Segmentation of On-line Handwritten Japanese Characters of Arbitrary Line Direction Using SVM

Bilan Zhu and Masaki Nakagawa
Tokyo University of Agriculture and Technology, 2-24-16 Naka-cho, Koganei, Tokyo 184-8588, Japan
E-mail: {zhubilan, nakagawa}@cc.tuat.ac.jp

Abstract This paper describes a method of producing segmentation point candidates for on-line handwritten Japanese text by a support vector machine (SVM) to improve text recognition. This method extracts multi-dimensional features from on-line strokes of handwritten text and applies the SVM to the extracted features to produces segmentation point candidates. This paper also shows the details of generating segmentation point candidates in order to achieve high discrimination rate by finding the optimal combination of the segmentation threshold and the concatenation threshold. We compare the method for segmentation by the SVM with that by a neural network (NN) and show the result that the method by the SVM bring about a better segmentation rate and character recognition rate.

Key words On-line recognition, Character recognition, Segmentation, SVM, Writing constraint

1. Introduction

On-line recognition was first employed in real products in 1980s for Japanese input with hard constraints such as character writing boxes. Due to the development of pen-based systems such as tablet PCs, electronic whiteboards, PDAs, pen and paper devices like Anoto pen and so on and the expansion of writing surfaces, handwritten text recognition rather than character recognition is being sought with less constraints since larger writing surfaces allow people to write more freely.

Japanese text is written horizontally, vertically or even diagonally on a piece of paper or blackboard. Diagonal lines of text do not appear often but the system must be prepared even for arbitrary line directions if it is to be used naturally as pen interfaces.

The model and system for separating freely written text into text line and estimating the line direction and character orientation was reported in [1]. If the initial segmentation is not good, however, it determines the upper limit of text recognition performance.

On-line recognition methods of format free Japanese text recognition reported so far incorporate segmentation, although most of them assume horizontal writing from left to right. Aizawa et al. reported real-time segmentation for on-line handwritten Japanese text by applying features preceding a segmentation point candidate to a NN in [2]. Okamoto et al. showed that several physical features are effective to segment on-line handwritten Japanese text deterministically [3]. Senda et al. proposed a linear discrimination method for segmentation. They presented a learning method of the discrimination function by the steepest gradient method [4]. We previously proposed a segmentation method for on-line handwritten Japanese text by a NN [5].

The SVM method [6], [7] for pattern recognition has recently been given increasing attention. It is a technique motivated by statistical learning theory and has been developed to construct a function for nonlinear discrimination by the kernel method. SVMs have been successful applied to numerous classification tasks. The key idea of SVMs is to learn the parameters of the hyperplane to classify two classes based on maximum margin from training patterns.

In this paper, we employ an SVM to determine segmentation point candidates for on-line handwritten Japanese text of arbitrary line direction. We compare the method for segmentation by the SVM with that by a NN. We incorporate the method into the segmentation by recognition scheme. We follow the stochastic model proposed in [8] to evaluate the likelihood composed of character pattern structure, character segmentation, character recognition and context and finally determine segmentation points and recognize text.

In this paper, section 2 presents the flow of processing. Section 3 describes text segmentation and a method for generating character segmentation point candidates. Section 4 presents evaluation. Section 5 concludes this paper.

2. Flow of Processing

A stroke denotes a sequence of pen-tip coordinates from pen-down to pen-up while an off-stroke denotes a vector from the pen-up to the next pen-down. On-line handwritten Japanese text is composed of several text lines separated by a large off-stroke from a previous line to a new line. Its detection is not difficult. We don’t go into this matter in this paper. Line direction of a handwritten Japanese text line is quantized into 4 directions as shown in Fig. 1.

We process each text line as follows:

Step1: Generation of segmentation point candidates

Each off-stroke is classified into a segmentation point, a non-segmentation point and an undecided point according to the features such as distance and overlap between adjacent strokes detailed later. A segmentation point should be between two characters while a non-segmentation point is within a character pattern. An undecided point is a point where segmentation or non-segmentation judgment cannot be made. A segmentation unit bounded by two adjacent segmentation points is assumed as a character pattern. An undecided point is treated as two ways of a segmentation point or a non-segmentation point. When it is treated

![Fig.1 Line direction of text.](image-url)
This paper will describe the details of the step 1. For the step 2 and step 3, refer to the literature [5], [8].

