int
நூ	1(ந)	As_Loop, Num_End, Num_Int
வூ	1(ல)	As_Loop, Num_End, Num_Int
ஏூ	1(எ)	Num_End, Num_Int, Num_Loop, Dz_L
றூ	1(ற)	As_Loop, Num_End, Num_Int
னூ	1(ன) 1(ண)	As_Loop, Num_End, Num_Int, Num_Loop
ழூ	1(ழ)	Num_End, Num_Int, Num_Loop, Dz_L
ரூ	1(ர)	Num_End, Num_Int, Num_Loop, Dz_L
ஞூ	1(ஞ)	Num_End, Num_Int, Num_Loop, Dz_L
நு	1(ற)	Num_End, Num_Int
ர	1(ா)	Num_Cross
ா	2(ர)	Num_Cross
ச	1(சு)	Num_Cross, Num_Int
சு	1(ச)	Num_Cross, Num_Int
எ	1(ஏ)	Num_Cross
ை	2(ன)	Num_Int, Num_End
ல	1(வ)	Num_Int
வ	1(ல)	Num_Int
ன	1(ண)	Num_Int, Num_Loop, Dz_L
ண	1(ன)	Num_Int, Num_Loop, Dz_L
ல்	1(ல)	Num_Obj
வ்	1(வ)	Num_Obj
ன்	1(ன) 1(ண்)	Num_Obj, Num_Loop, Num_Int, Dz_L
லி	1(வி)	Num_Int
வி	2(லி)	Num_Int
னி	1(ளி)	Num_Loop, Num_End, Dz_L
ளி	1(னி)	Num_Loop, Num_End, Dz_L
சீ	1(சி)	Num_Int, Num_Loop, Num_End, Dz_L
லீ	1(லி)	Num_Int, Num_Loop, Num_End, Dz_L
வீ	1(வி)	Num_Int, Num_Loop, Num_End, Dz_L
னீ	2(ளீ)	Num_Loop, Num_End, Dz_L

Pre-processing

The images are pre-processed by following two kinds of processes for removing the noise and resizing the images. Morphological operation or noise filtering can be used to remove the noises in a binary image. After removing noise, each character image is enclosed in a tight fit rectangular boundary using a bounding box, and it is followed by discarding the outside portion of the image using horizontal and vertical projection. Finally, the image is scaled to the size of 64 × 64 pixels.

Feature extraction

Most of the Tamil characters have a similar shape or slight variations. There are possibilities to misclassify the characters due to this similarity. Therefore, the selected features, which are going to be extracted should be able to represent the unique identification of different characters. The pre-processed images are the inputs for this process.

Table 2 contains the summary of misclassified printed Tamil characters using the proposed hybrid decision tree in Ramanan (2015) applied on UJTDchar. The first column and the second column in each block represent the actual, and the number of misclassifications along with the predicted characters, respectively.

The contribution of this study is to propose a different set of new features which are capable of correctly classifying those misclassified characters by Ramanan (2015) mentioned in Table 2 and improve the performance of the current best approach. Therefore, firstly, those actual characters' font shapes were carefully compared with the predicted characters' font shape (which were misclassified by Ramanan, 2015) to find out the differences among them. Seven new features were identified in addition to the features of Ramanan (2015). The new features are explained in the forthcoming section. The following section describes the full set of extracted features of this work which are the combination of newly added features through this study and the features proposed in Ramanan (2015).

Extracted Features

1. Density Features (DF): The following equation computes the density of a zone of the image.

$$\text{Density} = \frac{\text{Number of foreground pixels in a zone}}{\text{Total number of pixels in a zone}} \quad ...(01)$$

In this set, the pre-processed binarized input character images are divided into zones of different sizes as follows and then features are computed from each of those zones. Density features have extracted a total of 88 dimensions. Images are divided into two zones horizontally from top to bottom [Figure 2(a)]. Where the top zone is TZ and the bottom zone is BZ

a. Images are divided into two zones vertically from left to right [Figure 2(b)]. Where the left zone is LZ and the right zone is RZ.
b. Images are divided into nine zones [Figure 2(c)].
c. Images are divided into 15 zones diagonally from bottom left corner to top right corner by varying the size of the zones as 4 × 4, 8 × 8, …., 60 × 60 [Figure 2(d)].
d. Images are divided into 15 zones off- diagonally from top right corner to bottom left corner by varying the size of the zones as 4 × 4, 8 × 8, …., 60 × 60 [Figure 2(e)].
e. Images are divided into 15 zones diagonally from top left corner to bottom right corner by varying the size of the zones as 4 × 4, 8 × 8, …., 60 × 60 [Figure 2(f)].
f. Images are divided into 15 zones off- diagonally from bottom right corner to top left corner by varying the size of the zones as 4 × 4, 8 × 8, …., 60 × 60 [Figure 2(g)].
g. Images are divided into 15 zones diagonally starting from the inner centre towards the corner of the image. Size of the zones varies as 4 × 4, 8 × 8,., 60 × 60 [Figure 2(h)].

