Handwritten Chinese/Japanese Text Recognition Using Semi-Markov Conditional Random Fields

Xiang-Dong Zhou, Da-Han Wang, Feng Tian, Cheng-Lin Liu, *Senior Member*, *IEEE*, and Masaki Nakagawa, *Member*, *IEEE*

Abstract—This paper proposes a method for handwritten Chinese/Japanese text (character string) recognition based on semi-Markov conditional random fields (semi-CRFs). The high-order semi-CRF model is defined on a lattice containing all possible segmentation-recognition hypotheses of a string to elegantly fuse the scores of candidate character recognition and the compatibilities of geometric and linguistic contexts by representing them in the feature functions. Based on given models of character recognition and compatibilities, the fusion parameters are optimized by minimizing the negative log-likelihood loss with a margin term on a training string sample set. A forward-backward lattice pruning algorithm is proposed to reduce the computation in training when trigram language models are used, and beam search techniques are investigated to accelerate the decoding speed. We evaluate the performance of the proposed method on unconstrained online handwritten text lines of three databases. On the test sets of databases CASIA-OLHWDB (Chinese) and TUAT Kondate (Japanese), the character level correct rates are 95.20 and 95.44 percent, and the accurate rates are 94.54 and 94.55 percent, respectively. On the test set (online handwritten texts) of ICDAR 2011 Chinese handwriting recognition competition, the proposed method outperforms the best system in competition.

Index Terms—Character string recognition, semi-Markov conditional random field, lattice pruning, beam search

1 Introduction

The problem of Chinese/Japanese handwriting recognition has received considerable attention for its potential in many applications as well as the technical challenge. Compared to isolated character recognition [1], the recognition of character strings (handwritten text) is more challenging due to the difficulty of character segmentation. Many works have focused on the rather constrained domains with short strings, small vocabulary or strong lexical constraints, such as word recognition [2], [3], [4], disease name recognition [5], legal amount recognition in bank checks [6], and mail address recognition [7], [8], [9]. Works on Chinese handwriting recognition of general texts have been reported only in recent years. The ICDAR 2011 Chinese handwriting recognition competition reported the best results (character level correct rates (CR)) 94.33 percent on online text lines and 77.26 percent on offline text lines [10]. A recently published article reported CRs 91.39 and 92.72 percent on offline Chinese handwriting databases CASIA-HWDB and HIT-MW, respectively [11]. Some works of online handwritten Japanese text recognition have reported comparable accuracies on the TUAT Kondate database [12], [13].

Handwritten Chinese/Japanese text recognition (HCTR) is challenging due to the large character set (over 5,000 characters are frequently used), the divergence of writing styles, the ambiguity of character segmentation, and the weak lexical constraint in general texts [11]. For these reasons, HCTR is generally accomplished by an integrated segmentation and recognition approach based on character oversegmentation [14]. Fig. 1 shows the flowchart of this approach. The input string (text line image for offline data or pen-tip trajectory for online data) is oversegmented into a sequence of components according to the overlapping between strokes (see Fig. 2a), with the hope that each component is a character or part of a character. Subject to constraints of character width, consecutive components are combined to generate candidate character patterns, which constitute the segmentation candidate lattice (see Figs. 2b and 2c). On assigning each candidate pattern a number of candidate classes using a character classifier, we construct the segmentation-recognition candidate lattice (referred to as lattice for brevity). Each path in the lattice corresponds to a segmentation-recognition hypothesis, which is evaluated by a parameterized function combining the character recognition score, geometric and linguistic contexts, and the string recognition result is obtained by searching for the optimal path with maximum score.

- *X.-D. Zhou is with the Beijing Key Lab of Human-Computer Interaction, Institute of Software, Chinese Academy of Sciences, Beijing 100190, P.R. China. E-mail: xiangdong@iscas.ac.cn.*
- *F. Tian is with the Beijing Key Lab of Human-Computer Interaction and the State Key Laboratory of Computer Science, Institute of Software, Chinese Academy of Sciences, Beijing 100190, P.R. China. E-mail: tianfeng@iscas.ac.cn.*
- *D.-H. Wang and C.-L. Liu are with the National Laboratory of Pattern Recognition (NLPR), Institute of Automation, Chinese Academy of Sciences, Beijing 100190, P.R. China. E-mail: {dhwang, liucl}@nlpr.ia.ac.cn.*
- *M. Nakagawa is with the Department of Computer and Information Sciences, Tokyo University of Agriculture and Technology, Naka-cho 2-24-16, Koganei, Tokyo 184-8588, Japan. E-mail: nakagawa@cc.tuat.ac.jp.*

Manuscript received 4 June 2012; revised 12 Dec. 2012; accepted 9 Feb. 2013; published online 28 Feb. 2013.
Recommended for acceptance by R. Manmatha.
For information on obtaining reprints of this article, please send e-mail to: tpami@computer.org, and reference IEEECS Log Number TPAMI-2012-06-0422.
Digital Object Identifier no. 10.1109/TPAMI.2013.49.

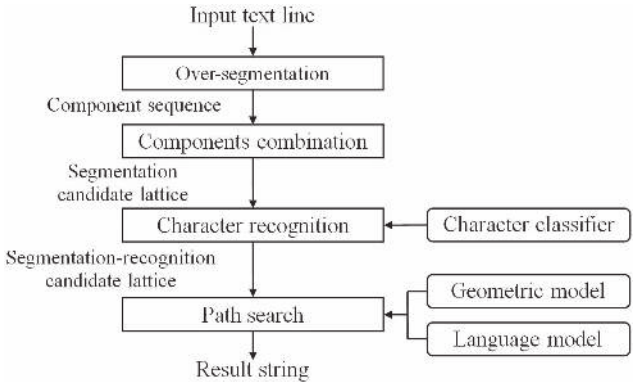


Fig. 1. Flowchart of integrated segmentation-recognition.

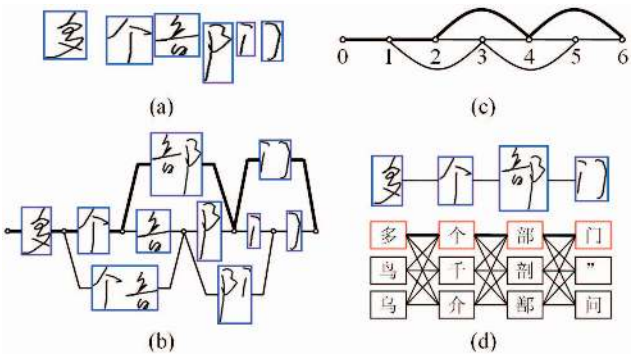


Fig. 2. Generation of segmentation-recognition candidate lattice: (a) component sequence; (b) candidate characters; (c) segmentation candidate lattice, where each node denotes a candidate segmentation point, each edge corresponds to a candidate character, and the bold lines indicate the desired segmentation; (d) candidate classes of the desired segmentation path.

The performance of integrated segmentation-recognition of character strings largely relies on the evaluation of candidate segmentation-recognition paths (hypotheses) in the lattice. In principle, character string recognition can be formulated as a Bayesian decision problem and solved by classifying to the string class of maximum a posteriori (MAP) probability. Due to the possibly infinite number of string classes, the probability has been heuristically approximated by decomposing into the character recognition scores and geometric and linguistic contexts [9], [11], [12], [15], [16], [17], [18], [19]. These heuristic methods more or less lose the optimality in fusing the contextual scores. Our previous method used the framework of conditional random field (CRF) for combining the scores and optimizing the weights [13], but could only combine context models of second-order (bigram) dependence.

In this paper, we propose a new method for character string recognition using the semi-Markov CRF (semi-CRF) [20]. We define semi-CRF on the lattice to directly estimate the a posteriori probability of a segmentation-recognition hypothesis, in which the information of character recognition, geometric and linguistic contexts are defined as feature functions. This model provides principled tools for both parameter learning and decoding under the MAP criterion and enables the fusing of high-order features (long range context models, such as the trigram language model). For alleviating the computational complexity of training with high-order features, we propose a forward-backward lattice pruning method to purge the implausible edges, and investigate beam search techniques for accelerating the decoding (path search). For improving the generalization performance of parameter learning in semi-CRF, we modify the negative log-likelihood (NLL) loss by adding a margin term, inspired by the margin-based maximum mutual information (MMI) training criterion in speech recognition [21], [22], [23]. In experiments on three online handwriting databases, the proposed method has yielded superior string recognition performance compared to the state-of-the-art methods.

The proposed method is an extension of our previous CRF-based method [13], which considers second-order language model only. The extension includes the use of trigram language models, the investigation of lattice pruning and beam search, the incorporation of margin term, extensive experimental results, and discussions. The remainder of this paper is organized as follows: Section 2 reviews the related works. Section 3 details the semi-CRF model defined on the candidate lattice. Section 4 introduces the forward-backward lattice pruning algorithm. Section 5 presents the decoding algorithms. Section 6 describes the feature functions employed. Section 7 presents our experimental results and Section 8 draws concluding remarks.

2 Related Work

Due to the large number of character classes and the infinite sentence classes of Chinese/Japanese texts, HCTR is preferably solved by segmentation-based approaches, which can be roughly divided into dissection methods and integrated segmentation-recognition methods [14]. Dissection methods [24], [25], [26], [27], [28] attempt to segment characters solely according to geometric layout features (character size/position and intercharacter relationship), before recognition with a character classifier. These methods are feasible only for neatly written texts, however. For continuous Chinese/Japanese handwriting, characters cannot be segmented unambiguously without character classification and linguistic context, due to irregular character size, ambiguous within-character and between-character gaps, as well as character touching. The integrated segmentation-recognition approach can overcome the ambiguity of character segmentation [29], and can be dichotomized into implicit segmentation and explicit segmentation methods [14].

Implicit segmentation methods [30], mostly combined with hidden Markov model (HMM) for character recognition, simply slice the string image into frames of equal length and label the sliced frames, which are concatenated into characters during recognition. This method does not exploit the character shape information sufficiently, and thus cannot yield satisfactory recognition performance. A two-stage approach takes implicit segmentation for character dissection using HMM, and then uses a second fine classifier for recognizing the segmented characters [31].

Explicit segmentation [7], [9], [11], [12], [15], [16], [17], [19], [32], [33], also called oversegmentation, tries to separate the input string at character boundaries, and usually results

in a component sequence with each component being a character or part of a character. Consecutive components are combined to generate candidate character patterns, which are assigned candidate classes using a character classifier, and the best segmentation-recognition result is achieved by path search. The evaluation of candidate segmentation-recognition paths (hypotheses) is a key issue in over-segmentation-based string recognition. It is usually done by combining the character classification scores, geometric and linguistic contexts heuristically as an approximation to the string class probability. A brief review of such methods and a recently proposed weighted evaluation function with maximum character accuracy training can be found in [11]. Particularly, strategies have been proposed to deal with the variable path length (number of characters) of segmentation hypotheses. Normalizing the accumulated path score with respect to the length helps overcome the bias of string class probability toward short strings [29], [34], but the resulting search problem cannot guarantee finding the optimal solution. Other measures include adding a path length penalty for compensating the accumulated path score [15] and weighting the character classification score with the number of constituent components [12], [18] or the character width [11]. The summation nature of accumulated path score enables optimal path search by dynamic programming. Generalizing the path score into a weighted form [11], [12], [15], [17] and optimizing the weights by discriminative training [11], [13] can improve the string recognition performance.

