Javanese Character Recognition
Using Hidden Markov Model

Anastasia Rita Widiarti, and Phalita Nari Wastu

Abstract—Hidden Markov Model (HMM) is a stochastic method which has been used in various signal processing and character recognition. This study proposes to use HMM to recognize Javanese characters from a number of different handwritings, whereby HMM is used to optimize the number of state and feature extraction. An 85.7% accuracy is obtained as the best result in 16-stated vertical model using pure HMM. This initial result is satisfactory for prompting further research.

Keywords—character recognition, off-line handwriting recognition, Hidden Markov Model.

I. INTRODUCTION

The richness of Javanese culture is stored in many ancient books written in Javanese handwritings. Unfortunately, only a few people could actually read those manuscripts. As these manuscripts have valuable contribution to knowledge, automatic off-line handwriting recognition systems are needed to provide bigger access to these manuscripts.

Fig. 1 The Legena of Javanese character [2]

Different methods have been applied in many handwriting recognition researches. One of the most successful and popular method is Hidden Markov Model, which works well not only for word or character recognition, but also for signal processing. In this study HMM is used to recognize Javanese handwritings. As mentioned by T.E. Behren [1], Javanese script has 103 characters, i.e. 20 main phonetic characters called legena or dentawyanyaja, 20 characters of pasangan, and 63 character of sandhangan which include all numbers and special characters used in poems and folksongs. The characters recognized in this study are the legena, as shown in Fig. 1.

The main goal of this study is to find the accuracy of HMM to recognize the shape of Javanese characters. The secondary goal is to figure out which feature extraction performs the best result among other feature extraction possibilities.

II. PREPROCESSING

To perform the complete tasks of off-line handwriting recognition, selected Javanese documents are scanned and pre-processed to filter the characters from background noise. The scanned documents are 1549x2340 pixel RGB images, which is then transformed into binary images. Segmentation into separated characters follows, continued by resizing the image to make all characters into the same size 72x72 pixels image.

III. FEATURE EXTRACTION

Feature is an instance used as a model in Hidden Markov Method. This instance is extracted during the training and testing. Extraction is done by taking the discrete pixel information from each vector. Because there is no previous work that provides information about the best Javanese handwriting feature extraction, the best form within possible extractions should be found. Vertical and horizontal feature extraction is then conducted, adopted from the articles by Nopsuwanchai and Povey [3], and by Theeramunkong at al [5]. A Thai feature extraction is adapted to use in this study based on the type similarity of Thai and Javanese characters. Four (4) feature extractions for the Javanese characters are:

1. Character divided into 1 horizontal vector (1H).
2. Character divided into 2 horizontal vectors (2H).
3. Character divided into 1 vertical vector (1V).
4. Character divided into 2 vertical vectors (2V).

Fig. 2 shows an example of four feature extraction on Ha character.

IV. HIDDEN MARKOV MODELS (HMM)

HMM is a stochastic system which is assumed from the Markov chain with unknown parameters, and the challenge is to find the hidden parameter(s) from observed parameters [4]. Continuous models are performed during the signal
processing, otherwise discrete HMM is done at the image processing as this study does.

Fig. 2 Feature extractions on Javanese character recognition. (a) 1H; (b) 2H; (c) 1V; (d) 2V;

After features are extracted, a Markov model is able to be generated. From the discrete data, HMM generates a chain that consists of a number of states with transition probabilities that make these states connect to others as shown in Fig. 3.

As explained by L. R. Rabiner [6], Hidden Markov Model can be generated from parameters:

\[ \lambda = (A, B, \pi) \]  

Where:

1. \( N \) is the number of state, denoted as \( S = \{S_1, S_2, S_3, \ldots, S_n\} \) and state at the time \( t \) is \( q_t \).
2. \( M \) is the number of observed symbol at any state. Individual symbols denoted as \( V = \{V_1, V_2, V_3, \ldots, V_m\} \).
3. Transition probability matrices \( A = \{a_{ij}\} \), where \( a_{ij} = P(q_{t+1} = S_j | q_t = S_i) \), \( 1 \leq i, j \leq N \)  
   \[ a_{ij} \geq 0 \] and \( \sum_{j=1}^{N} a_{ij} = 1 \)  
4. The observation symbol probability distribution in state \( j \), \( B = \{b_j(k)\} \) where \( b_j(k) = P(v_k \text{ in } t | q_t = S_j) \), \( 1 \leq j \leq N, 1 \leq k \leq M \)
5. The initial state distribution \( \pi = \{\pi_i\} \), where \( \pi_i = P(q_1 = S_i) \), \( 1 \leq i \leq N \)

At the appropriate value of \( N, M, A, B \) and \( \pi \), HMM can be used as a generator to give an observation sequence:

\[ O = O_1 O_2 O_3 \ldots O_t \]  

where each observation \( O_i \) is one of the symbols from \( V \), and \( T \) is the number of observations in the sequence.