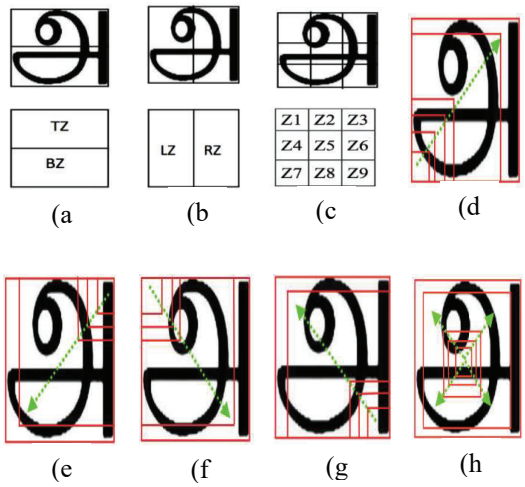

Figure 2: Extracted density features

Basic features (BF)

Two sets of basic features are extracted namely 'Basic Feature Type 1 (BF1)' and 'Basic Feature Type 2 (BF2)'.

i. BF1: In this type, a total of 12 basic features were extracted.

- *Number of horizontal lines present*
- *Number of vertical lines present*
- *Number of right diagonal lines present*
- *Number of left diagonal lines present*
- *Euler number*

As same as Ramanan (2015)

- Density of Loops (Dz_L)
- Aspect Ratio of Loops (As_Loop)
- Number of Endpoints (Num_End)
- Number of Loops (Num_Loop)
- Number of Objects (Num_Obj)
- Number of Intersecting Points (Num_Int)
- Number of Crossing Points at row 49 (Num_Cross)

Newly selected features

ii. BF2: In this type, a total of 10 basic features were extracted.

- *Number of horizontal lines present*
- *Number of vertical lines present*
- *Euler number*

As same as Ramanan (2015)

- Density of Loops (Dz_L)
- Aspect Ratio of Loops (As_Loop)
- Number of Endpoints (Num_End)
- Number of Loops (Num_Loop)
- Number of Objects (Num_Obj)
- Number of Intersecting Points (Num_Int)
- Number of Crossing Points at row 49 (Num_Cross)

Newly selected features

a. Density of Loops (Dz_L)
All the available loops in the character image are filled with white pixels, and the rest of the parts are removed [Figure 3(b)]. Finally, calculates the density of the resulted image.

b. Aspect Ratio of Loops (As_Loop)
All the available loops in the character image are filled with white pixels, and the rest of the parts are removed. The ratio of row (height) to column (width) of the resulted image with loops are calculated as the aspect ratio.

c. Number of Endpoints (Num_End)
This feature represents the total count of available endpoints in character [Figure 3(a)].

d. Number of Loops (Num_Loop)
This feature represents the total count of available loops in character [Figure 3(c)].

e. Number of Objects (Num_Obj)
This feature represents the total count of available objects in character [Figure 3(d)].

f. Number of Intersecting points (Num_Int)
This feature represents the total count of available intersecting points of a minimum of three pixels from at least two distinct directions [Figure 3(c)].

g. Number of Crossing Points at row 49 (Num_Cross)
This feature counts the number of foreground pixels at row 49 [Figure 3(e)]

h. Number of horizontal lines present
It represents the total count of available horizontal lines in the character [Figure 3(a)].

i. Number of vertical lines present
It represents the total count of available vertical lines in the character [Figure 3(a)].

j. Number of right diagonal lines and left diagonal lines present

It represents the total count of available right diagonal lines in the character [Figure 3(a)]. Same as the right diagonal lines, the total count of available left diagonal lines is also extracted for the type of BF1 feature set. In the case of these two features, if the diagonal lines are sharp without aliasing effects [Figure 3(a)] then it gives the accurate count. Otherwise, if it is with aliasing edges [Figure 3(f)], then it provides an inaccurate result. Therefore, these two types of features are omitted for the BF2 feature set to classify the characters, which have the aliasing edges.

k. Euler Number

This feature represents the value of the total number of objects in the image minus the total number of holes in those objects.

