Building a Compact On-line MRF Recognizer for Large Character Set using Structured Dictionary Representation and Vector Quantization Technique

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

Abstract

This paper describes a method for building a compact on-line Markov random field (MRF) recognizer for large handwritten Japanese character set using structured dictionary representation and vector quantization (VQ) technique. The method splits character patterns into radicals, whose models by MRF are shared by different characters such that a character model is constructed from the constituent radical models. Many distinct radicals are shared by many characters with the result that the storage space of model dictionary can be saved. Moreover, in order to further compress the parameters, we employ VQ technique to cluster parameter sets of the mean vectors and covariance matrixes for MRF unary features and binary features as well as the transition probabilities of each state into groups. By sharing a common parameter set for each group, the dictionary of the MRF recognizer can be greatly compressed without recognition accuracy loss.

1. Introduction

Due to the development and proliferation of pen-based or touch-based input devices such as tablet terminals, smart phones, electronic whiteboards and digital pens (e.g., Anoto pen), on-line handwritten character recognition is reviving even larger attentions than before.

Although character classifiers with high recognition accuracy have been reported [1-5], the demand for speeding up recognition and small memory consumption is very high for portable devices as well as desk-top applications for which handwriting recognition is incorporated as one of modules. Performance of these relatively small devices requires a handwriting recognition system as small as possible and recognition speed as fast as possible while maintaining high accuracy. Even for a desktop PC with relatively high device performance, recognition speed within 0.5 seconds per page and memory consumption within 10 MB are required in actual applications. Therefore, we need to refine the recognition scheme to reduce the system memory consumption.

Chinese, Japanese or Korean have thousands of different categories and their large character set is problematic not only in recognition rate but also in recognition speed and memory consumption.

We have proposed a robust on-line handwritten Japanese character recognition method using a MRF model [4, 5]. Although it realizes high recognition performance, however, the MRF recognizer requires about 20MB memory consumption due to the larger character set of 7,097 character categories within which included are 2,965 Kanji characters of the JIS level-1 standard and 3,390 Kanji characters of the level-2 standard as well as kana, English alphabet and so on. After combining the MRF recognizer with an off-line recognizer by the modified quadratic discriminant function (MQDF) and linguistic context processing at the string recognition step, the recognition system would require more larger memory consumption because of the large memory size for the MQDF dictionary and the linguistic context dictionary. Therefore, we need to compress the memory size as greatly as possible while maintaining high accuracy.

In segmentation-free HMM-based English word recognition, each letter is modeled as a HMM, word HMMs are constructed by concatenating letter HMMs during recognition [6-8]. We can regard each radical of Japanese characters as an English letter and each Japanese character as an English word, then we can

978-0-7695-4774-9/12 $26.00 © 2012 IEEE
DOI 10.1109/ICFHR.2012.188

apply the English word recognition method to recognize Japanese characters. However, the radicals in Japanese characters are different from the English letters largely, where the radicals with the same type from different Japanese characters may have different sizes, vertical-horizontal ratios and positions with the result that they have very different covariance matrixes among MRF models. It results in low recognition performance if we merge the radical MRF models with different covariance matrixes.

We presented an effective structured template-based on-line recognizer [9]. However, it did not need to consider the covariance matrixes of feature points, so that there was no problem such as MRF recognizer.

Nakai et al. [10-12] presented a structured HMM-based on-line Japanese recognizer, where substroke HMMs were used to construct the character model by concatenating them. However, only few character categories (1016 categories) were used so that it did not cause the problem that the radicals with the same type from different Japanese characters have very different covariance matrixes. Moreover, only sharing the common parameters of substrokes limits the compression effectiveness.

Long et al. [13] constructed a compact off-line MQDF recognizer by VQ technique. They split each two adjacent parameter elements of feature vectors as a set, and clustered the parameter sets into groups, and then by sharing a common parameter set for each group, the dictionary of the recognizer was greatly compressed. In our on-line MRF recognizer, there are many similar unary features and binary features among characters or radicals. Therefore, we can split the parameters of each mean vector, those of each covariance matrix for unary features and binary features, as well as those of the transition probabilities of each state as a set, and cluster the parameter sets into groups, with each group sharing a common parameter set. It can result in more effective compression.

