Off-line Handwritten Character Recognition using Hidden Markov Model

Gayathri P
Department of Computer Science & Engineering
SCMS School of Engineering and Technology
Ernakulam, Kerala, India
gayathrip1990@gmail.com

Sonal Ayyappan
Department of Computer Science & Engineering
SCMS School of Engineering and Technology
Ernakulam, Kerala, India
sonalayyappan@yahoo.com

Abstract— **In this paper, we are presenting a method for the recognition of Malayalam handwritten vowels using Hidden Markov Model (HMM). OCR is a method to detect characters in different sources. The goal of OCR is to classify optical patterns in an image to the corresponding characters. Recognition of handwritten Malayalam vowels is proposed in this paper. Images of the characters written by eighteen subjects are used for this experiment. Training and recognition are performed using Hidden Markov Model Toolkit. Recognition process involves several steps including image acquisition, dataset preparation, pre-processing, feature extraction, training and recognition. An average accuracy of about 81.38% has been obtained.**

Keywords - Handwritten character recognition; Hidden Markov Model; Optical Character Recognition; Binarization; Normalization; Pre-processing; Feature Extraction.

I. INTRODUCTION

Character recognition is the process of recognizing typed, printed or handwritten characters and converting them into machine-readable code. Handwriting recognition can be divided into two based on the input given to the recognition system. (a) On-line handwriting recognition. (b) Off-line handwriting recognition. In an off-line system, the writing in the documents is optically captured by a scanner and the complete characters in the writing are available as images. But, in an on-line recognition system the two dimensional coordinates are represented as a function of time. Character recognition can be used for automatic number plate recognition, converting handwriting in real time to control a computer, as a reading aid for blind etc.

We propose a method for off-line handwritten Malayalam alphabet recognition. The vowels in Malayalam language are used for conducting the experiment. Total of 2340 character images are collected. The database is created by collecting the characters written by 18 people. The recognition is performed using HTK [8].

Hidden Markov Models [11] are used to recognize character, word or text. The basic theory of Hidden Markov model was developed in 1960s. They are statistical models which are efficient for a wide spectrum of applications, especially speech processing.

Most of the official documents and forms used in the Kerala are in Malayalam. For example, the forms used for buying a land [14] are in Malayalam. So, there must be a system that is able to process this type of documents. There exists no efficient algorithm for the recognition of Malayalam characters. In the field of printed texts also there exists a little advancement for Malayalam language. Even if the official language of Kerala is Malayalam, only a little development was made in this area of Malayalam character recognition. Nowadays the Government of Kerala is promoting the development of Malayalam language and the scope of improvement in this field is promising.

Remaining sections are organized as: Section II gives some related works on character recognition, Section III contains our proposed methodology, handwritten character recognition (HCR), Section IV includes the results of the experiment conducted and finally the conclusion is given in section VI.

II. RELATED WORK

The authors in [1], proposed a method for off-line handwritten character recognition. The methodology proposed in this paper was based on a feature extraction technique by recursively subdividing the character image. They introduced a two-stage classification scheme. Handwritten character databases such as CEDAR and CIL and handwritten digit databases such as MNIST and CEDAR were used for conducting this experiment. The recognition result achieved for the method proposed in this paper was 94.73% for the CEDAR character database and 99.03% for the MNIST database.

In [3], the authors proposed a method for recognizing isolated and combinational handwritten characters in a noiseless environment. They classified the characters into three categories. An algorithm was devised to inveterate characteristic features to recognize the characters with perceptive accuracy by capturing the intensity variations. This algorithm recognized the antediluvian script of Malayalam characters that are connected in nature. The output is the editable version of the recognized Malayalam characters. This algorithm was tested for three sets of samples containing 402 letters and was conducted in a noiseless environment. An accuracy of 94% was obtained for this method.