3. Segmentation
First, we extract multi-dimensional features from off-strokes within a text line. Then, each off-stroke is classified into a segmentation point, a non-segmentation point and an undecided point by applying an SVM or a NN for the extracted features.

3.1 Selection of Off-stroke Features
First, we define the following terminology:

- \( B_b \): Bounding box of the immediately preceding stroke
- \( B_b' \): Bounding box of the immediately succeeding stroke
- \( B_b_{at} \): Bounding box of all the preceding strokes

\( B_{s_all} \): Average character size

\( D_{b_x} \): Distance between \( B_{b_{at}} \) and \( B_{b_s} \) to x-axis

\( D_{b_y} \): Distance between \( B_{b_{at}} \) and \( B_{b_s} \) to y-axis

\( S_{bac} \): Overlap area between \( B_b \) and \( B_b' \)

\( S_{acs} \): Overlap area between \( B_b_{at} \) and \( B_b_{s} \)

\( S_{acs} \): Overlap area between \( B_b_{s} \) and \( B_b' \)

\( D_{acs} \): Distance between centers of \( B_b \) and \( B_b' \)

\( Y \): Y coordinate of the top position of \( B_b_{at} \)

\( Y \): Y coordinate of the center of \( B_b \)

We examined the distributions of these features using training patterns, and deleted the features such as \( f_3 \) shown in Fig. 3(a) that two classes of segmentation points and non-segmentation points are not clearly divided while retained those such as \( f_2 \) shown in Fig. 3(b) that the two classes are divided to some extent. Moreover, some features have very similar effect. Employment of them at the
same time doesn’t affect the discrimination rate although it takes processing time. Therefore, we examined the correlation coefficient for each pair of features and selected either one from the pair that has 0.90 or more correlation coefficient. The finally selected features are shown in Table 1.

Table 1: Selected features

<table>
<thead>
<tr>
<th>Direction</th>
<th>Selected features</th>
<th>Number</th>
</tr>
</thead>
<tbody>
<tr>
<td>R</td>
<td>( f_1, f_2, f_3, f_4, f_5, f_6, f_7, f_8, f_9, f_{10}, f_{11}, f_{12}, f_{13}, f_{14}, f_{15}, f_{16}, f_{17}, f_{18}, f_{19}, f_{20}, f_{21} )</td>
<td>18</td>
</tr>
<tr>
<td>L</td>
<td>( f_1, f_2, f_3, f_4, f_5, f_6, f_7, f_8, f_9, f_{10}, f_{11}, f_{12}, f_{13}, f_{14}, f_{15}, f_{16}, f_{17}, f_{18}, f_{19}, f_{20}, f_{21} )</td>
<td>15</td>
</tr>
<tr>
<td>U</td>
<td>( f_1, f_2, f_3, f_4, f_5, f_6, f_7, f_8, f_9, f_{10}, f_{11}, f_{12}, f_{13}, f_{14}, f_{15}, f_{16}, f_{17}, f_{18}, f_{19}, f_{20}, f_{21} )</td>
<td>17</td>
</tr>
</tbody>
</table>

3.2. Neural Network

A three-layers NN can be used for distinguishing the two classes of segmentation points and non-segmentation points [9]. We constructed a NN that has an input layer composed of a feature vector \( \mathbf{v} \) from an off-stroke plus one additional input, a middle layer of \( n_{m} \) units and the single output. The output \( O \) is calculated as follows:

\[
O = \sum_{i=1}^{n_{m}} c_i \phi(\mathbf{w}_i \cdot \mathbf{v} + b_i)
\]

where \( \mathbf{w}_i \) represents all the network coefficients, \( c_i \) is the learning rate, \( \phi(\cdot) \) is the learning error, and \( \Delta \theta \) indicates the relative size of change in the network coefficients. \( \theta \) is updated at iteration \( t \) as:

\[
\theta(t + 1) = \theta(t) + \Delta \theta(t)
\]

Moreover, we use learning with momentum for speedup as follows:

\[
\theta(t + 1) = \theta(t) + (1 - \beta) \Delta \theta(t) + \beta \Delta \theta(t - 1)
\]

where \( \beta \) is set as 0.9.

