8th International Workshop on Frontiers in Handwriting Recognition (IWFHR-8), pp.78-83 (2002-08)

Context-dependent Substroke Model for HMM-based On-line Handwriting Recognition

Junko TOKUNO, Nobuhito INAMI, Shigeki MATSUDA,
Mitsuru NAKAI, Hiroshi SHIMODAIRA and Shigeki SAGAYAMA*

Graduate School of Information Science,
Japan Advanced Institute of Science and Technology
{j-tokuno,n-inami,matsuda,mit,sim}@jaist.ac.jp

*Graduate School of Information Science and Technology,
The University of Tokyo
sagayama@hil.t.u-tokyo.ac.jp

Abstract

This paper describes context-dependent substroke hidden Markov models (HMMs) for on-line handwritten recognition of cursive Kanji and Hiragana characters. As there are more than 6,000 distinctive characters including Kanji and Hiragana in Japanese, modeling each character by an HMM leads to an infeasible character-recognition system requiring huge amount of memory and enormous computation time. In order to tackle this problem, we have proposed the substroke HMM approach where a modeling unit "substroke" that is much smaller than a whole character is employed and each character is modeled as a concatenation of only 25 kinds of substroke HMMs. One of the drawback of this approach is that the recognition accuracy deteriorates in case of scribbled characters, and characters where the shape of the substrokes varies a lot. In this paper, we show that the context-dependent substroke modeling which depends on how the substroke connects to the adjacent substrokes is effective to achieve robust recognition of low quality characters. The Successive State Splitting (SSS) algorithm which was mainly developed for speech recognition is employed to construct the context dependent substroke HMMs. Experimental results show that the correct recognition rate improved from 88% to 92% for cursive Kanji handwritings and from 90% to 98% for Hiragana handwritings.

1. Introduction

The hidden Markov model (HMM) is the most popular technique for speech recognition and it has been successfully applied to European and American on-line handwriting recognition [1, 5, 6, 8]. In their study, the "whole character HMM" has been widely employed where each letter of the alphabet is modeled by one HMM and hence the number of HMMs is identical to the number of distinct letters that are to be recognized. Now in case of Japanese character recognition, where the task is to recognize more than 6,000 Kanji (Chinese-original) characters, apart from Hiragana, Katakana, Numeric and other character sets, the whole character HMM based approach [2, 9] is not practical as it requires huge number of training samples and enormous amount of memory which is proportional to the number of characters.

To tackle this problem, we have proposed "substroke-based HMM" approach [7] in which the size of the modeling unit and the number of models are much smaller than those of the whole character HMM. Actually, we have shown that any Kanji character can be represented as a concatenation of only 25 kinds of substroke HMMs. The substroke-based approach has advantages that the total size of the models is very small and the recognition speed is fast. Not only that, the proposed approach gives almost the same recognition accuracy, if compared to the whole character HMM based approaches where the Kanji characters are carefully written. However, the recognition accuracy for cursive handwritings and Hiragana handwritings are lower than the whole character HMMs. This is because, if compared to the carefully written Kanji characters which mainly consist short line segments, cursive handwritings and Katakana characters mainly consist of curved substrokes which are difficult to be modeled with the help of the proposed 25 substroke models containing short line segments.

Therefore, in order to achieve robust recognition of various types of characters, it is necessary to increase the number of substroke models. The shapes of handwritings are distorted by a number of factors, such as writing styles and writer's individualities. Furthermore, the shape of each substroke is dependent on both the character that the substroke belongs to and the adjacent substrokes. We call these factors "contexts". In the present study, we assume that the effect of the adjacent substrokes is the most dominating factor, and create "context-dependent substroke HMMs" based on it. Since these models are the same as the triphone models of speech recognition, several algorithms for constructing efficient context-dependent models avoiding the computational explosion problem have been adopted in handwriting recognition. By using the frequency of trigraph and the clustering

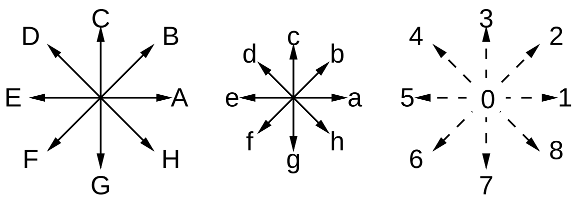


Figure 1. Substroke categories: A–H (a–h) are long (short) substrokes with pen down and 0–8 are the direction of pen up.

