USING HIDDEN MARKOV MODEL FOR CHINESE BUSINESS CARD RECOGNITION


*Department of Electronic Engineering, Fu Jen University
**Institute of Computer Science and Information Engineering, National Central University
***Department of Electrical Engineering, National Central University

ABSTRACT

Business card recognition is a difficult problem. Characters in business card are small with diverse font types. In this paper, an approach using left-right hidden Markov model is proposed for business card recognition. The hidden Markov model will output a top-10 candidate list as its recognition result. A postprocessing stage is followed to improve the recognition result. The postprocessing stage uses bigram table as linguistic information to search optimized recognition result from the top-10 candidate list. Our experiments are built on the recognition of company item and address item in Chinese business cards. Bigram table and hidden Markov models are trained with a telephony database. 100 address items and 30 company items are used for testing. Experimental results reveal the validity of our proposed method.

1. INTRODUCTION

Business cards convey much information. An automatic Chinese business card processing system can help us to convert the information. However, due to the huge number of character classes and high complexity of Chinese character, recognition of Chinese business card per se has been a very difficult task. One challenging problem of business card recognition is the small-sized characters in business card. Small characters will induce noisy and low-quality character images. It makes feature extraction be a hard work, especially for structural methods that extract strokes as features.

Multi-font is another problem that makes business card recognition difficult. Due to that a business card represents the person, diversity seems to be the nature of business card. Characters with versatile font types are usual. However, a business card recognition system can be trained with limited font types. Testing business cards with untrained font types will get low recognition rate.

In most systems, recognition rate is low so that they have to provide prior ten candidates of each character. The prior 10 candidates, or top-10 candidates, are the recognized characters with the most probabilities to be the correct character. With the top-10 candidates, users still need to pay great effort to manually choose the correct candidate.

Linguistic information can help the automatic choice of correct character from top-10 candidates. Literature [1] proposed a database matching scheme which uses a lexicon database to help choose the correct candidate. Chang [2] used linguistic information to correct errors, which may be produced by optical character recognition and input methods. Usually, the incorporation of linguistic information is based on the assumption of a statistical language model.

In this paper, a hidden Markov model based approach to recognize Chinese characters in business card is proposed. The system architecture is shown in Fig. 1. The HMM-based OCR presented in Section 2 recognizes characters. Postprocessing presented in Section 3 is adopted for improving the recognition rate of HMM-based OCR.

![Fig. 1 The flow diagram of the proposed approach.]

2. THE HIDDEN MARKOV MODEL

In recent years, hidden Markov models have been applied in several areas during the last 15 years, including speech recognition [3,4], language modeling, handwriting recognition, on-line signature verification [10], etc., due to its superior performance in solving the problems of segmentation, time warping, and stochastically randomness. For multi-font and hand-printed Chinese characters, it is observed that their distortion is quite similar to that of speech recognition, e.g., nonlinear shifting, multi-templates, etc., which is intrinsically stochastic. Many works for the application of HMM to character recognition have been present [5-9].

2.1 The hidden Markov model

A hidden Markov model is a doubly stochastic process with an underlying stochastic process that is not observable, but it can be observed through another stochastic process that
produces the sequence of observations. The hidden process consists of a set of states connected to each other by transition probability. The observed process consists of a set of outputs or observations. Each observation is contained in a state with some probability density function. The following will define the notations in the proposed HMM:

\( N \): The number of states in HMM, the set of states is denoted as \( \{ S_1, S_2, \cdots, S_N \} \).

\( M \): The number of observations that are produced in some state. Denoted as \( \tau_i = (x_1, x_2, \cdots, x_M) \), \( \tau_i \) represents the set of all observations that possibly appear, where \( x_1, x_2, \cdots, x_M \) are feature vectors.

\( O \): A sequence of observations. \( O = \{ o_1, o_2, \cdots, o_T \} \).

\( A \): The set of transition probabilities from a state to another state \( A = \{ a_{ij} \} \), where

\[
a_{ij} = P(q_t = S_j | q_{t-1} = S_i), 1 \leq i, j \leq N.
\]

\( B \): The set of observation probabilities. \( B = \{ b_j(k) \} \),

\[
b_j(k) = P(o_k = x_k | q_t = S_j), 1 \leq k \leq M, 1 \leq j \leq N.
\]

\( \pi \): The set of initial probabilities of states.

\[
\pi = \{ \pi_i \} \text{ where } \pi_i = P(q_1 = S_i), 1 \leq i \leq N.
\]

\( \lambda \) is known as the state transition probability distribution, and \( \pi \) as the symbol probability distribution. Now a Hidden Markov Model \( \lambda \) can be formulated as \( \lambda = (A, B, \pi). \) There are two problems in applying the above HMM.

