

Combining Character-Based Bigrams with Word-Based Bigrams in Contextual Postprocessing for Chinese Script Recognition

YUANXIANG LI

National University of Singapore

XIAOQING DING

Tsinghua University

and

CHEW LIM TAN

National University of Singapore

It is crucial to use contextual information to improve the recognition accuracy of Chinese script in an offline, handwritten Chinese character-recognition system. However, with the increase in the number of candidates given by a character recognizer, contextual postprocessing using a word-based bigram is time-consuming. This article presents a novel contextual postprocessing method that integrates character-based bigram postprocessing with word-based bigram postprocessing in light of the complementary action between Chinese characters and Chinese words. On the basis of isolated character recognition, character-based bigram postprocessing using a *forward-backward* search is first executed on a big candidate set, which improves both the accuracy and efficiency of the candidate set (the cumulative accuracy of the top ten candidates is greatly boosted). Then, to further improve accuracy, word-based bigram postprocessing (WBP) is executed on a small candidate set. This method obtains high accuracy while paying attention to postprocessing speed at the same time. Experimental results for three Chinese scripts (about 66,000 characters in total) demonstrate the effectiveness of our method: character-based bigram postprocessing improves accuracy from 81.58% to 94.50%, and the cumulative accuracy of the top ten candidates rises from 94.33% to 98.25%. After WBP, 95.75% accuracy is achieved, which is equivalent to the accuracy of WBP executed on a big candidate set. However, our method is more than 100 times faster than that of WBP.

Categories and Subject Descriptors: I.2.7 [Artificial Intelligence]: Natural Language Processing—*Language models*; I.2.8 [Artificial Intelligence]: Problem Solving, Control Methods, and Search—*Graph and tree search strategies* I.5.4 [Pattern Recognition]: Application—*Text processing*; I.7.5 [Document and Text Processing]: Document Capture—*Optical character recognition (OCR)*

General Terms: Algorithms, Languages, Performance

Additional Key Words and Phrases: Chinese character recognition, statistical language model, contextual post-processing, forward-backward search, efficiency of candidate set

1. INTRODUCTION

Recognizing handwritten Chinese characters is a challenging pattern recognition problem [Chen 1997]. Due to the large character set and wide variations in writing style, it is

This research was supported in part by National Science Foundation (grant 69972024) and National “863” Hightech Research and Development Plan (grant 863-306-ZT03-03-1) in P. R. China.

Authors' addresses: Yuanxiang Li and Chew Lim Tan, School of Computing, National University of Singapore, Kent Ridge 117543, Singapore; email: liyx@comp.nus.edu.sg, tancl@comp.nus.edu.sg; Xiaoqing Ding, State Key Lab. of Intelligent Technology and Systems, Department of Electronic Engineering, Tsinghua University, Beijing 100084, P. R. China; email: dingxq@mail.tsinghua.edu.cn.

Permission to make digital/hard copy of part of this work for personal or classroom use is granted without fee provided that the copies are not made or distributed for profit or commercial advantage, the copyright notice, the title of the publication, and its date of appear, and notice is given that copying is by permission of the ACM, Inc. To copy otherwise, to republish, to post on servers, or to redistribute to lists, requires prior specific permission and/or a fee.

© 2003 ACM 1073-0516/02/1200-0297 \$5.00

difficult to significantly improve the recognition accuracy of Chinese script in an offline, handwritten, isolated Chinese character-recognition system. Statistical language models (SLMs) have been used successfully for contextual postprocessing in order to increase accuracy in the recognition of Chinese script [Chang 1996; Lee and Tung 1997; Tung and Lee 1994; Wong and Chan 1999; Xia et al. 1996]. The technology of contextual postprocessing can be described as follows: Under the joint action of SLMs and candidate confidence, an efficient search strategy (such as the well-known Viterbi algorithm) is employed to select the most likely sentence from the candidate sets given by a character recognizer. In the Chinese language, a word is the basic syntax-meaningful unit; however, each character itself also has a definite meaning. Hence, a Chinese language model can be based on both words and characters. In word-based contextual postprocessing, the candidate set with m candidates should be small, since a large m may not only result in time-consuming processing, but also greatly increase the number of coincidental word formations, and thus produce bad results. In previous work [Wong and Chan 1999; Chang 1996], m was often chosen experimentally to be 6. On the other hand, in character-based contextual postprocessing, m was usually chosen to be 10 [Xia et al. 1996; Li et al. 1999].