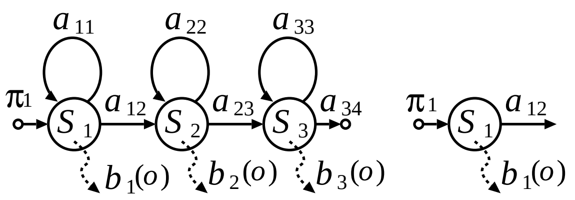


Figure 2. Substroke HMMs : (Left) pen down model, (Right) pen up model.

HMM states in order to reduce the number of parameters, the cursive handwriting recognition system with large vocabularies have improved [3, 4]. However, this method does not solve the optimization problem. On the other hand, the Successive State Splitting (SSS) algorithm [10] is employed in the present study, and it can find a pseudo-optimal HMM topology and the optimal parameters simultaneously in the sense of maximum likelihood criterion. By this algorithm, we control the total size of the models automatically and train those models more effectively.

2. Handwriting Recognition Algorithm Based on Substroke HMM

2.1. Input Features

The proposed system basically consists of a feature extractor, substroke models (HMMs), dictionaries and a decoder. During feature extraction, pen positions (x, y) in Cartesian coordinate system sampled from the pen tablet is used. Let (dx, dy) be the difference between the two consecutive pen-position samples and (r, θ) be its corresponding feature vector in Polar coordinate system, where r means the Euclidean distance between the two pen positions ($\sqrt{dx^2 + dy^2}$) and θ represents the direction of the feature vector. When the pen touches the tablet surface (pen-down), the feature vector (r, θ) represents the velocity vector of the pen positions sampled at every certain interval. When the pen leaves the tablet surface (pen-up), (r, θ) is obtained just after the pen touches the tablet again and the vector represents a displacement vector between the strokes observed just before and after the pen-up because the pen position is not sampled while it is in the air.

2.2. Substroke HMMs

Based on the knowledge of distinctive features of Kanji characters, we have defined 25 substrokes in all according to the direction and the length as shown in Fig. 1. Actually there are eight long strokes (A–H), eight short strokes (a–h), eight pen-up movement (1–8) and one pen-up-down movement (0). Each substroke is modeled by a left-to-right HMM as is depicted in Fig. 2. Three-state continuous-distribution HMM is employed for each pen-down substroke to model the changes of substroke velocity, while one-state HMM without self-loop probability is used for each pen-up substroke to model the displacement vector. Here, let $\lambda^{(k)} = (A^{(k)}, B^{(k)}, \pi^{(k)})$ be the set of HMM parameters of substroke k, where

$A^{(k)} = \{a_{ij}^{(k)}\}$	: the state-transition probability distributions from state S_i to S_j,
$B^{(k)} = \{b_j^{(k)}(o)\}$	: the probability distributions of observation symbols o at state S_j,
$\pi^{(k)} = \{\pi_i^{(k)}\}$	: the initial state probability distributions.

The observation probability distribution is represented by an M-mixture of Gaussian distributions:

$$b_i^{(k)}(\boldsymbol{o}) = \sum_{m=1}^{M} c_{im}^{(k)} \frac{\exp\left\{-\frac{1}{2}\left(\boldsymbol{o} - \boldsymbol{\mu}_{im}^{(k)}\right)^t \Sigma_{im}^{(k)^{-1}} \left(\boldsymbol{o} - \boldsymbol{\mu}_{im}^{(k)}\right)\right\}}{\sqrt{(2\pi)^n \left|\Sigma_{im}^{(k)}\right|}}, \quad (1)$$

where $\boldsymbol{\mu}_{im}^{(k)}$ is the mean vector, $\boldsymbol{\Sigma}_{im}^{(k)}$ is the covariance matrix, and $c_{im}^{(k)}$ is the weighting coefficient. Here, each Gaussian distribution is periodic with a 2π cycle with respect to the direction feature (θ). These model parameters can be trained by Viterbi training or Baum-Welch algorithm.