In this paper, we present a method for building a compact on-line MRF recognizer for large handwritten Japanese character set, where the radical models are shared by different characters, and a character model is constructed from the constituent radical MRF models. We investigate how to normalize the sizes and positions of radicals with the same types from different characters. Moreover, we employ VQ technique to cluster the parameter sets of the mean vectors and the covariance matrixes as well as the state transition probabilities for MRF into groups. By sharing a common parameter set for each group, the dictionary of the MRF recognizer can be greatly compressed without recognition accuracy loss.

The rest of this paper is organized as follows: Section 2 gives an overview of our online handwritten character recognition method. Section 3 presents the construction of our structured MRF dictionary. Section 4 describes the dictionary compression by VQ. Section 5 presents the experimental results and Section 6 draws our conclusion.

2. Recognition system overview

We linearly normalize an input pattern by converting the pen-tip trace pattern to a standard size, preserving the horizontal-vertical ratio.

After normalization, we extract feature points using the method developed by Ramner [14]. First, the start and end points of every stroke are picked up as feature points. Then, the most distant point from the straight line between adjacent feature points is selected as a feature point if the distance to the straight line is greater than a threshold value. This selection is done recursively until no more feature points are selected. This feature point extracting process is shown in Fig. 1(a).

The extracted feature points represent the structure of a pattern. They are effective and more efficient to process compared with processing all the pen-tip points, as was done in previous studies [6-8].

We set feature points from an input pattern as sites $\boldsymbol{S}=\{s_1, s_2, s_3, \cdots, s_I\}$ and states of a C as labels $\boldsymbol{L}=\{l_1, l_2, l_3, \cdots, l_J\}$. The method recognizes the input pattern by assigning labels to the sites to make the matching between the input pattern and each C such as $\boldsymbol{F}=\{s_1 = l_1, s_2 = l_1, s_3 = l_3, \cdots, s_9 = l_8, s_{10} = l_8\}$, as shown in Fig. 1(b). The notation $\boldsymbol{F}$ is a configuration and denotes a mapping from $\boldsymbol{S}$ to $\boldsymbol{L}$.

We employ the coordinates of feature points as unary features and the differences in coordinates between the neighboring feature points as binary features. We then use a MRF model to match the feature points with the states of each character class and obtain a similarity for each one. We then select the each character class with the largest similarity as the recognition result [4, 5].

The neighborhood system of MRF model is set according to the successive adjacent feature points in writing order. We define a linear-chain MRF for each C, as shown in Fig. 1(c), where each label has a state and each state has three transitions.

Therefore, the energy function is as follows:

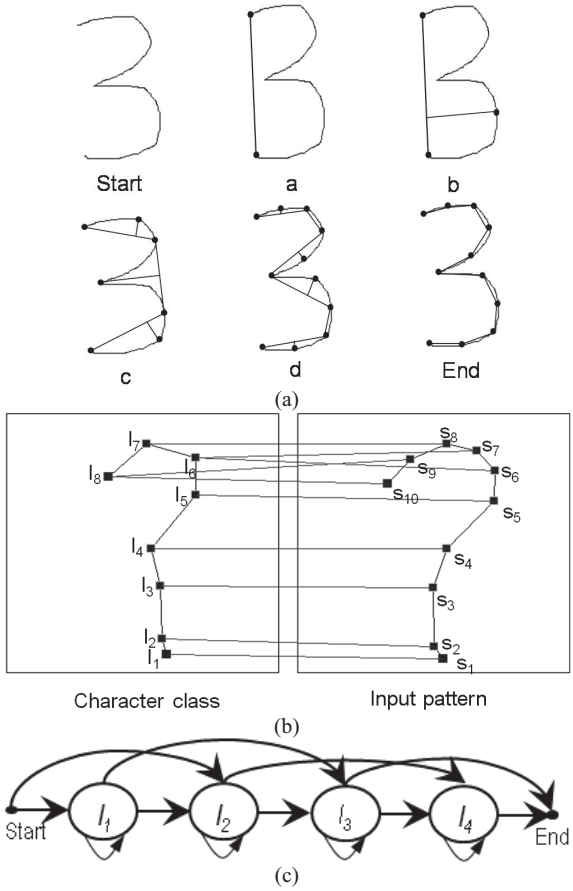

Figure 1. Feature points extraction and labeling and linear-chain MRF.