The authors in [4], proposed a method which aimed at training a simple neural network having three layers using back propagation algorithm. The characters are represented as a feature vector using free-man code. This feature vector acted as inputs to the neural network during the training and recognition. The output was the character expressed in Unicode format. The handwritten Malayalam character captured using a digital pen was converted into editable characters in the computer in one of the recognized fonts of the Malayalam language.

The authors in [6], developed a Character Recognition System, where a character matrix and a Network Structure key was created. They used a Feed Forward Algorithm which gives an insight into the entire working of the neural network. This was followed by a back-propagation algorithm. They perform the recognition of handwritten English characters using a multi-layer perceptron which has only one hidden layer. They obtained an accuracy of more than 70%.

In [7], the authors proposed a technique for the recognition of handwritten Devanagari characters. In this method, the feature vector is constituted by accumulating directional gradient changes inside different segments, number of intersection points in the character, type of spine present and type of shirorekha present. They divided the character image into different sized segments. For each segment they computed the gradient changes. These gradient values are normalized for performing training and testing of Neural Network. This method was applied on a database containing 1000 character samples and the recognition rate obtained was 88.12%.

The authors of [9] proposed a method for the recognition of handwritten characters using multi-layer feed forward neural network. Diagonal based feature extraction method was used in this paper. The neural network was trained using fifty datasets, each containing 26 alphabets and tested using 570 different handwritten characters.

III. PROPOSED METHODOLOGY

The Optical Character Recognition system automatically recognizes characters from a given document and also it provides the recognition of handwritten or printed characters, letters, texts, numerals and symbols into a machine-readable code.

A. Character Recognition System

The character recognition system proposed in this paper recognizes handwritten Malayalam vowels. The data are collected from 18 people. There are 13 vowels in Malayalam. List of the characters used in this experiment are shown in Table I. Figure 1 is the architecture of the character recognition system. We have two databases, the training and testing database. In order to train the model, the images in the training database are given to the preprocessing step. Then features are extracted from these preprocessed images. These extracted features are then used for training the model. Using these trained models classification of the images in the test database can be performed.

TABLE I. VOWELS IN MALAYALAM

Malayalam Characters
ah
aahh
e
ee
u
uu
ru
a
aa
i
o
oo
ou

I. Dataset Preparation

The dataset used in this experiment, consists of 2340 images of Malayalam vowels of various appearances. We used a flat-bed scanner for scanning the handwritten characters. Characters are extracted from this scanned image by using a fixed size rectangular window. This is divided into training and test sets each consisting of 1170 images. The data is collected from 18 people. For each alphabet 180 images are collected. Out of these 90 images are used in the training set and the remaining 90 for testing.

II. Training And Classification Process

a. Image Acquisition

The first step in character recognition is the acquisition of image. In this stage, the recognition system gets a scanned image as an input. The images used in this experiment are in *.pbm* format. The image can be acquired using a digital camera, scanner or any other digital input device. Here we use a scanner for acquiring the image.

b. Pre-Processing

It is a series of operations which is performed on a scanned input image in order to make it suitable for feature extraction. The different steps in pre-processing are,

i. **Binarization -** In this stage, the collected RGB images are converted into binary image. This is done by selecting a threshold value [5]. Figure 2a and 2b shows an RGB image and its corresponding binary image.
ii. **Normalization -** In this step, the size of the collected images are normalized, which means all images

