A Candidate Lattice Refinement Method for Online Handwritten Japanese Text Recognition

Jianjuan Liang, Bilan Zhu and Masaki Nakagawa
Department of Computer and Information Sciences
Tokyo University of Agriculture and Technology
Tokyo, 184-8588, Japan
jianjuanliang@gmail.com, {zhubilan, nakagawa}@cc.tuat.ac.jp

Abstract— This paper presents a candidate lattice refinement method for online handwritten Japanese text recognition. In the integrated segmentation-recognition framework, we first over-segment a character string pattern into primitive segments at least at their true boundaries so that each primitive segment may compose a single character or a part of a character. Then a candidate lattice is constructed based on the primitive segments. We search within the candidate lattice to obtain the optimal path as recognition result. In striving for high recognition accuracy, however, the approach must generate many candidate lattice nodes, which ultimately increase the recognition time. To solve this problem, we refine the candidate lattice to eliminate unnecessary nodes before path search and text recognition. For the refinement, we evaluate all segmentation hypotheses by combining the probability of a character verifier using noncharacter samples, the class-independent unary and binary geometric context, as well as character segmentation. We retain N-best paths by beam search to reduce the complexity of the candidate lattice. Experiments on horizontal text lines extracted from the Kondate database show that the proposed method keeps recognition accuracy while reducing recognition time to half.

Keywords-online handwritten text recognition; integrated segmentation and recognition; lattice refinement; noncharacter patterns.

I. INTRODUCTION

Online handwritten text recognition has been receiving large attention due to the development and growing popularity of pen-based input devices with large writing areas, such as tablet PCs, electronic whiteboards and digital pens (e.g., Anoto pen). Using these devices, people tend to write text continuously with little constraints. To meet the needs of real applications, however, there is still further work needed to improve the performance of handwritten text recognition.

Due to the variable space between characters and the fact that many characters are composed of multiple radicals with internal gaps in the languages of Chinese origin, online handwritten Japanese text recognition faces the difficulty of character segmentation compared to isolated character recognition. The integrated segmentation and recognition method has widely been used to overcome the ambiguity of character segmentation [1], which is classified into the segmentation-free method and the over-segmentation-based method [2]. The over-segmentation-based method can utilize the character shapes into recognition, therefore, it can be successfully applied into handwritten Chinese/Japanese text recognition [3] [4].

In the over-segmentation-based recognition framework, a handwritten text (character string) is first over-segmented into primitive segments, consecutive segments are concatenated into candidate character patterns. Each candidate pattern is associated with a number of candidate classes and confidence scores by a character classifier. Then, a segmentation-recognition candidate lattice is constructed by combining all candidate patterns and character classes. Finally, by integrating the scores of character recognition, geometric context and linguistic context, an optimal segmentation-recognition path is searched from the constructed lattice by Viterbi search.

In over-segmentation, we usually keep all possible segmentation points to guarantee a high recognition rate, though it complicates the lattice and consequently burdens the computation of string recognition, especially for the case of partial overlapped characters [5]. As a result, there are many candidate character patterns. Due to thousands of Japanese characters, Zhu et al. [6] proposed a coarse classifier for candidate patterns before the fine classification to speed up the online handwritten Japanese characters recognition.

Since candidate patterns include true character and noncharacter patterns in the lattice, the character classifier or path evaluation should be resistant to noncharacter patterns. Liu et al. [1] evaluated several classifiers for candidate patterns with noncharacter training in the context of handwritten numeral strings, and showed that training with noncharacter samples improves neural classifiers and support vector classifiers. However, noncharacter training shows limited influence on the performance of discriminative density models: the learning vector quantization classifier and the discriminative learning quadratic discriminant function classifier.

Moreover, Li et al. [7] proposed a probabilistic model to evaluate segmentation hypotheses of a text line by
combining an isolated character recognizer and a character verifier. The character verifier evaluates candidate patterns based on the modified quadratic discriminant function (MQDF) classifier using noncharacter samples. This method works well for handwritten offline Chinese text line recognition.

On the other hand, many studies have focused on the postprocessing of geometric context to reject noncharacter patterns. The geometric context, such as character size, character unary and binary position, disambiguates the uncertainty of character segmentation. The geometric context models, which are integrated with a character classifier and linguistic context, improve the recognition performance in handwritten Japanese text recognition [8], in handwritten Chinese text recognition [9] and in handwritten numeral string recognition [10].