CRFs [35] are discriminative graphical models for labeling structured data. The traditional linear-chain CRF [35] has been applied to handwriting recognition, such as [36] and [37]. Semi-Markov conditional random fields (semi-CRFs) [20] are conditionally trained semi-Markov chains. A semi-CRF outputs a segmentation S of the observation sequence X, together with the label sequence Y assigned to the segments (subsequences) of X. In other words, unlike the linear-chain CRF [35] which models $P(Y \mid X)$, the semi-CRF explicitly estimates $P(S, Y \mid X)$. For HCTR, if X is the component sequence after oversegmentation, the segments will be the candidate characters. Semi-CRFs have the advantages that they allow the use of segment features and between-segment dependences. This attribute is important for HCTR, since the state-of-the-art Chinese character classifiers, such as the modified quadratic discriminant function (MQDF) [38], usually take the holistic character features as input. In contrast, for western handwriting recognition, HMM and bidirectional long short-term memory usually manipulate frame-level (for offline data) or point-level (for online data) features [39]. Semi-CRFs were initially proposed for segmenting and labeling sequential data in natural language processing [20]. For handwriting recognition, Shetty et al. [40] used a similar model for lexicon-driven handwritten word recognition with the parameters learned by maximizing the pseudolikelihood. A first-order semi-CRF was employed in our former work for online handwritten Japanese character string recognition [13].

Despite the widespread use of CRFs, the incorporation of higher order dependence remains a significant challenge. For exact inference (i.e., forward-backward and Viterbi algorithms) of first-order linear-chain CRFs, the time complexity is $O(TJ^2)$, where T is the sequence length and J is the class number [35]. To reduce the computation cost in inference, Pal et al. [41] compressed the marginal distribution by minimizing the Kullback-Leibler divergence. Cohn [42] proposed a tied potential algorithm which constrains the labeling considered in each feature function. Jeong et al. [43] unified the above two methods by decomposing the output labels into active and inactive sets, with the parameters of the inactive set held as a constant. Feng et al. [37] investigated three beam search techniques to make the inference more efficient. For exact inference of first-order semi-CRFs, the time complexity is $O(TJ^2L)$, where L is the maximum segment length allowed [20]. To alleviate the computation, Okanohara et al. [44] proposed two techniques: the first is to filter candidate states with a naive Bayes classifier, and the second is to pack feature-equivalent states by using feature forests, i.e., the states that have the same end position and previous named entity tag are packed. Few works have investigated the inference of high-order linear-chain CRFs [45], [46] and high-order semi-CRFs [47], in which the features are usually assumed to be sparse.

In the community of speech recognition, many attempts have been made on large-margin training of HMMs (see [48] for a review) for improving the generalization ability. Do and Artières [49] have applied this principle to handwriting recognition. However, compared with conventional training criteria, such as MMI [50], minimum classification error [51] and minimum phone/word error (MPE/MWE) [52], margin-based training criteria usually have different objective functions and different optimization algorithms [21]. Heigold et al. [21], [22] demonstrated that the MMI and MPE training criteria can be extended directly to embed the margin term with the optimization algorithms unchanged. Similar extensions to MMI can also be found in [23]. In [53], a cost-sensitive loss is incorporated into the CRF learning objective by adding a special feature function.

Compared to other works on HCTR, our work based on semi-CRFs can flexibly incorporate various information (including character classification score and various contexts) in a probabilistic manner. Fast inference techniques are investigated in both training and decoding, and a margin term is incorporated into the loss function to improve the generalization ability.

3 Semi-CRFs for String Recognition

With the MAP criterion, given a string X (text line image for offline data or pen-tip trajectory for online data), string recognition is to find the optimal label sequence Y^* by maximizing the posterior probability $P(Y \mid X)$:

$$Y^* = \underset{Y}{\operatorname{argmax}} P(Y \mid X) = \underset{Y}{\operatorname{argmax}} \sum_{S:Y} P(S, Y \mid X), \qquad (1)$$

where $S{:}Y$ stands for a segmentation (character sequence) of X paired with the label sequence Y. In practice, to avoid summing over huge number of segmentation hypotheses, (1) is approximated by searching for the best segmentation-recognition pair (S^*, Y^*):

$$(S^*, Y^*) = \underset{(S,Y)}{\operatorname{argmax}} P(S, Y \mid X), \tag{2}$$

where (S, Y) stands for a segmentation (character sequence) and the corresponding label sequence of X, respectively. In character string recognition, (S, Y) corresponds to a segmentation-recognition path in the candidate lattice (cf., Fig. 2), and the task of string recognition is to find the optimal path with MAP probability.

3.1 Semi-CRFs on Candidate Lattice

Given a string X, we should first oversegment it into a component sequence and then construct the segmentation-recognition candidate lattice (cf., Fig. 2) on which our semi-CRF model is defined. The lattice reduces the number of possible labels of each candidate character, for the state set actually includes all the categories modeled by the character classifier. For a hypothesized path (S, Y) in the lattice of X, like the traditional semi-CRF [20], $P(S, Y \mid X)$ can be written as the normalized product of potential functions:

$$\begin{aligned} P(S, Y \mid X) &= \frac{1}{Z(X)} \prod_{c \in S} \Psi_c(X, Y_c) \\ &= \frac{1}{Z(X)} \exp\{-E(X, S, Y)\}, \end{aligned} \tag{3}$$

where $\Psi_c(X, Y_c)$ is the potential function on maximal clique c (consecutive characters in the lattice):

$$\Psi_c(X, Y_c) = \exp\left\{ \sum_{k=1}^{K} \lambda_k f_k(X_c, Y_c) \right\}. \tag{4}$$

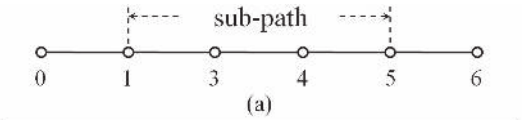
$f_k(X_c, Y_c)$ is the kth feature function, in which X and Y are both confined to the clique c. We also refer to Y_c as a labeling of clique c. Theoretically, semi-CRFs can take advantage of the whole string features, but for computational efficiency, X is usually confined to c. $\Lambda = \{\lambda_k \mid k = 1, \ldots, K\}$ are the weighting parameters to be learned. $E(X, S, Y)$ is the energy function:

$$E(X, S, Y) = -\sum_{c \in S} \sum_{k=1}^{K} \lambda_k f_k(X_c, Y_c). \tag{5}$$

$Z(X)$ is the partition function defined as the summation over all the paths in the lattice:

$$Z(X) = \sum_{(S', Y')} \prod_{c \in S'} \Psi_c(X, Y'_c). \tag{6}$$

A maximal clique is usually a sequence of m consecutive characters in the lattice, where m is called the maximal clique size, and is determined by the maximum order of features. For example, m is 2 if considering only the context of character pairs (e.g., bigram language model), and m is 3 for triple-character context (e.g., trigram language model). We also refer to a semi-CRF with maximal clique size m as $(m-1)$th order semi-CRF. Fig. 3a shows a full segmentation path and its subpath in the lattice illustrated in Fig. 2c, and Fig. 3b lists all the maximal cliques for $m = 3$. The cliques are formed by combining candidate characters with their preceding neighbors. This is to conform to the time-synchronous search which looks backward the character combinations [14]. Note that at the beginning of the path, some maximal cliques have sizes smaller than m due to the backward combination. Hereafter, if not stated otherwise, all cliques are maximal cliques.



(a)

Full path	(0,1), (0,1,3), (0,1,3,4),(1,3,4,5), (3,4,5,6)
Sub-path	(0,1,3), (0,1,3,4), (1,3,4,5)

(b)

Fig. 3. (a) A full segmentation path and its subpath in the lattice of Fig. 2c, where each number denotes the index of a candidate segmentation point, and each pair of neighboring numbers denotes a candidate character; (b) Maximal cliques ($m = 3$) on the full path and the subpath in (a).

3.2 Parameter Learning

In this section, we consider how to estimate the parameters Λ given N training samples: $\{(X^i, S^i, Y^i) \mid i = 1, \ldots, N\}$ (strings with segmentation points and character classes labeled). Following the standard MAP estimation, one minimizes the NLL loss with L_2-norm regularization:

$$\begin{aligned} L_{NLL}(\Lambda) &= -\sum_{i=1}^{N} \log P(S^i, Y^i \mid X^i) + \frac{C}{2} \|\Lambda\|^2 \\ &= \sum_{i=1}^{N} \big(E(X^i, S^i, Y^i) + \log Z(X^i) \big) + \frac{C}{2} \|\Lambda\|^2, \end{aligned} \tag{7}$$

where C is a positive constant balancing the loss term against the regularization term. Note that $P(S^i, Y^i \mid X^i)$, $E(X^i, S^i, Y^i)$ and $Z(X^i)$ involve parameters Λ. $L_{NLL}(\Lambda)$ is a convex function and can be optimized via gradient descent. The partial derivatives with respect to the weighting parameters are computed by

$$\frac{\partial E(X^i, S^i, Y^i)}{\partial \lambda_k} = -\sum_{c \in S^i} f_k(X^i_c, Y^i_c), \tag{8}$$

$$\frac{\partial \log Z(X^i)}{\partial \lambda_k} = \sum_{c, Y_c} f_k(X^i_c, Y_c) P(c, Y_c \mid X^i), \tag{9}$$

where the summation in (9) is calculated over all the cliques and labeling in the lattice. $P(c, Y_c \mid X^i)$ denotes the marginal probability that c is on the segmentation path and labeled as Y_c, that is, by fixing (c, Y_c), the paths through (c, Y_c) are arbitrary. Here, Y_c denotes a labeling of c. Similar to the linear-chain CRF [35], $P(c, Y_c \mid X^i)$ can be calculated by the forward and backward algorithms which will be detailed in Section 3.4.

We optimize the loss function by stochastic gradient descent, where the parameters are updated iteratively on each training sample. For $m = 3$, to reduce the computation cost, the lattice is first purged by a forward-backward lattice pruning step (cf., Section 4) using a pretrained first-order semi-CRF.