2.3. Recognition

A decoder recognizes an input pattern by referring to the character's substroke sequence which is obtained by expanding the definition in the hierarchically structured dictionary [7]. For example, the definition of the character "二" is 'a 6 A' which means that the two pen-down strokes 'a' and 'A' are connected with the pen-up model '6' in standard stroke order. Similarly, "子" is 'A f 0 G d 4 A' and "字" is 'g 5 g 3 A f 6 A f 0 G d 4 A', where "子" is a partial structure of "字" and both have a common substroke sequence. According to the description in the dictionary, the decoder concatenates the substroke HMMs to generate an HMM of each candidate character, and then calculates the probability that the input pattern is produced from the HMM. This operation is effectively done by the Viterbi search algorithm of a substroke network [7].

3. Context-dependent Substroke HMMs

3.1. Contextual Factors of Handwritings

The substrokes are highly influenced by various factors, for example, the relationship with a preceding substroke and a succeeding substroke, types of radicals or characters, input devices, writing speed and styles, writer's individualities and health condition. We call these factors "contexts". In the present study, we mainly focus on the preceding and succeeding substrokes among these contexts.

3.2. Context-dependent Models

Context-dependent model is a model trained by a set of training samples in a same context, while context-independent model is trained regardless of the context. Let 'S_c' be the current substroke, 'S_p' be the preceding substroke, and 'S_s' be the succeeding substroke, then context-dependent model of the substroke 'S_c' is labeled '$S_p/S_c/S_s$'. For example, Kanji character " 口 "is defined as a six-substroke sequence 'G 3 A G 5 A' in the dictionary. If the model is not context-dependent but just context-independent, both the 1st substroke and the 4th substroke are labeled 'G', and the 3rd substroke and the 6th substroke are labeled 'A'. On the other hand, in the context-dependent model, those substroke labels are different from each other, and Kanji " 口 " is defined as '\$/G/3 G/3/A 3/A/G A/G/5 G/5/A 5/A/\$', where '\$' represents the beginning or the ending of handwritings. If we assume that every different context is modeled separately, the total number of contextual combination is approximately 10,000 ($16 * 23^2$ for pen-down models and $9 * 16^2$ for pen-up models). This causes serious problems such as recognition-performance deterioration due to the shortage of training samples per model, and enormous amount of memory. To overcome this problem, it is necessary to reduce the number of models efficiently by sharing the similar states of context-dependent HMMs.

3.3. Substroke HM-Net

HM-Net (hidden Markov network) is a network that represents a structure of shared states of context-dependent HMMs, and each path in the network is equivalent to the corresponding context-dependent HMM. Since similar states of different HMMs are shared in the HM-Net, the total number of states can be reduced. Fig. 3 illustrates an HM-Net containing three context-dependent models; '\$/A/G', '6/A/G' and '3/A/G'. If the third state of each model has similar output probability distributions, they can share a common state. This sort of sharing of the 3rd states having the same succeeding substroke happens often for the left-to-right HMM with three states, because the 3rd state is strongly affected by the succeeding substroke rather than the preceding substroke. Also, '\$/A/G' and '6/A/G' are shared in the first state, and '\$/A/G' and '3/A/G' are shared in the second state. In this case, the number of states are reduced to 4 in all, and as a result, the context-dependent model '\$/A/G' is now represented by the shared state sequence '$s1$-$s3$-$s5$' in the HM-Net, so is '6/A/G' by '$s1$-$s4$-$s5$' and '3/A/G' by '$s2$-$s3$-$s5$'. It is infeasible to implement the above sharing scheme by a bottom-up clustering approach, because it requires all of the output probabilistic density functions of HMMs whose density functions are difficult to estimate due to the limitation of the number of training samples. Since top-down clustering approach does not suffer from this problem, the Successive State Splitting (SSS) algorithm[10], which is one of the most famous and effective top-down clustering algorithms for developing context-dependent HMMs in speech recognition, is employed in the present study.