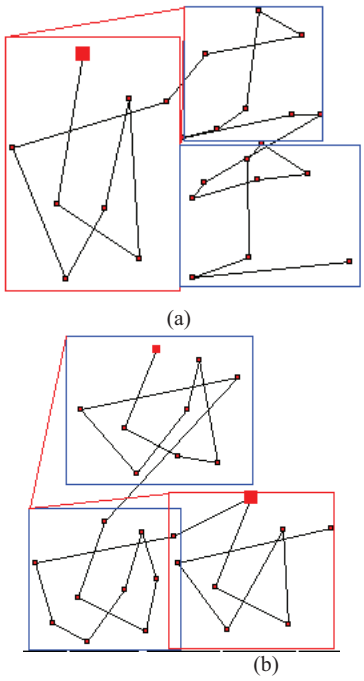

Figure 2. Interface to consruct structured MRF dictionary.

$$E(\mathbf{O},\mathbf{F}\mid C)=E(\mathbf{O}\mid\mathbf{F},C)+E(\mathbf{F}\mid C)$$
$$=\sum_{i=1}^{I}\left[-\log P(O_{s_i}\mid l_{s_i},C)-\log P(O_{s_is_{i-1}}\mid l_{s_i},l_{s_{i-1}},C)-\log P(l_{s_i}\mid l_{s_{i-1}},C)\right] \quad (1)$$

where l_{s_i} is the label of a C assigned to s_i, O_{s_i} is the unary feature vector extracted from site s_i, and $O_{s_is_j}$ is the binary feature vector extracted from the combination of s_i and s_j.

The smaller the energy function in (1) becomes, the larger the similarity between the input pattern and a C.

Each C has a linear-chain MRF, and the system uses the Viterbi search to match feature points of the input pattern with states for the MRF model of each C and to find the matching path with the smallest energy in (1) for each C.

The unary feature vector O_{s_i} comprises X and Y coordinates of s_i. The binary feature vector $O_{s_is_{i-1}}$ has two elements (dx: X coordinate of s_i - X coordinate of s_{i-1}, dy: Y coordinate of s_i - Y coordinate of s_{i-1}).

Gaussian functions are used to estimate $P(O_{s_i}\mid l_{s_i},C)$, and $P(O_{s_is_{i-1}}\mid l_{s_i},l_{s_{i-1}},C)$. $P(l_{s_i}\mid l_{s_{i-1}},C)$ is estimated as follows:

$$P(l_{s_i}\mid l_{s_{i-1}},C)=\frac{\text{Number of transitions from } l_{s_{i-1}} \text{ to } l_{s_i}}{\text{Number of sites assigned } l_{s_{i-1}}} \quad (2)$$
$$P(l_{s_1}\mid l_{s_0},C)=\frac{\text{Number of } s_1 \text{ assigned } l_{s_1}}{\text{Number of } s_1}$$

To train the MRF of each C, we first initialize the feature points of an arbitrary character pattern among the training patterns of the C as states of the MRF, set each unary feature vector of each feature point as the mean of the Gaussian function for each single-state, and set each binary feature vector between two adjacent feature points as the mean of the Gaussian function for each pair-state, and initialize the variances of those Gaussian functions and the state transition probabilities as 1. Then we use the Viterbi algorithm or the Baum-Welch algorithm to train the parameters of the MRF (the means and variances of Gaussian functions and the state transition probabilities). We repeat the training until the optimal parameters are obtained.

3. Construction of structured MRF dictionary

We first create a MRF recognizer at the character level. From each character MRF model, we can obtain the mean vectors of unary features that represent the mean coordinates of the feature points of training

patterns. Fig. 2 (a) and (b) show the mean coordinates of two characters [娃] and [姦]. Each small red rectangle shows a feature point. All the strokes are concatenated into a stroke for each character to achieve stroke-number independence. We created an interface to construct structured MRF dictionary as shown in Fig. 2. By a mouse right click at a feature point, we can cut the character pattern at the feature point and, by a mouse left click at a feature point we can concatenate the two radicals beside the feature point. We can split each character model into several parts by these operations. Then, for each part we can register it as a new radical to the radical dictionary by cutting it and normalizing it to the radical dictionary size. We can also search similar radicals from the registered radical dictionary then select a similar radical index to register it. Fig. 2 shows the split character models for characters [娃] and [姦].

