# The Reduction of Memory and the Improvement of Recognition Rate for HMM On-line Handwriting Recognition

A. Funada, D. Muramatsu and T. Matsumoto
Department of Electrical Engineering, and Bioscience
Waseda University, Tokyo 169-8555, Japan
E-mail : {fu-atsushi, daigo}@ruri.waseda.jp, takashi@mse.waseda.ac.jp

## Abstract

*The purpose of this project is two fold. The first purpose is to reduce the memory size of our previous handwriting recognition algorithm based on an HMM using Self-Organizing Map (SOM) density tying. The second is to improve recognition capability by incorporating additional information. SOM density tying reduced the dictionary size to 1/7 of the original size, with a recognition rate of* 90.45%*, only slightly less than the original recognition rate of* 91.51%*. Our additional feature increased recognition capability to* 91.34%*.*

## 1 Introduction

Recent developments in pen input devices including Personal Data Assistants (PDAs) require good on-line handwriting character recognition algorithms. The smaller the machines are, the more difficult the keyboards to use. Also, those who are not familiar with keyboards prefer pen input interface. However, small hand-held devices also have limited memory consumption.

The author's group proposed a new algorithm for on-line handwriting recognition, utilizing a discrete HMM [1]-[2]. The algorithm is fast, one-shot, and without Baum-Welch, with reasonable recognition capabilities. Two issues still require improvement. First is its memory size requirement when applied to Japanese character recognition, because of the structure of Japanese characters. The Japanese characters consist of 66 Hiragana characters together with many voiced and voiceless bilabial consonants, 66 Katakana characters together with many voiced and voiceless bilabial consonants, and several thousands Kanji characters, symbols and letters. The HMM proposed in [1]-[2] consists of several thousand models, thus giving rise to a large memory size, 7 Mbyte. A serious problem arises when one wants to install the algorithm on a small hand-held device. This paper presents one of the possible solutions, among others, with the use of a density tying scheme by Self-Organizing Map. The resulting memory reduction is 1/7, with little sacrifice (1.06%) of recognition capabilities. An early attempt of this work had been reported in [4].

This paper also discusses improved recognition capability by an additional feature so to compensate for the slightly degraded recognition rate. The new proposed feature consists of the coordinates of the stroke starting point and the stroke endpoint. The new feature improves recognition accuracy by approximately 1%.

## 2 Related Works

This section briefly reviews some of the works related to the present paper. Given a typical hardware available for on-line character recognition, there are not much varieties of features: pen position and pen up/down information, although some hardware provides pen inclination (pen tilt). Here we try our best to review those related works, however, it is far from exhaustive.

- [10] uses the pen direction features with 8 symbols instead of 16 and uses the density tying .
- In [6], the baseline feature is a 6 dimensional vector including the writing angle, as well as the pen up/down bit. Additional features including hight, space features are proposed and the feature vector is coded into an M symbols via VQ (Vector Quantization) with M being 256. Baum-Welch is used to train HMM.
- [7] proposes an HMM structure for recognizing mathematical symbols where feature vector is sim-

Proceedings of the 9th Int'l Workshop on Frontiers in Handwriting Recognition (IWFHR-9 2004)
0-7695-2187-8/04 $20.00 © 2004 IEEE

ilar to the one used in this paper. They use the Baum-Welch scheme.

## 3 Preprocessing

Raw data taken from digitizer is at first normalized by the length and we obtain

$$\big(x_1(t_i), x_2(t_i), p(t_i)\big) \in \mathbb{R}^2 \times \{0,1\}, \quad i = 1, ..., M \quad (3.1)$$

where $\big(x_1(t_i), x_2(t_i)\big)$ is the pen position and $p(t_i)$ is pen up/down information.

In order to formulate the problem in terms of a discrete HMM, we quantize the data. Consider

$$\begin{aligned} V_1(t) &:= v_{1k} \in \{1,2\}, \quad k = 1,2 \\ V_2(t) &:= v_{2l} \in \{1,2,...,L\}, \quad l = 1,2,...,L \end{aligned} \quad (3.2)$$

where $V_1(t)$ represents the pen down(= 1)/up(= 2) information that is already quantized, and $V_2(t)$ represents the angle information that is quantized into $L$ symbols.