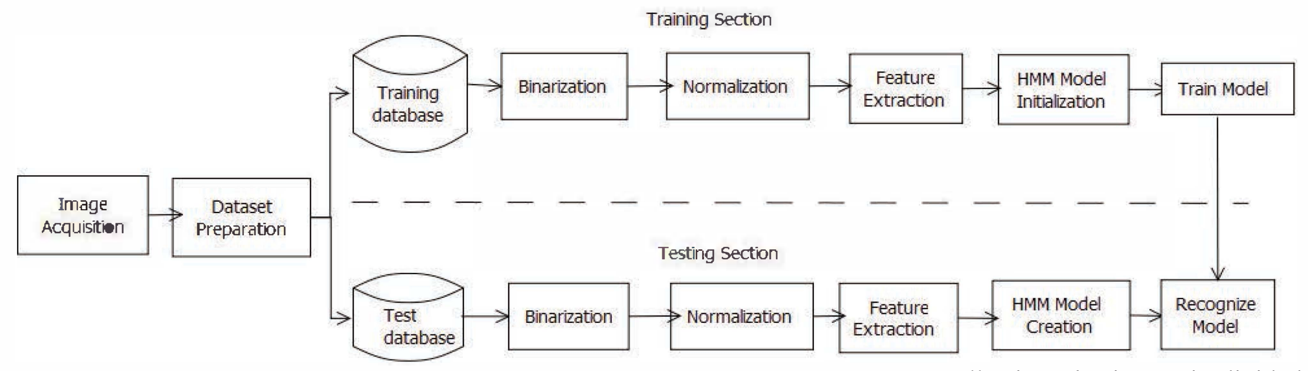

Fig 1. Character Recognition System

are converted into same size, 125 x 125. Since, some Malayalam alphabets occupy more space compared to English alphabets, all alphabets must be normalized to the size of the largest alphabet in the dataset.

Initially, the collected RGB images are converted into binary.

Fig.2a. RGB image of the letter ah

Fig.2b. Binary image of the letter ah

c. Feature Extraction

Feature extraction extracts the features from the input image. This is an important stage in character recognition because, its better functioning will improve the recognition rate and thereby reduces the misclassification rate. A total of 17 features are extracted from the character image. Thus a feature vector table is generated. The features are,

i. Angle between the base line and global center of gravity as shown in Figure 3. Global center of gravity is the center of gravity of the original image [10].

ii. Aspect ratio.

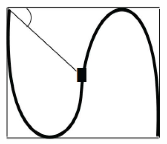

Fig.3. Angle between baseline and global center of gravity for the letter gha

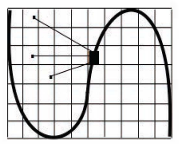

Fig.4. Distance between local center of gravities and global center of gravity for the letter gha

iii. Distance between local center of gravities in each cell and global center of gravity as shown in Figure 4. Local center of gravity is the center

:ell when the image is divided 2].

iv. Number of black pixels in each of the region above the diagonals of the four rectangles as shown in Figure 5.

v. Number of black pixels in each of the region below the diagonals of the four rectangles as shown in Figure 6.

vi. Number of black pixels in the region to the left of the vertical line which equally divides the image into 2.

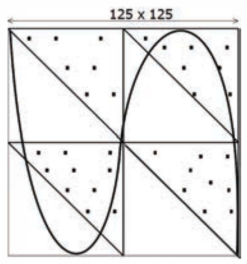

Fig.5 Region above the diagonal in each of the four cells for the letter gha

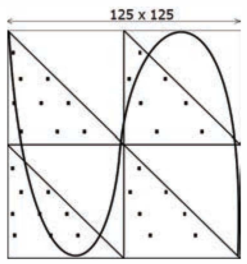

Fig.6 Region below the diagonal in each of the four cells for the letter gha

vii. Number of black pixels in the region to the right of the vertical line which equally divides the image into 2.

viii. Number of black pixels in the region above the horizontal line which equally divides the image into two.

ix. Number of black pixels in the region below the horizontal line which equally divides the image into two.

x. Number of black pixels above the main diagonal of the image.

xi. Number of black pixels below the main diagonal of the image.

xii. Intensity value of the diagonal elements (25 values) in each cell (containing 625 pixels) obtained by equally dividing the 125 x 125 image into an 5 x 5 grids as shown in Figure 7 [13].

xiii. Number of black pixels in each cell obtained by equally dividing the 125 x 125 image into 5 x 5 grids.