In fact, whether the candidate set includes the correct result (true candidate) or not is crucial for improving accuracy in script recognition. As is well known, if there is no correct result in the candidate set, it is impossible to correct errors in the recognizer, no matter how precise the SLMs. Apparently, it is not beneficial to improve the postprocessing performance if m is fixed. On the one hand, for well-recognized scripts, m need not be large. On the other hand, m must be large enough to include true candidate if scripts are poorly recognized. However, as m increases, the cost of postprocessing also increases. In our previous work [Li and Ding 2001], we demonstrated that the accuracy of word-based bigram postprocessing is higher than that of character-based bigram postprocessing, while the latter's speed is much faster than the former's. A question may naturally follow: Is it possible to integrate word-based bigram postprocessing with character-based bigram postprocessing, so that we can get high accuracy while paying attention to postprocessing speed at the same time?

In the Chinese language, characters and words lie at different levels. To some extent, there is a complementary action between Chinese characters and Chinese words, though Chinese words are composed of Chinese characters. Combining Chinese words with Chinese characters would reduce the uncertainty in Chinese sentences [Liu 2000]. Based on this notion, we attempt to combine Chinese words with Chinese characters for contextual postprocessing.

The remainder of this article is organized as follows. In Section 2 we propose the contextual postprocessing model that integrates Chinese characters with Chinese words. This kind of postprocessing integration can be divided into two levels: character-based postprocessing and word-based postprocessing. In Section 3, the forward-backward search algorithm is introduced to character-based bigram postprocessing, which is mathematically related to the Baum-Welch forward-backward training algorithm [Rabiner 1989]. Section 4 describes the postprocessing method that integrates character-based bigrams with word-based bigrams. Section 5 shows the experimental results, and Section 6 concludes.

2. CONTEXTUAL POSTPROCESSING MODEL INTEGRATING CHARACTERS AND WORDS

For a Chinese script recognition system, let S be a sentence including T characters (T words). S is represented as both

$$S_c = c_1 c_2 \wedge c_T, \text{ called a character string,}$$

and

$$S_w = w_1 w_2 \wedge w_{T'}, \text{ called a word string.}$$

$$X = x_1 x_2 \wedge x_T$$

be the feature sequence of character images in accordance with the character string S_c . Given X , the optimal sentence \hat{S} is the one that maximizes the *a posteriori* probability of the intersection between S_c and S_w . That is,

$$\hat{S} = \arg \max_{S_c S_w} P(S_c S_w | X) = \arg \max_{S_c S_w} P(S_c S_w) P(X | S_c S_w) \quad (1)$$

Here, since S_c includes all the information in S_w , we have

$$P(X | S_c S_w) = P(X | S_c).$$

Then equation (1) can be simplified as follows:

$$\begin{aligned} \hat{S} &= \arg \max_{S_c S_w} P(S_w S_c) P(X | S_c) \\ &= \arg \max_{S_c S_w} P(S_w | S_c) [P(S_c) P(X | S_c)] \\ &= \arg \max_{S_c S_w} P(S_w | S_c) [P(S_c | X) P(X)] \\ &= \arg \max_{S_c S_w} P(S_w | S_c) P(S_c | X) \end{aligned} \quad (2)$$

where $P(S_c)$ and $P(S_w | S_c)$ stand for character-based SLM and word-based SLM, respectively. $P(S_c | X)$ can represent the statistical model of character-based contextual postprocessing.

However, it is very difficult to solve equation (2) directly. We adopt a multilayer strategy to solve it, i.e., $P(S_c | X)$ is executed first, thereafter $P(S_w | S_c)$ is executed according to the results in $P(S_c | X)$. Hence equation (2) can be approximately computed by

$$\hat{S} \approx \arg \max_{S_w} P(S_w | \hat{S}_c) [\max_{S_c} P(S_c | X)] \quad (3)$$

where \hat{S}_c stands for the character string obtained by character-based contextual post-processing.

This kind of multilayer processing strategy can also be found in speech recognition [Schwartz and Austin 1991], where the most efficient knowledge sources were first used to produce a scored list of top m (N -Best) sentences, and then the other knowledge sources were used to reorder the sentences and select the most likely one.

In this article our aim is not to produce the top m sentences, but to produce the top m candidates for each original candidate set in character-based contextual postprocessing. In equation (3), character-based contextual postprocessing aims at, on the one hand, to improve accuracy, and, on the other hand, to improve the efficiency of the candidate set (that is, to increase the cumulative accuracy of the top m candidates). In order to speedily implement word-based contextual postprocessing, from the top m candidates with a high cumulative accuracy, we can construct a compact word graph (where the search space is not large) [Gu et al. 1991].

Although character-based trigram postprocessing has higher accuracy than character-based bigram postprocessing (as in Section 5.2), the latter's speed is far faster than the former's. In this article we adopt character-based bigrams in postprocessing where the candidate set may be fairly large.