3.3 Incorporation of Margin Term

Inspired by the work of Heigold et al. [22], to introduce the margin concept into the NLL loss function, we define the margin-posterior:

$$P_m(S^i, Y^i \mid X^i) = \frac{\exp\{-E(X^i, S^i, Y^i) - \rho A((S^i, Y^i), (S^i, Y^i))\}}{\sum_{(S,Y)} \exp\{-E(X^i, S, Y) - \rho A((S, Y), (S^i, Y^i))\}}. \tag{10}$$

Compared with the posterior formulated in (3), the margin-posterior includes a margin term $\exp(-\rho A((S, Y), (S^i, Y^i)))$, in which $A((S, Y), (S^i, Y^i))$ is the accuracy function measuring the gains of classifying the genuine path (S^i, Y^i) as a rival path (S, Y), and $\rho > 0$ is a predefined scaling factor. By replacing $P(S^i, Y^i \mid X^i)$ with $P_m(S^i, Y^i \mid X^i)$ in (7), we achieve the modified NLL loss. According to [53], parameters learned by the modified criterion not only try to classify the training data correctly, but also manage to enforce the margin between (S^i, Y^i) and (S, Y) (here it is $E(X^i, S, Y) - E(X^i, S^i, Y^i)$) to be larger than the cost of classifying (S^i, Y^i) as (S, Y) (here it is the scaled accuracy difference: $\rho(A((S^i, Y^i), (S^i, Y^i)) - A((S, Y), (S^i, Y^i)))$). Thus it is reasonable to assign a smaller cost to a path with more correct characters, and a larger cost to the one with less correct characters. So, like the work of Heigold et al. [22] and Povey et al. [23], we adopt the accuracy function proposed in [52] to evaluate the matching between (S, Y) and (S^i, Y^i):

$$A((S, Y), (S^i, Y^i)) = \sum_{q \in S} \tilde{A}((q, Y_q), (S^i, Y^i)), \tag{11}$$

where (q, Y_q) is an edge (character-label pair) on path (S, Y), and the raw accuracy $\tilde{A}((q, Y_q), (S^i, Y^i))$ is defined as

$$\tilde{A}((q, Y_q), (S^i, Y^i)) = \max_{q' \in S^i} \begin{cases} -1 + 2e(q, q'), & \text{if } Y_q = Y^i_{q'} \\ -1 + e(q, q'), & \text{otherwise.} \end{cases} \tag{12}$$

Here $(q', Y^i_{q'})$ is an edge on genuine path (S^i, Y^i), and $e(q, q')$ is defined as the common component number between q and q', divided by the component number of q'.

Benefiting from the summation form of $A((S, Y), (S^i, Y^i))$, the margin term can be decomposed onto each potential term (cf., (4)) by introducing an additional feature function:

$$f_{K+1}(X_c, Y_c) = -\rho \tilde{A}((q, Y_q), (S^i, Y^i)), \tag{13}$$

where $(c, Y_c) \in (S, Y)$ is a clique-labeling pair, and (c, Y_c) ends with (q, Y_q) (cf., Section 6). By adding this feature function to the potential of (4), we replace $P(S^i, Y^i \mid X^i)$ with $P_m(S^i, Y^i \mid X^i)$ in the NLL loss (cf. (7)). The margin-based training criterion can take advantage of the same optimization algorithms as the conventional NLL-based training, if we keep the weighting parameter of the additional feature function, namely λ_{K+1} (cf., (4)), fixed to 1. Note that the margin term is used only in training, i.e., the inference in decoding and lattice pruning remains unchanged.

3.4 Inference on Candidate Lattice

The inference of the proposed model is different from that of the high-order semi-CRF in [47], which considers only sparse features depending on a constrained set of consecutive segment labels. For HCTR, the features of consecutive characters and labels are both important, and the label combinations could be arbitrary. Our algorithm has the advantage of being able to handle high-order features having arbitrary label and segment combinations.

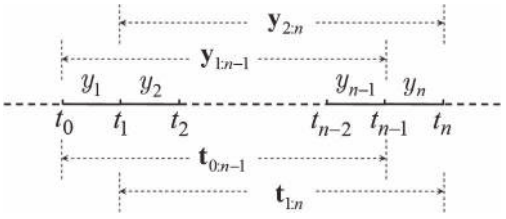


Fig. 4. Illustration of the calculation of forward and backward variables, where $\mathbf{t}_{0:n}$ is a maximal clique with size m.

Let $m \geq 2$ be the maximal clique size. Assume that the candidate segmentation points are indexed from 0 to T in the lattice (cf., Fig. 2c). We denote a subsegmentation path (character sequence) by $\mathbf{t}_{a:b}$, with $t_a, t_{a+1}, \ldots, t_b$ being $b - a + 1$ ordered candidate segmentation points (e.g., for the two-character subpath $\mathbf{t}_{0:2} = (1, 2, 4)$ in Fig. 2c, $t_0 = 1$, $t_1 = 2$ and $t_2 = 4$), and the labeling of $\mathbf{t}_{a:b}$ is denoted by $\mathbf{y}_{a+1:b}$, with $y_{a+1}, y_{a+2}, \ldots, y_b$ being $b - a$ labels for each character of $\mathbf{t}_{a:b}$. Let $(S^{t_b}_{t_a}, Y^{t_b}_{t_a})$ be an arbitrary subpath (character sequence and labeling) from candidate segmentation point t_a to t_b. The forward variables $\{\alpha_{\mathbf{t}_{1:n}}(\mathbf{y}_{2:n})\}$ are calculated on each $(\mathbf{t}_{1:n}, \mathbf{y}_{2:n})$ in the lattice, where $t_n = 1, \ldots, T$. When $t_1 = 0$, $2 \leq n \leq m$, otherwise $n = m$, that is, the size (character number) of $\mathbf{t}_{1:n}$ can be smaller than $m - 1$ when it starts from the first segmentation point. $\alpha_{\mathbf{t}_{1:n}}(\mathbf{y}_{2:n})$ is a summation of potential products over all the partial paths (character sequence and labeling) starting from segmentation point 0 and ending at $(\mathbf{t}_{1:n}, \mathbf{y}_{2:n})$ (cf., Fig. 4). The forward variables and the partition function $Z(X)$ can be calculated by the forward algorithm (sumproduct):

1. Initialization

$$\alpha_{\mathbf{t}_{1:n}}(\mathbf{y}_{2:n}) = \sum_{j=2}^{n} \Psi_{\mathbf{t}_{1:j}}(X, \mathbf{y}_{2:j}), \quad \text{for } t_1 = 0,\ 2 \leq n \leq m. \tag{14}$$

2. Recursion

$$\alpha_{\mathbf{t}_{1:n}}(\mathbf{y}_{2:n}) = \sum_{\substack{(S_0^{t_n}, Y_0^{t_n}): \\ (\mathbf{t}_{1:n}, \mathbf{y}_{2:n}) \in (S_0^{t_n}, Y_0^{t_n})}} \prod_{c \in S_0^{t_n}} \Psi_c(X, Y_c) = \sum_{t_0, y_1} \Psi_{\mathbf{t}_{0:n}}(X, \mathbf{y}_{1:n}) \alpha_{\mathbf{t}_{0:n-1}}(\mathbf{y}_{1:n-1}), \quad \text{for } t_1 \neq 0, n = m. \tag{15}$$

3. Termination

$$Z(X) = \sum_{\mathbf{t}_{1:n}, \mathbf{y}_{2:n}} \alpha_{\mathbf{t}_{1:n}}(\mathbf{y}_{2:n}), \quad \text{for } t_n = T. \tag{16}$$

In the above algorithm, $\Psi_{\mathbf{t}_{0:n}}(X, \mathbf{y}_{1:n})$ denotes the potential on maximal clique-labeling pair $(\mathbf{t}_{0:n}, \mathbf{y}_{1:n})$. $\{\alpha_{\mathbf{t}_{1:n}}(\mathbf{y}_{2:n})\}$ are calculated in a time-synchronous manner [14] sequentially from $t_n = 1$ to $t_n = T$.

Similarly, we can deduce the backward variables $\{\beta_{\mathbf{t}_{0:n-1}}(\mathbf{y}_{1:n-1})\}$ defined on each $(\mathbf{t}_{0:n-1}, \mathbf{y}_{1:n-1})$ in the lattice, where $t_0 = 0, \ldots, T-1$. When $t_{n-1} = T$, $2 \le n \le m$, otherwise $n = m$. $\beta_{\mathbf{t}_{0:n-1}}(\mathbf{y}_{1:n-1})$ is a summation of potential products over all the partial paths starting from $(\mathbf{t}_{0:n-1}, \mathbf{y}_{1:n-1})$ (cf., Fig. 4) and ending at segmentation point T. Also, $Z(X)$ can be calculated from the backward variables:

1. Initialization

$$\beta_{\mathbf{t}_{0:n-1}}(\mathbf{y}_{1:n-1}) = 1, \quad \text{for } t_{n-1} = T,\ 2 \le n \le m. \tag{17}$$

2. Recursion

$$\begin{aligned}\beta_{\mathbf{t}_{0:n-1}}(\mathbf{y}_{1:n-1}) &= \sum_{\substack{(S^T_{t_0}, Y^T_{t_0}):\\ (\mathbf{t}_{0:n-1}, \mathbf{y}_{1:n-1}) \in (S^T_{t_0}, Y^T_{t_0})}} \prod_{c \in S^T_{t_{n-1}}} \Psi_c(X, Y_c) \\ &= \sum_{t_n, y_n} \Psi_{\mathbf{t}_{0:n}}(X, \mathbf{y}_{1:n}) \beta_{\mathbf{t}_{1:n}}(\mathbf{y}_{2:n}), \\ &\quad \text{for } t_{n-1} \ne T, n = m.\end{aligned} \tag{18}$$

3. Termination

$$\begin{aligned}Z(X) = \sum_{\substack{\mathbf{t}_{0:n-1},\\ \mathbf{y}_{1:n-1}}} \beta_{\mathbf{t}_{0:n-1}}(\mathbf{y}_{1:n-1}) \prod_{c \in \mathbf{t}_{0:n-1}} \Psi_c(X, Y_c), \\ \text{for } t_0 = 0.\end{aligned} \tag{19}$$

In (19), $c \in \mathbf{t}_{0:n-1}$ denotes all the maximal cliques of $\mathbf{t}_{0:n-1}$ (cf., Fig. 3), and Y_c is a labeling of c determined by $\mathbf{y}_{1:n-1}$. $\{\beta_{\mathbf{t}_{0:n-1}}(\mathbf{y}_{1:n-1})\}$ is calculated in reverse order from $t_0 = T-1$ to $t_0 = 0$.

From the forward and backward variables, we can calculate the marginal probability on a subpath $(\mathbf{t}_{0:k}, \mathbf{y}_{1:k})$:

$$\begin{aligned}P(\mathbf{t}_{0:k}, \mathbf{y}_{1:k} \mid X) &= \sum_{(S,Y):(\mathbf{t}_{0:k}, \mathbf{y}_{1:k}) \in (S,Y)} P(S, Y \mid X) \\ &= \frac{1}{Z(X)} \sum_{\substack{\mathbf{t}_{2-m:1},\\ \mathbf{y}_{3-m:1}}} \sum_{\substack{\mathbf{t}_{k-1:k+m-2},\\ \mathbf{y}_{k:k+m-2}}} \alpha_{\mathbf{t}_{2-m:1}}(\mathbf{y}_{3-m:1}) \\ &\quad \times \beta_{\mathbf{t}_{k-1:k+m-2}}(\mathbf{y}_{k:k+m-2}) \prod_{c \in \mathbf{t}_{1:k+m-2}} \Psi_c(X, Y_c), \\ &\quad \text{for } m \ge 2,\end{aligned} \tag{20}$$

where $\{\alpha_{\mathbf{t}_{2-m:1}}(\mathbf{y}_{3-m:1})\}$ denotes all the forward variables ending at $(\mathbf{t}_{0:1}, y_1)$ (the first character and label of $(\mathbf{t}_{0:k}, \mathbf{y}_{1:k})$), and $\{\beta_{\mathbf{t}_{k-1:k+m-2}}(\mathbf{y}_{k:k+m-2})\}$ denotes all the backward variables starting from $(\mathbf{t}_{k-1:k}, y_k)$ (the last character and label of $(\mathbf{t}_{0:k}, \mathbf{y}_{1:k})$). $c \in \mathbf{t}_{1:k+m-2}$ denotes all the maximal cliques of $\mathbf{t}_{1:k+m-2}$ with $\mathbf{t}_{1:k+m-2} \in \mathbf{t}_{2-m:k+m-2}$ (cf., Fig. 3), and Y_c is a labeling of c determined by $\mathbf{y}_{3-m:k+m-2}$. Note that $\mathbf{t}_{2-m:-1}$, $\mathbf{y}_{3-m:0}$, $\mathbf{t}_{k+1:k+m-2}$ and $\mathbf{y}_{k+1:k+m-2}$ are variables. In (20), at the beginning of the component sequence, $\mathbf{t}_{2-m:1}$ should start from the segmentation point 0, and at the end of the component sequence, $\mathbf{t}_{k-1:k+m-2}$ should end at T. Fig. 5 illustrates the calculation of marginal probabilities. From (20), we can calculate the marginal probabilities on maximal cliques and further calculate the derivatives formulated in (9).