Table 2 Explains that new features can be applied to correctly classify those misclassified printed Tamil characters by Ramanan (2015)

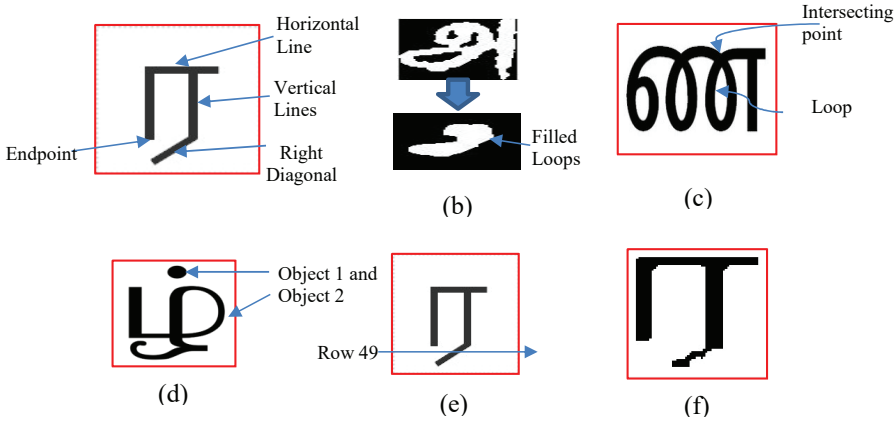

Figure 3: Images for basic features extraction

For example, consider the last two characters of னீ and ளீ; both these characters are having different number of loops, different number of end points, and different values for the density of loops. The first character (னீ) is having three loops, two end points and a different value for the total density of loops than the second character (ளீ). The second character is having only two loops and three end points. Because of these differences, these two characters can be discriminated clearly. Likewise, all misclassified characters can be discriminated clearly from one another.

2. *Histogram of Oriented Gradients (HOG)*

Four kinds of HOG features are extracted in the same way as in Ramanan (2015). They are described below.

i. snHOG

- Pre-processed character images are divided into nine sub-images.
- In each sub-image, nine features are extracted using 9-bins with non-overlapping [Figure 4(a)].
- From these nine sub-images, final feature vector with 81 dimensions is created.

ii. fHOG

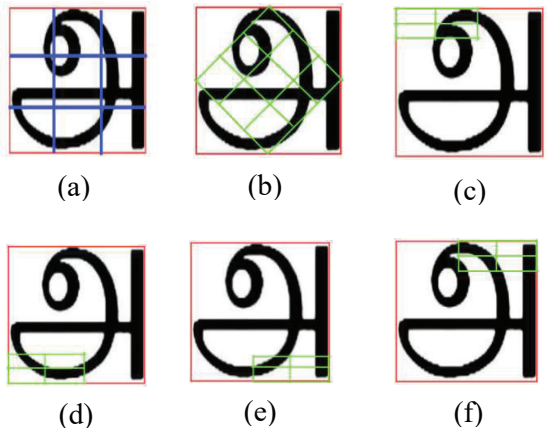

Figure 4: Image zones for HOG feature extraction

- The 64 × 64 pixel detection window was divided into two blocks horizontally and vertically, to create a total of four blocks and each block contains four cells (16 × 16 pixels)

- For each cell, a sum of 36 values per block are extracted using 9-bin histogram.
- Overlap size is defined by the block size divided by 2.
- The resulting feature vector with 324 dimensions is created.

iii. soHOG

- Pre-processed character images are divided into nine sub-images.
- Each sub-image is divided into two blocks horizontally and vertically, to create a total of four blocks and each block contains four cells (8 × 8 pixels)
- For each cell, a total of 36 values per block are extracted using 9-bin histogram
- Overlap size is defined by the block size divided by 2.
- The resulting feature vector with 1764 dimensions is created