Fig. 3 Figure of Markov chain generated from the “ha” character

A. Data Trains

Every legena character is modelled by the forward-backward algorithm or Baum-Welch algorithm. This algorithm is used to find means and variance as maximum likelihood to model all 20 legena characters, calculated from the observation sequence \( O \) and HMM \( \lambda = (A, B, \pi) \).

Forward variable is defined as partial observation from sequence state probability, denoted as \( O_1, O_2, \ldots, O_t \) (until time \( t \)) and state \( S_i \) at the time \( t \), with \( \lambda \) and \( \alpha \) as \( \alpha(i) \) as shown in Fig. 4. Backward variable is defined as partial observation from sequence state probability \( t+1 \) to the current state, where state \( S_i \) at the time \( t \), with \( \lambda \) and \( \alpha \) as \( \alpha(i) \) as shown in Fig. 5. This sequence state probability is denoted as:

\[ P(O | \lambda) = \sum_{i=1}^{N} \alpha_i(i) \beta_t(i) = \sum_{i=1}^{N} \alpha_i(i) \]  

Where:

\[ \gamma_t(i) = \frac{\alpha_i(i) \beta_t(i)}{P(O | \lambda)} \]  

A Vector Quantifier (VQ) is needed to map each observation vector to an indexed codebook [6]. Baum-Welch definitions to estimate the new model \( \lambda = (A, B, \pi) \) are:

\[ \gamma_t(i) = \frac{\alpha_i(i) \beta_t(i)}{P(O | \lambda)} \]  

\[ \gamma_t(i) = \frac{\alpha_i(i) \beta_t(i)}{P(O | \lambda)} \]
\[
\bar{a}_{ij} = \frac{\sum_{t=1}^{T-1} \zeta_t(i,j)}{\sum_{i=1}^{T} \gamma_t(i)}
\]
\[
\bar{b}_j(V_t) = \frac{\sum_{t=1}^{T} \gamma_t'(i)}{\sum_{i=1}^{T} \gamma_t(i)}
\]
with
\[
\zeta_t(i,j) = \frac{a_t(i)a_t(j)}{P(O_t | \lambda)}
\]

Where:
1. \( \pi_t \) is the expected frequency in state \( S_t \) at time \( t=1 \),
2. \( \bar{a}_{ij} \) is the expected number of transitions from state \( S_i \) to state \( S_j \) divided by the expected number of transitions from state \( S_i \),
3. \( \bar{b}_j(k) \) is the expected number of times in state \( j \) and observing symbol \( \nu_k \) divided by the expected number of times in state \( j \).

B. The Use of Viterbi Algorithm in Testing Phase
Best path can be found by using the Viterbi algorithm. Fig. 5 shows illustration of search best path, and path 12, 22 is the best path probability. The closest probability state sequence \( Q = \{q_1, q_2, ..., q_T\} \) is calculated to the observed sequence \( O = \{O_1, O_2, ..., O_T\} \) that can be defined as:
\[
\delta_t(i) = \max_{q_1q_2...q_t} \delta_{t-1}(i)P(O_t|O_{t-1}, q_t, \lambda)
\]
\[
\delta_T(i) \] is the best probability at time \( T \), which is calculated at first observation \( T \) and ends in state \( S_t \).

Fig. 5 Maximum path found by Viterbi algorithm. Bold line is the best path probability

V. IMPLEMENTATION
The system is used to recognize legena characters from the books written by Raden Ngabehi Yasadipura [7] and [8]. Data input is scanned images from these books, and the process can be seen in Fig. 6. Images that are inputted into the system are the pre-processed images, which are extracted from the training and testing as seen in Fig. 7.

VI. EXPERIMENTAL RESULT
This study uses 1000 Javanese characters as input consisting of 20 legena characters with 50 samples for each character. A 5-fold cross validation method is performed, which tests 200 sets of legena, and trains the remaining 800.

