# A word language model based contextual language processing on Chinese character recognition

Chen Huang*, Xiaoqing Ding, Yan Chen
State Key Laboratory of Intelligent Technology and Systems
Tsinghua National Laboratory for Information Science and Technology
Department of Electronic Engineering, Tsinghua University, Beijing 100084, P. R. China

## ABSTRACT

The language model design and implementation issue is researched in this paper. Different from previous research, we want to emphasize the importance of n-gram models based on words in the study of language model. We build up a word based language model using the toolkit of SRILM and implement it for contextual language processing on Chinese documents. A modified Absolute Discount smoothing algorithm is proposed to reduce the perplexity of the language model. The word based language model improves the performance of post-processing of online handwritten character recognition system compared with the character based language model, but it also increases computation and storage cost greatly. Besides quantizing the model data non-uniformly, we design a new tree storage structure to compress the model size, which leads to an increase in searching efficiency as well. We illustrate the set of approaches on a test corpus of recognition results of online handwritten Chinese characters, and propose a modified confidence measure for recognition candidate characters to get their accurate posterior probabilities while reducing the complexity. The weighted combination of linguistic knowledge and candidate confidence information proves successful in this paper and can be further developed to achieve improvements in recognition accuracy.

**Keywords:** n-gram, language model, word, perplexity, smoothing methods, quantization, contextual processing

## 1. INTRODUCTION

Language models have been widely used in many domains including speech recognition, optical character recognition, machine translation, spelling correction, etc. The radical task of language modeling is to estimate the probability of each occurred n-gram according to its count in training corpora. For a given character sequence $C = c_1 ... c_N$, we can express the probability p(C) of the N- character sequence as a chain of conditional probabilities:

$$p(C) = p(c_1)...p(c_N \mid c_{N-n+1}...c_{N-1}) = \prod_{i=1}^{N} p(c_i \mid c_{i-n+1}...c_{i-1}) \quad (1)$$

This model estimates the approximate probability of a given character by using the (n-1)$^{th}$ preceding character sequence instead of the whole long sequence of preceding characters. It is apparent that the larger n is, the higher the cost of computing conditional probabilities is. But high order language models can capture more prior context for predicting the next character[1] while increasing the computational cost. Bigram models where n is 2 have been used in previous work[2,3,4] due to efficiency concerns, especially in the case of language models for Chinese characters. Peter Heeman basically derived a phrase-based language model from word bigram probabilities at low order level rather than directly from high order n-grams. Yuanxiang Li mainly built bigram models on Chinese character corpus, and chose character based trigram models representing the high order models to be compared with the bigram models on character recognition performance, but the trigram models led to a significant increase in time complexity.

Usually, a single Chinese character cannot convey a meaning like words, so we propose a word-based language model to capture larger dependencies for more accurate prediction. If the character pair $w_i w_j$ appears in the training

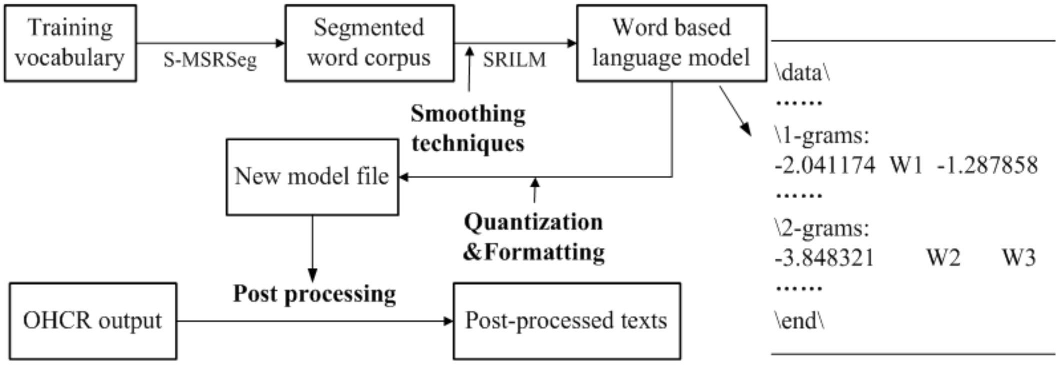
[*] Corresponding Author: huangchen@ocrserv.ee.tsinghua.edu.cn phone: +86 -10 -62772044 -244