iv. eHOG

- In the character images, five different areas are selected to create a feature vector named as eHOG with a total of 225 dimensions. The five areas are explained below
  a. Diamond HOG (*dHOG*): Images are divided into nine sub-images [Figure 4(b)], and a total of 81 values are extracted by applying the 9-bin histogram for each sub-image.
  b. Top-left HOG (*tlHOG*): Images are divided into four sub-images [Figure 4(c)] with the size of 16 × 34 pixels, and a total of 36 values are extracted using 9-bin histogram for each sub-image.
  c. Bottom-left HOG (*blHOG*): Images are divided into four sub-images [Figure 4(d)] with the size of 16 × 34 pixels, and a total of 36 values are extracted using 9-bin histogram for each sub-image.
  d. Bottom-right HOG (*brHOG*): Images are divided into four sub-images [Figure 4(e)] with the size of 16 × 34 pixels, and a total of 36 values are extracted using 9-bin histogram for each sub-image.
  e. Top-right HOG (*trHOG*): Images are divided into four sub-images [Figure 4(f)] with the size of 16 × 34 pixels, and a total of 36 values are extracted using 9-bin histogram for each sub-image.

eHOG with 225 dimensions is created by combining these five (*dHOG + tlHOG + blHOG + brHOG + trHOG) HOG* features.

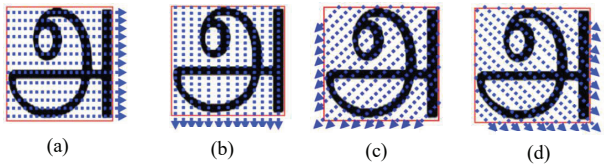

Figure 5: Images for transition feature extraction

3. *Transition Features (TF)*

In pre-processed character images, the number of transitions (Saba *et al*., 2011) is computed from the background (1) to foreground (0) in four directions in the same way as in Ramanan (2015). A resulting feature vector (TF) with a total of 374 values is created.

i. Row-wise Transition: The number of transitions is counted along each of the rows [Figure 5(a)]. A total of 64 values is extracted.
ii. Column-wise Transition: The number of transitions is counted along each of the columns [Figure 5(b)]. A total of 64 values is extracted.
iii. Diagonal Transition: The number of transitions is counted along the diagonal lines [Figure 5(c)]. A otal of 123 values are extracted.
iv. Off-diagonal Transition: The number of transitions is counted along the off-diagonal lines [Figure 5(d)]. Total of 123 values is extracted.

Feature sets are extracted from the dataset using Algorithm 1.

Algorithm 1: Feature Extraction

Input:	Set of images (Tamil characters)
Output:	Different feature vectors (concatenation of basic, density, HOG, and transition features)
Step1:	Binarize the input image
Step2:	Pre-process the binarized image (remove noise and resize it into 64 × 64 pixels)
Step3:	Extract basic, density, HOG, and Transition features
Step4:	Repeat steps 1 to 3 until reaching the last image of the image set
Step5:	End of program

After the extraction of all of these features, ten types of feature sets are formed as FS1, FS2, FS3, …, FS10 by having a different concatenation of features. The details of these feature sets are specified in Table 3.

Table 3: Formed feature sets

No	Feature vector	Concatenation of features
1	FS1	fHOG + snHOGZ
2	FS2	fHOG + soHOG + DF
3	FS3	fHOG + snHOGZ + soHOG + eHOG + DF
4	FS4	fHOG + BF2
5	FS5	fHOG + DF + BF2
6	FS6	fHOG + DF + TF + BF2
7	FS7	fHOG + DF + TF
8	FS8	fHOG + BF + DF + soHOG
9	FS9	fHOG + BF2 + DF + soHOG
10	FS10	fHOG + DF + BF

Two (2) pools of feature sets were created using these feature vectors, namely, feature set Pool 1 and feature set Pool 2. Feature set Pool 1 contains all these ten (FS1, FS2, FS3, ….., FS10) feature vectors. Feature set pool 2 includes only six (6) feature vectors (FS1, FS4, FS5, FS6, FS7, FS10)

Classification

The extracted feature vectors are analysed using the OVA (One-Versus-All) SVM, and the result is compared with the accuracy of (Ramanan, 2015). In this study also, the same classes of Tamil Characters (124 unique classes) used in (Ramanan, 2015) shown in Table 4 have been used.

All of the experiments were carried out on a PC running with an Intel Core i5 2.4GHz processor and 2GB RAM under Matlab R2017a platform. The LIBSVM tool (Chih-Chung & Chih-Jen, n.d.) was used.