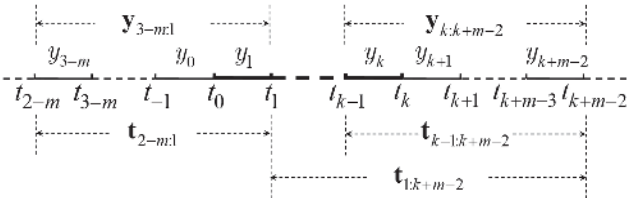


Fig. 5. Illustration of the calculation of marginal probability $P(\mathbf{t}_{0:k}, \mathbf{y}_{1:k})$.

Replacing the summation by maximization in (15) will yield the Viterbi-like (max-product) recursion, with which we can find the most probable segmentation-recognition path defined in (2), together with another type of forward variables $\{\hat{\alpha}_{\mathbf{t}_{1:n}}(\mathbf{y}_{2:n}) \mid t_n = 1, \ldots, T\}$:

1. Initialization

$$\begin{aligned}\hat{\alpha}_{\mathbf{t}_{1:n}}(\mathbf{y}_{2:n}) &= \sum_{j=2}^{n} \Psi_{\mathbf{t}_{1:j}}(X, \mathbf{y}_{2:j}), \\ &\quad \text{for } t_1 = 0,\ 2 \le n \le m.\end{aligned} \tag{21}$$

2. Recursion

$$\begin{aligned}\hat{\alpha}_{\mathbf{t}_{1:n}}(\mathbf{y}_{2:n}) &= \max_{t_0, y_1} \Psi_{\mathbf{t}_{0:n}}(X, \mathbf{y}_{1:n}) \hat{\alpha}_{\mathbf{t}_{0:n-1}}(\mathbf{y}_{1:n-1}), \\ \xi_{\mathbf{t}_{1:n}}(\mathbf{y}_{2:n}) &= \operatorname*{argmax}_{t_0, y_1} \Psi_{\mathbf{t}_{0:n}}(X, \mathbf{y}_{1:n}) \hat{\alpha}_{\mathbf{t}_{0:n-1}}(\mathbf{y}_{1:n-1}), \\ &\quad \text{for } t_1 \ne 0, n = m.\end{aligned} \tag{22}$$

3. Termination

$$(\mathbf{t}^*_{1:n}, \mathbf{y}^*_{2:n}) = \operatorname*{argmax}_{\mathbf{t}_{1:n}, \mathbf{y}_{2:n}} \hat{\alpha}_{\mathbf{t}_{1:n}}(\mathbf{y}_{2:n}), \text{for } t_n = T. \tag{23}$$

4. Backtracking

$$(t^*_0, y^*_1) = \xi_{\mathbf{t}^*_{1:n}}(\mathbf{y}^*_{2:n}), \quad \text{for } t^*_0 \ge 0,\ t^*_n \le T. \tag{24}$$

Once the recursion reaches the end segmentation point T, the best path can be obtained by traversing $\xi^*_{\mathbf{t}^*_{1:n}}(\mathbf{y}^*_{2:n})$ backwards until $t^*_0 = 0$.

Also, by replacing the summation in (18) with maximization, we can calculate another type of backward variables $\{\hat{\beta}_{\mathbf{t}_{0:n-1}}(\mathbf{y}_{1:n-1}) \mid t_0 = 0, \ldots, T-1\}$:

1. Initialization

$$\hat{\beta}_{\mathbf{t}_{0:n-1}}(\mathbf{y}_{1:n-1}) = 1, \text{ for } t_{n-1} = T,\ 2 \le n \le m. \tag{25}$$

2. Recursion

$$\begin{aligned}\hat{\beta}_{\mathbf{t}_{0:n-1}}(\mathbf{y}_{1:n-1}) &= \max_{t_n, y_n} \Psi_{\mathbf{t}_{0:n}}(X, \mathbf{y}_{1:n}) \hat{\beta}_{\mathbf{t}_{1:n}}(\mathbf{y}_{2:n}), \\ &\quad \text{for } t_{n-1} \ne T, n = m.\end{aligned} \tag{26}$$

With the modified forward and backward variables, we can calculate the posterior probability of the best path

traversing a subpath by replacing the summation with maximization in (20). Here, we consider only the best path through an edge $(\mathbf{t}_{0:1}, y_1)$ for $m = 2$:

$$P(S^*, Y^*, \mathbf{t}_{0:1} \in S^*, y_1 \in Y^* \mid X) = \max_{\substack{(S,Y):\\ (\mathbf{t}_{0:1}, y_1) \in (S,Y)}} P(S, Y \mid X) = \frac{1}{Z(X)} \hat{\alpha}_{\mathbf{t}_{0:1}}(y_1) \hat{\beta}_{\mathbf{t}_{0:1}}(y_1). \quad (27)$$

In (27), $Z(X)$ is a constant with trained parameters. The value $\hat{\alpha}_{\mathbf{t}_{0:1}}(y_1)\hat{\beta}_{\mathbf{t}_{0:1}}(y_1)$ will be used as a score of $(\mathbf{t}_{0:1}, y_1)$ for pruning the lattice in parameter learning.

For exact inference (i.e., forward-backward and Viterbi-like algorithms) on a dense lattice (before pruning) of the above $(m-1)$th order semi-CRFs, the time complexity is $O(TL^m J^m)$, where L is the maximum component number allowed to form a candidate character, and J is the candidate class number for each character. We can see that the computation cost increases exponentially with the maximal clique size m. For fast inference, we consider lattice pruning in training and use beam search in decoding.

4 Lattice Pruning

For semi-CRFs, the use of high-order features ($m > 2$) can potentially lead to an exponential blowup in the computation cost of inference. In our experiments, when $m = 3$ (for trigram language model), the training becomes computationally intractable on long string samples with a dense lattice. A practical way to solve this problem is to reduce the lattice complexity by removing implausible edges. The difficulty of lattice pruning lies in that the edges are not independent, i.e., to remove an edge will break the paths through it.

Inspired by the work of Sixtus and Ortmanns [54], we propose a forward-backward lattice pruning algorithm. By (27), each edge $(\mathbf{t}_{0:1}, y_1)$ in the lattice can be assigned a score $\hat{\alpha}_{\mathbf{t}_{0:1}}(y_1)\hat{\beta}_{\mathbf{t}_{0:1}}(y_1)$, which is also the score of the best path traversing it. Considering that a high-score path is unlikely to go through a low-score edge, to remove a low-score edge will unlikely break a high-score path. Denote the best path score by Q_{max}, an edge is reserved if the following condition is held:

$$\log Q_{max} - \log(\hat{\alpha}_{\mathbf{t}_{0:1}}(y_1)\hat{\beta}_{\mathbf{t}_{0:1}}(y_1)) \le \gamma_p, \quad (28)$$

where $\gamma_p > 0$ is the pruning threshold. With this method, the paths with higher scores are retained, while those with lower scores are discarded. Note that in training, the ground-truthed path is reserved without pruning. In our implementation, we first prune the lattice using a pre-trained first-order semi-CRF, then perform parameter learning with trigram language model on the pruned lattice.

5 Decoding

According to (2), decoding is to search for the best segmentation-recognition path in the lattice. In the Viterbi-like decoding (cf., Section 3.4), the forward variables are calculated in a time-synchronous manner from $t_n = 1$ to $t_n = T$. To make the search more efficient, we use beam search to purge $\{(\mathbf{t}_{1:n}, \mathbf{y}_{2:n}) \mid t_n = \tau\}$ at each candidate segmentation point τ by comparing the forward variables $\{\hat{\alpha}_{\mathbf{t}_{1:n}}(\mathbf{y}_{2:n}) \mid t_n = \tau\}$. Only the survived partial paths are extended in the following time step, and the number of reserved partial paths is called the beam width at τ. Although beam search does not guarantee finding the optimal solution, in practice, the pruning error is negligible when the beam width is properly selected [37]. We investigate three beam search techniques based on N-best, ratio threshold and K-L divergence, that were initially utilized on the inference of first-order linear-chain CRFs [37].

At each candidate segmentation point τ, we first sort the forward variables $\{\hat{\alpha}_{\mathbf{t}_{1:n}}(\mathbf{y}_{2:n}) \mid t_n = \tau\}$ in descending order. The N-best based beam search reserves at most top γ_b elements of $\{(\mathbf{t}_{1:n}, \mathbf{y}_{2:n}) \mid t_n = \tau\}$ according to the sorted forward variables, and the others are eliminated.

Ratio threshold-based beam search compares each element of $\{\hat{\alpha}_{\mathbf{t}_{1:n}}(\mathbf{y}_{2:n}) \mid t_n = \tau\}$ with the largest one $A_{max}(\tau)$, and $(\mathbf{t}_{1:n}, \mathbf{y}_{2:n})$ is reserved if the following condition is held:

$$\log A_{max}(\tau) - \log \hat{\alpha}_{\mathbf{t}_{1:n}}(\mathbf{y}_{2:n}) \le \gamma_r, \quad \text{for } t_n = \tau, \quad (29)$$

where γ_r is a empirically selected positive threshold.

The basic idea of K-L divergence-based beam search is to approximate a distribution with a mixture of Kronecker delta functions by minimizing the K-L divergence between the two distributions [41]. The distribution of $\{\hat{\alpha}_{\mathbf{t}_{1:n}}(\mathbf{y}_{2:n}) \mid t_n = \tau\}$ is calculated by

$$P(\hat{\alpha}_{\mathbf{t}_{1:n}}(\mathbf{y}_{2:n}) \mid t_n = \tau) = \frac{\hat{\alpha}_{\mathbf{t}_{1:n}}(\mathbf{y}_{2:n})}{\sum_{(\mathbf{t}'_{1:n}, \mathbf{y}'_{2:n}): t'_n = \tau} \hat{\alpha}_{\mathbf{t}'_{1:n}}(\mathbf{y}'_{2:n})}. \quad (30)$$

After sorting $\{P(\hat{\alpha}_{\mathbf{t}_{1:n}}(\mathbf{y}_{2:n})) \mid t_n = \tau\}$ in descending order, we select $(\mathbf{t}_{1:n}, \mathbf{y}_{2:n})$ from the top until the summation of the probabilities defined in (30) satisfies

$$-\log \sum_{(\mathbf{t}_{1:n}, \mathbf{y}_{2:n}): t_n = \tau} P(\hat{\alpha}_{\mathbf{t}_{1:n}}(\mathbf{y}_{2:n}) \mid t_n = \tau) \le \gamma_k, \quad (31)$$

where γ_k is a predefined positive threshold. To avoid breaking all the paths especially when γ_k is a small value, $(\mathbf{t}_{1:n}, \mathbf{y}_{2:n})$ with the largest probability at each candidate segmentation point will be definitely reserved.