For the learning rate \( \eta \), we initialize it as a large value, and update it at each iteration \( t \) as follows:

\[
\eta(t + 1) = \eta(t) + (1 - \beta) \eta(t) + \beta \eta(t - 1)
\]

where \( \beta \) is set as 0.9.

The learning speed can be remarkably improved by the above method.

For the number of units for the middle layer \( n_{m} \) we will test several numbers and select the number that makes the smallest learning error.

3.3. Support Vector Machine

The key idea of SVMs is to separate two classes with the hyperplane that has the maximum margin. Finding this hyperplane \( \alpha \mathbf{x} + b = 0 \) can be translated into the following optimization problem:

\[
\begin{align*}
\text{minimize} & : \frac{1}{2} \| \mathbf{w} \|^2 + C \sum_{i=1}^{n \delta} \xi_i \\
\text{subject to:} & : \xi_i \geq 0, y_i (\alpha \mathbf{x}_i + b) \geq 1 - \xi_i
\end{align*}
\]

where \( \frac{1}{2} \| \mathbf{w} \|^2\) is for the maximum margin, \( \xi_i \) is the learning error of a training pattern \( i \), \( C \) is the trade-off between learning error and margin, \( \mathbf{x}_i \) is the feature vector of a training pattern \( i \), \( y_i \) is the target value of a training pattern \( i \), \( l \) is the number of training patterns, respectively.

Then, the feature vectors are mapped into an alternative space by choosing kernel \( K(\mathbf{x}_i, \mathbf{x}_j) = \phi(\mathbf{x}_i) \cdot \phi(\mathbf{x}_j) \) for nonlinear discrimination. Consequently, it leads to the following quadratic optimization problem:

\[
\begin{align*}
\text{minimize} & : W(\alpha) = \sum_{i=1}^{n} a_i + \frac{1}{2} \sum_{i=1}^{n} \sum_{j=1}^{n} y_i y_j a_i a_j K(\mathbf{x}_i, \mathbf{x}_j) \\
\text{subject to:} & : \sum_{i=1}^{n} y_i a_i = 0, \forall i; 0 \leq a_i \leq C
\end{align*}
\]

where, \( \mathbf{a} \) is a vector of \( l \) variables and each component \( a_j \) corresponds to a training pattern \( (\mathbf{x}_i, y_i) \). The solution of the optimization problem is the vector \( \mathbf{a}^* \) for which \( W(\mathbf{a}^*) \) is minimized and the constraints of the eq. (7) are fulfilled. The classification of an unknown pattern \( \mathbf{z} \) is made based on the sign of the function:

\[
G(\mathbf{z}) = \sum_{i=1}^{n} a_i y_i K(\mathbf{x}_i, \mathbf{z}) + b
\]

We set the target value of segmentation points as 1 and that of non-segmentation points as -1. We obtain the separating hyperplane by solving this optimization problem shown in eq. (7) for training patterns using SVMs [10] that can efficiently handle problems with many thousand support vectors, converges fast with minimal memory requirements.
We could do so if it were only a classification of two classes for segmentation points and non-segmentation points. However, this does not allow the later processing to apply likelihood factors such as character recognition or context to better segment handwritten text.