Table 4: The simplified 124 unique classes of Tamil character

class	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25
char	அ	ஆ	இ	ஒ	ஓ	ஐ	த்	ந்	ழ்	ற்	ஞ்	டீ	ஸீ	தீ	நீ	ழீ	றீ	ரீ	டு	ஸி	தி	நி	ழி	றி	ரி
class	26	27	28	29	30	31	32	33	34	35	36	37	38	39	40	41	42	43	44	45	46	47	48	49	50
char	ஞ	ஙூ	சூ	ணு	ணூ	தூ	நூ	லு	லூ	ளு	ளூ	றூ	னு	னூ	ஹூ	மு	மூ	ரு	ரூ	ழு	ழூ	பூ	யூ	வூ	கு
class	51	52	53	54	55	56	57	58	59	60	61	62	63	64	65	66	67	68	69	70	71	72	73	74	75
char	ஏ	த	ந	ழ	ற	டி	து	நு	பு	யு	வு	று	ஙு	ஙு	ர	ஈ	ெ	ா	க	ச	ப	ம	சு	உ	ஊ
class	76	77	78	79	80	81	82	83	84	85	86	87	88	89	90	91	92	93	94	95	96	97	98	99	100
char	எ	ை	ங	ய	ல	வ	ள	ன	ண	ட	கூ	டீ	ஃ	∴	ர்	க்	ங்	ச்	ட்	ண்	ப்	ம்	ய்	ல்	வ்
class	101	102	103	104	105	106	107	108	109	110	111	112	113	114	115	116	117	118	119	120	121	122	123	124	
char	ள்	ன்	கி	ஙி	சி	ணி	பி	மி	யி	லி	வி	னி	ளி	கீ	ஙீ	சீ	ணீ	பீ	மீ	யீ	லீ	வீ	னீ	ளீ	

RESULTS AND DISCUSSION

Experimental setup

Dataset

Ramanan (2015) had prepared a vast dataset as part of his research. Two kinds of datasets in his preparation have been made publicly available (Datasets of printed Tamil characters and printed documents - University of Jaffna, n.d.). The same dataset was used in this study. They are mentioned below.

- *UDTchar* consist of 12400 printed Tamil individual character samples cropped from different printed Tamil scanned documents. In which, 124 unique symbols with 100 images per symbol are available.
- *UJTDdocF* prepared with 20 different font faces of printed Tamil scanned documents which consist a total of 30 pages, and each page has an average of 275 words. Each page complexed by different font styles, and font sizes having bold and italics formatting.

We used 30 % of UJTDchar for testing and 70 % for training the system, similar to the current best approach (Ramanan, 2015).

Evaluation criteria

The recognition rate is computed as the percentage of correctly recognised characters against the number of characters in the document.

$$\text{Recognition rate} = \frac{\text{Total number of correctly recognised characters}}{\text{Total number of characters in the document}} \times 100 \quad ...(2)$$

Testing Results

In the classification part, 10-fold cross validation (SVM classifier-MATLAB crossval) for training and testing the data of each feature sets was used. Cross validation is a multi-time experiment, and the average accuracy of those folds for each feature set was calculated. The highest average accuracy for OVA algorithm was 94.87 % for the feature set FS8 in Pool 1, and 94.30 % for the Feature set FS10 in Pool 2. When considering the UDT algorithm, it would gain 97.07 % of highest average accuracy for Pool 1 feature sets and 96.35 % of highest average accuracy for Pool 2 feature sets. Table 6 shows the calculated average accuracy of each feature set.

The result gained for the feature sets of Pool 1 of this study is compared with the outcome of Ramanan (2015). The proposed approach through this study gives 94.87 % recognition rate for the algorithm of OVA SVM, but the method in Ramanan (2015) gave only 91.73 % of accuracy for this same dataset.

The resulting accuracy proves that the proposed feature selection achieves an increased accuracy over the accuracy of Ramanan (2015). In addition to this, an analysis was carried out on the outputs to observe the result that would be obtained if the feature sets of Pool 1 were processed with the algorithm of UDT SVM proposed in Ramanan (2015). The result of Ramanan (2015) was 93.13 %, but the proposed feature sets through this study would result in 97.07 % of accuracy.

The misclassified character classes of Ramanan (2015) is compared with the results of this study. It shows that more than twenty classes give better results than Ramanan (2015) and it is included in Table 5

Feature sets of Pool 1 include a total of 2868 (basic feature – 12, density feature – 88, hog – 2394, transition feature – 374) values of features. But Ramanan (2015) used only 1619 (basic feature – 05, density feature – 88, hog – 1154, transition feature – 372) values of features.