Document Recognition and Retrieval XVII, edited by Laurence Likforman-Sulem, Gady Agam,
Proc. of SPIE-IS&T Electronic Imaging, SPIE Vol. 7534, 75340N · © 2010 SPIE-IS&T
CCC code: 0277-786X/10/$18 · doi: 10.1117/12.838718

corpus, they are identified as a single token $w_i\_w_j$ [2] which is taken to be the basis of computing the language model probabilities. The training corpus is then rewritten with words, resulting in a word language model (LM) built at the lexical level. However, it is more complicated than the character model, and increases the perplexity, a common metric to evaluate a language model. So we use a word-based bigram language model to compromise between efficiency and accuracy, thus avoiding unacceptable time complexity and meanwhile directly optimizing the linguistic structure at the lexical level without approximation from morphemes. There has been work[eg,4,5] in Chinese word-based bigram language models, but none has considered the language model accuracy together with the model structure optimization. In this paper, we combine the two parts and implement the resulted language model for contextual language processing.

Smoothing is a way of preventing zero probabilities due to data sparseness, which is more severe in word based models. There are several well known smoothing methods including Additive Smoothing, Good-Turing, Jelinek-Mercer Smoothing, Katz, Witten-Bell, Absolute Discount[6], etc. Based on previous work, we develop an improved Absolute Discount algorithm outperforming all the others in reducing perplexity. Our word based language model is trained on selected Chinese Gigaword Second Edition[7] corpus using the toolkit of SRILM[1]. The flowchart of the system is shown in Fig.1, where we use S-MSRSeg[8] developed at Microsoft Research Asia as a Chinese word segmenter.

How to implement the new generated language model for contextual processing is a problem. Yuanxiang Li[4] directly put the LM in use, we here take into account its size and practical searching efficiency in the toolkit of SRILM. We follow the steps of quantizing the coefficients and designing an efficient storage format to compress data and optimize the processing speed. This better solves the space occupying problem of the word based language model and lowers time complexity compared with previous work[4,9]. Finally, we use the word based model to post-process the output of a simplified online handwritten Chinese character recognition system (OHCR) with a new confidence measure and a weighted Viterbi algorithm, obtaining an improvement in recognition accuracy.



```
\data\
……
\1-grams:
-2.041174  W1  -1.287858
……
\2-grams:
-3.848321      W2      W3
……
\end\
```

Fig. 1. Flowchart of training a word-based language model for contextual processing

## 2. THEORETICAL FRAMEWORK OF LANGUAGE MODEL

### 2.1 Modified Absolute Discount method

Perplexity is a common metric to evaluate a language model, and it is interpreted as the average number of bits to encode each word in the test set. If a given language model assigns probability $p(C)$ to a character sequence C, the cross-entropy $H_p(C)$ of this model on data C is defined as:

$$H_p(C) = -\frac{1}{W_C}\log_2 p(C) \tag{2}$$

[1] SRILM is a toolkit for building and applying statistical language models, it has been under development since 1995 in SRI Speech Technology and Research Laboratory. The basic tools for LM estimation and evaluation are ngram -count and ngram, respectively.

where $W_C$ is the length of the character sequence C. Then the perplexity $PP(C)$ of the test set is related to cross-entropy by the equation:

$$PP(C) = 2^{H_p(C)} \tag{3}$$

Clearly, lower perplexity correlates to better performance. In SRILM the perplexity is represented as a base 10 exponent ranging from 200 to 400 on our corpus.