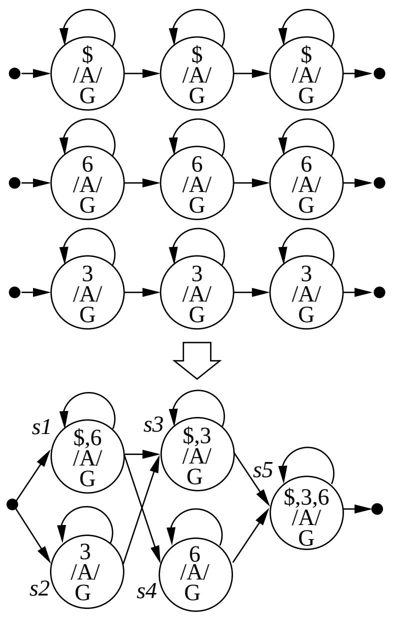


Figure 3. Context-dependent HMMs (Upper) and HM-Net (Lower).

3.4. The Successive State Splitting Algorithm

The SSS is a top-down clustering and model learning algorithm, with a mechanism of capacity control it automatically gives pseudo optimal HM-Net topology representing a state-sharing structure of context-dependent models.

When applying the SSS algorithm to our present work, context-independent substroke HMMs are used as the initial model for clustering, though, in the original SSS algorithm, only a single context-independent one-state HMM is used as the initial model.

The outline of the SSS algorithm is illustrated in Fig. 4, and its detailed procedure is as follows.

Step 1: Training of the initial models

Since we have defined 25 substrokes in section 2.2, we set up 25 substroke context-independent HMMs for the initial models for the clustering. Each state of HMMs has a 2-mixture Gaussian density function with diagonal covariance matrices, and each substroke HMM is trained with all the training samples regardless to the context.

After training, the parameter M is set to indicate the current number of states used all together. At this initial stage, M is set to 57 as we employed 3-state HMMs for 16 pen-down strokes and 1-state HMMs for 9 pen-up strokes.

Step 2: Determine a state to split

For each state $S_{(i)}$, calculate the normalized distribution size d_i of (2) and let $S_{(m)}$ be the state to split which gives the maximum d_i among the all.

$$d_i = n_i \times \sum_{k=1}^{K} \frac{\sigma_{ik}^2}{\sigma_{Tk}^2} \tag{2}$$

where

$$\sigma_{ik}^2 = \lambda_{i1}\sigma_{i1k}^2 + \lambda_{i2}\sigma_{i2k}^2 + \lambda_{i1}\lambda_{i2}(\mu_{i1k} - \mu_{i2k})^2,$$

K denotes the dimension of the feature vector, λ_{i1}, λ_{i2} represent weight coefficients of state i; $\sigma_{i1k}^2, \sigma_{i2k}^2$ denote the variances of k-th feature element at the state which has two output density functions; n_i denotes the number of training samples assigned to the state; and σ_{Tk}^2 denotes the k-th variance of all samples.

Step 3: Split of the state

The $S_{(m)}$ is split into two states, $S'_{(m)}$ and $S_{(M)}$, each of which has a single Gaussian density function corresponding to one of the respective two Gaussian densities on the split domain. Although the original SSS algorithm takes both contextual and temporal split into account, we consider only the split on the contextual domain in this paper. The training data passing through $S_{(m)}$ are divided into two paths through $S'_{(m)}$ and $S_{(M)}$.

Step 4: Re-estimate the distribution

$S'_{(m)}$ and $S_{(M)}$ are retrained to reconstruct the 2-mixture Gaussian densities for each new state. Other states affected by the split operation are also re-trained to optimize the model parameters. After this, $S'_{(m)}$ is renamed as $S_{(m)}$ and M is increased by 1. Step2 through Step4 are repeated until M reaches the specified number of states.

4. Experimental Evaluation

Handwriting database used in this evaluation is the JAIST IIPL (Japan Advanced Institute of Science and Technology, Intelligence Information Processing Laboratory) database consisting of several kinds of data sets. Among them, for the present study, we used the cursive handwriting dataset (ϵ_1 set), of which some examples are depicted in Fig. 5. The dataset contains 93,704 samples that were collected from 68 writers and covers 1,016 Japanese characters of old and new educational Kanji, 83 Hiragana, 86 Katakana and 62 Alphanumeric with free stroke order.