Table 5: Results comparison table of the Analysis to the algorithm UDT SVM of this study and HDT of (Ramanan, 2015)

No	Class No	Misclassified Count By HDT in [56]	Misclassified Count By UDT of this study
1	1	1	0
2	2	1	0
3	5	2	1
4	7	1	5
5	8	1	0
6	10	1	0
7	12	1	1
8	15	1	1
9	17	1	0
10	22	1	1
11	24	1	1
12	25	1	1
13	29	1	1
14	30	1	0
15	32	1	0
16	34	1	0
17	36	1	0
18	37	1	0
19	39	2	1
20	42	1	1
21	44	1	1
22	46	1	1
23	58	1	2
24	65	1	1
25	68	2	1
26	70	1	0
27	73	1	1
28	76	1	1
29	77	2	1
30	80	1	0
31	81	1	0
32	83	1	2
33	84	1	0
34	99	1	0
35	100	1	0
36	102	2	2
37	110	1	0
38	111	2	0
39	112	1	0
40	113	2	4
41	116	1	1
42	121	1	0
43	122	1	2
44	123	2	2

Table 6: Calculated average accuracy of each feature set using OVA

No	Feature vector	Average accuracy
1	FS1	91.73 %
2	FS2	92.02 %
3	FS3	92.74 %
4	FS4	93.15 %
5	FS5	93.90 %
6	FS6	94.22 %
7	FS7	93.63 %
8	FS8	94.87 %
9	FS9	94.22 %
10	FS10	94.30 %

The total count of used features is larger than that of the selected approach. Therefore, the speed of recognition of the present system might be slower than (Ramanan, 2015). As there is a research question to propose an efficient set of features to improve the performance, it is needed to consider the improvement on accuracy and speed as well. Through the feature sets of Pool 1, only an improvement on accuracy was achieved, and not on the speed of recognition. Among the extracted features, HOG has the most significant amount of vector dimensions. The total count of used feature vectors can be reduced by removing the feature vectors from the pools which are created by the concatenation of HOG. Therefore, another pool (Pool 2) of feature vectors which contains FS1, FS4, FS5, FS6, FS7, FS10 only was prepared. An accuracy of 94.30 % was reached for OVA classification algorithm and a better increase in accuracy through Pool 2 was also achieved. Moreover, the total count of used features in this study (1164) is smaller than the selected approach, which is 1619. Therefore, the recognition speed also could potentially be higher than the selected approach. UDT SVM classification algorithm gave us an accuracy of 96.35% for Pool 2.

CONCLUSION

This paper has presented a better set of features, which can be used to recognise the misclassified printed Tamil characters in Ramanan (2015) with the highest accuracy. The proposed work includes concatenated feature sets of basic, density, transition, and HOG features of printed Tamil character images.

In the pre-processing stage, the process of skew angle detection and correction is performed using an algorithm composed of an axes-parallel bounding box and works regardless of the content of the documents. Therefore, this algorithm works in the presence of graphical images, tables, charts, etc. with no angle limitation. As a result of testing with the dataset *UJTDdocF* 100 % accuracy was achieved. In the segmentation stage, the proposed algorithm produces error free segmentation for the testing datasets of *UJTDdocF* documents which do not contain the overlapping and touching Tamil characters.

Furthermore, in the state-of-the-art finding process, a thorough review and analysis of the proposed approaches of feature selection for printed Tamil character recognition was conducted. All the experiments of this study was processed with large datasets publicly available (Tobacco-800 Complex Document Image Dataset and Groundtruth, n.d.; Datasets of printed Tamil characters and printed documents - the University of Jaffna, n.d.). Moreover, in the classification stage, the proposed feature vectors of Pool 1 are tested using the OVA technique. The recognition rate was 94.87 %, which shows an increase of 3.14 % than the results of Ramanan (2015). Thus, the current approach per page can reduce a total of 63 character misclassifications. The analysis results for UDT-based SVM gives an accuracy of 97.07 %, which shows an increase of 3.94 % than the results of (Ramanan, 2015). Therefore, a total of 80 character misclassifications can be solved by this proposed work using the feature sets of Pool 1. Even though the accuracy is high, the drawback of Pool 1 is that the recognition speed might be lower than the selected approach because of the increased count on total used features.