3. FORWARD-BACKWARD SEARCH IN CHARACTER-BASED BIGRAM CONTEXTUAL POSTPROCESSING

Generally speaking, character-based bigram postprocessing is implemented by the well-known Viterbi search algorithm [Xia et al. 1996], which can find the sentence with the highest probability (i.e., the single best state sequence [Rabiner 1989]) along a single path. However, the Viterbi search algorithm does not modify the candidate sets given by the recognizer, so it cannot improve the efficiency of the candidate set.

Forward-backward search is an important algorithm for computing the hidden Markov model's (HMM) parameters [Rabiner 1989]; the forward-backward search algorithm selects the most probable state every time (it is also widely used for path search in speech recognition [Ney and Ortmanns 1999]). Here, we use the principal idea of forward-backward search as a basis for implementing character-based bigram postprocessing in order to boost the efficiency of the candidate set. In the following, we first introduce the basic idea of forward-backward search and then show the derivation step in detail.

3.1 Basic Idea

Unlike the Viterbi search algorithm, the basic idea of the forward-backward search algorithm for implementing character-based bigram postprocessing is as follows: Each character c_t ($1 \leq t \leq T$) in the character string $S_c = c_1 c_2 \wedge c_T$ has maximal probability, that is, for each position t , the optimal candidate \hat{c}_t is selected step-by-step from the candidate set $\{c_{t,1}, c_{t,2}, \wedge c_{t,m_t}\}$ of c_t using its contextual information; m_t is the number of candidates.

The optimal candidate \hat{c}_t can be represented as follows:

$$\hat{c}_t = \arg \max_{1 \leq i \leq m_t} P(c_t = c_{t,i} | X) \quad 1 \leq t \leq T \quad (4)$$

where

$$P(c_t = c_{t,i} | X)$$

is the probability of the i^{th} candidate $c_{t,i}$, given X . For conciseness, let

$$\gamma_t(i) = P(c_t = c_{t,i} | X).$$

Noting that

$$X = x_1 x_2 \Lambda x_t x_{t+1} \Lambda x_T$$

we have

$$\begin{aligned} \gamma_t(i) &= P(x_1, \Lambda, x_t, c_t = c_{t,i}) P(x_{t+1}, \Lambda, x_T | c_t = c_{t,i}) \\ &\quad / \sum_{j=1}^{m_t} P(x_1, \Lambda, x_t, c_t = c_{t,j}) P(x_{t+1}, \Lambda, x_T | c_t = c_{t,j}) \end{aligned} \quad (5)$$

In order to calculate the probability $\gamma_t(i)$, forward and backward probability is introduced.

3.2 Forward and Backward Probability

Consider the forward probability $\alpha_t(i)$ defined as

$$\alpha_t(i) = P(x_1, x_2, \Lambda, x_t; c_t = c_{t,i}) \quad (6)$$

i.e., the joint probability of the partial feature sequence from the beginning to position t and current candidate $c_{t,i}$ at position t . We can compute $\alpha_t(i)$ recursively, as follows:

(1) Initialization

$$\alpha_1(i) = P(c_{1,i}) P(x_1 | c_{1,i}), \quad 1 \leq i \leq m_1; \quad (7)$$

(2) Recursion

$$\begin{aligned} \alpha_{t+1}(j) &= P(x_1, x_2, \Lambda, x_t, x_{t+1}; c_{t+1} = c_{t+1,j}) \\ &= \left[\sum_{i=1}^{m_t} \alpha_t(i) P(c_{t+1,j} | c_{t,i}) \right] P(x_{t+1} | c_{t+1,j}), \quad 1 \leq t \leq T-1, \quad 1 \leq j \leq m_{t+1} \end{aligned} \quad (8)$$

In a similar manner, we consider a backward probability $\beta_t(i)$ defined as

$$\beta_t(i) = P(x_{t+1}, x_{t+2}, \Lambda, x_T | c_t = c_{t,i}) \quad (9)$$

i.e., the conditional probability of the partial feature sequence from position $t+1$ to the end, given candidate $c_{t,i}$ at position t . We can also compute $\beta_t(i)$ recursively, as follows:

(1) Initialization

$$\beta_T(i) = 1, 1 \leq i \leq m_T \quad (10)$$

(2) Recursion

$$\beta_t(i) = \sum_{j=1}^{m_{t+1}} P(c_{t+1,j} | c_{t,i}) P(x_{t+1} | c_{t+1,j}) \beta_{t+1}(j) \quad T-1 \geq t \geq 1, \quad 1 \leq i \leq m_t \quad (11)$$

After introducing $\alpha_t(i)$ and $\beta_t(i)$, equation (5) can then be written as

$$\gamma_t(i) = \alpha_t(i) \beta_t(i) / \sum_{j=1}^{m_t} \alpha_t(j) \beta_t(j) \quad (12)$$