Absolute Discount method has low perplexity and can be further improved in SRILM. It involves interpolating high and low order models, the higher order distribution will be calculated just subtracting a static discount D from each bigram with non-zero count[6]. The probability for a bigram can be calculated by interpolating the bigram and a lower order unigram as follows:

$$P^*(w_i \mid w_{i-1}) = \frac{\max\left\{c(w_{i-1}^{i}) - D, 0\right\}}{\sum_{w_i} c(w_{i-1}^{i})} + (1 - \lambda(w_{i-1}))P(w_i) \tag{4}$$

where D denotes a constant discount ( $0 < D \le 1$ ). In order to make the distribution add up to 1, we take ：

$$1 - \lambda(w_{i-1}) = \frac{D}{\sum_{w_i} c(w_{i-1}^{i})} N_{1+}(w_{i-1}\bullet) \tag{5}$$

The $N_{1+}(w_{i-1}\bullet)$ denotes the number of bigrams with one or more counts[6]. We propose to optimize D in $(0,1]$ and get a global optimization to minimize the perplexity, rather than estimating D to be $n_1/(n_1 + 2n_2)$ , where $n_1$ and $n_2$ are the total number of n-grams with one and two counts, respectively.

The modified absolute discount smoothing method is aimed at alleviating the problem of serious zero count in Chinese word language model with relatively low perplexity, and we can further lower the perplexity of the language model with the aid of SRILM. We limit the corpus by a specific lexicon and replace the words out of the lexicon with a universal token <unk>. After using a single token to represent the uncommon words, the new corpus has lower uncertainty and changed distributions of n-grams, leading to the reduced perplexity of smoothing methods as well. The comparison and experimental results are shown in section 4.

### 2.2 Language model compression

We are to compress the trained language model since it has substantial memory requirements. It is mainly composed of logarithm (base 10) of probabilities for unigrams and transition probabilities for bigrams, as well as back of weights (BOW) for interpolating bigrams with unigrams. For a bigram model, only unigrams have BOWs. SRILM uses standard ARPA[10] (Fig.1) format for n-gram backoff LM. The basic principle of using the backoff weight is that $P(w_i \mid w_{i-1})$ equals to $P^*(w_i \mid w_{i-1})$ if $w_{i-1}\_w_i$ exists, otherwise $bow(w_{i-1}) * P^*(w_i)$ , where $P^*(w_i \mid w_{i-1})$ and $P^*(w_i)$ are the smoothed probabilities. However, the large size of the resulted LM (320MB in computer memory) restricts its implementation in practical use.

Previous work shows that language models can be compressed by up to 60% of their original size with no significant loss in performance[9]. There are mainly two steps in our compression part. We first use the instruction 'gtnmin'[11] in SRILM to set a threshold number of word occurrence in the nth gram, and the words occurring less than the threshold will be omitted in the corpus. This method can largely reduce the size of the language model with no significant performance degradation. In our case we set gt2min to prune bigrams, since bigrams are severely sparse. Then we quantize the probabilities and BOWs non-uniformly to get smallest mean square quantization error $\sigma_q^2$ :

$$\sigma_q^2 = \sum_{k=1}^{L} \int_{x_k}^{x_{k+1}} (x - y_k)^2 \, p(x) dx \tag{6}$$

Where L is the number of quantization levels, $x_k$ is the judging threshold and $y_k$ is taken to be the quantization level. The experimental results are shown in section 4.

### 2.3 Language model storage structure

A well organized model storage structure can further reduce its occupation of computer memory and improve the searching efficiency as well. We describe a tree storage structure that branches out from the unigram nodes at the first level of the tree into the bigram nodes at the second level. This structure mainly includes a word table and a probability look up table to build up the mapping function between them. The first advantage of such structure is that we largely save space of storage by building a transition function among the word table to describe bigrams, which outperforms the space occupying approach of simply listing all the unigrams and bigrams. The second advantage is that the searching efficiency is therefore improved as a result of the tree mapping. We do not have to compare the table items one by one to find a required bigram, the hierarchical storage structure makes searching rapid with all items indexed and organized.