6 Feature Functions

In the proposed model, the feature functions evaluate the scores of candidate character classification and the compatibilities of geometric and linguistic contexts, thus there are three types of feature functions. In (4), for each potential function on clique-labeling pair $(c, Y_c) = (\mathbf{t}_{0:m}, \mathbf{y}_{1:m})$, the feature functions are defined on the suffixes of (c, Y_c): $\{(\mathbf{t}_{j:m}, \mathbf{y}_{j+1:m}) \mid j = 0, \ldots, m-1\}$. If a suffix is shared by multiple clique-labeling pairs, the feature functions on it will be duplicated.

6.1 Character Classification

The feature functions for candidate character classification can be categorized into class-specific and class-unspecific ones. The class-unspecific one is defined as

$$f^{cu}(X_c, Y_c) = (t_m - t_{m-1}) \log g(\mathbf{t}_{m-1:m}, y_m), \quad (32)$$

where $g(\mathbf{t}_{m-1:m}, y_m)$ is the transformed classifier output (confidence) for character $\mathbf{t}_{m-1:m}$ on class y_m. From each

candidate character, we extract 512D features of local direction histogram, with the trajectory normalized by the moment normalization method in original direction [55]. The feature dimensionality is further reduced to 160, and on the reduced vector, we select 200 candidate classes according to the euclidean distance to class means. On the candidate classes, the character classifier, such as MQDF, is computed to give the recognition score [55]. Based on the work of Liu et al. [56] and [11], the classifier output is converted to a confidence according to the Dempster-Shafer theory of evidence. After taking logarithm, the value is further scaled with the component number $(t_m - t_{m-1})$ of the character to balance the effect of path length [11].

Inspired by the features used in natural language tasks [57], we define class-specific features as

$$f_i^{cs}(X_c, Y_c) = \delta(y_m, \omega_i) f^{cu}(X_c, Y_c), \quad \text{for } i = 1, \ldots, |\Omega|,\ \omega_i \in \Omega, \tag{33}$$

where $\Omega = \{\omega_i \mid i = 1, \ldots, |\Omega|\}$ denotes the classes modeled by the character classifier, and y_m is the last class of Y_c as in (32). Compared with the 1D class-unspecific feature function f^{cu}, whose weighting parameter in (4) is shared by all the character classes, each $f_i^{cs}, i = 1, \ldots, |\Omega|$, is weighted independently. We hope to improve the discrimination ability of the proposed model by learning class-specific weights in the energy function. Note that in (33), although the dimensionality of $\{f_i^{cs}\}$ is $|\Omega|$, only the one with $y_m = \omega_i$ is nonzero given the input (X_c, Y_c), that is why they are referred to as class-specific.

6.2 Geometric Context

We use four geometric feature functions, which are the logarithm of transformed classifier outputs via sigmoidal confidence transformation [56], and are categorized as class-dependent if depending on character class (or class pair) or class-independent otherwise [13].

Considering that alphanumeric characters, punctuation marks, and Chinese characters usually exhibit distinct outline features (e.g., size, position, aspect ratio, and within-character gap), we design two class-dependent geometric feature functions, with one defined on single characters (unary), and another defined on character pairs (binary), respectively [13]. Since the category number of Chinese/Japanese characters is very large and many different characters have similar geometric features, we cluster the character classes into six superclasses as in [11] and [17]. Then two quadratic discriminant functions (QDFs) for 6 and 36 superclasses are trained to give the scores for unary geometry and binary geometry, respectively.

In addition, two class-independent geometric feature functions are designed to indicate whether a candidate pattern is a valid character (unary), and whether a gap is a between-character gap (binary) or not, respectively [13]. Both are two-class problem, so we use two linear SVMs to give the scores. The details of the unary and binary geometric feature extraction can be found in [13].

6.3 Linguistic Context

In handwriting recognition, the statistical language model (SLM) provides a prior probability $P(\mathbf{y}_{1:N})$ for a label sequence $\mathbf{y}_{1:N}$, which attempts to reflect how frequently $\mathbf{y}_{1:N}$ occurs as a sentence [58]. Generally, SLM is trained on a large text corpus. To make the estimation practicable, proper approximation is required. An m-gram language model is based on the assumption that a label sequence follows an $(m-1)$th order Markov process:

$$P(\mathbf{y}_{1:N}) = \prod_{n=1}^{N} p(y_n \mid \mathbf{y}_{1:n-1}) \approx \prod_{n=1}^{N} p(y_n \mid \mathbf{y}_{n-m+1:n-1}). \tag{34}$$

The character bigram ($m = 2$) and trigram ($m = 3$) are considered in our experiments. Here, m is identical to the maximal clique size. We use the SRI language model toolkit [59] to give the parameters of m-gram models. By the toolkit, the default smoothing technique (Katz smoothing) and the entropy-based pruning are used. The thresholds of the pruning for character bigram and character trigram are empirically set as 5×10^{-8} and 10^{-7}, respectively [18]. The logarithm of the m-gram probability $\log p(y_n \mid \mathbf{y}_{n-m+1:n-1})$ is used as a feature function for each clique with labeling $\mathbf{y}_{n-m+1:n}$.

7 Experiments

We evaluated the proposed method on unconstrained online handwritten text lines of a Chinese handwriting database CASIA-OLHWDB [60] and a Japanese handwriting database TUAT Kondate [12]. For character classification, unless otherwise stated, the default classifier is MQDF and the feature dimensionality is reduced to 160D by Fisher linear discriminant analysis (FDA) [55].

7.1 Databases and Experimental Setting

The CASIA-OLHWDB database contains both isolated characters and unconstrained handwritten texts, with 816 writers' data for training and 204 writers' data for testing. The training set contains 3,129,496 isolated character samples of 7,356 classes (7,185 Chinese characters, 10 digits, 52 English letters and 109 frequently used symbols) and 4,072 pages of handwritten texts (41,710 text lines, including 1,082,220 characters of 2,650 classes). Four out of five of the merged training samples (isolated characters and those segmented from the texts falling in the 7,356 classes) were used for training the character classifier, and the remaining 1/5 were used for confidence parameter estimation. The test string set contains 10,510 text lines from 1,020 text pages, including 269,674 characters of 2,631 classes.

For Japanese, the horizontal text lines extracted from the TUAT Kondate database [12] were used in our experiments. The training set contains 10,174 text lines, including 104,093 characters of 1,106 classes. And the test set contains 3,511 text lines, including 35,766 characters of 791 classes (some labeling errors of the test set are corrected in our work, so the statistics are slightly different from those of [12]). The character classifier was trained on the TUAT Nakayosi database (1,695,689 isolated characters of 4,438 classes) [61] together with the isolated characters extracted from the Kondate training set falling in the 4,438 classes.

Note that for general-purpose recognition, the classifiers for both Chinese and Japanese model large numbers of classes (7,356 and 4,438, respectively), though the test text lines have much less classes of characters.

TABLE 1
Statistics of Character Types on Segmented Characters of Test Strings

All	Chinese	Symbol	Digit	Letter
269,674	234,803	26,764	6,933	752
35,766	12,432	16,082	5,576	1,676

Upper: CASIA-OLHWDB; lower: TUAT Kondate.

For both the CASIA-OLHWDB database and the TUAT Kondate database, the QDFs and the SVMs for geometric feature functions were trained on the features extracted from the respective training text lines. Again, 4/5 of data were used for training classifiers and the remaining 1/5 were used for confidence parameter estimation. The Chinese language models were trained on a text corpus from the CLDC (Chinese Linguistic Data Consortium), containing about 50 million characters [18]. The Japanese language models were estimated from the text corpus of the Japanese Mainichi Newspaper [17]. For each database, the semi-CRF parameters were learned on the training text lines.

The text lines of both databases have been annotated with segmentation points and character labels. Table 1 lists the character numbers for the characters segmented from the test strings, where "Chinese" denotes Chinese characters or Japanese Kanji, "Symbol" includes both symbol characters and kana characters for Japanese. In addition, the test text lines in CASIA-OLHWDB have 422 characters that are out of the classes modeled by the character classifier and thus cannot be correctly recognized.

7.2 Performance Metrics

Following [30] and [11], the string recognition performance is evaluated by character level CR and accurate rate (AR). We also use the term character error rate (CER), which equals $1 - AR$, i.e., the total number of errors divided by the total number of characters in the transcript. And we denote the error rates of the three error types (substitution, deletion and insertion) by SUB, DEL, and INS, respectively.

To evaluate the quality of the segmentation-recognition candidate lattice, we consider lattice edge density (LED) and lattice error rate (LER). LED is used to measure the complexity of the lattice, which is defined as the total number of edges divided by the total number of characters in the transcript. LER is used to measure the errors caused by lattice construction, which provides a lower bound of CER. LER is defined as the total number of lattice errors divided by the total number of characters in the transcript, where the lattice errors are defined as the minimum edit-distance between the transcript and any label sequence in the lattice. Similar measures for the evaluation of word graph quality in speech recognition can be found in [62].

7.3 Experimental Results

We first evaluate the effects of feature functions and the margin term, then show the performance of lattice pruning in semi-CRF training, and evaluate the three beam search techniques in decoding. At last, we compare the performance using different character classification methods.

We implemented the methods in MS Visual C++ 2008 and tested on a PC with Intel Quad Core 2.83-GHz CPU and 4-GB RAM. In training, the string samples were processed iteratively for five cycles in stochastic gradient decent. The candidate class number J was set as 12 in training, and 10 in testing, which are large enough for HCTR problem [11]. The scaling factor ρ (cf. (13)) in the margin term was set as 2. In parameter learning with trigram language model ($m = 3$), the lattice was first pruned with a pretrained first-order semi-CRF ($m = 2$) with bigram language model, and the pruning threshold γ_p (cf. (28)) was set as 12. In decoding, the default beam search technique is ratio threshold based, and γ_r was set as 10. (In Section 7.3.4, we will see that with comparable recognition accuracies, this method runs faster than the other two). ρ, γ_p, and γ_r were separately tuned on the training set by cross-validation. Since the regularization constant C and the margin scale ρ are not completely independent of each other [22], we fixed C to 0.01 and tuned ρ. If not stated otherwise, the following experiments adopt these default settings.