Problem 1: How to find the probability \( P(O | \lambda) \)?

Problem 2: Given the model \( \lambda \) and an observation sequence \( O = \{ o_1, o_2, \cdots, o_T \} \), how to maximize the probability \( P(O | \lambda) \)?

Problem 1 is called the training problem. It is usually resolved by training \( \lambda \) with a set of training examples. Problem 2 is called the recognition problem. The observation sequence \( O \) is called testing sample, and the maximization of \( P(O | \lambda) \) is named a recognition process.

To decrease computation cost, the initial probability \( \pi \) and state transition probability \( A \) is usually defined as 1. The observation probability \( B \) is defined as a mixture of Gaussian distribution \( D_k \), which is formulated as follows:

\[
D_k(\tau_i) = (2\pi)^{-M/2} | \Sigma_k |^{-1/2} \exp \left\{ -\frac{1}{2} (\tau_i - \mu_k)^T \Sigma_k^{-1} (\tau_i - \mu_k) \right\}
\]

\( n \) is the dimension of feature vector; \( \tau_i \) is the feature vector of a training sample or testing sample; \( \mu_k \) is the mean vector of the mixture \( i \) of the state of the model; \( \Sigma_k \) is the covariance matrix of the mixture \( i \) of the state of the model. Practically, we calculate the natural logarithm of the observation probability. The function can be described as follows:

\[
\ln D_k(\tau_i) = \left( \frac{n}{2} \right) \ln (2\pi) + \frac{1}{2} \ln | \Sigma_k | + \frac{1}{2} (\tau_i - \mu_k)^T \Sigma_k^{-1} (\tau_i - \mu_k)
\]

Replacing \( \ln D_k(\tau_i) \) with \( b_j(\tau_i) \), we obtain an equation to calculate the observation probabilities \( B \).

Given a set of training observation sequences, a procedure called segmental K-mean algorithm, or Baum-Welch algorithm, can iteratively and automatically adjust \( b_j(\tau_i) \). The algorithm, which is an implementation of EM (expectation-maximization) algorithm in HMM case, guarantees that HMM converges to a local maximum according to the maximum likelihood estimation (MLE) criterion. The segmental K-mean algorithm is described as follows:

Step 1. At first, equally assign the observations to each state. Calculate the mean vector \( \mu_k \) and the covariance vector \( \Sigma_k \) of each state. The initial value of \( B \) is obtained from the Equation 2.

Step 2. Utilize Viterbi algorithm to adjust the mappings between observations and states, then calculate the new mean vector \( \mu_k \) and new covariance vector \( \Sigma_k \) to obtain new value of \( B \).

Step 3. Replace \( \mu_k \) with \( \mu_k' \), \( \Sigma_k \) with \( \Sigma_k' \), and \( B \) with \( B' \).

Step 4. When the range-abilities of all \( a_{ij} \) are smaller than a threshold, it stops iteration; otherwise return to Step 2.

From the above procedure, we get a trained HMM model \( \lambda \). Generally we adopt a left-right Hidden Markov Model, which is also called Bakis model. Its principal ideal is that a state will transit only to the next state or stay in the original state as time increases.

2.2 Feature extraction

When using an HMM to train or recognize Chinese character images, we need to extract features to form feature vectors. These feature vectors can then be utilized to obtain the observation probability \( B \). In speech recognition, we divide speech signal into a sequence of frames and obtain a feature vector for each frame. In our work, we have to construct a sequence of frames for each character image.

A binary character image is preprocessed with noise removal, and then is normalized to a fixed \( W \times N \) image size. We obtain a sequence of frames from the normalized image. Each frame is a narrow strip of one pixel width by projecting the image along both horizontal and vertical directions. We scan the character image from left to right and top to down. In each horizontal scan, we obtain a narrow strip which we call it a frame. Then we scan the image form top to down, and obtain other narrow strips as frames, too. The result is a combined frames. Features are computed for each frame as follows:

(1) Feature 1: The number of transition of pixel value from 0 to 1 and 1 to 0.

(2) Feature 2: The sum of transition positions.

(3) Feature 3: The density of pixels with value 0 with respect to the whole character image.