In this study, based on the system of Zhu et al. [3] (hereafter we call it the initial system), we focus on refining the constructed candidate lattice for handwritten Japanese text recognition to improve recognition speed. The proposed recognition method can be resistant to noncharacter patterns. In the lattice refinement, we evaluate segmentation paths in the lattice by combining the information of the candidate pattern verifier and class-independent geometric context, as well as character segmentation. A candidate pattern is verified whether it is true character or not by an SVM classifier using noncharacter samples. Then, we maintain N-best segmentation paths by beam search for normal text recognition. We search within the refined candidate lattice to obtain an optimal segmentation-recognition path by Viterbi search. Experiments on horizontal Japanese text lines extracted from the Kondate database [11], demonstrate the effectiveness of our lattice refinement.

The rest of this paper is organized as follows: Section II gives an overview of our handwritten Japanese text recognition system. Section III describes the lattice refinement. Section IV shows the results. Finally, section V concludes with our remarks.

II. OVERVIEW OF OUR RECOGNITION SYSTEM

For online handwritten Japanese text recognition, our recognition system has four major steps: over-segmentation, candidate lattice construction, candidate lattice refinement and path search, as shown in Fig. 1. The input is a handwritten text line composed of a sequence of strokes.

We define an off-stroke is a vector from the last point of a previous stroke to the first point of the next stroke. Each processing step will be described in detail in the following subsections.

A. Over-segmentation

In over-segmentation, we use two-stage classification scheme to minimize misclassification of each off-stroke.

A segmentation point separates two characters at the off-stroke, while a non-segmentation point indicates the off-stroke is within a character. An undecided point is interpreted as either segmentation point or non-segmentation point.

The group of consecutive strokes between two adjacent segmentation/undecided points is a primitive segment, and one or more consecutive primitive segments form a candidate character pattern.

In the first stage, each off-stroke is classified into two classes: non-segmentation point and hypothetical segmentation point, according to geometric features. To let hypothetical segmentation points include true segmentation points as much as possible, we adjust the threshold. We can get strict and loose over-segmentation, as shown in Fig. 2. We generally choose loose over-segmentation to take the accuracy unless speed is required.

In the second stage, we select some hypothetical segmentation points as segmentation points using an SVM classifier on 11 geometric features extracted from an off-stroke, as shown in Table 1. The remaining hypothetical segmentation points are set as undecided points.

Table II shows terminologies to derive geometric features, where acs is estimated by measuring the longer side length of the bounding box of each stroke, sorting the lengths of all the strokes and taking the average of the larger 1/3 of them.

![Figure 1. Flow of handwritten Japanese text recognition system.](image)

![Figure 2. Over-segmentation of the same text line: (a) strict case, (b) loose case.](image)
TABLE I. FEATURES EXTRACTED FROM AN OFF-STROKE.

<table>
<thead>
<tr>
<th>No.</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>f₁</td>
<td>DBₓ₁ / acs</td>
</tr>
<tr>
<td>f₂</td>
<td>Dₓ₂ / acs, Dₓ₀ / acs</td>
</tr>
<tr>
<td>f₃</td>
<td>Dₓ₃ / acs, Dₓ₄ / acs</td>
</tr>
<tr>
<td>f₄</td>
<td>Dₓ₅ / acs</td>
</tr>
<tr>
<td>f₅</td>
<td>Lₓ₆ / acs</td>
</tr>
<tr>
<td>f₆</td>
<td>sine(Lₓ₇), cosine(Lₓ₇)</td>
</tr>
<tr>
<td>f₇</td>
<td>DBₓ / the maximum DBₓ in text line</td>
</tr>
<tr>
<td>f₈</td>
<td>D / acs</td>
</tr>
</tbody>
</table>

TABLE II. TERMS TO DERIVE FEATURES.

<table>
<thead>
<tr>
<th>Term.</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>acs</td>
<td>Average character size of text line</td>
</tr>
<tr>
<td>BBₓ₀</td>
<td>Bounding box of immediately preceding stroke</td>
</tr>
<tr>
<td>BBₓ₁</td>
<td>Bounding box of immediately succeeding stroke</td>
</tr>
<tr>
<td>BBₓ₂</td>
<td>Bounding box of all preceding strokes</td>
</tr>
<tr>
<td>BSₓ₀</td>
<td>Bounding box of primitive segment which includes immediately preceding stroke</td>
</tr>
<tr>
<td>BSₓ₁</td>
<td>Bounding box of primitive segment which includes immediately succeeding stroke</td>
</tr>
<tr>
<td>DBₓ</td>
<td>Distance between BBₓ₀ and BBₓ₁ in x-axis</td>
</tr>
<tr>
<td>Dₓ₂</td>
<td>Distance between Bₓ₀ and Bₓ₁ in x-axis</td>
</tr>
<tr>
<td>Dₓ₃</td>
<td>Distance between centers of Bₓ₀ and Bₓ₁ in x-axis</td>
</tr>
<tr>
<td>Dₓ₄</td>
<td>Distance between centers of Bₓ₀ and Bₓ₁ in y-axis</td>
</tr>
<tr>
<td>Dₓ₅</td>
<td>Distance between centers of Bₓ₀ and Bₓ₁ in y-axis</td>
</tr>
<tr>
<td>Dₓ₆</td>
<td>Distance between centers of Bₓ₀ and Bₓ₁ in y-axis</td>
</tr>
<tr>
<td>Lₓ₇</td>
<td>Length of off-stroke</td>
</tr>
</tbody>
</table>

For SVM classification, we set the target value of segmentation point as 1 and that of non-segmentation point as -1, and train it using training patterns of off-strokes.