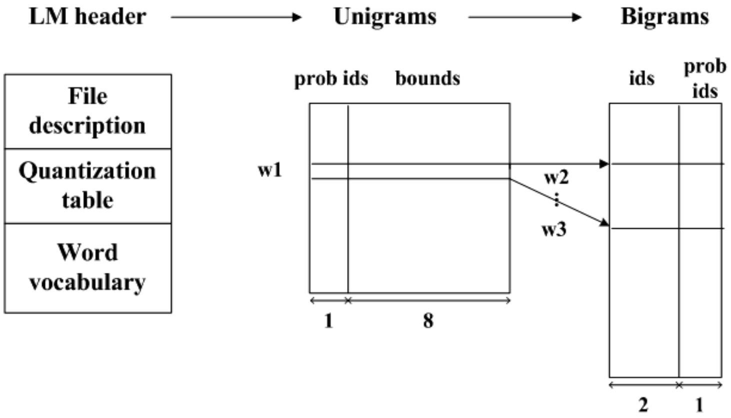


Fig. 2. Tree storage structure of a word based bigram language model

The storage framework is mainly composed of a LM header, unigram nodes and bigram nodes as shown in Fig.2. The quantization table and word vocabulary are stored in the header, both of which are cited with according indices rather than the original items in the following nodes. We quantize the probabilities within 1 Byte, and rank words in increasing order by their coding to make the fast searching algorithm possible. The prob ids in unigrams are the indices of probabilities stored in quantization table, and they belong to words stored in the LM header accordingly. After filling the unigrams with specific words and the according probabilities, we should also equip the possible following words with the word ids and prob ids to construct mapping from unigrams to bigrams. Based on above settings, the prob ids in Fig.2 generally use 1-byte representation, and ids at the second level use 2-byte, compared with 4-byte float probabilities and 2*k Bytes for the word that has k characters. Since there are 73858 words in our model, 2 Bytes are not enough to represent all the bigram node ids. So we lay off 2 more bytes to indicate the words whose ids are beyond 65535, maintaining that the ids can still be represented by 2 Bytes.

To build the mapping between the two levels, we let bounds at unigram level indicate the range of the next word ids at bigram level. For example, in Fig.2 the 8-byte bounds for 'w1' include the storage location of its first following bigram, the total number of following bigrams and the index from which ids are beyond 65535 (ids in bigrams are also ranked increasingly). They take 4 bytes, 2 bytes and 2 bytes respectively. According to this mapping rule, we can easily find the words following the unigram 'w1', like 'w2' and 'w3', as well as according transition probabilities in prob ids. It is remarkable that in SRILM toolkit not all bigrams use universal token <unk> to represent the unseen bigram nodes at the second level, making the search of look up table unclose. In fig.2, if 'w1' has no item <unk> among its following words, we should add the token to its bigram nodes and assign a transition probability according to the principle of backoff weighting mentioned in section 2.2. This new storage structure proves efficient in implementation and can also be easily extended to trigram models or higher order ones.

# 3. METHODOLOGY OF POST PROCESSING

## 3.1 Confidence measure

Concerned with language model post-processing, the recognition information is needed to be combined with linguistic knowledge. Adaptive Confidence Transform (ACT)[3] and Logistic Regression Model (LRM) are both statistical methods for recognition candidate posterior probability estimation, but they require large training set. In order to avoid being lack of samples, we propose a new empirical method based on Lee Y-S[12]'s work. Suppose there are k candidates $c_j(x), j=1,...,k$ for character x, followed by distance $d_j, j=1,...,k$. The candidate confidence $CM_j$ is related to the normalized distance $d_j$ given as follows:

$$CM_j = s_j / \sum_{j=1}^{k} s_j, \quad s_j = \frac{1}{d_j - d_1 + C}, j = 2,...,k \tag{7}$$

where recognition distances are ranked in descending order and C is a constant that minimizes $H(x)$, the entropy of posterior probability distributions of recognition candidates of x, which is:

$$H(x) = -\sum_{i=1}^{n} CM_i \log CM_i \tag{8}$$

The exact minimization of the entropy function is computationally expensive and there is no explicit solution of C. We use a gradient descent[13] strategy to find a global minimum of entropy $H(x)$, we can obtain:

$$\frac{\partial H(x)}{\partial C} = \sum_j \frac{score_j^2 * \sum_i score_i - score_j * \sum_i score_i^2}{(\sum_i score_i)^2} * \log(\frac{e * score_j}{\sum_i socre_i}) \tag{9}$$

where $score_j = 1/(d_j - d_1 + C)$. Based on this basic gradient term and a particular step size, we can run the gradient descent method for iterations to get the optimized C. The modified empirical method of confidence measure lowers the requirement of large training set and calculation cost. It proves efficient in confidence transform from distances since it makes OHCR output tend to achieve the most possible result by minimizing the entropy of the candidate set.