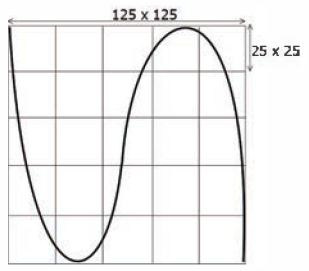

Fig.7. 125 x 125 size image of the letter gha is divided into a 25 x 25 grid

xiv. Number of black pixels in the main diagonal of the image.
xv. Number of black pixels in the second diagonal of the image.
xvi. Number of black pixels in the vertical line passing through the center of the image.
xvii. Number of black pixels in the horizontal line passing through the center of the image.

d. Training and Classification

The FVT created during feature extraction is used for classification. In the 17 features extracted, the third feature has 25 values, fourth and fifth has 4 values each, 12th feature has 625 values, 13th has 25 values and all others have only a single value. Hence, the feature chain created consists of 695 (25+4+4+625+25+12) feature values. In the FVT some intensity values are constant for all the classes. Since HTK does not support constant values, these values are removed from the FVT. Therefore, the FVT contains only 238 feature values. These values must be normalized before performing training. Training and classification can be performed using Hidden Markov Model Toolkit (HTK).

B. Hidden Markov Models

HMM [2] is a statistical model based on Markov model. Now HMM is a wide spread model in various fields such as speech recognition, hand written characters recognition, gesture recognition, face recognition etc. It is a model with a finite set of states and each state is associated with a probability. Transitions between the states depend on a set of probabilities called the transition probabilities. The transition from one state to the next is a Markov process since the next state depends only on the current state and the fixed probabilities. But the state through which the model pass through is unknown to us. We only know the observation sequence, probability of transitioning from one state to another, the probability that for the current state what are the chances that given observation sequence will be generated.

a. Hidden Markov Model Toolkit (HTK)

HTK [8] was initially developed for speech recognition technology. But recently, it has been used for character recognition. Figure 8 shows the overall description of different stages in training and recognition using HTK.

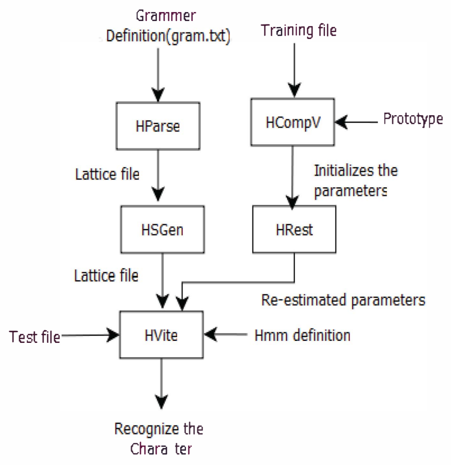

Fig.8. Training and recognition procedure using HTK

TABLE II. OPTIONS USED BY DIFFERENT HTK TOOLS

Options	Description
-A	Print command line arguments
-C	To specify configuration file
-D	Display configuration variables
-H	Load HMM macro file that is specified
-M	Store output HMM macro files in the directory specified
-S	Use command line script file specified
-T	Set trace level to the number specified
-m	To update the mean
-f	The value to be multiplied with the global variance to create variance floor.
-i (HRest)	Maximum number of iterations
-i (HVite)	To specify the output file
-w	To specify the lattice file

Table II contains the list of all options used by different HTK tools. Different phases of training and recognition using HTK are,

i. Data preparation

In this stage we will define a word network using HTK's Standard Lattice Format (SLF). The toolkit also provides a grammar definition format, and a tool that can build the word network automatically called `HParse`. We write a grammar in a file called `grammer.txt`. The content of the file `grammer.txt` is,

```
$WORD = a | aa | aahh | ah | e | ee | i
 | o | oo | ou | ru | u | uu;([$WORD])
```

The command for creating the word network is,

```
HParse -A -D -T 1 grammer.txt
          network.slf
```

