Tamil Handwritten Character Recognition Using Kohonon's Self Organizing Map

Dr.J.Venkatesh[†] **and C. Sureshkumar**[††]

Anna University Coimbatore, J.K.K.M College of Technology, India.

Abstract
The extracted features for recognition are converted to Self Organizing Maps (SOM) where the characters are classified using Kohonen Network algorithm. This character recognition finds applications in document analysis where the handwritten document can be converted to editable printed document. This approach can be extended to recognition and reproduction of hand written documents in South Indian languages. Hand written Tamil Character recognition refers to the process of conversion of handwritten Tamil character into printed Tamil character. The scanned image is segmented into paragraphs using spatial space detection technique, paragraphs into lines using vertical histogram, lines into words using horizontal histogram, and words into character image glyphs using horizontal histogram. Each image glyph is subjected to feature extraction procedure, which extracts the features such as character height, character width, number of horizontal lines(long and short), number of vertical lines(long and short), horizontally oriented curves, the vertically oriented curves, number of circles, number of slope lines, image centroid and special dots.

Keywords:
Character recognition, Unicode, Self Organizing Maps (SOM)

1. Introduction

Tamil is an ancient language with a rich literary tradition and Ancient India was popular in several fields such as medicine, astronomy and business. Ancient people recorded their knowledge in various fields in palm leaves. The handwritten text written in palm leaves decayed over a period of time. It is very difficult to preserve them in the same form. This paper proposes a new approach for converting handwritten Tamil script using unicode. The style of writing and the font were different compared to present day scripts. Lot of software tools is available only to read present day printed Tamil text with better recognition and accuracy.

1.1 Tamil Language

Tamil is a South Indian language spoken widely in TamilNadu in India. Tamil has the longest unbroken literary tradition amongst the Dravidian languages. Tamil is inherited from Brahmi script. The earliest available text is the Tolkaappiyam, a work describing the language of the classical period. There are several other famous works in Tamil like Kambar Ramayana and Silapathigaram but few supports in Tamil which speaks about the greatness of the language. For example, Thirukural is translated into other languages due to its richness in content. It is a collection of two sentence poems efficiently conveying things in a hidden language called Slaydai in Tamil. Tamil has 12 vowels and 18 consonants. These are combined with each other to yield 216 composite characters and 1 special character (aayutha ezhuthu) counting to a total of (12+18+216+1) 247 characters.

1.2 Vowels

Tamil vowels are called uyireluttu (uyir – life, eluttu – letter). The vowels are classified into short (kuril) and long (five of each type) and two diphthongs, /ai/ and /auk/, and three "shortened" (kuril) vowels. The long (nedil) vowels are about twice as long as the short vowels. The diphthongs are usually pronounced about 1.5 times as long as the short vowels, though most grammatical texts place them with the long vowels.

1.3 Consonants

Tamil consonants are known as meyyeluttu (mey - body, eluttu - letters). The consonants are classified into three categories with six in each category: vallinam - hard, mellinam - soft or Nasal, and itayinam - medium. Unlike most Indian languages, Tamil does not distinguish aspirated and unaspirated consonants. In addition, the voicing of plosives is governed by strict rules in centamil. Plosives are unvoiced if they occur word-initially or doubled. Elsewhere they are voiced, with a few becoming fricatives intervocalically. Nasals and approximants are always voiced. As commonplace in languages of India Tamil is characterized by its use of more than one type of coronal consonants. Retroflex consonants include the retroflex approximant , which among the Dravidian languages is also found in

Manuscript received December 5, 2009
Manuscript revised December 20, 2009

Malayalam (example Kozhikode), disappeared from Kannada in pronunciation at around 1000 AD (the dedicated letter is still found in Unicode), and was never present in Telugu. Dental and alveolar consonants also contrast with each other, a typically Dravidian trait not found in the neighboring Indo-Aryan languages.

1.4 Tamil Unicode

The Unicode Standard is the Universal Character encoding scheme for written characters and text. It defines the uniform way of encoding multilingual text that enables the exchange of text data internationally and creates the foundation of global software. The Tamil Unicode range is U+0B80 to U+0BFF [3]. The Unicode characters are comprised of 2 bytes in nature. The Unicode is designed for various other Tamil characters.