Moreover, the proposed feature sets of pool 2 were classified with OVA algorithm. The recognition rate was 94.30 % which shows an increase of 2.57 % than the results of Ramanan, (2015). The analysis results for UDT-based SVM gives an accuracy of 96.35 %, which shows an increase of 3.22 % than the selected approach. Due to these increases in accuracy, 52 character misclassifications by OVA, and 65 character misclassifications by UDT SVM can be avoided than Ramanan (2015). Also, the feature vectors of Pool 2 give an improvement on recognition speed. The total count of used feature vector dimensions of Pool 2 is lower than the selected approach. Therefore, the recognition speed is potentially higher than in Ramanan (2015).

Finally, the results prove that the proposed feature sets through this research give better results on accuracy and speed compared to the proposed feature sets of Ramanan (2015). The performance of printed Tamil character

recognition was improved through the proposed set of features. According to the study, better character recognition accuracy can be achieved if the proposed feature sets are processed with the hybrid decision tree proposed in Ramanan (2015). In this approach the accuracy can be higher than the reported accuracy of 98.80 % in Ramanan (2015). Future work will focus to expand to recognise the handwritten Tamil characters.

REFERENCES

A Brief Introduction to Tamil Language and Culture - தமிழ் மொழி. (n.d.). Available at *https://tamilnation.org/literature/geetha.htm*, Accessed 12 November 2017.

Aparna K. & Chakravarthy V. (2003). A complete OCR system development of Tamil magazine documents. *Tamil Internet Conference*, pp. 45–51. Chennai, India. Available at *http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.4.8406&rep=rep1&type=pdf*

Aparna K. & Ramakrishnan A. (2002). A complete Tamil optical character recognition system. In: *Document Analysis Systems* (eds. D. Lopresti, J. Hu & R. Kashi) V. DAS 2002. Lecture Notes in Computer Science, volume 2423, pp. 53–57. Springer, Berlin, Heidelberg, Germany. DOI: https://doi.org/10.1007/3-540-45869-7_6

Apte A. & Gado H. (2010). Tamil character recognition using structural features. Available at *http://cecas.clemson.edu/~stb/ece847/projects/Tamil_Character_Recognition.pdf*

Chih-Chung C. & Chih-Jen L. (2011). LIBSVM: a library for support vector machines. *ACM Transactions on Intelligent Systems and Technology* **2**(3): 27:1–27. DOI: https://doi.org/10.1145/1961189.1961199

Data sets of printed Tamil characters and printed documents - University of Jaffna. (n.d.). Available at *http://www.jfn.ac.lk/index.php/data-sets-printed-tamil-characters-printed-documents/*, Accessed 10 February 2017.

Devi V.A. & Baboo S.S. (2014). Embedded optical character recognition on Tamil text image using Raspberry Pi. *International Journal of Computer Science Trends and Technology* **2**(4): 11–15.

Dhamayanthi N., Thangavel P., Kalyanasundram K. & Nathan S.S. (2000). Handwritten Tamil character recognition using neural network. *Tamil Internet Conference*, pp. 171-176, July 22-24, Available at *http://www.infitt.org/ti2000/tamilinaiyam/conference_hub.html*

Dhanya D. & Ramakrishnan A.G. (2001). Simultaneous Recognition of Tamil and Roman Scripts. *Tamil Internet Conference*, pp. 64–68.

HathiTrust Digital Library | Millions of books online. (n.d.). Available at *https://www.hathitrust.org/*, Accessed 22 March 2018.

Hewavitharana S. & Fernando H. (2002). A two stage classification approach to Tamil handwriting recognition. *Tamil Internet Conference*, pp. 118–124, California, USA. Available at *http://infitt.org/ti2002/papers/25SANJIK.PDF*

Kannan R.J. & Prabhakar R. (2008). Accuracy Augmentation of Tamil OCR using algorithm fusion. *International Journal of Computer Science and Network Security* **8**(5): 3–8.

Kannan R. J., Prabhakar R. & Suresh R.M. (2008). Off-line cursive handwritten Tamil character recognition. *Proceedings-2008 International Conference on Security Technology, SecTech 2008* **4**(6): 159–164. DOI: https://doi.org/10.1109/SecTech.2008.33

Karthieswari R., Sreethivya M., Kumar D.D. R. & Soman K.P. (2014). Tamil characters classification using random kitchen sink algorithm (RKS). *Proceedings of the 2014 International Conference on Interdisciplinary Advances in Applied Computing*, pp. 1–5. DOI: https://doi.org/10.1145/2660859.2660916

Pugazhenthi D. & Arul Vallarasi S. (2015). Offline character recognition of printed tamil text using template matching method of Bamini Tamil font. *Indian Journal of Science and Technology* **8**(35): 1–4. DOI: https://doi.org/10.17485/ijst/2015/v8i35/86811

Raja S. & John M. (2013). A novel Tamil character recognition using decision tree classifier. *IETE Journal of Research* **59**(5): 569. DOI: https://doi.org/10.4103/0377-2063.123763

Ramakrishnan A.G. & Mahata K. (2000). A complete OCR for printed Tamil text. DOI: https://doi.org/10.13140/RG.2.1.4593.5209

Ramanan M. (2015). an approach for recognising printed characters using a hybrid decision tree and a post-processing error correction technique to Tamil OCR *M.Phil. Thesis*. Jaffna, Sri Lanka.