After all character MRF models are split and are registered, we match all training patterns with character MRF models, and then we can cut each training pattern into radicals at the places where character MRF models are split. From the bounding boxes of radicals of training patterns, we can obtain the mean bounding box as shown in Fig. 2.

As shown in Fig. 2, we can see that the radicals with the same type such as [女] from different characters may have different sizes and positions. We need to normalize the sizes and positions of radical patterns. We can normalize them by the following two methods:

Method 1: Normalize each radical pattern according to the bounding box of the radical pattern.

Method 2: Normalize each radical pattern according to the mean bounding box of the radicals of training patterns

In the method 1, there is a problem to extract radical patterns from character patterns. This is very difficult for Japanese character. Without robust extraction method, we need to hypothetically segment radicals at every feature point, which results in very larger processing time.

To solve the problem, we normalize radical patterns by the mean bounding boxes of radicals appearing in n the same character class as shown in Fig.3, where the orange patterns show the input radical patterns. We cut out radical patterns, and then normalize them by the mean bounding box, then use the normalized radical patterns to train the radical MRF models.

When recognizing, we set the radical MRF models into the character models according to the mean bounding boxes by normalizing the mean vectors and the covariance matrixes with the mean bounding box to construct the character models.

Each radical MRF may come from various radical patterns in many character categories that have very different sizes and positions. We consider that different sizes will bring different variances, which may result in low recognition performance if we merge the radical MRF models with too different variances. Therefore, we propose a method that uses the k-mean method to cluster the sizes of each radical into groups and splits the radical into multiple classes. We optimize the number of the groups.

3. Dictionary compression by VQ

We propose a dictionary compression method inspired by the construction method for a compact off-line MQDF recognizer using VQ technique [13]. In our on-line MRF recognizer, there are many similar unary features and binary features among character models or radical models. We use a linear-chain MRF for MRF model, where each label has a state and each state has three transitions. As shown in Fig. 4, each mean vector has two parameter elements, and each covariance matrix has four parameter elements for both of unary features and binary features. Each state

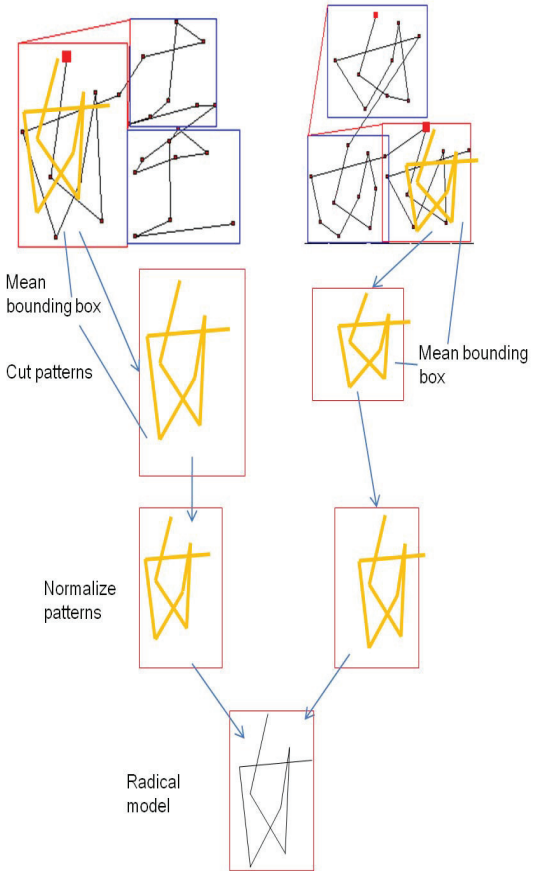

Figure 3. Radical pattern normalization.