## 3.2 Weighted Viterbi algorithm

The post-processor's task is to select the most possible sentence from the sentence set, which includes all sentence combinations from the recognition candidates of OHCR. We use the Viterbi algorithm to integrate the candidate confidence information of each online handwritten Chinese character with the word based language model to post-process the OHCR output. Let $c_1c_2...c_T$ be a sentence of Chinese characters given by OHCR, and $X = x_1x_2...x_T$ be the sequence of Chinese character images, where T is the length of the sequence. For the word language model, the sentence S can be expressed as a sequence of $T'$ words $S = w_1w_2...w_{T'}$ generated from T character candidates $v_1v_2...v_T$, where $v_i$ is the chosen candidate of character $c_i$. Fig.3 depicts how to select the optimized path S.

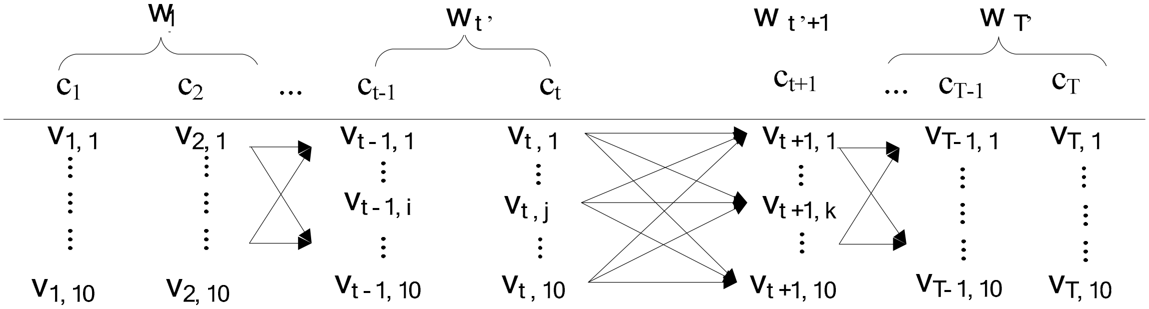


Fig. 3. Viterbi algorithm for word based language model

In Fig.3 we assume that there are 10 candidates $v_{i,j}, j = 1,...,10$ for each character $c_i$. For the word based language model, we rewrite the character sentence as a word set S at the lexicon level, then the optimization object of Viterbi algorithm is transformed from character set $v_1 v_2 ... v_T$ to a $T'$ word set as shown above. We take $w_{t'}$ and $w_{t'+1}$ for example in Fig.3 to show the process of path selection. We assume $w_{t'}$ contains two characters and $w_{t'+1}$ contains only one character, the Viterbi algorithm finds the best path that maximizes the product of $w_{t'+1}$'s recognition posterior probability and the transition probability from $w_{t'}$ to it.

The selected sentence is then rewritten as follows:

$$\tilde{S} = \arg\max_{S}[P(w_1)^{1-\lambda} CM(w_1)^{\lambda}] * \left[\prod_{i=2}^{T'} P(w_i \mid w_{i-1})^{1-\lambda} CM(w_i)^{\lambda}\right] \tag{10}$$

where $\lambda$ is the weight; and the word probability $CM(w_i) = \prod_{j=1}^{k_i} P(c_j^{w_i} \mid x_j^{w_i})$, in which $P(c_j^{w_i} \mid x_j^{w_i})$ is the posterior probability of the j^th^ character in word $w_i$ that contains $k_i$ characters. $\lambda$ is chosen to weight between contextual linguistic knowledge and the candidate confidence information to select a most possible sentence, which is more adaptive than previous work.

## 4. EXPERIMENTAL WORK

In this paper, the language model is trained on selected Chinese Gigaword Second Edition corpus, which includes about 610 million characters and involves sports, science, economics, politics and so on. The entire corpus includes three distinct international sources of simplified Chinese newswire: Agence France Press (afp_cmn), Xinhua News Agency (xin_cmn) and Zaobao Newspaper (zbn_cmn). There are 280 text files in total, specifically 75 for afp_cmn, 192 for xin_cmn and 13 for zbn_cmn, respectively. We use the provided corpus to train a language model, and then compress the data without severe performance degradation, which facilitates the post-processing of character recognition results according to the flowchart as shown in Fig.1.