B. Candidate lattice construction

The combination of all possible candidate character patterns is represented by a candidate lattice, as shown in Fig. 3, where each node (rectangle) denotes a candidate character pattern. Each path from start segmentation point to end one in the lattice denotes a segmentation hypothesis for a handwritten text line.

Figure 3. Candidate lattice based on the over-segmentation as shown in Fig. 2: (a) strict case, (b) loose case.

C. Candidate lattice refinement

Due to many noncharacter patterns in the constructed candidate lattice, especially for the loose over-segmentation, noncharacters complicate the computation of text recognition. To speed up the text recognition, we keep only N-best segmentation paths by beam search with combining the probability of a character verifier using noncharacter samples, the class-independent unary and binary geometric context, as well as character segmentation. We use the maintained N-best segmentation paths to reconstruct a downsized candidate lattice with few candidate character patterns, as shown in Fig. 4, that is refined from the lattice as shown in Fig. 3 (b) where the thickly marked path is the correct segmentation path. Sec. III describes the processing of lattice refinement in detail.

D. Path search and recognition

Each candidate character pattern in the refined candidate lattice, is associated with a number of candidate classes with confidence scores by character classification. Then, all retained segmentation paths and recognition candidate classes are represented by a segmentation-recognition candidate lattice.

We utilize the path evaluation criterion proposed by Zhu et al. [3] to re-evaluate paths in the refined candidate lattice and search for the optimal string result by the Viterbi algorithm. This criterion combines the scores of character recognition, character size, inner gap, single-character position, pair-character position, and linguistic context, as well as character segmentation, with weighting parameters estimated by the genetic algorithm.

III. CANDIDATE LATTICE REFINEMENT

This section describes the process of candidate lattice refinement.

Given a handwritten text pattern, which is over-segmented into a sequence of candidate character patterns \( X = x₁x₂\cdots xₘ \). Using the following evaluation criterion, we can get N-best segmentation paths in the candidate lattice.
The following subsections will describe each probability in Eq. (1).

A. Candidate character verification

\[ P(\text{true} \mid x_i) \] is the probability that a candidate pattern \( x_i \) is a true character pattern. We obtain this probability using an SVM classifier, since there are only two classes: true character patterns and noncharacter patterns. Both of them are extracted from real handwritten text patterns. Figure 5 shows some examples of noncharacter patterns.

For training the SVM classifier, we extract directional features described in [12]. Since the extracted directional feature has 512 dimensions, we further use principal component analysis (PCA) to reduce the feature dimension.

In SVM classification, we set the target value of true character pattern as 1 and that of noncharacters as -1. We use the similar method described in [3] to transform the output of the SVM classifier to probability values.

B. Class-independent geometric features

\[ P(p_i^u \mid x_i) \] and \[ P(p_i^b \mid x_{i-1}, x_i) \] are the probability of a candidate pattern as a true character, and that of an over-segmentation gap is a valid between-character gap, respectively.

For class-independent unary geometric model, we extract 12 geometric features from a candidate pattern. For class-independent binary geometric model, we extract 14 geometric features from two adjacent primitive segments. These features are the same as in [9].

The class-independent unary geometric model measures whether a candidate pattern is a true character or not. For this two-class problem, we use an SVM classifier trained with the true character and noncharacter samples.
A. Settings

For the over-segmentation which is used for both the path evaluation in the lattice refinement search and that in the refined candidate lattice search, we collect the samples of two classes (segmentation points and non-segmentation points) from handwritten Japanese text lines of the training set as training data, and train the SVM classifier for off-strokes using the selected 11 features.

For the lattice refinement, we collect samples of two classes (true character patterns and noncharacter patterns) from handwritten Japanese text lines in the training set. The collection process is as follows: we mark the ground-truth boundaries of characters in text lines, then, we use the over-segmentation step to over-segment each text line so that a candidate lattice is constructed. For each candidate pattern in this lattice, if its boundary is the same as the ground-truth, it is treats as true character pattern, otherwise, it is regarded as a noncharacter pattern.