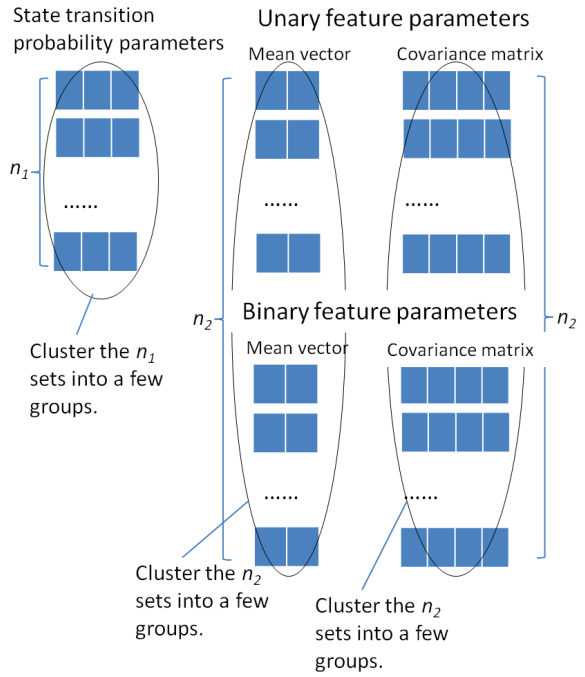

Figure 4. Clustering parameters by VQ.

has three transition probabilities so that it has three parameter elements. We consider the parameters of each mean vector, those of each covariance matrix and those of three transition probabilities of each state as a set. Then we cluster the parameter sets into groups, with each group sharing a common parameter set that is the canter of the group. Therefore, we only need to store the indexes of the groups and the center parameters of the groups, so that it can realize more effective compression.

4. Experiments

We evaluate the compact recognizer for 2,965 Kanji characters of the JIS level-1 standard. We trained the character recognizer by using an on-line Japanese handwriting database called Nakayosi [15]. On the other hand, the performance test was made on an on-line Japanese handwriting database called Kuchibue [15]. Table 1 shows the details of the databases. Each character class (character category) has a different number of sample patterns, and kana and symbol have more patterns (see Table 1). To maintain balance, we selected 100 patterns at random from each character class of the Kuchibue database and used the same number of sample patterns for each character class to evaluate the performance. The experiments were implemented on an Intel(R) Xeon(R) CPU W5590 @ 3.36 GHz 3.36 GHz (2 processers) with 12 GB memory.

TABLE I. STATISTICS OF CHARACTER PATTERN DATABASES.

		Nakayosi_t	Kuchibue_d
#writers		163	120
#characters /each writer	Total	11,962	10,403
	Kanji/Kana/ Symbol/alpha numerals	5,643/5,068/ 1,085/166	5,799/3,723/ 816/65
#character categories /each writer	Total	4,438	3,356
	Kanji/Kana/ Symbol/alpha numerals	4058/169 149/62	2976/169/ 146/62
#average category characters	Total	2.3	3.6
	Kanji/Kana/ Symbol/alpha numerals	1.4/22.0 5.5/1.0	1.9/30.0/ 7.4/2.7

We first compared the performance of three models: the original MRF without structured dictionary representation, the structured MRF (Str-MRF1), and the structured MRF which splits each radical into multiple classes by clustering the sizes for each radical with a k-mean method (Str-MRF2). Table 2 shows the results where C_r is the character recognition rate, *speed* is the average time to recognize a character.

TABLE II. COMPARISON OF STRUCTED MODELS

Performance \ Method		Original MRF	Str-MRF1	Str-MRF2
Test	*Cr* (%)	97.45	95.20	96.62
	memory	8MB	690KB	2.9MB
	speed	178.8us	178.9us	178.7us

From these results, we can see that Str-MRF1 largely degrades the character recognition accuracy, although it compresses the memory space largely. Str-MRF2 yields better recognition accuracy than Str-MRF1, and it also compresses the memory space from the original MRF. As for the speed, no loss has been observed.

Then, we compared the performance of the dictionary compression method by VQ. We can cluster the parameter sets into 255 groups, and save each group index at 1 byte data (VQ1). We can also cluster the parameter sets into the optimal number of groups, and save each group index at 2 byte data (VQ2). Moreover, after constructing the structured MRF, we can apply the VQ method to the structured dictionary to further compress the dictionary. For original MRF, we use 4 byte data to store each parameter of the state transition probabilities and the covariance matrixes, and use 2 byte data to store each parameter of the mean vectors. Table 3 shows the results.