### 4.1 The language model and smoothing methods

Table.1. Perplexities (base 10 exponent) of different smoothing methods obtained from SRILM

| | Good-Turing | Witten-bell | J-M Smoothing | Absolute Discount | Additive |
|---|---|---|---|---|---|
| PP (base 10) | 220.667 | 219.859 | 220.696 | 220.569 | 399.887 |

Table.1 shows perplexities of several smoothing methods obtained from the toolkit of SRILM, we can see only Additive Smoothing algorithm performs badly and the other four have nearly equal perplexities. The test set is randomly selected from the three newswire proportional to their file count, including about 11.3 million Chinese characters in total. The rest text files are taken to be the training set.

As discussed in section 2.1, we choose the Absolute Discount smoothing method and find its global minimum as a function of constant D. The allocation of training corpus and test corpus remains the same, and results are shown in Fig.4. We can see that perplexity of this smoothing method has its global minimum 210.94 (base 10) when D is between 0 and 1, 0.632 exactly. This value is lower than the empirical one[6] since the smoothing method is not just limited within unigrams or bigrams, it treats the corpora as a whole statistical combination of ngrams instead. The global minimum is attained when the whole distribution achieves least uncertainty. The optimized perplexity is about 5 as a base 2 exponent, suggesting that we can use an average of 5 bits to encode a word in this language model. The word based language model achieves relatively low perplexity by the Absolute Discount smoothing method, and can be further improved by the toolkit SRILM.

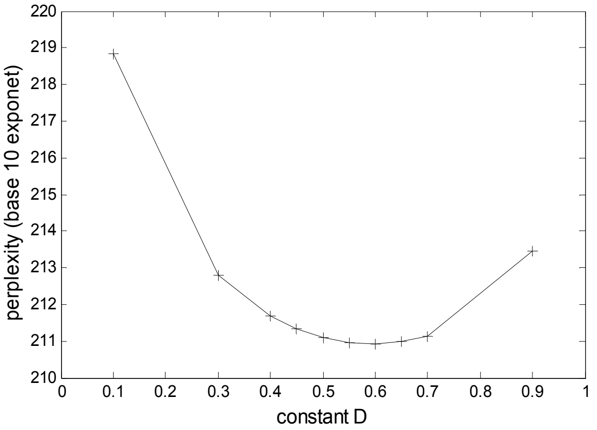


Fig. 4. Perplexity of Absolute Discount smoothed language model as a function of constant D

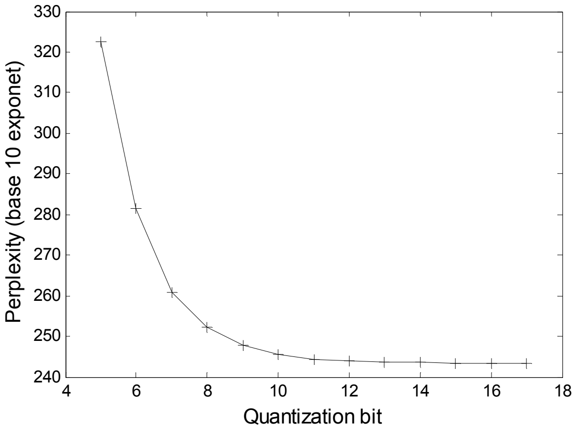


Fig. 5. Language model perplexity as a function of quantization bit

Following section 2.1, we tested the effect of adding token <unk> to the word corpus on the perplexity of the language model. In SRILM, the instruction '-unk' is often associated with '-vocab'. It functions on the corpus with a specific lexicon. Words not found in the lexicon will be added into it[10]. We compare the performance of the language model using these two methods with that of the original one as shown in Table.2. Here we randomly choose 2 text files from each newspaper to be the test set, and take the remaining to be the training set. We can see that '–vocab' leads to an obvious performance degradation because it results in a bigger lexicon that increases the uncertainty when estimating probabilities. We use token <unk> to greatly decrease the perplexity since it eliminates the uncertainty by using a single token to represent the uncommon words. The instruction '-unk' together with '-vocab' makes the word language model attain a lower perplexity than the original one.