Ramanan M., Ramanan A. & Charles E.Y.A. (2015). A hybrid decision tree for printed Tamil character recognition using SVMs. *15th International Conference on Advances in ICT for Emerging Regions, ICTer 2015 - Conference Proceedings*, August 2015, 176–181. DOI: https://doi.org/10.1109/ICTER.2015.7377685

Saba T., Rehman A. & Sulong G. (2011). Improved statistical features for cursive character recognition. *International Journal of Innovative Computing* **7**(9): 5211–5224.

Available at *https://pdfs.semanticscholar.org/0f93/3b8c3d275249bd1c9eb2b467939c7dfcb391.pdf*

Seethalakshmi R., Sreeranjani T. R., Balachandar T., Abnikant S., Markandey S., Ritwaj R. & Sarvesh K. (2005). Optical character recognition for printed Tamil text using Unicode. *Journal of Zhejiang University Science* **6A**(11): 1297–1305. DOI: https://doi.org/10.1631/jzus.2005.A1297

Shafii M. (2014). Optical character recognition of printed Persian/Arabic documents. *Electronic Theses and Dissertations*. Available at *https://scholar.uwindsor.ca/etd/5179*

Shanthi N. & Duraiswamy K. (2007). Performance comparison of different image sizes for recognizing unconstrained handwritten Tamil characters using SVM. *Journal of Computer Science* **3**(9): 760–764. DOI: https://doi.org/10.3844/jcssp.2007.760.764

Shivsubramani K., Loganathan R., Srinivasan C.J., Ajay V. & Soman K.P. (2007). Multiclass hierarchical SVM for

recognition of printed Tamil characters. In: *Proceedings of the IJCAI 2007 Workshop on Analytics for Noisy Unstructured Text Data (AND 07),* Hyderabad, India.

Siromoney G., Chandrasekaran R. & Chandrasekaran M. (1978). Computer recognition of printed Tamil characters. *Pattern Recognition* **10**(4): 243–247. DOI: https://doi.org/10.1016/0031-3203(78)90032-8

Subramanian A. & Kubendran B. (n.d.). OCR for printed Tamil characters. Available at *http://www.angelfire.com/in/anbu/vt/tamilocr*/. Accessed 23 May 2017.

Sundar K.A. & John M. (2013). A high precision printed character recognition method for Tamil script. *2013 IEEE International Conference on Computational Intelligence and Computing Research*, 26–28 December, Enathi, India. DOI: https://doi.org/10.1109/ICCIC.2013.6724285

Suresh R.M. (2008). Printed and handwritten tamil characters recognition using fuzzy technique, *Proceedings of The International MultiConference of Engineers and Computer Scientists*, 19–21 March, Hong Kong, pp. 702–706.

Sutha J. & Ramaraj N. (2007). Neural network based offline Tamil handwritten character recognition system. *International Conference on Computational Intelligence and Multimedia Applications (ICCIMA 2007)*, Sivakasi, India, 13–15 December, pp. 446–450. DOI: https://doi.org/10.1109/ICCIMA.2007.86

SVM classifier - MATLAB crossval - MathWorks. Available at *https://ch.mathworks.com/help/stats/classificationsvm.crossval.html*, Accessed 8 August 8 2017.

Tamil Language (n.d.). Available at *https://en.wikipedia.org/wiki/Tamil_language*, Accessed 11 November 2017.

Tobacco-800 Complex Document Image Dataset and Groundtruth (n.d.). Available at *http://legacydirs.umiacs.umd.edu/~zhugy/tobacco800.html*, Accessed 8 August 2017.

World Digital Library Home (n.d.). Available at *https://www.wdl.org/en/*, Accessed 22 March, 2018.

நூலகம் (n.d.). Available at *http://www.noolaham.org/wiki/index.php/*முதற்_பக்கம் Accessed 22 March 2018.