In the above discussion, $P(c_{t+1,j} | c_{t,i})$ is the transition probability of a character-based bigram; $P(c_{1,i})$ is the first character probability in a sentence; $P(x_{t+1} | c_{t+1,j})$ is the conditional probability, and can be represented as follows by applying the Bayes rule:

$$P(x_{t+1} | c_{t+1,j}) = P(c_{t+1,j} | x_{t+1}) * P(x_{t+1}) / P(c_{t+1,j}) \quad (13)$$

where $P(c_{t+1,j})$ is the *a priori* probability determined by the recognizer, and can be regarded as equally probable for all j [Chen 1997]; whereas $P(x_{t+1})$ is the common item in both the numerator and denominator in equation (12), and can be omitted. Therefore, while solving equation (12) with practicable computation, the probability item $P(x_{t+1} | c_{t+1,j})$ can be replaced with the *a posteriori* probability $P(x_{t+1,j} | x_{t+1})$ of candidate $c_{t+1,j}$. $P(c_{t,j} | x_t)$ is equivalent to the confidence value of candidate $c_{t,j}$ [Li and Ding 2002]. The *logistic regression model* [Ho et al. 1994; Li and Ding 2002] can directly convert the distance measure of a candidate into its confidence value:

$$P(c_{t,i} | x_t) = (1 + \exp(\beta_0^i + \sum_{j=1}^p \beta_j^i d_j))^{-1}, \quad 1 \leq i \leq m_t \quad (14)$$

where β_j^i is the regression coefficient, and can be estimated by the *maximum likelihood estimation* [Hosmer and Lemeshow 1989] through the recognition results of some training sets. d_j is the recognition distance of the j^{th} candidate ($d_1 \leq d_2 \leq \Lambda \leq d_m$); p is the order of the regression model.

3.3 New Candidate Set

We rerank the candidates in the candidate set $\{c_{t,1}, c_{t,2}, \Lambda, c_{t,m_t}\}$ according to the probability computed by equation (12), so that we can get a new candidate set $\{\hat{c}_{t,1}, \hat{c}_{t,2}, \Lambda, \hat{c}_{t,m_t}\}$. Owing to the effects of using contextual information, the true candidate is often included in the top m candidates in the new candidate set. The relevant probability sequence of new candidates $\{\hat{\gamma}_t(1), \hat{\gamma}_t(2), \Lambda, \hat{\gamma}_t(m_t)\}$ could be regarded as the new candidates' confidence value.

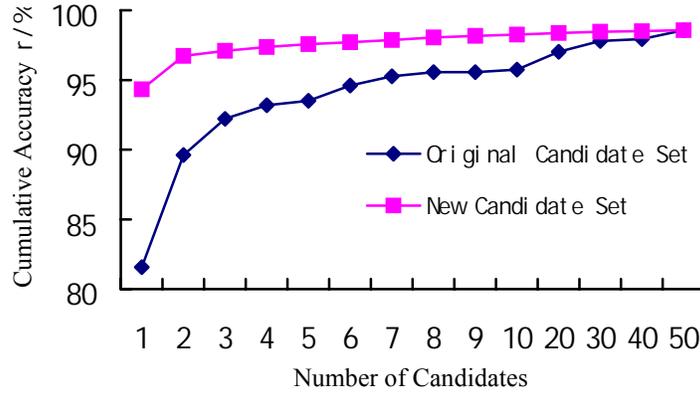


Fig.1. Comparing cumulative accuracy for different candidate sets.

Based on the forward-backward search, experimental results for *Script B* are shown in Figure 1 (the number of original candidates is 50, and the cumulative accuracy is 98.56%; see Section 5).

As can be seen in Figure 1, character-based bigram postprocessing using the forward-backward search not only improves the recognition accuracy, but modifies the original candidate set to increase the efficiency of each candidate set as well. Most of the true candidates in the new candidate sets are included in the top 10 candidates.

The cumulative accuracy¹ of the top 10 candidates improves from 95.73% to 98.24%, i.e., the error rate for the top 10 candidates is reduced by 58.78%. In other words, the cumulative accuracy of the top 10 new candidates approximates the cumulative accuracy of the top 50 original candidates. The high efficiency of the candidate set is very beneficial for constructing a compact word graph for word-based contextual post-processing.

4. CONTEXTUAL POSTPROCESSING INTEGRATING CHARACTER-BASED BIGRAMS WITH WORD-BASED BIGRAMS

After completing character-based bigram postprocessing using the forward-backward search algorithm, we may select the top m candidates of the new candidate set $\{\hat{c}_{t,1}, \hat{c}_{t,2}, \dots, \hat{c}_{t,m}\}$ with greater efficiency in order to further execute word-based postprocessing.