Simplified notation $t$ is used instead of $t_i$. Therefore, at each resampled time $t$, there are two symbols that $V_1(t)$ can take, whereas there are $L$ symbols that $V_2(t)$ can take.

Fig. 3.1 depicts the case with $L = 16$, which is used in our experiments. The length information of the trajectory is naturally preserved by repetition of the symbol sequence $(v_{1k}, v_{2l})$. Let $l_0 > 0$ be an (empirical) "unit" length of a trajectory $\boldsymbol{x}(t)$ on $\mathbb{R}^2$. If a particular vector $\boldsymbol{x}(t_i + 1) - \boldsymbol{x}(t_i)$ has length $l$ with an associated symbol $(v_{1k}, v_{2l})$, then this particular symbol sequence is repeated every defined unit length $l_0$.

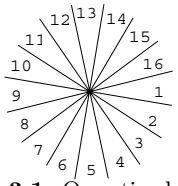


**Fig. 3.1:** Quantized direction

**Fig. 3.2:** Japanese character

With $L = 16$ and with appropriate $l_0$, the trajectory given in Fig. 3.2 is quantized as

$$\begin{aligned} \{V_1(t), V_2(t)\} = \{&(1,4),(1,4),(2,8),(2,8),(2,8), \\ &(1,16),...,(1,16),(2,8),(2,8),(2,8), \\ &(1,16),(1,16),(1,16),(2,8),(2,8),...\} \end{aligned}$$

where we are slightly abusing our notation for time. Namely, we specify an appropriate unit time length $t_i := kt$ for some inter $k$, so that $t$ should have been $t_i$. We maintain this notation since the new notation might cause confusion.

## 4 Discrete HMM for On-line Handwritten Character Recognition

### 4.1 HMM Structure for On-line Handwritten Character Recognition

In order to make an HMM precise, let us first recall the output symbols as (3.2) defined in the previous section, and let

$$\boldsymbol{O}(t) := \Big(V_1(t), V_2(t)\Big), \quad t = 1, 2, ..., T \quad (4.1)$$

be the observed output sequence

An HMM of a character

$$\boldsymbol{\mathcal{H}} = \boldsymbol{\mathcal{H}}\Big(\{a_{ij}\}, \{b^1_{ik}\}, \{b^2_{il}\}, \boldsymbol{\pi}, N\Big) \quad (4.2)$$

is defined by the joint distribution of $\{Q(t), \boldsymbol{O}(t)\}_{t=1}^T$ given $\boldsymbol{\mathcal{H}}$;

$$\begin{aligned} &P\big(\{Q(t), \boldsymbol{O}(t)\}_{t=1}^T \mid \boldsymbol{\mathcal{H}}\big) = \\ &\pi_{Q(1)} \prod_{t=1}^{T-1} a_{Q(t+1)Q(t)} \prod_{t=1}^{T} b^1_{Q(t)V_1(t)} b^2_{Q(t)V_2(t)} \end{aligned} \quad (4.3)$$

where $\{Q(t)\}$ stands for a hidden state at each sequence $t$, $\{a_{ij}\}$ state transition probability, $\{b^1_{ik}\}$,$\{b^2_{il}\}$ output emission probabilities, $\{\pi_i\}$ initial state probability, and $N$ number of states.

In order to tune HMM to our current problem, we use the left-to-right model.

### 4.2 Recognition

Learning and recognition are closely related in HMM. In our recognition algorithm, the proposed recognition scheme decides that

$$\underset{\boldsymbol{\mathcal{H}}}{\operatorname{argmax}}\, P\Big(\{\boldsymbol{O}(t)\}_{t=1}^T, Q(T) = q_N \mid \boldsymbol{\mathcal{H}}\Big) \quad (4.4)$$

is the most probable character.

### 4.3 Learning

#### 4.3.1 Generation of the First Model

Given the first data set

$$\boldsymbol{O}_1(t) = \Big(V_1(t), V_2(t)\Big), \quad t = 1, ..., T_1 \quad (4.5)$$

make a division between $\boldsymbol{O}_1(t)$ and $\boldsymbol{O}_1(t+1)$ if

(i) $V_1(t) \neq V_1(t+1)$, i.e., if the pen up/down information changes; or

(ii) $|V_2(t) - V_2(t+1)| > \theta_0$, i.e., if the angle variation exceeds threshold, where $\theta_0 > 0$ is an empirical value;

and then associate the states with $\boldsymbol{O}_1(t)$ and $\boldsymbol{O}_1(t+1)$, and determine $N$.