Then we create a file called `hmmlist.txt` which contains the list of all characters and a file called `dictionary.txt` which contains the sorted list of characters to be trained. The content of `dictionary.txt` is,

```
a [a] a
aa [aa] aa
aahh [aahh] aahh
ah [ah] ah
e [e] e
ee [ee] ee
i [i] i
o [o] o
oo [oo] oo
ou [ou] ou
ru [ru] ru
u [u] u
uu [uu] uu
```

The generated word network can be tested using the following command.

```
HSGen -A -D -n 4 -s network.slf
         dictionary.txt
```

ii. Model Training

The first task in training is the definition of a prototype for each of the model to be trained, that includes the model topology, transition parameters and output distribution parameters. This depends on the number of states and the feature extracted from each character. In our application, there are 238 features and 4 states for each of the character to be recognized. Part of the prototype defined for the model *a* is given below.

```
~o <VecSize> 238 <USER>
~h "a"
<BeginHMM>
   <NumStates> 4
   <State> 2
      <Mean> 238
         0.0 . . . 0.0
         .
         .
      <Variance> 238
         1.0 . . . 1.0
         .
         .
   <State> 3
      <Mean> 238
         0.0 . . . 0.0
         .
         .
      <Variance> 238
         1.0 . . . 1.0
         .
         .
      <TransP> 4
      0.0   0.5   0.5   0.0
      .
      .
      0.0   0.0   0.0   0.0
<EndHMM>
```

The names of the FVTs are given in the files `trainlist.txt` and `testlist.txt`. These csv files must be converted into HTK format before its usage. Using the prototype definition HTK initializes the parameters of an HMM. The `HCompV` command is used for this.

```
HCompV -A -D -T 1 -S trainlist_a.txt -M
model/hmm0flat -H model/proto/hmm_a.txt
            -f 0.01 -m a
```

`HRest` command performs a re-estimation of the parameters, using an embedded training using the Baum-Welch algorithm. The format of `HRest` command is,

```
HRest -A -D -T 1 -i 50 -m 1 -S
trainlist_a.txt -M model/hmm1 -H
model/hmm0flat/vFloors_a.txt -H
model/hmm0flat/hmm_a.txt a
```

After the re-estimation process, the HMM model is written into `hmmsdef.mmf` file which contains the definition of all the trained models and this is then used for recognition. Part of the HMM definition file is given below.

```
~o
<STREAMINFO> 1 238
<VECSIZE> 238<NULLD><USER><DIAGC>
~h "a"
<BEGINHMM>
<NUMSTATES> 4
<STATE> 2
<MEAN> 238
 1.885979e+00 . . . 2.408333e+02
 .
 .
 3.411808e+03 . . . 2.761553e+03
 <TRANSP> 4
 0.0000e+00 . . . 0.000000e+00
 .
 .
 0.0000e+00 . . . 0.000000e+00
<ENDHMM>
```

iii. Recognition

Recognition is performed using the recognition tool `HVite`. This is a general purpose Viterbi word recognizer. `HVite` command uses the word network, the dictionary, the list of HMMs and the `hmmsdef.mmf` file for recognition. `HVite` writes the result into a Master Label File with an extension .mlf. The format of the `HVite` command is,

```
HVite -A -D -T 1 -H hmmsdef.mmf -i
results/reco_a.mlf -w network.slf
dictionary.txt hmmlist.txt -S
testlist_a.txt
```

IV. CLASSIFICATION RESULT

We extracted 17 features from the character images. The FVT is then normalized and converted into standard HTK format. This is then given as input to HTK for training and testing. HTK will classify the unknown samples in the test database. Both the train and test database contains 90 character images. The accuracy obtained for each class is shown in Table III. Accuracy of each class is computed using equation (1).

$$Accuracy = \frac{total\,number of\ samples correctly classified}{total\,number of\ samples in\,the\,class} \tag{1}$$

TABLE III. ACCURACY IN RECOGNIZING THE CHARACTERS

Alphabets	**Accuracy**
ah	83%
aahh	67%
e	81%
ee	81%
uh	90%
uuhh	87%
ru	78%
a	84%
aa	75%
i	87%
oh	82%
oohh	83%
ou	80%

- Most of the *aahh*s are misclassified as *ah*. That is why the accuracy of *aahh* is very less.
- Some *aa*s are misclassified as *a* and some *uuhh*s are misclassified as *uh*. This is because their shapes are similar.