Since the word string S_w consists of the character string \hat{S}_c , we have $P(S_w | \hat{S}_c) = P(S_w)$ in equation (3). Combining equations (3) and (4), the optimal word string \hat{S} (the optimal sentence needed) can be computed by

$$\hat{S} \approx \arg \max_{S_w} P(S_w) \prod_{t=1}^T \hat{\gamma}_t \quad (15)$$

¹ Cumulative accuracy = $(1.0 \times \text{the number of correct Chinese characters in the top 10 candidates} / \text{total Chinese characters}) \times 100\%$

Here, we adopt a word-based bigram as the word-based SLM, i.e.,

$$P(S_w) = P(w_1) \prod_{i=2}^{T'} P(w_i | w_{i-1}).$$

So equation (15) can be rewritten as follows:

$$\begin{aligned} \hat{S} &\approx \arg \max_{S_w} P(w_1) \prod_{i=2}^{T'} P(w_i | w_{i-1}) \prod_{t=1}^T \hat{\gamma}_t \\ &= \arg \max_{S_w} [P(w_1)\varphi(w_1)] * [\prod_{i=2}^{T'} P(w_i | w_{i-1})\varphi(w_i)] \end{aligned} \quad (16)$$

where $\hat{\gamma}_t$ stands for the confidence value of the new candidate in the t^{th} position;

$$\varphi(w_i) = \prod_{t=t_{i-1}+1}^{t_i} \hat{\gamma}_t, \quad t_i = \sum_{j=1}^i |w_j|$$

$|w_j|$ stands for the j^{th} word length; $P(w_i | w_{i-1})$ is the transition probability of a word-based bigram; and $P(w_i)$ is the first word probability in a sentence. The optimal sentence in equation (16) can be searched by the Viterbi algorithm [Rabiner 1989].

5. EXPERIMENTS

In our experiment, ‘‘THOCR’97 Synthetical and Integarated Chinese Character Recognition System’’ [Chen 1997] is used as the offline, handwritten, isolated Chinese character recognizer. The SLMs are trained by the *People’s Daily* newspaper corpus (1993–1994), which includes about 40,000,000 Chinese characters. The *People’s Daily* newspaper corpus is very comprehensive, and the SLMs trained by it can be widely applied to different domains. There are 3,763 characters and 78,993 words in the lexicon. The objects of postprocessing are three scripts (*Script A*, *Script B*, and *Script C*, handwritten by 30 writers), whose recognition accuracy² (namely the first candidate’s accuracy) without postprocessing is 92.32%, 81.58%, and 70.84%, respectively. Each script includes about 22,000 characters, involving news, politics, and computers selected from the Internet (they are not in the corpus). In the experiment, in order to deal with the sparse data in the SLMs, we use Witten-Bell smoothing [Chen and Goodman 1999], which can be depicted as follows:

$$P_{WB}(s_i | s_{i-1}) = \frac{n(s_{i-1}s_i) + N_{1+}(s_{i-1}\cdot)P(s_i)}{N_{1+}(s_{i-1}\cdot) + \sum_{s_i} n(s_{i-1}s_i)} \quad (17)$$

$$P(s_i) = (n(s_i) + \varepsilon) / \sum_j n(s_j)$$

² Recognition accuracy = (1.0 – the number of incorrect Chinese characters / total Chinese characters) × 100%

where

$$N_{1+}(s_{i-1}) = |\{s_i : n(s_{i-1}s_i) > 0\}|$$

is the number of unique words (or characters) that follow the history s_{i-1} in the corpus; $n(s_i)$ and $n(s_{i-1}s_i)$ represent the number of times unigram s_i and bigram $s_{i-1}s_i$ occur in the training corpora; $\varepsilon = 0.01$ avoids zero probability. For word-based bigrams, s_i stands for a Chinese word; for character-based bigrams, s_i stands for a Chinese character.

In order to compute the candidate confidence value, 50 training sets with an average accuracy of 87.40% are used to estimate the regression coefficients in equation (14). Each training set consists of 3,755 offline, handwritten, Chinese characters. For the first candidate, we have

$$P(c_{t,1} | x_t) = (1 + \exp(-0.647 + 0.439d_1 - 0.325d_2 - 0.086d_3))^{-1} \quad (18)$$

Equation (18) states that the smaller the d_1 (and the larger d_2 or d_3), the more reliable the first candidate. For the second candidate

$$P(c_{t,2} | x_t) = (1 + \exp(-0.221 - 0.377d_1 + 0.392d_2))^{-1}$$

for other candidates ($i \leq 3$), and the confidence value has a similar formula:

$$P(c_{t,i} | x_t) = (1 + \exp(\beta_0^i + \beta_1^i d_1 + \beta_1^i d_i))^{-1} \quad (\beta_1^i < 0, \beta_i^i < 0)$$

which means that the smaller the d_1 (and the larger d_i), the less reliable the nonfirst candidate.