TABLE 2
Effects of Feature Functions (Percent)

Features	AR	CR	Chinese	Symbol	Digit	Letter
cu	24.09	63.25	65.42	44.26	67.17	64.10
cu+g	85.92	87.93	89.38	80.12	76.49	67.42
cu+cti	90.91	92.12	93.68	80.68	90.09	82.71
cu+g+cti	92.88	93.66	95.31	81.40	91.79	85.24
cu+cs+g+cbi	92.84	93.55	94.84	84.48	91.26	84.84
cu+cs+g+cti	93.66	94.26	95.51	85.64	91.72	87.23
cu	65.61	80.74	85.67	74.04	94.13	64.00
cu+g	84.02	86.38	93.90	80.63	91.65	68.18
cu+cti	91.21	93.04	97.77	90.14	91.27	91.63
cu+g+cti	91.56	93.12	97.36	89.52	93.98	93.42
cu+cs+g+cbi	92.65	93.97	97.57	91.01	95.56	90.31
cu+cs+g+cti	93.58	94.62	97.95	92.04	95.06	93.12

The right columns show the CRs for different character types. Upper: CASIA-OLHWDB; lower: TUAT Kondate.

7.3.1 Effects of Feature Functions

To evaluate the feature functions, the margin term was dropped in training. The effects of different combinations of feature functions are shown in Table 2, where "cu" and "cs" denote class-unspecific and class-specific, respectively, both are for character recognition (cf., Section 6.1). "g" denotes geometric feature functions, "cbi" denotes character bigram, and "cti" denotes character trigram. We can see that when using the class-unspecific feature function ("cu") only, the performance is remarkably improved by combining geometric models ("$\mathrm{cu} + \mathrm{g}$"). The incorporation of language model (LM) ("$\mathrm{cu} + \mathrm{cti}$") is much more effective than the geometric model. Better results are given by combining "cu" with both geometric and language models, and the best result is given by further combining class-specific feature functions ("cs"). As expected, when combining with the same feature functions "$\mathrm{cu} + \mathrm{cs} + \mathrm{g}$", the trigram LM ("cti") outperforms the bigram LM ("cbi") by capturing long-range dependences. Nevertheless, when the trigram LM is used without "cs," its performance is comparable or inferior to that of bigram combined with "cs." This again justifies the benefit of class-specific feature functions.

7.3.2 Effects of Margin Term

To evaluate the effects of the margin term in training, we first set the margin scale ρ to the default value and compared the recognition results with and without the margin term, where $m = 3$ and all the feature functions are used. Table 3

TABLE 3
Results (Percent) with and without (w/o) the Margin Term in Training

Margin	AR	CR	Chinese	Symbol	Digit	Letter
w/o	93.66	94.26	95.51	85.64	91.72	87.23
with	93.75	94.34	95.58	85.67	92.25	87.37
w/o	93.58	94.62	97.95	92.04	95.06	93.12
with	94.08	95.06	98.22	92.51	95.71	93.90

Upper: CASIA-OLHWDB; lower: TUAT Kondate.

shows that margin-based training can yield improvement of the recognition performance, especially for the TUAT Kondate database, which has less training samples. For the CASIA-OLHWDB with larger training set, the gain of margin-based training is limited. This result is in agreement with the conclusion drawn in [22], that for large vocabulary tasks, the margin term is less important if overfitting is not so severe. Fig. 6a shows that the accuracies are insensitive to the margin scale provided ρ is not too small.

7.3.3 Effects of Lattice Pruning in Training

As mentioned in Section 3.4, the time complexity of inference increases exponentially with m. When $m = 3$, the training is computationally intensive on long string samples, such as those in the CASIA-OLHWDB database with more than 20 characters per string. To accelerate training, we resorted to the forward-backward lattice pruning method, which reduces the lattice complexity while reserving the most rival paths using a pretrained first-order semi-CRF.

Fig. 6b shows the effects of different pruning threshold γ_p on test string sets, where $m = 3$ and the initial LED on training set is 83.49 for CASIA-OLHWDB. We can see that the default value $\gamma_p = 12$ performs sufficiently well in respect of the recognition accuracies. Increasing γ_p, though incorporates more rival paths, does not improve the performance. For CASIA-OLHWDB, with the default γ_p, the corresponding LED on training set is 3.06 and the training time is 2.35 s/str (averaged over the training string number times the iteration number), while the time cost without lattice pruning is over 175.69 s/str (calculated on 1,000 samples). This justifies the great benefit of lattice pruning by reducing the LED.

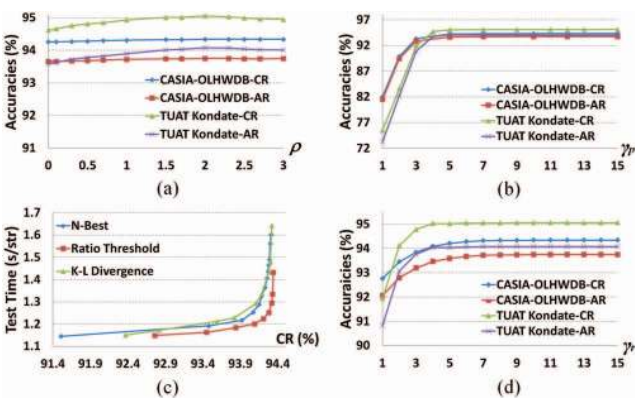


Fig. 6. (a) Effects of the margin scale. (b) Effects of lattice pruning in training. (c) Decoding time versus CRs for three beam search techniques. (d) Effects of pruning threshold for ratio threshold-based beam search.

TABLE 4
Effects of Different Character Classification Methods (Percent)

Classifier	Top1	Top10	AR	CR	Chinese	Symbol	Digit	Letter
FDA+MQDF	85.08	97.75	93.75	94.34	95.58	85.67	92.25	87.37
DFE+MQDF	86.09	98.04	94.36	94.89	96.22	85.57	92.62	86.97
DFE+DLQDF	91.19	98.71	94.54	95.20	96.50	85.96	93.28	86.97
FDA+MQDF	86.22	98.19	94.08	95.06	98.22	92.51	95.71	93.90
DFE+MQDF	86.82	98.38	94.55	95.44	98.50	92.91	96.30	94.08
DFE+DLQDF	87.29	97.68	93.15	94.02	97.97	91.28	96.19	83.85

Upper: CASIA-OLHWDB; lower: TUAT Kondate.

7.3.4 Comparison of Decoding Methods

The parameters (γ_b, γ_r, and γ_k) of three beam search techniques are used to control the beam width. With increased beam width, beam search approximates the Viterbi-like decoding asymptotically. Fig. 6c plots the decoding time versus the CR on the CASIA-OLHWDB test strings by varying the three parameters, where $m = 3$. We can see that with comparable accuracies, ratio threshold-based beam search runs faster than the other two techniques. Fig. 6d shows the accuracies over γ_r on test strings. It can be seen that the default value $\gamma_r = 10$ performs sufficiently well with respect to the recognition accuracies (the corresponding decoding time is 1.43 s/string for CASIA-OLHWDB and 1.06 s/string for TUAT Kondate, respectively). Increasing γ_r, though improves the coverage of correct path, does not improve the performance.

7.3.5 Comparison of Character Classifiers

In the proposed model, character classification is used in both candidate class selection and feature functions. Before character classification, the feature dimensionality was first reduced to 160D (cf., Section 6.1). To achieve higher string recognition performance, we consider different dimensionality reduction and classification methods: discriminative feature extraction (DFE) [63] and discriminative learning QDF (DLQDF) [64] in addition to the baseline FDA and MQDF. DFE optimizes the feature subspace (initialized by FDA) under a discriminative learning objective, and DLQDF is a discriminatively updated version of MQDF [64]. The training set for DFE and DLQDF is the same as that for FDA and MQDF.

Table 4 shows the recognition results on test string sets using different combinations of dimensionality reduction (FDA, DFE) and classification (MQDF, DLQDF), where trigram language models ($m = 3$) and margin-based training are employed. For segmented character recognition, the "Top1" row gives the CRs, and "Top10" gives the cumulative accuracies of top 10 ranks (the default candidate class number J in decoding is 10). From Table 4 we can see that DFE and DLQDF generally achieves higher CRs ("Top1") than FDA and MQDF due to discriminative learning. However, the string recognition performance (AR and CR) owes much to the cumulative accuracy ("Top10") rather than the top-1 accuracy, which is the reason that on CASIA-OLHWDB, superior string recognition rates are obtained by $\mathrm{DFE} + \mathrm{DLQDF}$, while on TUAT Kondate, $\mathrm{DFE} + \mathrm{MQDF}$ performs better.

7.4 Error Analysis

We analyze recognition errors under the default setting that uses $\mathrm{FDA} + \mathrm{MQDF}$ for character classification. Table 5 lists the error rates on the test string sets when $m = 3$. Note that LER is a lower bound of CER. To investigate the effects of

TABLE 5
LERs and CERs (Percent) on Test Strings

LER	CER	SUB	DEL	INS
2.51	6.25	4.77	0.89	0.59
0	6.19	4.82	0.75	0.62
1.03	5.92	4.07	0.88	0.98
0	5.89	4.03	0.79	1.07

$LER = 0$ *means decoding on the perfect lattice.*
Upper: CASIA-OLHWDB; lower: TUAT Kondate.

lattice errors, we also gave the results by decoding on the perfect lattices, i.e., the correct path was inserted when constructing the lattice such that $\mathrm{LER} = 0$, and all the errors were caused by the path evaluation criterion (the energy defined in (5) can be taken as a path evaluation function). From Table 5, we can see that the CER is reduced just slightly on perfect lattices. Among the three types of character errors (substitution, deletion and insertion), the substitution errors are dominating. Lattice errors owe much to the performance of the character classifier (the true class label is not included in the candidate classes output by the character recognizer). Except for the lattice errors, another part of character errors are caused by the imperfection of the path evaluation function.

7.5 Comparison with Previous Methods

The negative energy in (5) can be taken as a scoring function for candidate segmentation-recognition paths. For Chinese handwritten text recognition, we first compare with two path evaluation functions proposed in [11], which are the linear combinations of weighted character recognition scores, geometric and linguistic contexts on each path. In the first function, the transformed classification confidence score of each character is weighted with the constituent component number (WSN), and in the second function, the classification score is weighted with the normalized character width (WCW). The results are listed in Table 6, where the trigram language model was used. For fair comparison, the parameters of WSN and WCN were learned by minimizing the NLL loss incorporating the margin term, and the ratio threshold based beam search was used for decoding. From the results, we can see that the proposed method performs better than the best path evaluation functions in [11]. The improvement owes much to the class-specific feature functions, without which the negative energy defined in (5) is similar to the first path evaluation function (WSN).

To compare with the method of Zhu et al. [12] on Japanese character string recognition, which uses the same TUAT Kondate database as in our experiments, we calculate the segmentation measure F and the character recognition rate *crr* defined in [16] (the percentage of characters which are both segmented and recognized correctly). The results are listed in Table 7, where trigram language models ($m = 3$) were used. It can be seen that though the proposed method has lower segmentation accuracy than the method of Zhu et al., it gives much higher character recognition rate. The improvement owes much to the class-specific features and the margin-based training.