TABLE III. COMPARISON OF VQ'S

Performance \ Method		Original MRF dictionary	VQ1	VQ2	VQ2+ Str-MRF2
Test	*Cr* (%)	97.45	96.50	97.55	96.72
	memory	8MB	975KB	1.72MB	750KB
	speed	178.8us	196.6us	196.6us	196.6us

From these results, we can see that VQ2 achieves better character recognition accuracy, although it consumes slightly more memory space compared to VQ1. Combining the str-MRF2 with VQ2 yields better recognition accuracy and smaller memory space than VQ1. As for the speed, little loss has been observed from VQ.

5. Conclusion

This paper presented a method for building a compact on-line MRF recognizer for large handwritten Japanese character set using structured dictionary representation and VQ technique. The radical models were shared by different characters, and a character model was constructed from the constituent radical MRF models. Moreover, we employed VQ technique to further compress the memory space without recognition accuracy loss.

References

[1] B. Zhu, X.-D. Zhou, C.-L. Liu and M. Nakagawa: A robust model for on-line handwritten Japanese text recognition, Int'l J. Document Analysis and Recognition (IJDAR), 13(2), pp.121-131, 2010.

[2] H. Oda, B. Zhu, J. Tokuno, M. Onuma: A. Kitadai and M. Nakagawa: A compact on-line and off-line combined recognizer. Proc. 10th IWFHR, pp. 133-138, 2006.

[3] C.-L. Liu and X.-D. Zhou: Online Japanese character recognition using trajectory-based normalization and direction feature extraction, Proc. 10th IWFHR, pp.217-222, 2006.

[4] B. Zhu and M. Nakagawa: A MRF model with parameters optimization by CRF for on-line recognition of handwritten Japanese characters, Proc. Document Recognition and Retrieval XVIII (DRR) that is part of IS&T/SPIE Electronic Imaging, San Jose, USA, 2011.

[5] B. Zhu and M. Nakagawa: On-line handwritten Japanese characters recognition using a MRF model with Parameter Optimization by CRF, Proc. 11th ICADR, pp. 603-607, 2011.

[6] S. Gunter and H. Bunke: HMM-based handwritten word recognition: on the optimization of the number of states, training Iterations and Gaussian components, Pattern Recognition, 37, pp. 2069-2079, 2004.

[7] M. Liwicki and H. Bunke: HMM-based on-line recognition of handwritten whiteboard notes, Proc. 10th IWFHR, pp. 595-599, 2006.

[8] S. Jaeger, S. Manke and J. Reichert: Online handwriting recognition: the Npen++ recognizer, Int'l J. Document Analysis and Recognition, Vol. 3, pp. 169-180, 2001.

[9] M. Nakagawa and L.V. Tu: Structural learning of character patterns for on-line recognition of handwritten Japanese characters, Advances in Structural and Syntactic Pattern Recognition, P. Perner, P. Wang, and A. Rosenfeld, eds., pp. 180-188, Springer-Verlag, 1996.

[10] M. Nakai, N. Akira, H. Shimodaira and S. Sagayama: Substroke approach to HMM-based on-line kanji handwriting recognition, Prof. 6th ICDAR, pp. 491-495, 2001.

[11] M. Nakai, T. Sudo, H. Shimodaira and S. Sagayama: Pen pressure features for writer-independent on-line handwriting recognition based on substroke HMM, Proc. 16th ICPR, 3, pp. 220-223, 2002.

[12] M. Nakai, H. Shimodaira and S. Sagayama: Generation of hierarchical dictionary for stroke-order free kanji handwriting recognition based on substroke HMM, Proc. 7th ICDAR, pp. 514-518, 2003.

[13] T. Long and L. Jin: Building compact MQDF classifier for large character set recognition by subspace distribution sharing, Pattern Recognition, 41, pp. 2916-2925, 2008.

[14] U. Ramer: An iterative procedure for the polygonal approximation of plan closed curves, Computer Graphics and Image Processing, 1(3), pp. 244-256, 1972.

[15] M. Nakagawa and K. Matsumoto: Collection of on-line handwritten Japanese character pattern databases and their analysis, Int'l J. Document Analysis and Recognition (IJDAR), 7(1), pp.69-81, 2004.