5.1 Relationship Between Processing Time and Number of Candidates

With an Intel Pentium III 800 PC, we obtain the relationship between postprocessing time and the number of original candidates m for *Script B*, as shown in Figure 2.

As can be seen in Figure 2, character-based bigram postprocessing is much faster. Its processing time increases linearly with increasing m , while word-based bigram postprocessing appears very slow, and its processing time rises exponentially with increasing m . In fact, postprocessing speed is mainly decided by three factors: the complexity of looking up SLM parameters; the complexity of the search algorithm; and the complexity of constructing a word lattice. Apparently, the parameters of a character-based bigram are far smaller than those of a word-based bigram, and the search space of character-based bigram postprocessing is also far smaller than that of word-based bigram postprocessing. Furthermore, character-based bigram postprocessing does not need to construct a word lattice. As we know, the complexity of rapidly constructing a word graph rises with increasing m [Gu et al. 1991].

It is worth noting that word-based bigram postprocessing with the original m candidates is very slow if m is large. In Figure 2, word-based bigram postprocessing takes 79 minutes with $m=50$, while character-based bigram postprocessing only takes 78 seconds with the same candidate sets.

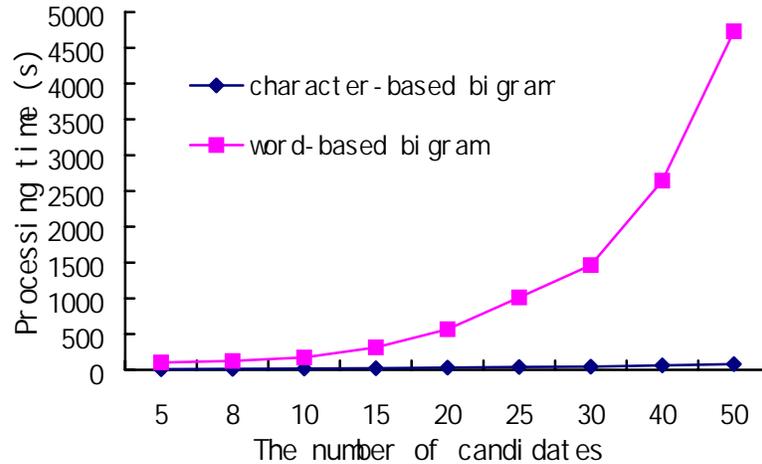


Fig. 2. Postprocessing time as a function of the number of candidates.

5.2 Comparing Postprocessing Performance

While doing character-based bigram postprocessing (CBP), the number of candidates m depends on the script's recognition accuracy without postprocessing, which can be estimated by equation (18). The mean value of all the first candidates' confidence values in a sample is equal to the expected value of accuracy in character recognition [Lin et al. 1998]. In fact, compared to the true recognition accuracy of *Script A*, *Script B*, and *Script C*, the estimated results based on equation (18) are fairly consistent: the estimated values are 93.38%, 83.56%, and 72.45%, respectively.

There is an experimental relationship curve between the recognition accuracy without postprocessing and the cumulative recognition accuracy of the top m candidates [Li and Ding 2001]. Apparently, if the script's recognition accuracy is high, m should be small; otherwise m should be large. Here the number of candidates m is estimated as 20, 50, and 100 for *Script A*, *Script B*, and *Script C*, respectively.

While doing contextual postprocessing to integrate character-based bigrams with word-based bigrams, the efficiency of the top 10 new candidates is fairly high (also shown in Table II), so 10 is selected as the number of candidates in word-based bigram postprocessing (WBP) for all of the scripts.

Table I shows the experimental results of seven postprocessing methods coded as $M1$ to $M7$: i.e., $M1$ - CBP using Viterbi search (VS) with 10 original candidates; $M2$ - CBP using forward-backward search (FBS) with 10 original candidates; $M3$ - character-based trigram postprocessing (CTP) using VS with 10 original candidates; $M4$ - WBP using VS with 10 original candidates; $M5$ - CBP using FBS with m original candidates; $M6$ - WBP using VS with m original candidates; $M7$ - WBP using VS with 10 new candidates given by $M5$, which is our proposed method and actually integrates $M4$ with $M5$.