After state clarification, we assign $\{a_{ij}\}$,$\{b^1_{ik}\}$,$\{b^2_{il}\}$, $\{\pi_i\}$ as in Section 4.3.4 with $C = 1$, and generate the first model.

### 4.3.2 Model Generation

So far, our algorithm has presumed that the number of $\boldsymbol{\mathcal{H}}$ coincides with the number of different characters. Modification is necessary because a character may be written in different stroke orders, and because a character may have significantly different shape variations.

Thus, an automatic model generation procedure is necessary. In the following step, we propose a model generation criterion, normalized log likelihood ratio. (See [1]-[**?**])

Let $\boldsymbol{\mathcal{H}}_1\Big(\{a_{ij}\},\{b^1_{ik}\},\{b^2_{il}\},\{\pi_i\},N\Big)$ be the HMM obtained by the previous steps. Let $\{\boldsymbol{O}_c(t)\}_{t=1}^{T_c}$ be another training set for the same character.

$$r_c = \frac{\frac{-logP\left(\{\boldsymbol{O}_1(t)\}_{t=1}^{T_1}, Q(T_1)=q_N | \boldsymbol{\mathcal{H}}_1\right)}{T_1}}{\frac{-logP\left(\{\boldsymbol{O}_c(t)\}_{t=1}^{T_c}, Q(T_c)=q_N | \boldsymbol{\mathcal{H}}_1\right)}{T_c}} \tag{4.6}$$

- $r_c < r_{th}$
  We create a new model $\boldsymbol{\mathcal{H}}_2$ based on $\{\boldsymbol{O}_c(t)\}_{t=1}^{T_c}$, using the previous steps.

- $r_c \geq r_{th}$
  $\{\boldsymbol{O}_c(t)\}_{t=1}^{T_c}$ is included in the same model as $\boldsymbol{\mathcal{H}}_1$, so we go to Sections 4.3.3 and 4.3.4.

This model generation algorithm has a significant consequence in that it prevents combinatorial explosion of the number of stored HMM's for different stroke orders.

### 4.3.3 Most Probable State Transition

Let $\{\boldsymbol{O}_c(t)\}_{t=1}^{T_c}, c = 2, ..., C$ be the rest of the data sets for training. This data also assigns the state. Suppose the first data is as shown in Fig.3.2 and the new data is as in Fig.4.1. Because of the stroke connection in Fig.4.1, which is not present in Fig.3.2, the two data sets' results are different numbers of states, giving rise to difficulty in learning. Our next step in the learning scheme is to use the A posteriori decoding algorithm to make an appropriate correspondence between models with different numbers of states.

**Fig. 4.1:** The same character as in Fig.3.2 with different strokes

For $\{\boldsymbol{O}_c(t)\}_{t=1}^{T_c}, c = 2, ..., C$ let

1. Set $t = 1$, $state = 1$ and $Q_c(1) = q_1$.
2. Set $t \rightarrow t+1$ and determine $Q_c(t)$.

$$\begin{aligned} Q_c(t) &= \underset{i=state,state+1}{\operatorname{argmin}} P\big(Q_c(t) = q_i | \{O_c(t)\}_{t=1}^{T_c}, \mathcal{H}\big) \\ &\qquad (state \neq N) \\ Q_c(t) &= q_N, \quad (state = N) \end{aligned} \tag{4.7}$$

3. Set $state = Q_c(t)$ and repeat 2 to 3.

### 4.3.4 Determination of HMM Parameter

Let $n\big(\boldsymbol{O}_c, q_i\big)$ be the number of repetitions of the symbol $(v_{1k}, v_{2l})$ associated with $q_i$. Let $n\big(\boldsymbol{O}_c, q_i, q_i\big)$ be the number of repetitions of the symbol $(v_{1k}, v_{2l})$ while state stays at $q_i$. Let $n\big(\boldsymbol{O}_c, q_i, v_{1k}\big)$ be the number of times where $V_1(t) = v_{1k}$ occurs, and let $n\big(\boldsymbol{O}_c, q_i, v_{2l}\big)$ be the number of times where $V_2(t) = v_{2l}$ takes place, and determine.