2. Character Recognition

The schematic block diagram of handwritten Tamil Character Recognition system consists of various stages as shown in figure. They are Scanning phase, Preprocessing, Segmentation, Feature Extraction, Classification, Unicode mapping and recognition and output verification.

2.1 Character Recognition Functions I

This phase includes the scanning, preprocessing, segmentation and feature extraction.

2.2 Scanning

A properly printed document is chosen for scanning. It is placed over the scanner. A scanner software is invoked which scans the document. The document is sent to a program that saves it in preferably TIF, JPG or GIF format, so that the image of the document can be obtained when needed. This is the first step in OCR. The size of the input image is as specified by the user and can be of any length but is inherently restricted by the scope of the vision and by the scanner software length.

2.3 Preprocessing

This is the first step in the processing of scanned image. The scanned image is pre processed for noise removal. The resultant image is checked for skewing. There are possibilities of image getting skewed with either left or right orientation. Here the image is first brightened and binarized. The function for skew detection checks for an angle of orientation between ±15 degrees and if detected then a simple image rotation is carried out till the lines match with the true horizontal axis, which produces a skew corrected image.

2.4 Skew Detection and Correction

Knowing the skew of a document is necessary for many document analysis tasks. Calculating projection profiles, for example, requires knowledge of the skew angle of the image to a high precision in order to obtain an accurate result. In practical situations, the exact skew angle of a document is rarely known, as scanning errors, different page layouts, or even deliberate skewing of text can result in misalignment. In order to correct this, it is necessary to accurately determine the skew angle of a document image or of a specific region of the image, and, for this purpose, a number of techniques have been presented in the literature. Postal found that the maximum valued position in the Fourier spectrum of a document image corresponds to the angle of skew.

2.5 Segmentation

After pre-processing, the noise free image is passed to the segmentation phase, where the image is decomposed into individual characters and the algorithm for segmentation is given below:

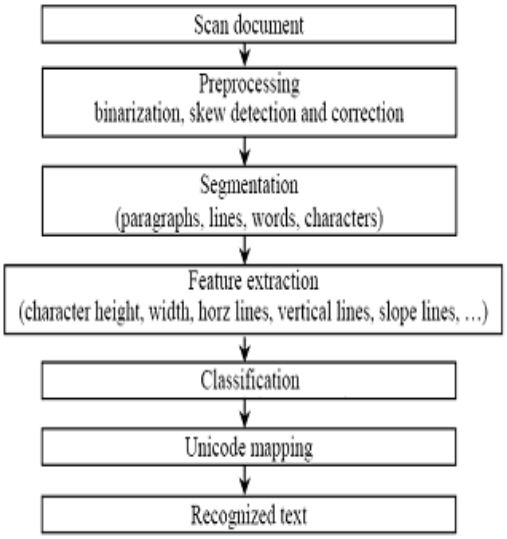

Fig.1 Handwritten Character Recognition System

Fig.2 Histograms for skewed and skew corrected images

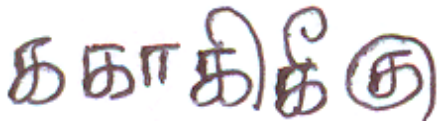

Fig. 3 Original Text

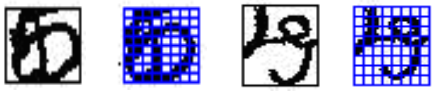

Fig. 4 Character Segmentation

Algorithm:

(1) The binarized image is checked for inter line spaces.

(2) If inter line spaces are detected then the image is segmented into sets of paragraphs across the interline gap.

(3) The lines in the paragraphs are scanned for horizontal space intersection with respect to the background. Histogram of the image is used to

Algorithm for Horizontal Line Detection:

C [i] [j] =img [i−1] [j−1]× (−1) + img[i−1][j] × (−1)
+img [i−1] [j+1] × (−1) + img[i] [j−1] × 2
+img [i] [j] ×2+img[i] [j+1] ×2
+img [i+1] [j−1] × (−1) + img [i+1] [j] × (−1)
+img [i+1] [j+1]× (−1)

If (c[i] [j]>0)
Increment q[i];
If (q[i] ≥smallThreshold)
Small horizontal line is detected
Else
If (q[i] ≥longThreshold)
Long Horizontal line is detected;

detect the width of the horizontal lines [6]. Then the lines are scanned vertically for vertical space intersection. Histograms are used to detect the width of the words. Then the words are decomposed into characters using character width computation.