V. CONCLUSION

We have developed a Hidden Markov Model (HMM) based system for handwritten character recognition for Malayalam vowels. The system consists of three stages; the first one is the preprocessing, in which the character images are converted into binary images and the image is then normalized. Then features are extracted from these normalized images. The other part consists of the Hidden Markov Model Toolkit operations that are used to initialize the HMM models and to recognize a particular model. The developed system has been trained and tested using 13 Malayalam vowels and obtained an average accuracy of about 81.38%. The accuracy obtained for HTK can be improved by applying some feature selection methods on the generated FVT. Feature selection methods are used for dimensionality reduction. This will remove the redundant features from the feature space and thus improves the accuracy. In future, this work can be extended by introducing some feature selection methods.

REFERENCES

[1]. Georgios Vamvakas, Basilis Gatos, Stavros J. Perantonis, “Handwritten character recognition through two-stage foreground sub-sampling”, Pattern Recognition, 2010.

[2]. L. R. Rabiner, “A tutorial on hidden Markov models and of selected applications in speech recognition”, Proceedings the IEEE, Vol. 77, pp. 257 to 286, February 1989.

[3]. Abdul Rahiman M and Rajasree M S, “An Efficient Character Recognition System for Handwritten Malayalam Characters Based on Intensity Variations”, International Journal of Computer Theory and Engineering, Vol. 3, No. 3, June 2011.

[4]. Amritha Sampath, Tripti C, Govindaru V, “Freeman Code Based Online Handwritten Character Recognition For Malayalam Using Backpropagation Neural Networks”, Advanced Computing: An International Journal (ACIJ), Vol.3, No.4, July 2012.

[5]. Md. Abul Hasnat, “Research Report on Bangla OCR Training and Testing Methods”, 2010.

[6]. Vijay Patil, Sanjay Shimpi, “Handwritten English Character Recognition Using Neural Network”, Elixir Comp. Sci. Engg. 41, 5587-5591, 2011.

[7]. Sandhya Arora, Latesh Malik, Debotosh Bhattacharjee, Mita Nasipuri, “A Novel Approach For Handwritten Devnagari Character Recognition”, 2010.

[8]. The HTK Book (for HTK Version 3.4) available at http://htk.eng.cam.ac.uk/docs/docs.shtml.

[9]. J.Pradeep, E.Srinivasan and S.Himavathi, “Diagonal Based Feature Extraction For Handwritten Alphabets Recognition System Using Neural Network”, International Journal of Computer Science Information Technology (IJCSIT), Vol 3, No 1, Feb 2011.

[10]. Amit Kishore Shukla , Pulkit Mohan , Gaurav Ojha , Manoj Wariya , “Off-line signature verification system using grid and tree based feature extraction”, International Conference on Issues and Challenges in Intelligent Computing Techniques (ICICT), 2014.

[11]. Mark Stamp, “A Revealing Introduction to Hidden Markov Models”, 2012.

[12]. Gayathri P, Sonal Ayyappan, “Handwritten Character Recognition”, In the proceedings of IEEE’s International Conferences For Convergence of Technology (I2CT), Pune, April 6-8, 2014.

[13]. Gayathri P, Sonal Ayyappan, “Robust Feature Extraction Scheme for Handwriting Recognition”, In the proceedings of IEEE’s Fourth International Conference on Advances in Engineering & Technology (ICAET), Nagapattinam, May 2-3, 2014.

[14]. http://keralaregistration.gov.in/pearlpublic/downloads/GO_formats.pdf?tok=j833uk3ks4us42