Fig. 4 shows the distribution of the outputs of the NN trained for text lines of the direction $R$. We can set the concatenation threshold $th_\text{c}$ and the segmentation threshold $th_\text{s}$ for both the sides of $th$ and judge values smaller than $th_\text{s}$ as concatenation (non-segmentation) points, values larger than $th_\text{s}$ as segmentation points, and the others as undecided points to obtain the highest segmentation rate for the step 3 in Section 2. The widths $th - th_\text{c}$ and $th_\text{s} - th$ are not certainly equal, because the distribution of the outputs for the two classes of non-segmentation points and segmentation points are unbalanced as shown in Fig. 4. Therefore, we take the segmentation measure after applying the step 3 for all the combinations of $th_\text{c}$ and $th_\text{s}$ using the training patterns and take the combination of $th_\text{c}$ and $th_\text{s}$ producing the best segmentation measure. We consider two kinds of the segmentation measure. The one is the point classification rate $Cp$ that shows how much segmentation and non-segmentation points are correctly classified according to eq. (9). The other is the $f$ measure according to eq. (10) where $r$ is recall and $p$ is precision. The former seeks the best classification rate of segmentation and non-segmentation points while the latter takes the balance between the recall and the precision. We search for the optimal combination of $th_\text{c}$ and $th_\text{s}$ from the training patterns and apply them to the testing patterns.

\[
Cp = \frac{\text{number of correctly classified segmentation and non-segmentation points}}{\text{number of segmentation and non-segmentation points}}
\]

\[
f = \frac{2}{1+\frac{1}{p}}
\]

\[
r = \frac{\text{number of correctly classified segmentation points}}{\text{number of true segmentation points}}
\]

\[
p = \frac{\text{number of correctly classified segmentation points (including false)}}{\text{number of correctly classified segmentation points}}
\]

4. Experiments

We extracted text lines from the database of character-orientation and line-direction free handwritten on-line text HANDS-Kondate_t_bf-2001-11 collected from 100 people and with their character orientations normalized, i.e., a text line rotated so that characters have normal orientation but the text line direction is arbitrary. These text lines were classified into the 4 line directions.

Moreover, we divide the text lines for each line direction further into 4 groups of 25 persons’ patterns each. We follow the cross validation method to measure the recognition rate and select one group among the 4 groups as the testing set $i (i = 1 \text{ to } 4)$ and merge all the remaining groups (25 x 3 persons’ patterns) as the training set $i$. For each testing set $i$, we use the training set $i$ to train or obtain the parameters for SVMs or NNs as well as the concatenation threshold $th_\text{c}$ and the segmentation threshold $th_\text{s}$, and then evaluate the performance on the testing set $i$.

For discussing the recognition rates on training patterns we take the average of the 4 training sets for each line direction and for discussing those on testing patterns we take the average of the 4 testing sets for each line direction.

Table 2 shows the total number of the 4 sets of training patterns and that of the 4 sets of testing patterns for each direction of text line patterns as well as some statistics, where $N_{\text{i}}^\text{tr}$, $N_{\text{i}}^\text{te}$, $N_{\text{ac}}$ and $N_{\text{sd}}$ denote the number of true segmentation points, the number of true non-segmentation points, the average number of characters in a text line and the average number of characters written by one person, respectively.

4.1. Setting Parameters

For each line direction, we examined NNs which have the number of units for the middle layer $n_{\text{mu}}$ as 2, 4, 6, 8, and 10, and trained the parameters for these NNs using each training set until getting the smallest learning error. We selected the NNs that made the smallest learning error. The middle layer $n_{\text{mu}}$ of these selected NNs according to the training sets are shown in table 3.

For the SVMs, we used the following radial basis function kernel:

\[
K(x, x_\text{i}) = \exp\left(\frac{\|x - x_\text{i}\|^2}{2\sigma^2}\right)
\]

We obtained $\sigma$ and $C$ shown in eq. (7) by examining several values in experiments using each training set. Then, we obtained the parameters of the separating hyperplanes for the SVMs using the same training set again. The details of the numbers of support vectors for the trained SVMs according to each training set are shown in table 4.