We also obtain the postprocessing time for the various methods using an Intel Pentium III 800 PC. The WBP with the original 10 candidates only needs 3 minutes. For *Script A*,

Table I. Comparing Postprocessing Performance of the Seven Methods

	<i>Script A</i>	<i>Script B</i>	<i>Script C</i>	Average	Error correct rate ³	Average processing time
Before post-processing	92.32%	81.58%	70.84%	81.58%	-	-
<i>M1</i> : CBP + original 10 candidates (VS)	98.49%	93.34%	84.38%	92.07%	56.95%	19 seconds
<i>M2</i> : CBP + original 10 candidates (FBS)	98.47%	93.35%	84.32%	92.05%	56.82%	20 seconds
<i>M3</i> : CTP + original 10 candidates (VS)	98.69%	93.95%	84.94%	92.53%	59.43%	463 seconds
<i>M4</i> : WBP + original 10 candidates (VS)	98.73%	94.01%	84.92%	92.55%	59.55%	187 seconds
<i>M5</i> : CBP + original <i>m</i> candidates (FBS)	98.76%	94.32%	90.43%	94.50%	70.14%	101 seconds
<i>M6</i> : WBP + original <i>m</i> candidates (VS)	99.16%	96.23%	92.54%	<u>95.98%</u>	78.16%	<u>500 minutes</u>
<i>M7</i> : WBP + new 10 candidates (VS)	99.10%	96.08%	92.08%	<u>95.75%</u>	76.93%	<u>265 seconds</u>

a WBP with 20 original candidates needs about 6 minutes, while a WBP with 50 original candidates needs 79 minutes for *Script B*. For *Script C*, the WBP with 100 original candidates needs about 24 hours, which is very time-consuming. However, CBP with *m* original candidates only needs 29 seconds for *Script A*, 78 seconds for *Script B*, and 195 seconds for *Script C*. Finally, note that our method for integrating CBP with WBP (*M7*) only needs 197 seconds for *Script A*, 250 seconds for *Script B*, and 348 seconds for *Script C*. The average postprocessing time for the seven methods is also shown in Table I.

From Table I, experimental results are characterized by the following: For CBP, the performance of the forward-backward search (*M2*) is comparable to that of the Viterbi search (*M1*). With the same candidates, CTP (*M3*) and WBP (*M4*) show comparable accuracy, which is higher than that of CBP; CTP is very slow in comparison to CBP.

Based on *M5*, WBP with 10 new candidates (*M7*) can further improve the recognition accuracy from 94.50% to 95.75% (the error is reduced by 22.73%), which validates the complementary action between Chinese characters and Chinese words. In comparison to traditional CBP (*M1*), the error is greatly reduced by 46.41%.

With the increase in the number of candidates, WBP (*M6*) becomes very time-consuming, and is far slower than CBP (*M5*), although WBP can obtain a higher accuracy than CBP. However, *M7*, which actually integrates *M4* with *M5*, can obtain a recognition

³ Error correction rate = $(1.0 - \text{the number of errors after postprocessing} / \text{the number of errors before post-processing}) \times 100\%$

Table II. Comparing Cumulative Accuracy

	Cumulative accuracy for the 10 original candidates	Cumulative accuracy for the 10 new candidates	Error reduction rates for the top 10 candidates
<i>Script A</i>	99.31%	99.71%	57.97%
<i>Script B</i>	95.73%	98.24%	58.78%
<i>Script C</i>	87.94%	96.81%	73.55%
Average	94.33%	98.25%	69.14%

accuracy of approximately 95.98% of *M6*, while the processing speed of *M7* is more than 100 times faster than that of *M6*.

In our experiment, by limiting the number of new candidates (to only 10), word-based bigram contextual postprocessing can be speeded up, while character-based bigram contextual postprocessing is very fast, even for the big candidate set. Therefore, our post-processing method for integrating character-based bigrams with word-based bigrams performs well, has high accuracy, and is fast. Our method is fairly effective (e.g., *Script B* and *Script C*), especially when the script is poorly recognized.

While executing character-based bigram postprocessing with m original candidates (*M5*), the forward-backward search not only improved recognition accuracy, but also boosted the cumulative accuracy of the top 10 candidates. Table II lists the cumulative accuracy for 10 original candidates and 10 new candidates. In comparison to the 10 original candidates, the cumulative accuracy of the 10 new candidates increased from 94.33% to 98.25%. The rate of error reduction among the 10 candidates reached 69.14%. Since the efficiency of the top 10 candidates was very high, it is very beneficial to use an advanced language model to further improve accuracy in recognizing script.