Therefore, we get three features for each frame. These features are not effected by the stroke width of character. They
are robust features in business card recognition. A character with different font types has the similar features. Each training image is binarized, noise-removed, and then normalized to a fixed 56×56 image size. The projection of the normalized image in both horizontal and vertical directions results in 112 frames. Therefore, we got 336 features since each frame generates three features.

2.3 Training

The training of the HMM model means the maximization process of the likelihood of the training data. Each Chinese character is a class. Each class has a trained HMM model whose model parameters are estimated from various image instances of the character class. In this paper, each character class is given five image instances, which are obtained from five font types of the character class. There are Song font, HEI font, Kai font, Ming font, and Li-Su font.

For each character class, we obtain 112 frames from each font type. We equally assign frames of each font type into states. Each state thus has 70 frames, which results from that 112 frames divide 8 states and multiply 5 font types. Then \( r_{i\alpha} \) and \( \delta_{i\alpha} \) are calculated, and segmental K-means and Viterbi algorithms are executed to change the assignments of frames, until \( r_{i\alpha} \) is changed within a threshold. The final symbol probability distribution \( B \) is obtained as the training result.

2.4 Recognition

In recognition stage, we need neither initial model nor iteration. HMMs will find out the maximum likelihood by Viterbi algorithm.

If there are \( T \) frames for the testing character image, we assign the first frame \( \alpha_{1} \) to state \( S_{1} \), then adopt the Viterbi algorithm to obtain the probability \( b_{1}(\alpha_{1}) \). \( \delta_{i}(1) \) is equal to \( b_{1}(\alpha_{1}) \). In the best path of the Viterbi algorithm, the second frame is assigned to the first state, then this position records the probability that the second frame in the first state \( b_{1}(\alpha_{2}) \). \( \delta_{i}(1) \) multiplies \( b_{1}(\alpha_{2}) \). If we assign the last frame in the last state, then \( \delta_{i}(N) \) equals \( \delta_{i}(N) \) multiplies \( b_{1}(\alpha_{1}) \). Finally, compare all the model probability. We obtain top-10 candidates which belongs to the character classes with higher probabilities.

3. POSTPROCESSING

To improve the recognition rate is to use the clues in the top-10 candidates of the character sequence [2]. We can adopt linguistic information, such as lexicon and mutual information of characters, to draw new first candidate in the top-10 candidates of each character. The way we proposed is to build a statistical language model, then we develop a search algorithm based on the language model to achieve the task.

3.1 Bigram table

Chinese sentence is composed of words. A word includes several characters and constitutes the smallest lexical unit. Therefore, some character has strong neighbor relationship with another character, because both characters form a word.

Bigram table is the table to describe the neighbor relationship between any two characters. \( P_{B}(i,j) \) is the number of co-occurrence for the two characters \( i \) and \( j \). We use Equation 3 to calculate mutual information as the probability of two neighbor characters.

\[
CO_{occur}(C_{i}, C_{j}) = \frac{P(C_{i}, C_{j})}{P(C_{i})P(C_{j})}
\]

3.2 Top-10 candidate table

The character recognition of business card will results in top-10 candidates for each character. For a sequence of \( N \) characters, we will obtain a top-10 candidate table \( P_{i} \) with \( N=13 \). \( P_{i}(i,j) \), \( 1 \leq j \leq 10 \), denotes an element in the table.

3.3 Viterbi algorithm

To find a best path in the top-10 candidate table is to find a sequence of candidates that have the best neighbor relationship. We devise a Viterbi algorithm to obtain the optimal sequence. The key point of Viterbi algorithm is to use the Markov property to make it more efficient. The Markov property says that the most probable path for the rest of a sequence can depend only on the state in which it starts, not on anything else about the path that got there. It means we need not search all possible paths that lead to final state; what we need is to keep track of the most probable path that ends in that state. Thus, Viterbi algorithm is an instance of dynamic programming. We define the symbol as follows:

\( T \): The number of characters.
\( N \): The number of candidates.
\( C_{i} \): The character in position \( i \). It also represents the top-10 candidates of the character. \( t=1,2,\ldots,T \).
\( \delta_{i} \): The maximum probability of the searched path in state \( i \) at time \( t \).

First, we find \( P_{i}(i,j) \) from the bigram table element \( P_{k}(k,l) \), where \( k \) is the character in position \( i \) of \( P_{i} \), and \( l \) the character in position \( j \) of \( P_{i} \).