2.6 Feature extraction

The next phase to segmentation is feature extraction where individual image glyph is considered and extracted for features. Each character glyph is defined by the following attributes:

(1) Height of the character. (2) Width of the character. (3) Numbers of horizontal lines present—short and long. (4) Numbers of vertical lines present—short and long. (5) Numbers of circles present. (6) Numbers of horizontally oriented arcs. (7) Numbers of vertically oriented arcs. (8) Centroid of the image. (9) Position of the various features. (10) Pixels in the various regions.

3. Character Recognition Functions II

The second phase of the Character Recognition functions consists of classification and Unicode mapping and recognition strategies.

3.1 Classification

The various classification methods are as follows:

3.1.1 A typical rule based Classifier

The height of the character and the width of the character, various distance metrics are chosen as the candidates for classification when conflict occurs. Similarly, the classification rules are written for other characters [5]. This method is a generic one since it extracts the shape of the characters and need not be trained. When a new glyph is given to this classifier block it extracts the features and compares the features as per the rules and then recognizes the character and labels it.

3.1.2 Self-Organizing Maps:

A self-organizing map (SOM) is a type of artificial neural network that is trained using unsupervised learning to produce a low dimensional (typically two dimensional), discretized representation of the input space of the training samples, called a map. Self-organizing maps are different than other artificial neural networks in the sense that they use a neighborhood function to preserve the topological properties of the input space. This makes SOM useful for visualizing low-dimensional views of high-dimensional data, akin to multidimensional scaling [12].

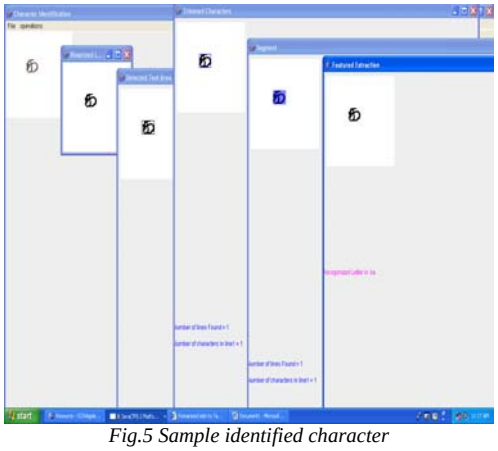

Fig.5 Sample identified character

The model was first described as an artificial neural network by the Finnish professor Teuvo Kohonen, and is sometimes called a Kohonen map. Like most artificial

neural networks, SOMs operate in two modes: training and mapping. Training builds the map using input examples. It is a competitive process, also called vector quantization. Mapping automatically classifies a new input vector. The self-organizing map consists of components called nodes or neurons. Associated with each node is a weight vector of the same dimension as the input data vectors and a position in the map space. The usual arrangement of nodes is a regular spacing in a hexagonal or rectangular grid. The self-organizing map describes a mapping from a higher dimensional input space to a lower dimensional map space [11].

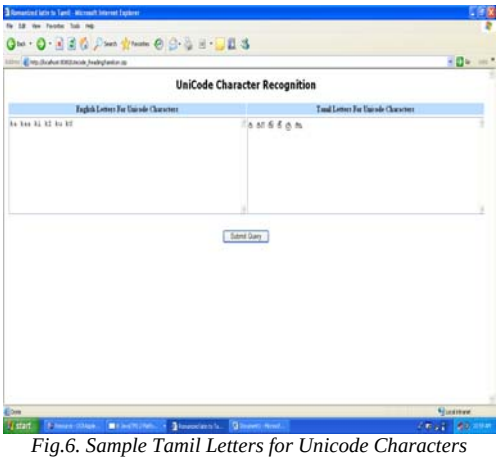

Fig.6. Sample Tamil Letters for Unicode Characters

The procedure for placing a vector from data space onto the map is to find the node with the closest weight vector to the vector taken from data space and to assign the map coordinates of this node to our vector. While it is typical to consider this type of network structure related to feed forward networks where the nodes are visualized as being attached, this type of architecture is fundamentally different in arrangement and motivation. Useful extensions include using toroidal grids where opposite edges are connected and using large numbers of nodes. It has been shown that while self-organizing maps with a small number of nodes behave in a way that is similar to K-means, larger self-organizing maps rearrange data in a way that is fundamentally topological in character. It is also common to use the U-matrix. The U-matrix value of a particular node is the average distance between the node and its closest neighbors. In a rectangular grid for instance, we might consider the closest 4 or 8 n large SOMs display properties which are emergent. Therefore, large maps are preferable to smaller ones. In maps consisting of thousands of nodes, it is possible to perform cluster operations on the map itself.