6. CONCLUSION

In this article we presented a novel contextual postprocessing method that combines Chinese characters with Chinese words, and showed the implementation steps in detail. Based on isolated character recognition, character-based bigram postprocessing using the forward-backward search is first executed on a big candidate set, which not only improves recognition accuracy but greatly boosts the cumulative accuracy of the top 10 candidates as well. Then, word-based bigram postprocessing using the Viterbi search is executed on a small candidate set (containing 10 new candidates) to further improve recognition accuracy. Experimental results on offline, handwritten, Chinese scripts validate the existence of complementary action between characters and words in the Chinese language. The proposed method effectively improves recognition accuracy, while at the same time paying attention to processing speed. In comparison to word-based bigram postprocessing executed on a big candidate set, our method obtains comparable accuracy and improves the speed more than 100 times.

In addition, due to the high efficiency of the top 10 candidates, we can construct a compact word graph and employ an advanced language model (such as the word-based trigram model and the semantic-based n-gram model) while executing word-based contextual postprocessing to further improve accuracy in the recognition of Chinese script. These are subjects for future research.

ACKNOWLEDGMENTS

The authors would like to thank the reviewers for their valuable comments and suggestions.

REFERENCES

- CHANG, C.-H. 1996. Simulated annealing clustering of Chinese words for contextual text recognition. *Pattern Recogn. Lett.* 17, 3, 30-36.
- CHEN, Y. B. 1997. Research on hand-printed Chinese character recognition. Ph.D. dissertation, Tsinghua University, China (in Chinese).
- CHEN, S. F. AND GOODMAN, J. 1999. An empirical study of smoothing techniques for language modeling. *Comput. Speech Lang.* 13, 4, 359-394.
- GU, H.-Y., TSENG, C.-Y., AND LEE, L.-S. 1991. Markov modeling of mandarin Chinese for decoding the phonetics sequence into Chinese characters. *Comput. Speech Lang.* 5, 4, 363-377.
- HO, K.T., HULL, J.J., AND SRIHARI, S.N. 1994. Decision combination in multiple classifier systems. *IEEE Trans. Pattern Anal. Mach. Intell.* 16, 1, 66-75.
- HOSMER, D.W. AND LEMESHOW, S. 1989. *Applied Logistic Regression*. Wiley, New York.
- LEE, H.-J. AND TUNG, C.-H. 1997. A language model based on semantically clustered words in a Chinese character recognition system. *Pattern Recogn.* 30, 8, 1339-1346.
- LI, Y. X. AND DING, X. Q. 2001. Multiple candidate characters in the post-processing for off-line handwritten Chinese character recognition. In *Proceedings of the 2001 International Conferences on Info-tech and Infonet (ICII2001)*, Conference C, Beijing), 438-443.
- LI, Y. X. AND DING, X. Q. 2002. Evaluation of character candidate confidence measure using logistic regression model. *Pattern Recogn. Artif. Intell.* 15, 2, 160-166 (in Chinese).
- LI, Y. X., DING, X. Q., AND LIU C. S. 1999. Post-processing study of Chinese document recognition based on HMM. *J. Chinese Inf. Process.* 13, 4, 29-34 (in Chinese).
- LIN, X. F., DING, X. Q., CHEN, M., ET AL. 1998. Adaptive confidence transform based classifier combination for Chinese character recognition. *Pattern Recogn. Lett.* 19, 10, 975-988.
- LIU, J. 2000. Research on large vocabulary mandarin Chinese continuous speech recognition system. *Acta Electron. Sinica* 28, 1, 85-91 (in Chinese).
- NEY, H. J. AND ORTAMMNS, S. 1999. Dynamic programming search for continuous speech recognition. *IEEE Signal Process. Mag.* (Sept.), 64-83.
- RABINER, L. R. 1989. A tutorial on hidden Markov models and selected applications in speech recognition. *Proc. IEEE* 77, 2, 257-286.
- SCHWARTZ, R. AND AUSTIN, S. 1991. A comparison of several approximate algorithms for finding multiple (N-Best) sentence hypotheses. In *Proceedings of 1991 International Conferences on Acoustics, Speech, Signal and Processing (ICASSP1991)*, Toronto, Canada), 701-704.
- TUNG, C.-H. AND LEE, H.-J. 1994. Increasing character recognition accuracy by detection and correction of erroneously identified characters. *Pattern Recogn.* 27, 9, 1259-1266.
- WONG, P.-K. AND CHAN, C. 1999. Post-processing statistical language models for a handwritten Chinese character recognizer. *IEEE Trans. Syst. Man Cybern.* 29, 2, 286-291.
- XIA, Y., MA, S. P., CHANG, X. G., ET AL. 1996. The method of automatic post-processing based statistical probabilities for Chinese recognition text. *Pattern Recogn. Artif. Intell.* 9, 2, 172-178 (in Chinese).

Received January 2002; revised February 2003; accepted March 2003.