Table. 2. Perplexities of different smoothing methods using '-unk' and '-vocab'

| | Good-Turing | Witten-bell | J-M Smoothing | Absolute discount | Additive |
|---|---|---|---|---|---|
| -unk -vocab | 211.276 | 209.954 | 211.329 | 211.103 | 347.201 |
| -vocab | 490.151 | 486.07 | 490.276 | 489.635 | 844.013 |
| none | 304.57 | 301.26 | 304.801 | 304.432 | 519.826 |

## 4.2 Model quantization and formatting

To compress the language model, we use gtnmin mentioned in section 2.2 to prune space occupying words. The language model after procession with the instruction '-unk -vocab' is 320MB in computer memory, and the perplexity is 5.0706 as a base 2 exponent. We use gt2min to prune bigrams occurring under 10 times in corpus, the result of perplexity and new file size as a function of the threshold is shown below in Table.3.

Table. 3. Perplexity and file size as a function of the threshold of bigram pruning

| | unlimited | Min2 | Min3 | Min4 | Min5 | Min6 | Min7 | Min8 | Min9 | Min10 |
|---|---|---|---|---|---|---|---|---|---|---|
| PP(base 2) | 5.0706 | 5.0951 | 5.1186 | 5.1389 | 5.1572 | 5.1734 | 5.1887 | 5.2031 | 5.2160 | 5.2281 |
| File size(MB) | 320 | 163 | 114 | 90 | 75 | 65 | 58 | 52 | 48 | 44 |

As expected, the perplexity of the resulted language model gradually rises when the threshold for pruning bigrams increases. A higher threshold means much more bigrams are discarded out of the language model, which leads to the degradation of the model performance and meanwhile the decrease in file size. Finally, we choose to reduce the size of the language model to 44MB while increasing the base 2 perplexity to 5.2281.

Non-uniform quantization is then applied to the pruned LM, the results are shown in Fig.5. As can be seen from Fig.5, when quantization bit is more than 8, perplexity decreases little. So we quantize the data within 1Byte, with the perplexity finally being 5.3092 (base 2).

We have mentioned in section 2.3 that we can further organize the model file as a tree structure. This new storage format is mainly a mapping function among a single word table; it not only reduces the file size by eliminating the space occupying bigram words, but also improves the searching efficiency based on the hierarchical structure. We finally make the model have its size down to 6.8MB, which is stored in the form of two-level tree. As a result, we get a language model compressed by 97.88% with little degradation, which outperforms previous work[4,9] with comparable lexicon size.

### 4.3 Contextual language processing

In the case of post-processing of OHCR, we have got the output of a simplified online handwritten Chinese character recognition system. The recognized character generally has 10 candidates with according recognition distances, which are arranged in ascending order. Since recognition distances are not all in a common metric space, we should transform the distances to the posterior probabilities. We propose an improved confidence measure mentioned in 3.1 to obtain both accuracy and small dependency on large training set. Then we combine the confidence measure and linguistic knowledge, and use the Viterbi algorithm for both the character based (CB) model and the word based (WB) model on a test corpus. The Viterbi algorithm integrates the recognition information and the language model transition probabilities to select an optimized path that represents the most possible sentence, and we expect the use of language model to improve the recognition accuracy obtained only from the character recognition system. The performance of this approach on different language models is shown in Fig.6. In this experiment, there are 6215 online handwritten Chinese characters in total, whose recognition accuracy without post-processing is 82.2%.

We can see from Fig.6 that both language models improve the recognition accuracy, but the WB model enhances the performance more for it captures more contextual information. When weight $\lambda$ in formula (10) increases to 1, the candidate confidence measure gains prior over linguistic knowledge. This weakens the language model's function of contextual processing, making the recognition accuracy reach the one without post-processing. Results are obtained when C in formula (7) remains 1, and the highest recognition accuracy for the WB model is 92.66% when $\lambda \leq 0.2$, which is higher than the maximum of CB model. The language model has a function of wiping out words that carry no meaning or correcting the mismatch of words obtained from machine. However, it is still auxiliary in the recognition.