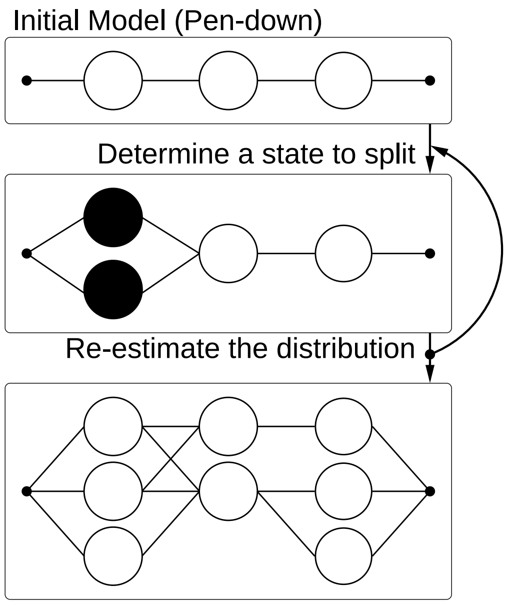


Figure 4. Successive State Splitting Algorithm for HM-Net generation.

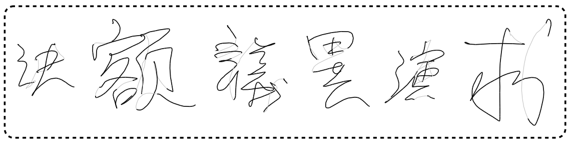


Figure 5. Samples of cursive handwriting dataset: “决”, “額”, “議”, “異”, “演”, “求”.

4.1. Experiment 1: Performance of Context-dependent Substroke HMMs

In order to evaluate the context-dependent substroke HMMs, we at first carried out a recognition experiment of 1,016 Japanese educational Kanji characters. In the experiment, each state of HMM had a single Gaussian distribution with diagonal-covariance. Thirty seven writers of ϵ_1 set were used for constructing the HM-Net and the remaining 31 writers were used for evaluation.

The recognition performance as a function of the total number of states in the HM-Net is shown in Fig. 6, where the recognition performance given at 57 states indicates the baseline performance when the context-independent models were used. The dotted-line indicates the closed recognition performance in which evaluation data set is identical to the training set, whereas the straight-line indicates the open recognition performance in which testing data set and training set are different. We can see from the result that the recognition accuracy increases as the number of states increases. Especially, the increasing rate is relatively higher when the number of states is smaller than 107. This is because that the SSS algorithm splits the sate that gives the

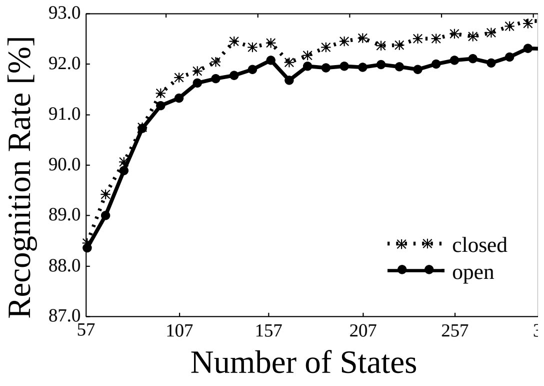


Figure 6. Recognition rate of writer-independent HM-Net with standard stroke order.

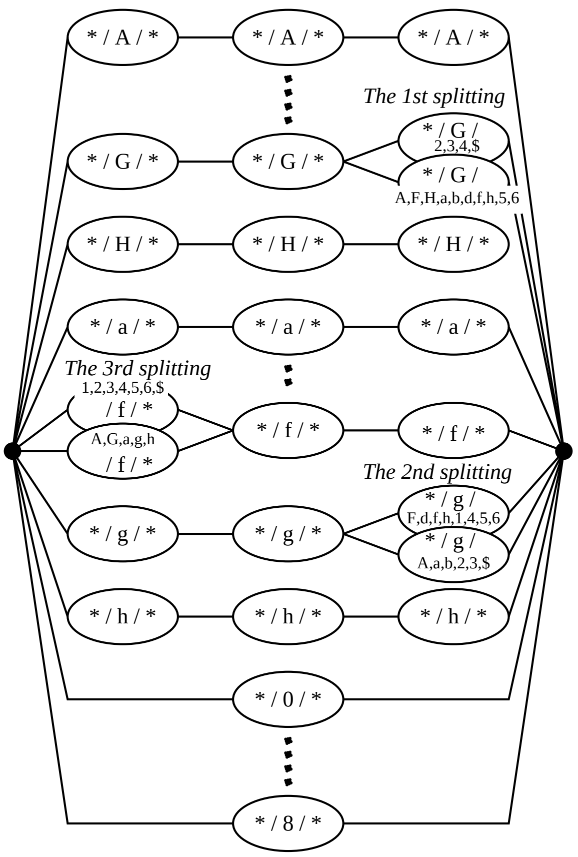


Figure 7. First three splitting by SSS algorithm for writer-independent HM-Net.