$$a_{ij} = 0 \quad (i \neq j,\ i \neq j+1), \quad a_{NN} = 1 \tag{4.8}$$

$$a_{ii} := \frac{\sum_{c=1}^{C} n\big(\boldsymbol{O}_c, q_i, q_i\big)}{\sum_{c=1}^{C} n\big(\boldsymbol{O}_c, q_i\big)}, \quad a_{i+1,i} := 1 - a_{ii} \tag{4.9}$$

$$(i = 1, ..., N-1), \quad (4.8, 4.9)$$

$$b^1_{ik} := \frac{\sum_{c=1}^{C} n\big(\boldsymbol{O}_c, q_i, v_{1k}\big)}{\sum_{c=1}^{C} n\big(O_c, q_i\big)}, \tag{4.10}$$

$$b^2_{il} := \frac{\sum_{c=1}^{C} n\big(\boldsymbol{O}_c, q_i, v_{2l}\big)}{\sum_{c=1}^{C} n\big(\boldsymbol{O}_c, q_i\big)} \tag{4.11}$$

$$(k = 1, 2, \quad l = 1, ..., L), \quad (4.10, 4.11)$$

$$\boldsymbol{\pi} := (1, 0, ..., 0) \tag{4.12}$$

### 4.3.5 Smoothing

The simple procedure described in Section 4.3.4 often gives rise to a sparse $\{b^2_{il}\}$. Smoothing described below circumvents this problem.

Given $\{b^{2OLD}_{il}\}$ obtained in Section 4.3.4, update them by

$$b^{2NEW}_{il} = \sum_{n=1}^{L} w_{ln} b^{2OLD}_{in} \tag{4.13}$$

where $w_{ln}$ must satisfy

$$0 < w_{ln}, \quad \sum_{l=1}^{L} w_{ln} = 1 \tag{4.14}$$

In this report,

$$w_{ln} \quad := \quad \int_{g(l,n)-\pi/L}^{g(l,n)+\pi/L} f(\theta)\, d\theta \tag{4.15}$$

is used, where

$$g(l,n) = \begin{cases} \frac{\pi}{L/2}|l-n| & \big(|l-n| < L/2\big) \\ \frac{\pi}{L/2}\Big(L - |l-n|\Big) & \big(|l-n| \geq L/2\big) \end{cases} \tag{4.16}$$

$$f(\theta) = \frac{1}{Z(\alpha,\sigma)} \left( \frac{\alpha}{\sqrt{2\pi}\sigma} e^{-\frac{\theta^2}{2\sigma^2}} + \frac{1-\alpha}{2\pi} \right) \tag{4.17}$$

$$Z(\alpha,\sigma) = \int_{-\pi}^{\pi} \left( \frac{\alpha}{\sqrt{2\pi}\sigma} e^{-\frac{\theta^2}{2\sigma^2}} + \frac{1-\alpha}{2\pi} \right) d\theta \tag{4.18}$$

Function $g(l,n)$ is the angle between $v_{2l}$ and $v_{2n}$ (see Fig.3.1), whereas function $f(\theta)$ is the sum of the "Gaussian" and the uniform distribution $1/2\pi$ divided by normalization constant $Z(\alpha,\sigma)$.

## 5 Memory-Size Reduction, Self-Organizing Map Density Tying

One of the drawbacks of the HMM described above is its memory requirement when the number of character classes and their associated models become large as applied to the Japanese characters. The Self-Organizing Map (SOM) [5] Density Tying described below is a solution to this problem. This discipline consists of the mapping phase and the tying phase. In the mapping phase, the second emission probabilities $\{b_{il}^2\}$ generated by the HMM are mapped into one of the neurons in an unsupervised manner. In the tying phase, the generated $\{b_{il}^2\}$s are tied by the map organized in the mapping phase. Therefore, by a clever choice of the number of neurons, a significant reduction is possible as reported in Section 7.