Step 1. Initialization

\( \delta_{1}(j) = \max_{k=1}^{T} P_{1}(P_{1}(i,1), P_{1}(j,2)) \), \( 1 \leq j \leq N \)

\( \psi_{1}(j) = \arg \max_{k=1}^{T} P_{1}(P_{1}(i,1), P_{1}(j,2)) \), \( 1 \leq j \leq N \)

Step 2. Recursion

\( \delta_{i}(j) = \max_{k=1}^{T} P_{i}(P_{i}(i,t), P_{i}(j,t+1)) \),

where \( 2 \leq t \leq T-1, 1 \leq j \leq N \)

\( \psi_{j}(j) = \arg \max_{k=1}^{T} P_{i}(P_{i}(i,t), P_{i}(j,t+1)) \),

where \( 2 \leq t \leq T-1, 1 \leq j \leq N \)

Step 3. Termination

\( \delta_{N}(j) = \arg \delta_{N}(j) \), \( 1 \leq j \leq N \)

Step 4. Path backtracking

\( S^{*} = \psi_{t}(S^{*}+1) \), \( t=T-2, T-3, \ldots, 1 \)

\( P' = \max_{1 \leq j \leq N} \sum_{i=1}^{T} \delta_{i}(S^{*}(j)) \)

\( \delta_{i}(j) \) is the maximum probability of the search path in state \( i \) at time \( t \). \( P' \) is the total probabilities of observations. \( \psi_{i}(j) \) records the path position when in state \( i \) at time \( t \). At time \( t=1 \), the path arrives at state \( S^{*} \) and
4. EXPERIMENTAL RESULTS

The approach is experimented in Pentium-II 300 personal computer with 300 MBytes RAM and Windows 98. The program is implemented using Matlab. Business card images are scanned with 300 dpi resolution. We cut the address and company item from Chinese business cards, and segment each character automatically and correct it factually. Laplacian operator is modified to binarize the card images. We use 8 states in our HMM character recognition because it got the best result in the experiments.

4.1 Company item in Chinese business card

The hidden Markov model is trained with 300 Chinese character models. Testing samples come from thirty pieces of Chinese business cards. These company items have different kinds of fonts types. There are 243 characters in these testing company items. All the characters are included in the models.

In our HMM recognition results, there are 96 characters in top-1 candidate, and 172 characters in top-10 candidates. There are 61 lost candidates. Therefore, our accuracy rate of top-10 candidates $R_{top\_10}$ is 70.8%.

The above accuracy rates do not include the postprocessing method. After including the postprocessing stage described in Section 3, the number of correct characters in top-1 candidate is increased to 157. The improvement ratio of the post processing stage is (157-96)/96 = 63.8%. The overall recognition rate $R_{top\_1}$ of our approach is 157/243 = 64.6%.

One commercial OCR software is applied in the recognition of company item of Chinese business cards. In certain testing examples, the accuracy rate $R_{top\_1}$ of the commercial software is 66.67%. The accuracy rate $R_{top\_1}$ of our approach is 55.56%. Our approach has higher recognition rate. For the characters with great variety fonts, our method provides better performance.

Low recognition rate is still a bottleneck of business card recognition. It comes from the small-size problem in business card. Characters in business card are small, which increase the noise effect and decrease the efficiency of feature extraction. Font diversity is another problem. Artistic characters appear usually in company item. It makes the training of HMM with 5 font styles be not enough. Feature extraction needs also to be discussed for recognition rate. Features extracted from a sequence of frames may be similar among different characters.

4.2 Address item in Chinese business card

To build bigram table of address, 12656 addresses are collected from the yellow page of Taipei City. There are 194905 characters in the database. After the creation of bigram table, it is found that there are only 660 characters occurred in the address database. It is 5.6 % with respect to 13051 Chinese characters.

To test our business card recognition system, 100 address items are obtained from 100 pieces of Chinese business card. These addresses locate in Taipei. There are 1484 testing characters. After OCR processing, the number of top-1 candidate is 1164. The number of top-10 candidates is 1385. 99 characters are lost. The accuracy rate $R_{top\_10}$ is 93.33%. By using our bigram and Viterbi algorithm, the number of top-1 candidate is 1388. The overall recognition rate $R_{top\_1}$ of our approach is 93.53%.

5. CONCLUSIONS

Optical Chinese character recognition for business card is an extremely challenging work. We propose a hidden Markov model to recognize omni-font Chinese characters. A postprocessing method is combined with the hidden Markov model to improve the recognition rate.

From our experiments, the recognition rate after the hidden Markov model is better than commercial softwares. The postprocessing can further improve the recognition result. However, the improvement performs better for address item in business card, because address item has more strong relationship between neighbor characters.

6. REFERENCES