TABLE 6
Comparison with the Methods of Wang et al. [11] on CASIA-OLHWDB Database (Percent)

Method	AR	CR	Chinese	Symbol	Digit	Letter
WSN	92.99	93.71	95.35	81.65	91.24	85.51
WCW	92.69	93.13	95.15	78.31	88.98	81.91
Proposed	93.75	94.34	95.58	85.67	92.25	87.37

TABLE 7
Comparison with the Method of Zhu et al. [12] on TUAT Kondate Database

Method	F	*crr* (%)
Zhu et al.	0.9934	92.80
Proposed	0.9866	93.98

Finally, we compare with the best results achieved by Vision Objects Ltd. (VO) in ICDAR 2011 Chinese handwriting recognition competition [10] on the same test data of online handwritten texts. The test data of 60 writers (3,432 text lines, including 91,576 characters of 1,375 classes) is provided by the same group as the CASIA-OLHWDB database [60]. Three semi-CRF-based string recognition models using different character classification methods (cf., Section 7.3.5) are compared with the system of VO, which adopts multilayer perceptron (MLP) as the character classifier. As recommended by the competition, all the three models are trained with the samples of the whole CASIA-OLHWDB database. From Table 8, we can see that even with $\mathrm{DFE} + \mathrm{MQDF}$ and a relatively weaker linguistic model (LM), the achieved AR is already higher than that of VO. With $\mathrm{DFE} + \mathrm{DLQDF}$, the proposed method achieved higher string recognition performance in respect of both AR and CR.

8 Conclusion

We proposed a semi-CRF-based HCTR method, which evaluates the candidate segmentation-recognition paths by fusing the scores of character recognition, linguistic and geometric contexts in a principled MAP framework. Experimental results on the unconstrained Chinese handwriting database CASIA-OLHWDB and the Japanese TUAT Kondate database justify the effectiveness of each feature function, especially the class-specific feature function. Incorporating a margin term in training was shown to benefit the recognition performance, and the proposed forward-backward lattice pruning algorithm significantly speeds up training. To reduce the decoding time, we tested three beam search techniques and the ratio threshold-based technique was shown to be the most effective. Our string recognition results on three test sets are superior to those of other state-of-the-art methods. The error analysis reveals

TABLE 8
Comparison with the Best Results of ICDAR 2011 Chinese Handwriting Competition (Online Handwritten Texts)

System	Classifier	LM	AR (%)	CR (%)
Proposed	FDA+MQDF	char tri-gram	93.17	93.69
Proposed	DFE+MQDF	char tri-gram	93.76	94.21
Proposed	DFE+DLQDF	char tri-gram	94.06	94.62
VO	MLP	word tri-gram	93.56	94.33

that there remains a room for further improvement by upgrading the character classification accuracy and the path evaluation function. CRF is a principled framework for integrating multi-source contextual information in path evaluation. Higher order feature functions, such as high-order language models, can be exploited in the future work.

ACKNOWLEDGMENTS

This work was supported by the National Natural Science Foundation of China under grants nos. 61273269, 60933010, 61232013, and 61170182. The authors would like to thank Qiu-Feng Wang, Yan-Ming Zhang, and Liang Huang for valuable discussions.

REFERENCES

[1] C.L. Liu, S. Jaeger, and M. Nakagawa, "Online Recognition of Chinese Characters: The State-of-the-Art," *IEEE Trans. Pattern Analysis and Machine Intelligence,* vol. 26, no. 2, pp. 198-213, Feb. 2004.

[2] Y. Hotta, H. Takebe, M. Suwa, and S. Naoi, "Accuracy Improvement for Handwritten Japanese Word Recognition by Combination of Character and Word Recognizer," *Proc. Eighth Int'l Conf. Document Analysis and Recognition,* pp. 685-689, 2005.

[3] Z.B Yao, X.Q Ding, and C.S Liu, "On-Line Handwritten Chinese Word Recognition Based on Lexicon," *Proc. 18th Int'l Conf. Pattern Recognition,* pp. 320-323, 2006.

[4] T. Long and L.W. Jin, "A Novel Orientation Free Method for Online Unconstrained Cursive Handwritten Chinese Word Recognition," *Proc. 19th Int'l Conf. Pattern Recognition,* pp. 1-4, 2008.

[5] B.L. Zhu and M. Nakagawa, "Trie-Lexicon-Driven Recognition for On-Line Handwritten Japanese Disease Names Using a Time-Synchronous Method," *Proc. 11th Int'l Conf. Document Analysis and Recognition,* pp. 1130-1134, 2011.

[6] H.S. Tang, E. Augustin, C.Y. Suen, O. Baret, and M. Cheriet, "Spiral Recognition Methodology and Its Application for Recognition of Chinese Bank Checks," *Proc. Ninth Int'l Workshop Frontiers in Handwriting Recognition,* pp. 263-268, 2004.

[7] C.L. Liu, M. Koga, and H. Fujisawa, "Lexicon-Driven Segmentation and Recognition of Handwritten Character Strings for Japanese Address Reading," *IEEE Trans. Pattern Analysis and Machine Intelligence,* vol. 24, no. 11, pp. 1425-1437, Nov. 2002.

[8] C.H. Wang, Y. Hotta, M. Suwa, and S. Naoi, "Handwritten Chinese Address Recognition," *Proc. Ninth Int'l Workshop Frontiers in Handwriting Recognition,* pp. 539-544, 2004.

[9] Q. Fu, X.Q. Ding, T. Liu, Y. Jiang, and Z. Ren, "A Novel Segmentation and Recognition Algorithm for Chinese Handwritten Address Character Strings," *Proc. 18th Int'l Conf. Pattern Recognition,* pp. 974-977, 2006.

[10] C.L. Liu, F. Yin, Q.F. Wang, and D.H. Wang, "ICDAR 2011 Chinese Handwriting Recognition Competition," *Proc. 11th Int'l Conf. Document Analysis and Recognition,* pp. 1464-1469, 2011.

[11] Q.F. Wang, F. Yin, and C.L. Liu, "Handwritten Chinese Text Recognition by Integrating Multiple Contexts," *IEEE Trans. Pattern Analysis and Machine Intelligence,* vol. 34, no. 8, pp. 1469-1481, Aug. 2012.

[12] 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,* vol. 13, no. 2, pp. 121-131, 2010.

[13] X.D. Zhou, C.L. Liu, and M. Nakagawa, "Online Handwritten Japanese Character String Recognition Using Conditional Random Fields," *Proc. 10th Int'l Conf. Document Analysis and Recognition,* pp. 521-525, 2009.

[14] M. Cheriet, N. Kharma, C.L. Liu, and C.Y. Suen, *Character Recognition Systems: A Guide for Students and Practitioners.* John Wiley & Sons, Inc., 2007.

[15] S. Senda and K. Yamada, "A Maximum-Likelihood Approach to Segmentation-Based Recognition of Unconstrained Handwriting Text," *Proc. Sixth Int'l Conf. Document Analysis and Recognition,* pp. 184-188, 2001.

[16] M. Nakagawa, B. Zhu, and M. Onuma, "A Model of On-Line Handwritten Japanese Text Recognition Free from Line Direction and Writing Format Constraints," *IEICE Trans. Information and Systems,* vol. E88-D, no. 8, pp. 1815-1822, 2005.

[17] X.D. Zhou, J.L. Yu, C.L. Liu, T. Nagasaki, and K. Marukawa, "Online Handwritten Japanese Character String Recognition Incorporating Geometric Context," *Proc. Ninth Int'l Conf. Document Analysis and Recognition,* pp. 48-52, 2007.

[18] Q.F. Wang, F. Yin, and C.L. Liu, "Integrating Language Model in Handwritten Chinese Text Recognition," *Proc. 10th Int'l Conf. Document Analysis and Recognition,* pp. 1036-1040, 2009.

[19] N.X. Li and L.W. Jin, "A Bayesian-Based Method of Unconstrained Handwritten Offline Chinese Text Line Recognition," *Int'l J. Document Analysis and Recognition,* vol. 16, no. 1, pp. 17-31, Mar. 2013.

[20] S. Sarawagi and W. Cohen, "Semi-Markov Conditional Random Fields for Information Extraction," *Neural Information Processing Systems,* vol. 17, pp. 1185-1192, 2005.

[21] G. Heigold, T. Deselaers, R. Schlüter, and H. Ney, "Modified MMI/MPE: A Direct Evaluation of the Margin in Speech Recognition," *Proc. 25th Int'l Conf. Machine Learning,* pp. 384-391, 2008.

[22] G. Heigold, P. Dreuw, S. Hahn, R. Schlüter, and H. Ney, "Margin-Based Discriminative Training for String Recognition," *IEEE J. Selected Topics in Signal Processing—Statistical Learning Methods for Speech and Language Processing,* vol. 4, no. 6, pp. 917-925, Dec. 2010.

[23] D. Povey, D. Kanevsky, B. Kingsbury, B. Ramabhadran, G. Saon, and K. Visweswariah, "Boosted MMI for Model and Feature-Space Discriminative Training," *Proc. IEEE Int'l Conf. Acoustics, Speech, and Signal Processing,* pp. 4057-4060, 2008.

[24] L.Y. Tseng and R.C. Chen, "Segmentation Handwritten Chinese Characters Based on Heuristic Merging of Stroke Bounding Boxes and Dynamic Programming," *Pattern Recognition Letters,* vol. 19, no 10, pp. 963-973, 1998.

[25] Y. Lu, C.L. Tan, P.F. Shi, and K.H. Zhang, "Segmentation of Handwritten Chinese Characters from Destination Addresses of Mail Pieces," *Int'l J. Pattern Recognition and Artificial Intelligence,* vol. 16, no. 1, pp. 85-96, 2002.

[26] S. Zhao, Z. Chi, P. Shi, and H. Yan, "Two-Stage Segmentation of Unconstrained Handwritten Chinese Characters," *Pattern Recognition,* vol. 36, no 1, pp. 145-156, 2003.

[27] X. Wei, S. Ma, and Y. Jin, "Segmentation of Connected Chinese Characters Based on Genetic Algorithm," *Proc. Eighth Int'l Conf. Document Analysis and Recognition,* pp. 645-649, 2005.

[28] Z. Liang and P. Shi, "A Metasynthetic Approach for Segmenting Handwritten Chinese Character Strings," *Pattern Recognition Letters,* vol. 26, no. 10, pp. 1498-1511, 2005.

[29] C.L. Liu, H. Sako, and H. Fujisawa, "Effects of Classifier Structures and Training Regimes on Integrated Segmentation and Recognition of Handwritten Numeral Strings," *IEEE Trans. Pattern Analysis and Machine Intelligence,* vol. 26, no. 11, pp. 1395-1407, Nov. 2004.

[30] T.H. Su, T.W. Zhang, D.J. Guan, and H.J. Huang, "Off-Line Recognition of Realistic Chinese Handwriting Using Segmentation-Free Strategy," *Pattern Recognition,* vol. 42, no. 1, pp. 167-182, 2009.

[31] Z.W. Jiang, X.Q. Ding, C.S. Liu, and Y.W. Wang, "A Novel Short Merged Off-Line Handwritten Chinese Character String Segmentation Algorithm Using Hidden Markov Model," *Proc. 11th Int'l Conf. Document Analysis and Recognition,* pp. 668-672, 2011.

[32] H. Murase, "Online Recognition of Free-Format Japanese Handwritings," *Proc. Ninth Int'l Conf. Pattern Recognition,* pp. 1143-1147, 1988.

[33] X. Gao, P.M. Lallican, and C. Viard-Gaudin, "A Two-Stage Online Handwritten Chinese Character Segmentation Algorithm Based on Dynamic Programming," *Proc. Eighth Int'l Conf. Document Analysis and Recognition,* pp. 735-739, 2005.