1. Experiment I
Components:
a. 20 character of legena, which are ha, na, ca, ra, ka, da, ta, sa, wa, la, ma, ga, bha, tha, nga, pa, dha, ja, ya, nya
b. Total character is 20x50 = 1000.
c. Characters are resized to be 72 x 72 binary images (0 and 1 pixel information).
d. Characters are divided into 72 horizontal vectors (1H).
e. The data inputted are characters which have been thinned and also character which have not been thinned.

As result of experiment I look in the Table II, and the result of 5-fold cross validation see Table I.

<table>
<thead>
<tr>
<th>TABLE I</th>
<th>CONFUSION MATRIX FOR THE 18-STATED 1H FEATURE</th>
</tr>
</thead>
<tbody>
<tr>
<td>ha</td>
<td>na</td>
</tr>
<tr>
<td>26</td>
<td>0</td>
</tr>
<tr>
<td>na</td>
<td>0</td>
</tr>
<tr>
<td>ra</td>
<td>1</td>
</tr>
<tr>
<td>ka</td>
<td>0</td>
</tr>
<tr>
<td>da</td>
<td>0</td>
</tr>
<tr>
<td>sa</td>
<td>0</td>
</tr>
<tr>
<td>wa</td>
<td>0</td>
</tr>
<tr>
<td>la</td>
<td>0</td>
</tr>
<tr>
<td>ma</td>
<td>0</td>
</tr>
<tr>
<td>ga</td>
<td>0</td>
</tr>
<tr>
<td>bha</td>
<td>0</td>
</tr>
<tr>
<td>tha</td>
<td>0</td>
</tr>
<tr>
<td>nga</td>
<td>0</td>
</tr>
<tr>
<td>pa</td>
<td>0</td>
</tr>
<tr>
<td>dha</td>
<td>0</td>
</tr>
<tr>
<td>nga</td>
<td>0</td>
</tr>
</tbody>
</table>
TABLE II
Accuracy at 1H Features

<table>
<thead>
<tr>
<th>Number of State</th>
<th>Accuracy (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>15</td>
<td>81</td>
</tr>
<tr>
<td>16</td>
<td>81</td>
</tr>
<tr>
<td>17</td>
<td>84.4</td>
</tr>
<tr>
<td>18</td>
<td>85.4</td>
</tr>
<tr>
<td>19</td>
<td>83.4</td>
</tr>
<tr>
<td>20</td>
<td>83.2</td>
</tr>
<tr>
<td>21</td>
<td>83.8</td>
</tr>
<tr>
<td>22</td>
<td>81.7</td>
</tr>
</tbody>
</table>

TABLE III
Accuracy at 2H, 1V and 2V Features

<table>
<thead>
<tr>
<th>Number of State</th>
<th>2H Feature (%)</th>
<th>1V Feature (%)</th>
<th>2V Feature (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>15</td>
<td>68.8</td>
<td>85.0</td>
<td>74.1</td>
</tr>
<tr>
<td>16</td>
<td>69.1</td>
<td>85.7</td>
<td>76.2</td>
</tr>
<tr>
<td>17</td>
<td>71.2</td>
<td>83.2</td>
<td>74.3</td>
</tr>
<tr>
<td>18</td>
<td>71.2</td>
<td>85.1</td>
<td>75.6</td>
</tr>
<tr>
<td>19</td>
<td>71.8</td>
<td>83</td>
<td>79.9</td>
</tr>
<tr>
<td>20</td>
<td>70.8</td>
<td>83.3</td>
<td>81.5</td>
</tr>
<tr>
<td>21</td>
<td>72.6</td>
<td>79.6</td>
<td>86.4</td>
</tr>
<tr>
<td>22</td>
<td>71.5</td>
<td>77.5</td>
<td>81.7</td>
</tr>
</tbody>
</table>

Hidden Markov Models yields a good accuracy in Javanese-script character recognition, and it has good raw data handle and flexibility to input some parameters. This research should be continued for character recognition at the level of words, sentences, and even documents. Meanwhile, prototype has not yet been implemented for the rest of 103 Javanese characters.

ACKNOWLEDGMENT

This research was supported by Indonesian Director General of Higher Education (DIKTI) and Sanata Dharma University of Yogyakarta Indonesia.

REFERENCES