Moreover, we took the distribution of the outputs of the NN and the SVM for each training set. The result for the training set $1$ of the direction $R$ is shown in Fig. 4 and Fig. 5. The outputs of the NNs and the SVMs for other training sets have similar features for each line direction. We can see the distribution of the outputs of the SVM is small from $-1 \text{ to } 1$, because the training patterns having the outputs from $-1 \text{ to } 1$ are regarded as having training errors and the

Table 2: Training and testing sets for each direction

<table>
<thead>
<tr>
<th>Direction</th>
<th>R</th>
<th>L</th>
<th>D</th>
<th>U</th>
</tr>
</thead>
<tbody>
<tr>
<td>Test lines</td>
<td>4242</td>
<td>1414</td>
<td>485</td>
<td>160</td>
</tr>
<tr>
<td>English letters</td>
<td>11973</td>
<td>5991</td>
<td>1889</td>
<td>1297</td>
</tr>
<tr>
<td>Numerals</td>
<td>65200</td>
<td>42750</td>
<td>6009</td>
<td>10351</td>
</tr>
<tr>
<td>Kanas</td>
<td>16328</td>
<td>54429</td>
<td>2253</td>
<td>751</td>
</tr>
<tr>
<td>Chinese characters</td>
<td>15144</td>
<td>58481</td>
<td>1135</td>
<td>445</td>
</tr>
<tr>
<td>Other characters</td>
<td>48024</td>
<td>19898</td>
<td>2371</td>
<td>77</td>
</tr>
<tr>
<td>$N_{\text{i}}^\text{tr}$</td>
<td>408360</td>
<td>2070594</td>
<td>3019</td>
<td>1172</td>
</tr>
<tr>
<td>$N_{\text{i}}^\text{te}$</td>
<td>1223712</td>
<td>335143</td>
<td>11589</td>
<td>3953</td>
</tr>
<tr>
<td>$N_{\text{ac}}$ (average)</td>
<td>10.6</td>
<td>10.6</td>
<td>8.1</td>
<td>8.9</td>
</tr>
<tr>
<td>$N_{\text{sd}}$ (average)</td>
<td>14927</td>
<td>14927</td>
<td>15.4</td>
<td>15.4</td>
</tr>
</tbody>
</table>
SVM has been trained to have the smallest sum of the training errors.

Then, we measured $C_p$ and the $f$ measure according to eq. (9) and eq. (10) after applying the step 3 in Section 2 using each training set for all the combinations of $th$ and $th_s$ at every 0.01 step from 0.0 to 0.5 for $th$ and from 0.5 to 1.1 for $th_s$ for the NNs, and at every 0.02 step from −1.1 to 0 for $th_s$, and from 0 to 1.1 for $th_s$ for the SVMs, respectively. We took the combination of $th$ and $th_s$ producing the best segmentation measure $C_p$ or $f$. The details for the parameters $th$ and $th_s$ set according to the result on each training set are shown in table 5 and table 6.

The average result of the 4 training sets and that of the 4 testing sets according to these parameters are shown in table 7, table 8, where $cnp$, $cup$, and $csp$ denote off-strokes classified into non-segmentation points, those classified into undecided points and those classified into segmentation points, respectively.

The $f$ measure of NN and SVMs

We compare the performance of the SVMs and that of the NNs on the training sets and the testing sets employing a Pentium (R) 4 3.40 GHz CPU with 0.99 GB memory.

Table 9 to 12 show the average result of the 4 trainings and that of the 4 testing sets after applying the step 3 in Section 2, where $C_p$, $f$, $Rc$, $Teq$, $Tr$, $Tav$, $Tav_{seg}$, $Tav_{rec}$, $T_{f_	ext{ave}}$ denote the point classification rate, the $f$ measure, the character recognition rate after applying the step 3 in Section 2, the time for training the parameters for the NNs or the SVMs using the training patterns, the average time for classifying an off-stroke into the three classes, the average time for processing a text line by the three steps mentioned in Section 2, respectively.