[34] S. Tulyakov and V. Govindaraju, "Probabilistic Model for Segmentation Based Word Recognition with Lexicon," *Proc. Sixth Int'l Conf. Document Analysis and Recognition,* pp. 164-167, 2001.

[35] J. Lafferty, A. McCallum, and F. Pereira, "Conditional Random Fields: Probabilistic Models for Segmenting and Labeling Sequence Data," *Proc. 18th Int'l Conf. Machine Learning,* pp. 282-289, 2001.

[36] T.M.T. Do and T. Artières, "Conditional Random Fields for Online Handwriting Recognition," *Proc. 10th Int'l Workshop Frontiers in Handwriting Recognition,* pp. 197-202, 2006.

[37] S. Feng, R. Manmatha, and A. McCallum, "Exploring the Use of Conditional Random Field Models and HMMs for Historical Handwritten Document Recognition," *Proc. Second Int'l Conf. Document Image Analysis for Libraries,* pp. 30-37, 2006.

[38] F. Kimura, K. Takashina, S. Tsuruoka, and Y. Miyake, "Modified Quadratic Discriminant Functions and Its Application to Chinese Character Recognition," *IEEE Trans. Pattern Analysis and Machine Intelligence,* vol. 9, no. 1, pp. 149-153, Jan. 1987.

[39] A. Graves, M. Liwicki, S. Fernández, R. Bertolami, H. Bunke, and J. Schmidhuber, "A Novel Connectionist System for Unconstrained Handwriting Recognition," *IEEE Trans. Pattern Analysis and Machine Intelligence,* vol. 31, no. 5, pp. 855-868, May 2009.

[40] S. Shetty, H. Srinivasan, M. Beal, and S.N. Srihari, "Segmentation and Labeling of Documents Using Conditional Random Fields," *Proc. SPIE Document Recognition and Retrieval XIV,* pp. 6500U-1-11, 2007.

[41] C. Pal, C. Sutton, and A. McCallum, "Sparse Forward-Backward Using Minimum Divergence Beams for Fast Training of Conditional Random Fields," *Proc. Int'l Conf. Acoustics, Speech, and Signal Processing,* pp. v581-v584, 2006.

[42] T. Cohn, "Efficient Inference in Large Conditional Random Fields," *Proc. 17th European Conf. Machine Learning,* pp. 606-613, 2006.

[43] M. Jeong, C.Y. Lin, and G.G. Lee, "Efficient Inference of CRFs for Large-Scale Natural Language Data," *Proc. Conf. Short Papers ACL-IJCNLP,* pp. 281-284, 2009.

[44] D. Okanohara, Y. Miyao, Y. Tsuruoka, and J. Tsujii, "Improving the Scalability of Semi-Markov Conditional Random Fields for Named Entity Recognition," *Proc. 21st Int'l Conf. Computational Linguistics and 44th Ann. Meeting of the Assoc. for Computational Linguistics,* pp. 465-472, 2006.

[45] N. Ye, W.S. Lee, H.L. Chieu, and D. Wu, "Conditional Random Fields with High-Order Features for Sequence Labeling," *Proc. 22nd Ann. Conf. Neural Information Processing Systems,* pp. 1393-1400, 2009.

[46] X. Qian, X. Jiang, Q. Zhang, X. Huang, and L. Wu, "Sparse Higher Order Conditional Random Fields for Improved Sequence Labeling," *Proc. 26th Ann. Int'l Conf. Machine Learning,* pp. 849-856, 2009.

[47] V.C. Nguyen, N. Ye, W.S. Lee, and H.L. Chieu, "Semi-Markov Conditional Random Field with High-Order Features," *Proc. 28th Int'l Conf. Machine Learning,* 2011.

[48] D. Yu and L. Deng, "Large-Margin Discriminative Training of Hidden Markov Models for Speech Recognition," *Proc. First Int'l Conf. Semantic Computing,* pp. 429-438, 2007.

[49] T.M.T. Do and T. Artières, "Maximum Margin Training of Gaussian HMMs for Handwriting Recognition," *Proc. 10th Int'l Conf. Document Analysis and Recognition,* pp. 976-980, 2009.

[50] L.R. Bahl, P.F. Brown, P.V. DeSouza, and R.L. Mercer, "Maximum Mutual Information Estimation of Hidden Markov Model Parameters for Speech Recognition," *Proc. IEEE Int'l Conf. Acoustics, Speech, and Signal Processing,* pp. 49-52, 1986.

[51] B.H. Juang and S. Katagiri, "Discriminative Learning for Minimum Error Classification," *IEEE Trans. Signal Processing,* vol. 40, no 12, pp. 3043-3054, Dec. 1992.

[52] D. Povey, "Discriminative Training for Large Vocabulary Speech Recognition," PhD dissertation, Cambridge Univ., 2003.

[53] M.Y. Kim, "Large Margin Cost-Sensitive Learning of Conditional Random Fields," *Pattern Recognition,* vol. 43, no. 10, pp. 3683-3692, 2010.

[54] A. Sixtus and S. Ortmanns, "High Quality Word Graphs Using Forward-Backward Pruning," *Proc. IEEE Int'l Conf. Acoustics, Speech, and Signal Processing,* vol. 2, pp. 593-596, 1999.

[55] C.L. Liu and X.D. Zhou, "Online Japanese Character Recognition Using Trajectory-Based Normalization and Direction Feature Extraction," *Proc. 10th Int'l Workshop Frontiers in Handwriting Recognition,* pp. 217-222, 2006.

[56] C.L. Liu, "Classifier Combination Based on Confidence Transformation," *Pattern Recognition,* vol. 38, no. 1, pp. 11-28, 2005.

[57] C. Sutton and A. McCallum, "An Introduction to Conditional Random Fields for Relational Learning," *Introduction to Statistical Relational Learning,* L. Getoor and B. Taskar, eds., MIT Press, 2006.

[58] A. Vinciarelli, S. Bengio, and H. Bunke, "Offline Recognition of Unconstrained Handwritten Texts Using HMMs and Statistical Language Models," *IEEE Trans. Pattern Analysis and Machine Intelligence,* vol. 26, no 6, pp. 709-720, June 2004.

[59] A. Stolcke, "SRILM—An Extensible Language Modeling Toolkit," *Proc. Seventh Int'l Conf. Spoken Language Processing,* pp. 901-904, 2002.

[60] C.L. Liu, F. Yin, D.H. Wang, and Q.F. Wang, "CASIA Online and Offline Chinese Handwriting Databases," *Proc. 11th Int'l Conf. Document Analysis and Recognition,* pp. 37-41, 2011.

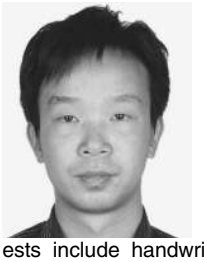
[61] K. Matsumoto, T. Fukushima, and M. Nakagawa, "Collection and Analysis of On-Line Handwritten Japanese Character Patterns," *Proc. Sixth Int'l Conf. Document Analysis and Recognition,* pp. 496-500, 2001.

[62] S. Ortmanns, H. Ney, and X. Aubert, "A Word Graph Algorithm for Large Vocabulary Continuous Speech Recognition," *Computer Speech Language,* vol. 11, pp. 43-72, 1997.

[63] C.L. Liu, R. Mine, and M. Koga, "Building Compact Classifier for Large Character Set Recognition Using Discriminative Feature Extraction," *Proc. Eight Int'l Conf. Document Analysis and Recognition,* pp. 846-850, 2005.

[64] C.L. Liu, "High Accuracy Handwritten Chinese Character Recognition Using Quadratic Classifiers with Discriminative Feature Extraction," *Proc. 18th Int'l Conf. Pattern Recognition,* pp. 942-945, 2006.

Xiang-Dong Zhou received the BS degree in applied mathematics, the MS degree in management science and engineering from National University of Defense Technology, Changsha, China, and the PhD degree in pattern recognition and artificial intelligence from the Institute of Automation, Chinese Academy of Sciences, Beijing, China, in 1998, 2003, and 2009, respectively. He was a postdoctoral fellow at Tokyo University of Agriculture and Technology from March 2009 to March 2011. From 2011, he has been a research assistant at the Institute of Software, Chinese Academy of Sciences, Beijing, China. His research interests include handwriting recognition and ink document analysis.



Da-Han Wang received the BS degree in automation science and electrical engineering from Beihang University, Beijing, China, in 2006 and the PhD degree in pattern recognition and intelligent systems from the Institute of Automation, Chinese Academy of Sciences, Beijing, China, in 2012. He is now a postdoctoral research fellow at the School of Information Science and Engineering, Xiamen University, Fujian, China. His research interests include handwriting recognition, text detection and recognition, and computer vision.

Feng Tian received the PhD degree in the Institute of Software, Chinese Academy of Sciences in 2003. He is a professor in the Institute of Software, Chinese Academy of Sciences where he manages the pen-based and multimodal user interface research group in Intelligence Engineering Lab. His research interests include theories, interaction techniques and tools in natural user interface, pen-based UI, multimodal UI, and other new UI styles. He has published more than 60 papers in HCI field, including ACM CHI, ACM CSCW, ACM IUI, ACM TIST, and so on. He also serves as the chair of ACM SIGCHI China chapter now.

Cheng-Lin Liu received the BS degree in electronic engineering from Wuhan University, China, the ME degree in electronic engineering from Beijing Polytechnic University, China, and the PhD degree in pattern recognition and intelligent control from the Chinese Academy of Sciences, Beijing, China, in 1989, 1992, and 1995, respectively. He is a professor at the National Laboratory of Pattern Recognition, Institute of Automation of Chinese Academy of Sciences, Beijing, China, and is now the deputy director of the laboratory. He was a postdoctoral fellow at Korea Advanced Institute of Science and Technology and later at Tokyo University of Agriculture and Technology from March 1996 to March 1999. From 1999 to 2004, he was a research staff member and later a senior researcher at the Central Research Laboratory, Hitachi, Ltd., Tokyo, Japan. His research interests include pattern recognition, image processing, neural networks, machine learning, and especially the applications to character recognition and document analysis. He has published more than 170 technical papers at prestigious international journals and conferences. He is on the editorial board of journals *Pattern Recognition*, *Image and Vision Computing*, and *International Journal on Document Analysis and Recognition*. He is a fellow of the IAPR, and a senior member of the IEEE.

Masaki Nakagawa received the BSc and MSc degrees from the University of Tokyo, in 1977 and 1979, respectively. During the academic year 1977-1978, he followed the computer science course at Essex University in the United Kingdom and received the MSc degree with distinction in computer studies in July 1979. He received the PhD degree in information science from the University of Tokyo in December 1988. Since April 1979, he has been working at the Tokyo University of Agriculture and Technology. Currently, he is a professor of media interaction in the Department of Computer and Information Sciences. For the past 10 years, he has been collaborating with industry to advance pen-based human interactions composed of online handwriting recognition, pen-based interfaces, and applications, especially educational applications on an interactive electronic whiteboard. He has been serving on several committees of the Japanese government on Industry and Universities partnership and those on IT-oriented and IT-supported learning. He is a fellow of the IAPR, and a member of the IEEE and the IEEE Computer Society.