### 5.1 Mapping Phase

Consider an array of neurons, a matrix $\boldsymbol{v}_{mn}$

$$\boldsymbol{v}_{mn} := \{u_{mn,1}, ..., u_{mn,L}\}, \quad m = 1, ..., R \\ n = 1, ..., C \tag{5.1}$$

1. Set $t = 0$ and initialize $\boldsymbol{v}_{mn}(0)$ randomly.
2. Set $t \rightarrow t+1$ and select input data $\boldsymbol{x}(t)$ randomly

$$\boldsymbol{x}(t) := \{b_{i_{rand},1}^2, ..., b_{i_{rand},L}^2\}_{m_{rand}} \in \mathbb{R}^L \\ m_{rand} := \forall[1, ..., M], \quad i_{rand} := \forall[1, ..., N_m] \tag{5.2}$$

$M$ : the number of HMM models
$N_m$ : the state number of the selected HMM

3. Select the winner neuron index $(I, J)$ ;

$$(I, J) = \underset{m=1,...,R,\ n=1,...,C}{\operatorname{argmin}} \parallel \boldsymbol{x}(t) - \boldsymbol{v}_{mn}(t) \parallel \tag{5.3}$$

4. Update the reference vector $\boldsymbol{v}_{mn}(t)$ by the Gaussian Neighborhood Function

$$\boldsymbol{v}_{mn}(t+1) = \boldsymbol{v}_{mn}(t) + h_{cm}(t)\big(\boldsymbol{x}(t) - \boldsymbol{v}_{mn}(t)\big) \tag{5.4}$$

where

$$h_{cm}(t) := \alpha(t) exp\Big(-\frac{\parallel r_c - r_m \parallel^2}{2\sigma^2(t)}\Big) \tag{5.5}$$

$$\parallel r_c - r_m \parallel^2 := (I-m)^2 + (J-n)^2 \tag{5.6}$$

$$\alpha_1(t) = \alpha_1(0)\frac{M_1}{M_1 + t}, \quad log\sigma_1(t) = log\sigma_1(0) - \frac{t}{S}$$

5. Repeat 2 to 4

### 5.2 Tying Phase

Let $\boldsymbol{b}_i^2 := (b_{i1}^2, ..., b_{iL}^2)$. Our density tying scheme is performed by

$$ref_i := \underset{\boldsymbol{v}_{mn}}{\operatorname{argmin}} \parallel \boldsymbol{b}_i^2 - \boldsymbol{v}_{mn} \parallel \tag{5.7}$$

where $\boldsymbol{v}_{mn}$ is the vector obtained in the mapping phase. This way, the large number of the $\boldsymbol{b}_i^2$ vectors is compressed into the $\boldsymbol{v}_{mn}$ vectors.

One new HMM will be defined by

$$\boldsymbol{\mathcal{H}}^{NEW} := \boldsymbol{\mathcal{H}}\Big(\{a_{ij}\}, \{b_{ik}^1\}, \{ref_i\}, \boldsymbol{\pi}, N\Big), \\ ref_i = 1, ..., R \times C \tag{5.8}$$

Therefore, a significant memory-size reduction can be achieved by an appropriate choice of $R \times C$. In the experiments reported in Section 7, $R = C = 33$, and the resulting memory-size reduction is approximately one-seventh of the original size.

## 6 Improvement of Recognition Capability

One possible improvement of our original algorithm is the recognition rates on those characters with relatively few strokes. Typically, such characters are Hiragana characters as well as Western letters. In some of the characters with relatively few strokes, the direction information (3.2) and its associated number of repetitions tend to be similar to each other which give rise to similar likelihood values, which, in turn, give rise incorrect recognition. In order to overcome this problem, we need some more information. The following position information improves the recognition capability.

Let $(X_s, Y_s), (X_e, Y_e)$ and $(X_g, Y_g)$ be the starting

position, the end position and the center of a character, and let

$$\boldsymbol{O}_2 := (X_s, Y_s, X_g, Y_g, X_e, Y_e) \tag{6.1}$$

Our new recognition algorithm uses (6.2) in stead of (4.4).

$$\underset{\boldsymbol{\mathcal{H}}}{\operatorname{argmax}}\big(P\big(\{\boldsymbol{O}(t)\}_{t=1}^{T}, Q(T) = q_N \mid \boldsymbol{\mathcal{H}}\big) \times P\big(\boldsymbol{O}_2 \mid \boldsymbol{\mathcal{H}}\big)\big) \tag{6.2}$$

where $P\big(\boldsymbol{O}_2 \mid \boldsymbol{\mathcal{H}}\big)$ is the likelihood over $\boldsymbol{O}_2$, which is also computed by the SOM:

1. Map all $\boldsymbol{O}_2$'s in the data sets into one of the neurons with SOM. ($R = 16, C = 16$ See 5.1) Here, we must equalize the numbers of each character used in SOM.