highest "gain" when the state is divided, and hence the state splitting at the early clustering stage would contribute to improve the recognition performance.

Fig. 7 shows an example of how the state splitting proceeds. We can see that the first splitting took place at the 3rd state of the model '$*/G/*$', where '$*$' denotes any contexts and thus '$*/G/*$' represents a context-independent model for substroke 'G'. After the splitting, the 3rd state has been split into two states according to the succeeding context groups, the 1st group contains 'A', 'F', 'H', 'a', 'b', 'd', 'f', 'h', '5', '6', and the 2nd group contains '2', '3', '4', '\$'. Since the pen movement directions of the substroke 'G' and the following pen-up ('2', '3', '4') are almost opposite, it is thought that these pen-ups affect the shape of 'G' differently from other contexts. In the 2nd splitting, the last state of 'g' is split by the succeeding contexts. These results match the fact that 'G' and 'g' are the most frequently appearing substrokes in Kanji.

4.2. Experiment 2: Context-dependent HMMs vs. Mixture Gaussian Density HMMs

Generally speaking, the context-dependent model is effective to model temporal variations, while the mixture Gaussian-density model in Eq. (1) is effective to model spatial variations. It would be useful to investigate which factor, temporal or spatial, really dominates the variations of the handwriting characters. To that end, we carried out an experimental comparison of the context-dependent HMMs (HM-Net) and the context-independent mixture Gaussian density HMMs. The experimental conditions were the same with that of the writer-independent HM-Net of Experiment 1.

Fig. 8 shows correct recognition rate as a function of the total number of Gaussian distributions, in which corresponding number of states and number of mixtures are shown as well. As a result, the HM-Net achieved better recognition performance with smaller number of Gaussian distributions than the context-independent mixture Gaussian density HMMs. It can be concluded that, in the cursive handwriting dataset used for this experiment, substroke context is a main factor for variations of handwritten characters.

4.3. Experiment 3: Context-dependent HMMs vs. Heuristic Macro HMMs

So far, we have employed the context-dependent models to model (i) the dependency between substrokes and (ii) curve substrokes that mainly appear in Hiragana handwritings. In case of (ii), there would be another approach in which curve substrokes are modeled not only by the straight-line models but curve stroke models or larger modeling units ("macro") explicitly. Table 1 shows additional curve substrokes and macro models, both of which are heuristically defined.

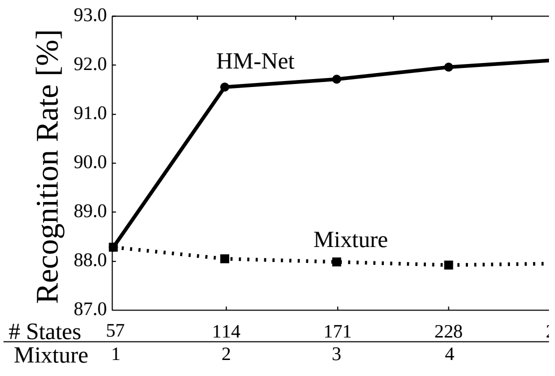
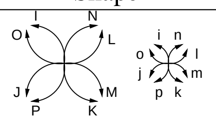
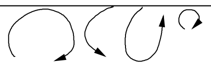


Figure 8. Comparison of recognition rate of HM-Net and mixture Gaussian density HMM

Table 1. Additional curve substroke models and macro models for defining Hiragana characters.

Model	Shape	Models	States
Curve Stroke		16	48
Macro		15	171
Character	あ, て, ぴ	72	1,487

It should be noted that the context-dependent model generated by the SSS algorithm is expected to have curve substroke like models automatically. For example, the curve substroke 'K' in Table 1 would be equivalent to the context-dependent model 'G/H/A'.