3.1.3 Learning algorithm

The goal of learning in the self-organizing map is to cause different parts of the network to respond similarly to certain input patterns. The weights of the neurons are initialized either to small random values or sampled evenly from the subspace spanned by the two largest principal component eigenvectors. With the latter alternative, learning is much faster because the initial weights already give good approximation of SOM weights [8]. The network must be fed a large number of example vectors that represent, as close as possible, the kinds of vectors expected during mapping. The examples are usually administered several times. The training utilizes competitive learning. When a training example is fed to the network, its Euclidean distance to all weight vectors is computed. The neuron with weight vector most similar to the input is called the best matching unit (BMU). The weights of the BMU and neurons close to it in the SOM lattice are adjusted towards the input vector. The magnitude of the change decreases with time and with distance from the BMU. The update formula for a neuron with weight vector Wv(t) is

$$d_j = \sum_{i=0}^{N-1} (x_i(t) - w_{ij}(t))^2 \qquad (1)$$

Where α (t) is a monotonically decreasing learning coefficient and D (t) is the input vector. The neighborhood function (v, t) depends on the lattice distance between the BMU and neuron v.

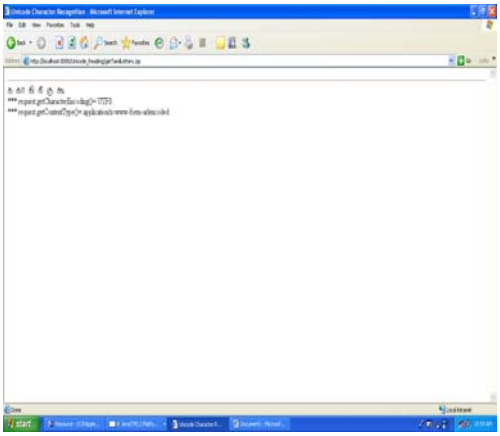

Fig.7. Mapped Unicode Characters

In the simplest form it is one for all neurons close enough to BMU and zero for others, but a gaussian function is a common choice, too. Regardless of the functional form, the neighborhood function shrinks with time. At the beginning when the neighborhood is broad,

the self-organizing takes place on the global scale. When the neighborhood has shrunk to just a couple of neurons the weights are converging to local estimates. This process is repeated for each input vector for a (usually large) number of cycles λ. The network winds up associating output nodes with groups or patterns in the input data set. If these patterns can be named, the names can be attached to the associated nodes in the trained net. During mapping, there will be one single winning neuron the neuron whose weight vector lies closest to the input vector [10]. This can be simply determined by calculating the Euclidean distance between input vector and weight vector. While representing input data as vectors has been emphasized in this article, it should be noted that any kind of object which can be represented digitally and which has an appropriate distance measure associated with it and in which the necessary operations for training are possible can be used to construct a self-organizing map. This includes matrices, continuous functions or even other self-organizing maps.

3.1.4 Preliminary Definitions

Consider a 10×10 array of nodes each of which contains a weight vector and is aware of its location in the array. Each weight vector is of the same dimension as the node's input vector. The weights are initially set to random values. Now we need input to feed the map [9]. (The generated map and the given input exist in separate subspaces.) We will create three vectors to represent colors. Colors can be represented by their red, green, and blue components. Consequently our input vectors will have three components, each corresponding to a color space.

4. Algorithm for Kohonon's SOM

(1) Assume output nodes are connected in an array, (2) Assume that the network is fully connected - all nodes in input layer are connected to all nodes in output layer. (3) Use the competitive learning algorithm.

$$\left|\omega_i - x\right| \le \left|\omega_k - x\right| \qquad \forall k$$

$$w_k(new) = w_k(old) + \mu\, \aleph(i,k)\,(x - w_k) \quad (2)$$

Randomly choose an input vector x , Determine the "winning" output node i, where w_i is the weight vector connecting the inputs to output node.