Since there are hundreds of thousands of noncharacter samples, we use a part of them to train the candidate character verifier, and a small part of them to train the class-independent unary geometric model. For the class-independent binary geometric model, we use a small part of text lines of the training set to train it. For the character verifier, we reduce the feature dimension from 512 to 10 by PCA to balance the accuracy and speed. Moreover, we adjust empirically the weighting parameters in the Eq. (1).

For the path evaluation in the refined candidate lattice search, the isolated character recognizer combines on-line and off-line character recognizers by a linear function [13], where the combining parameters are trained by Nakayosi database [14], we keep top 10 candidate character classes for every candidate character pattern. We also use Nakayosi database to train class-dependent geometric feature functions: unary and binary position features, size features and inner-gap features.

As for the linguistic context, we use tri-gram model. It is trained on the year 1993 volume of the ASAHI newspaper and the year 2002 volume of the NIKKEI newspaper. We estimate the smoothing parameters by Nakayosi database.

As for comparison with the initial system, we use the same terms to evaluate the performance of text line recognition as follows: the character recognition rate ($C_r$), which is the percentage of characters which are both correctly segmented and recognized, the segmentation measure ($F$) defined in Eq. (2), which combines recall and precision rates, and the average recognition time of a text line ($T_{av-rec \_tl}$).

The experiments are made on a PC with Intel(R) Core™ i7 3.40GHz CPU and 8GB RAM.

\begin{equation}
F = \frac{2}{\frac{1}{R} + \frac{1}{P}}
\end{equation}

\begin{align*}
R &= \frac{\text{number of correctly detected segmentation points}}{\text{number of true segmentation points}} \\
P &= \frac{\text{number of correctly detected segmentation points}}{\text{number of detected segmentation points (including false)}}
\end{align*}

B. Result and analysis

We compare the proposed system for online handwritten Japanese text recognition with the initial system on the same testing set. The difference between them is that whether it uses the processing of candidate lattice refinement or not. Table IV shows the results.

From the results, we can see that the character recognition rate of the proposed method can be close to that of the initial system, with higher speed by keeping 70-best segmentation paths. Moreover, the proposed method can achieve higher character recognition rate of 93.50% by keeping 100-best segmentation paths.

### TABLE IV. RESULTS OF TEXT LINE RECOGNITION ON THE TESTING SET.

<table>
<thead>
<tr>
<th>Performance Method</th>
<th>N-best</th>
<th>$C_r$ (%)</th>
<th>$F$</th>
<th>$T_{av-rec_tl}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>Initial system</td>
<td>1</td>
<td>81.72</td>
<td>0.9470</td>
<td>0.070 (s)</td>
</tr>
<tr>
<td></td>
<td>3</td>
<td>86.91</td>
<td>0.9668</td>
<td>0.072 (s)</td>
</tr>
<tr>
<td></td>
<td>5</td>
<td>88.90</td>
<td>0.9745</td>
<td>0.080 (s)</td>
</tr>
<tr>
<td></td>
<td>10</td>
<td>90.76</td>
<td>0.9820</td>
<td>0.090 (s)</td>
</tr>
<tr>
<td></td>
<td>30</td>
<td>92.39</td>
<td>0.9882</td>
<td>0.105 (s)</td>
</tr>
<tr>
<td></td>
<td>50</td>
<td>92.92</td>
<td>0.9902</td>
<td>0.114 (s)</td>
</tr>
<tr>
<td></td>
<td>100</td>
<td>93.50</td>
<td>0.9923</td>
<td>0.126 (s)</td>
</tr>
</tbody>
</table>

**Figure 6.** Examples of misrecognition of handwritten Japanese text, the result is under the handwritten text, and the ground truth is in brackets.
Figure 6 shows some misrecognition examples of handwritten Japanese text. The recognition errors are mainly due to the following reasons: First, the character classifier cannot give the correct class in the top 10 candidates. Second, the correct segmentation-recognition path cannot be found in the path search and text recognition, although the refined lattice includes the correct segmentation path. Third, the correct segmentation path cannot be found in N-best paths of the lattice refinement.

V. CONCLUSION

In this paper, we proposed a method to refine the lattice for online handwritten Japanese text recognition. In the lattice refinement, we evaluate segmentation hypotheses by combining the scores of the character verifier using noncharacter samples, the class-independent unary and binary geometric context, as well as character segmentation. The experiments on horizontal Japanese text lines showed the effect of the candidate lattice refinement.

In the future work, we will do the following works to make our proposed text recognition method robust. First, we will optimize the parameters in the evaluation criterion of lattice refinement by the genetic algorithm. Second, to better verify candidate character patterns, we cluster true characters into several super-classes according to some geometric features such as size and position, then classify them with noncharacters using another classifier such as MQDF.

ACKNOWLEDGMENT

This research is being partially supported by Grant-in-aid for Scientific Research C-15K00225 and NEDO under the number 27J1103.

REFERENCES