2. Compute $P\big(\boldsymbol{O}_2 \mid \boldsymbol{\mathcal{H}}\big)$ by (6.3).

$$P\big(\boldsymbol{O}_2 \mid \boldsymbol{\mathcal{H}}\big) \propto P\big(\boldsymbol{\mathcal{H}} \mid \boldsymbol{v}_{IJ}\big) = \frac{n(\boldsymbol{\mathcal{H}}, Neuron_{I,J})}{n(Neuron_{I,J})}$$
$$(I, J) := \underset{m=1,\ldots,R,\ n=1,\ldots,C}{\operatorname{argmin}} \parallel \boldsymbol{O}_2 - \boldsymbol{v}_{mn} \parallel \tag{6.3}$$

- $n(\boldsymbol{\mathcal{H}}, Neuron_{I,J})$ : number of times that a particular character model $\boldsymbol{\mathcal{H}}$ is mapped into $Neuron_{I,J}$.

- $n(Neuron_{I,J})$ : number of times that a character model is mapped into $Neuron_{I,J}$.

## 7 Experiment

Database Kuchibue [3] contains Kanji, Hiragana, Katakana, Western alphabets, numerals, and symbols. In our experiment, we used 2,976 Kanji (JIS First Level), 66 Hiragana, 66 Katakana, and 52 Western letters. Each dataset consists of 10,618 characters where 5,643 are Kanji, 4,372 are Hiragana, 487 are Katakana, and 116 are Western letters. Of the 120 datasets, Kuchibue 1 to 100(100 datasets) were used for learning with the following parameter (see Sections 3, 4, and 5): $L = 16$, $l_0 = 20$, $\theta_0 = 1$, $r_{th} = 0.45$, $\alpha_1(0) = 0.5$, $\sigma_1(0) = 5$, $M_1 = 25000$, $S = 1000$. The number of states $N$ is uniquely defined by (i) and (ii) of Section 4.3.2. In this experiment, $N$ was between 2 and 48.

Our algorithm described in the previous section created 3,724 models (Kanji 3,487, Hiragana 71, Katakana 76, Western letters 90). A recognition experiment was performed against Kuchibue 101 - Kuchibue 120 (20 datasets). Table 7.1 shows the numbers of the models created.

We have conducted following four experiments.

(1) The original HMM algorithm scheme described in Section 4.

(2) Experiment (1), in addition to algorithm described in Section 6.

(3) Experiment (1), in addition to memory-size reduction with the SOM density tying proposed in Section 5. ($R = 33,\ C = 33$) See 5.1)

(4) Experiment (2), in addition to memory-size reduction with the SOM density tying. ($R = 33,\ C = 33$) See 5.1)

Table 7.3 shows the average recognition rates in each experiment, and Table 7.2 shows the dictionary size and the recognition speed in each experiment. 1st[%] denotes the rate recognized correctly.

Fig.7.1 shows examples of characters for which the recognition capabilities were improved in Experiment (2), and Fig.7.2 shows examples of characters whose recognition capabilities were not improved in Experiment (2).

**Table 7.1:** Generated Models

| character category | number of classes | number of generated models |
|---|---|---|
| Kanji | 2,976 | 3,487 |
| Hiragana | 66 | 71 |
| Katakana | 66 | 76 |
| Letters | 52 | 90 |
| Overall | 3,160 | 3,724 |

**Table 7.2:** Dictionary Size and Recognition Speed

| Experiment | Size [MB] | Speed [sec/char] |
|---|---|---|
| (1) | 7.10 | 0.5821 |
| (2) | 7.41 | 0.5892 |
| (3) | 1.13 | 0.5905 |
| (4) | 1.44 | 0.6012 |

**Table 7.3:** Recognition Rates

| character category | 1st [%] | | | |
|---|---|---|---|---|
| | (1) | (2) | (3) | (4) |
| Kanji | 93.78 | 94.21 | 92.42 | 92.89 |
| Hiragana | 90.12 | 91.26 | 89.46 | 90.69 |
| Katakana | 83.07 | 85.63 | 82.27 | 84.95 |
| Letters | 68.58 | 69.05 | 66.59 | 67.54 |
| Overall | 91.51 | 92.33 | 90.45 | 91.34 |

## 8 Discussion

There are significant variations of recognition rates among the individual Kuchibue data sets (Table 7.3).