In order to compare the two approaches, automatically-created context-dependent models and manually-created substroke and macro models, the following recognition experiments on 72 Hiragana recognition task were carried out; 1) context-dependent substroke HM-Net (the number of states is variable), 2) combination of primarily substroke HMMs and curve substroke HMMs (105 states), 3) combination of primarily substroke HMMs and macro HMMs (228 states), 4) whole character HMMs (1,487 states). The dataset used in this experiments were Hiragana samples of 60 writers written in standard stroke order. Data from 50 writers were used for training and the remaining 10 writers were used for evaluation.

Fig. 9 depicts the comparison of the recognition performance based on the above four conditions. we can see from the figure that the automatically generated HM-Net performs the best among them and its recognition performance is almost comparable to the character-based HMM.

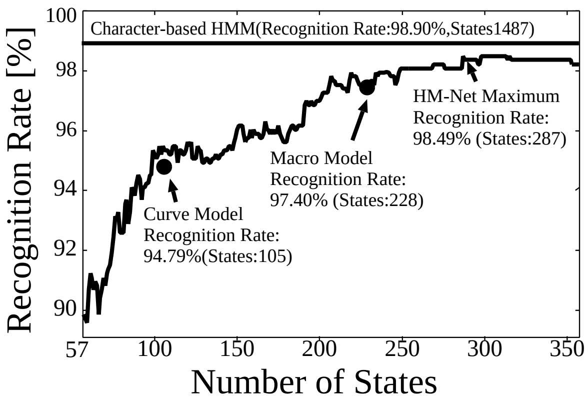


Figure 9. Recognition Rate of Hiragana characters.

5. Conclusion

We have developed an on-line handwriting recognition method based on context-dependent substroke HMMs. Through the experiments, it has been shown that the context-dependent substroke HMMs are effective for recognition of cursive Kanji and Hiragana handwritings. Moreover, we have shown that our proposed model is superior to conventional context-independent mixture Gaussian density HMMs.

References

[1] J. Hu, M. K. Brown, and W. Turin. "HMM Based On-Line Handwriting Recognition". *IEEE Trans. Pattern Analysis and Machine Intelligence*, 18(10):1039–1045, Oct. 1996.

[2] H. Itoh and M. Nakagawa. "An On-line Handwritten Character Recognition Method based on Hidden Markov Model" (in Japanese). *Technical report of IEICE*, PRMU97-85:95–100, July 1997.

[3] A. Kosmala and G. Rigoll. "Tree-Based State Clustering Using Self-Organizing Principles for Large Vocabulary On-Line Handwriting Recognition". *Proc. ICPR'98*, pages 1313–1315, Aug. 1998.

[4] A. Kosmala, J. Rottland, and G. Rigoll. "Improved On-Line Handwriting Recognition Using Context Dependent Hidden Markov Models". *Proc. ICDAR'97*, 2:641–644, Aug. 1997.

[5] A. Kundu and P. Bahl. "Recognition of Handwritten Script: A Hidden Markov Model Based Approach". *Proc. ICASSP '88*, 2:928–931, Apr. 1988.

[6] R. Nag, K. H. Wong, and F. Fallside. "Script Recognition Using Hidden Markov Models". *Proc. ICASSP '86*, 3:2071–2074, Apr. 1986.

[7] M. Nakai, N. Akira, H. Shimodaira, and S. Sagayama. "Substroke Approach to HMM-based On-line Kanji Handwriting Recognition". *Proc. ICDAR'01*, pages 491–495, Sept. 2001.

[8] T. Starner, J. Makhoul, R. Schwartz, and G. Chou. "On-Line Cursive Handwriting Recognition Using Speech Recognition Methods". *Proc. ICASSP'94*, 5:125–128, Apr. 1994.

[9] K. Takahashi, H. Yasuda, and T. Matsumoto. "On-line Handwritten Character Recognition Using Hidden Markov Model" (in Japanese). *Technical report of IEICE*, PRMU96-211:143–150, Mar. 1997.

[10] J. Takami and S. Sagayama. "A Successive State Splitting Algorithm for Efficient Allophone Modeling". *Proc. ICASSP'92*, 1:573–576, Mar. 1998.