5. Unicode Mapping

The Unicode standard reflects the basic principle which emphasizes that each character code has a width of 16 bits. Unicode text is simple to parse and process and Unicode characters have well defined semantics [3] [7]. Hence Unicode is chosen as the encoding scheme for the current work. After classification the characters are recognized and a mapping table is created in which the unicodes for the corresponding characters are mapped.

6. Character recognition

The scanned image is passed through various blocks of functions and finally compared with the recognition details from the mapping table from which corresponding unicodes are accessed and printed using standard Unicode fonts so that the Character Recognition is achieved.

7. Conclusion

The Tamil Character Recognition is implemented using a Java Neural Network. A complete tool bar is also provided for training, recognizing and editing options. Character Recognition is aimed at recognizing handwritten Tamil document. The input document is read preprocessed, feature extracted and recognized and the recognized text is displayed in a picture box. Tamil is an ancient language. Maintaining and getting the contents from and to the books is very difficult. Character Recognition eliminates the difficulty by making the data available in handwritten format. In a way Character Recognition provides a paperless environment. Character Recognition provides knowledge exchange by easier means. If a knowledge base of rich Tamil contents is created, it can be accessed by people of varying categories with ease and comfort.

References

[1] B. Scholkopf, P. Simard, A. Smola, and V. Vapnik, "Prior knowledge in support vector kernels," in Advances in Neural Inf. Proc. Systems, vol. 10. MIT Press, 1998, pp. 640–646.

[2] G. Seni, V. Krispasundar "Generalizing Edit Distance to Incorporate Domain Information" Pattern Recognition, vol. 29 no.3, pp. 405-414, 2002

[3] B. Heisele, P. Ho, and T. Poggio, "Face recognition with support vector machines: global versus component-based approach," in ICCV, 2001, pp. 688–694.

[4] O. Chapelle, P. Haffner, and V. Vapnik, "SOM for histogram-based image classification," IEEE Transactions on Neural Networks, 1999.

[5] F. Jing, M. Li, H. Zhang, and B. Zhang, "Self Organizing Map for region-based image retrieval," in Proc. IEEE International Conference on Multimedia and Expo, 2003.
[6] S. Belongie, C. Fowlkes, F. Chung, and J. Malik, "Spectral partitioning with indefinite kernels using the nystrom extention," in ECCV, part III, Copenhagen, Denmark, may 2002,
[7] C. Wallraven, B. Caputo, and A. Graf, "Recognition with local features: the kernel recipe," in Proceedings of the International Conference on Computer Vision, vol. I, 2003, p. 257ff.
[8] T. Evgeniou, M. Pontil, and T. Poggio, "Regularization networks and support vector machines," Advances in Computational Mathematics, vol. 13, pp. 1–50, 2000.
[9] V. Vapnik, The nature of statistical learning Theory. John Wiley and sons, New York, 1995.
[10] D. M. J. Tax and R. Duin, "Uniform object generation for optimizing one-class classifiers," Journal of Machine Learning Research, Special Issue on Kernel methods, no. 2, pp. 155–173, 2002.
[11] C. Papageorgiou and T. Poggio, "A trainable system for object detection," International Journal of Computer Vision, vol. 38, no. 1, pp. 15–33, 2000.
[12] G. Guodong, S. Li, and C. Kapluk, "character recognition by neural network," in Proc. IEEE International Conference on Automatic Face and Gesture Recognition, 2000, pp. 196–201.
[13] K. Jonsson, J. Matas, J. Kittler, and Y. Li, "Learning Support Vectors for face verification and recognition," in Proc. IEEE Int Conf on AutomaticFace and Gesture Recognition, 2000.

Prof. Dr. J. Venkatesh received a MBA degree in 1997 and a PhD in System Information in 2008, both from the University of Bharathiar, Tamilnadu, India. He is working as an Assistant Professor at Anna University Coimbatore, Tamilnadu, India, specialised in the field of Systems and Production. He published many papers on computer vision applied to automation, motion analysis, image matching, image classification and view-based object recognition and management oriented empirical and conceptual papers in leading journals and magazines. His present research focuses on statistical learning and its application to computer vision and image understanding and problem recognition.

Mr. C . Sureshkumar received a ME degree in Computer Science in 2006, from the University of Anna, Tamilnadu, India. He is a part-time PhD research scholar in the Department of Computer Science and Engineering, Anna University Coimbatore. His main interests in research are Hand written Tamil Character recognition refers to the process of conversion of handwritten Tamil character into printed Tamil character.