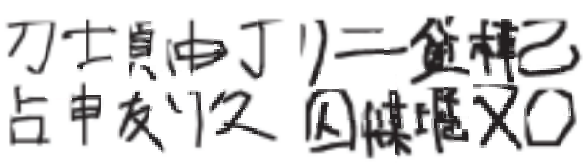

**Fig. 7.1:** Improved characters

**Fig. 7.2:** Not improved charcters

These variations appear to be attributed to the fact that this database contains a great variety of casualness in writing those characters.

From Experiment (1), the proposed HMM algorithm appears to be extremely robust against stroke number variations while maintaining reasonable robustness against stroke order variations.

Based on Experiment (2), with the proposed algorithm in Section 6, we improved the recognition rate by 1%, compared with experiment 1. (See Table 7.3)

According to Experiment (3), with SOM Density Tying, we successfully reduced the dictionary size to 1/7 of the original, keeping the first recognition rate 90.45%, 1% decline from the original. (See Tables 7.3 and 7.2.)

From Experiment (4), with the proposed algorithm in Section 6 and SOM Density Tying, we raised the recognition rate by 1%, over that in experiment 3. (See Table 7.3)

The memory reduction rate is a function of the matrix size and the number of states. One of the reviewers points out an interesting conjecture to the effect that the memory reduction rate may be approximately $(R \times C)/\sum_m^M N_m$. In Experiment (3)-(4), $R \times C = 1089$, $\sum_m^M N_m = 87715$. So, $(R \times C)/\sum_m^M N_m$ is 0.012, while the memory reduction rate was 1/7 (See Table 7.2). It would be interesting to study relationship between the two quantities.

## References


[1] K. Takahashi and T. Matsumoto : "HMM On-line Handwriting Recognizer with Sparse Data", IEEJ National Convention, March, (1996)

[2] K. Takahashi, H. Yasuda and T. Matsumoto : "A Fast HMM Algorithm for On-line Handwritten Character Recognition", ICDAR (Fourth International Conference on Document Analysis and Recognition), vol.1, pp.369-375, August, (1997)

[3] M. Nakagawa : "TUAT Nakagawa Lab. HANDS-kuchibue_d-97-06", Tokyo University of Agriculture and Technology, (1996)

[4] R. Kohle and T. Matsumoto : "Pruning Algorithm for HMM On-line Handwriting Recognition (in Japanese)", Technical Report of IEICE, PRMU97-5:33,39, May, (1997)

[5] T. Kohonen : "Self-Organizing Maps", Springer, (2001)

[6] H. Shu : "On-Line Handwriting Recognition Using Hidden Markov Models", S.B., Electrical Engineering and Computer Science Massachusetts Institute of Technology, (1996)

[7] Y. Sakamoto, M. Xie, R. Fukuda and M. Suzuki : "On-Line Recognition of Handwriting Mathematical Expression via Network", The third Asian Technology Conference in Mathematics, August, (1998)

[8] R. Plamondon and N. Shirai : "On-line and Off-line Handwriting Recognition:A Comprehensive Survey", IEEE Transactions on Pattern Analysis and Machine Intelligence, vol 22, NO.1, pp.63-84, January, (2000)

[9] P.M. Lallican, C. Viard-Gaudin and S. Knerr : "From Off-line to On-line Handwriting Recognition", The 7th International Workshop on Frontiers in Handwriting Recognition, pp.303-312, Amsterdam, September, (2000)

[10] M. Nakai, N. Akira, H. Shimodaira and S. Sagayama : "Substroke Approach to HMM-based On-line Kanji Handwriting Recognition ", The 6th International Conference on Document Analysis and Recognition, pp.491-495, Seattle, September, (2001)

[11] L. Koerich, R. Sabourin and Y. Suen : "Fast Two-level Viterbi Search Algorithm for Unconstrained Handwriting Recognition", The 2002 IEEE International Conference on Acoustics, Speech, and Signal Processing, pp.IV-3537 - IV-3540, Orlando, May, (2002)

[12] Jay J. Lee, Jah W. Kim and Jin H. Kim : "Data Driven Design of HMM Topology For On-Line Handwriting Recognition". The 7th International Workshop on Frontiers in Handwriting Recognition, pp.239-248, Amsterdam, September, (2000)