Table 11 shows the comparison of the two methods for text of the direction $D$

Table 12: Comparison of the two methods for text of the direction $U$

Eq. (12) shows a formula of the average time for processing a text line. The terms $N_{wgg}$ and $N_{adv}$ denote the average number of off-strokes in a text line and the average number of undecided points in a text line, respectively. The terms $T_{wgg}$, $T_{f_	ext{ave}}$, $T_{gg}$ and $T_{gg}^*$, are the average time for extracting the features from an off-stroke, the average time of character recognition in a text line, the average time for constructing the candidate lattice for a text line and the average time to search into the candidate lattice for a text line, respectively. The latter three terms depend on how many...
consecutive undecided points appear, and they have approximately the order of two to the power of $N_{udp}$.

$$T_{av\_seg} = N_{udp}T_{xy} + N_{udp}T_{av\_seg} + T_{bc} + T_{st} \quad (12)$$

$$T_{bc} = O(2^{N_{udp}})$$

$$T_{st} = O(2^{N_{udp}})$$

From Table 5 to 12 and eq. (12), we consider as follows:

1. For the direction R, L and D, the result of the segmentation measure and the character recognition rate by the SVMs are better than that by the NNs. For the direction U, although the result of the segmentation measure and the character recognition rate by the SVMs are a little behind than those by the NNs for the training patterns, the result of the segmentation measure and the character recognition rate by the SVMs are much better than those by NNs for the testing patterns, probably because the NNs were over-trained. Therefore, we can consider that the SVMs have brought about better segmentation performance and character recognition rate for all the directions.

2. The best NN has three layers with the middle layer as shown in Table 3. The larger the number of units for the middle layer $n_{u1}$ is, the smaller the learning error should be, but it is practically difficult to find the global minimum for the learning error.

3. The distribution of the outputs is very small form 1 to 1 for the SVMs as shown in Fig. 5, which provides reliable margin to discriminate segmentation points and non-segmentation points.

4. For the direction R, the classification time $T_{av\_seg}$ by the SVMs is about 8,835 times longer than that by the NNs because the SVMs must count the sum of the support vectors according to eq. (8), but the average time $T_{av\_seg}$ for processing a text line by the SVMs is only about 9 times longer than that by the NNs. This is because the segmentation by the NNs has a larger number of undecided points, which incurs longer time for character recognition, constructing the candidate lattice and searching into the candidate lattice as shown in Table 7, Table 8 and eq. (12). We consider that the average time $T_{av\_seg}$ for processing a text line of the direction R by the SVMs is acceptable. The result for the direction D is similar to that for the direction R. For the direction L and U, there are no great difference between $T_{av\_seg}$ by the SVMs and that by the NNs, because the difference between the number of undecided points by the SVMs and that by the NNs is not so large as shown in Table 7, Table 8.

5. The training time $T_{train}$ by the NNs is much shorter than that of the SVMs, when there are a large amount of training patterns (for the direction R and D), while $T_{train}$ by the SVMs is much shorter that of the neural network, when there are a small amount of training patterns (for the direction L and U).

6. The larger the number of training patterns is, the more support vectors the trained SVMs have as shown in Table 2 and Table 4. The larger the number of the support vectors is, the longer the classification time $T_{seg}$ is for the SVMs, because the SVMs must count the sum of the support vectors according to the eq. (8) with the result that the classification time $T_{seg}$ by the SVMs for the direction R is longest.

7. In Table 7 and Table 8, CSP is very low while $cup$ has high values by the NNs for true segmentation points of the direction R and D. It is because the output of the NNs distributes with large overlap between the two classes of true segmentation points and true non-segmentation points, with the result that the segmentation threshold $th$ are set excessively within the distribution of true segmentation points. This is not so fatal for recognition, however, since candidates are included in $cup$, although it entails longer recognition time.

8. The evaluation methods by $Cp$ and the $f$ measure implied the consistent result, i.e., SVMs' superiority over NNs.

5. Conclusion

This paper described a segmentation method of on-line handwritten Japanese text. We extracted multi-dimensional features from off-strokes in on-line handwritten text and applied a NN and an SVM to produce segmentation point candidates. The SVM brought about better segmentation performance and character recognition rate, although its processing time is behind the NN.

Acknowledgments

This research is being partially supported by Grant-in-Aid for Scientific Research under the contract number (B)17300031 and the MEXT Research and Education Fund for Promoting Research on Symbiotic Information Technology.

References