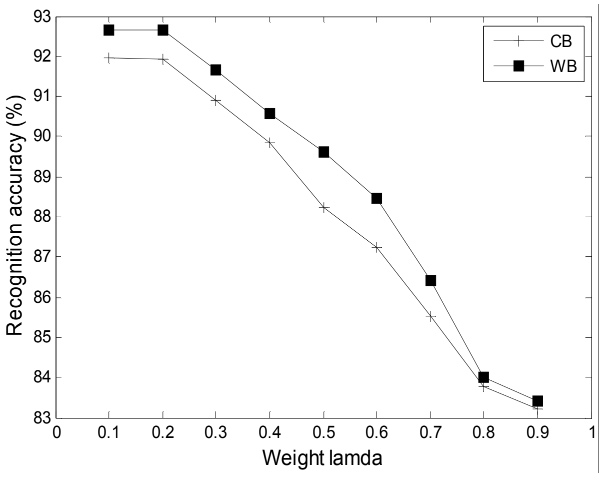


Fig. 6. Recognition accuracy as a function of weight $\lambda$ in formula (10)

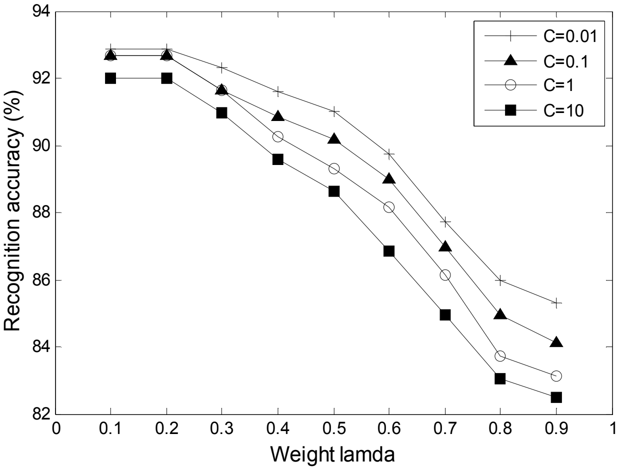


Fig. 7. Recognition accuracy as a function of weight $\lambda$ in formula (10) with different C in formula (7)

To find the optimized constant C in formula (7), we compare the performance of the word based language model under different C as shown in Fig.7. We can see when C is below empirical value 1, recognition accuracy rises, reaching at 92.87% with constant C being about 0.01. Table.4 shows the recognition accuracy as a function of weight $\lambda$ when the constant C remains 0.01, the maximum is achieved with a low weight for the confidence measure. This value of C is approximately the same as the one obtained with the gradient descent method mentioned in 3.1.

Table. 4. Recognition accuracy as a function of weight $\lambda$ when C is 0.01

| $\lambda$ | 0.1 | 0.2 | 0.3 | 0.4 | 0.5 | 0.6 | 0.7 | 0.8 | 0.9 |
|---|---|---|---|---|---|---|---|---|---|
| Recognition accuracy(%) | 92.87 | 92.87 | 92.36 | 91.51 | 90.75 | 89.76 | 87.21 | 85.93 | 83.11 |

Finally, this word based language model greatly increases the recognition accuracy by 10.67%. It not only has the advantage of being packed in small size, but also achieves lower time complexity than previous work[4]. We mainly make use of the data compression and the improved tree storage structure to boost the processing speed, which is 14% higher than the previous work. The post processing system together with the word language model proves robust and efficient in contextual language processing.

## 5. CONCLUSION AND FUTURE WORK

In this paper, we propose an improved Absolute Discount smoothing method that outperforms others in perplexity reduction. In SRILM toolkits, using the instruction of gtnmin greatly reduces the space that the language model occupies in computer memory. We further quantize probabilities and BOWs in the language model and reorganize it as a tree structure, which leads to a decrease of 97.88% in size with tolerable performance degradation. The new tree structure for storage also proves robust in contextual post-processing, and reducing time complexity as well. Experimental results show that the joint actions of entropy optimization based confidence measure and weighted Viterbi algorithm improve online handwritten Chinese character recognition accuracy from 82.2% to 92.87%.

The word based language model is more effective than the character based language model in post-processing of Chinese character recognition, and linguistic knowledge contributes more than character candidate confidence in improving the accuracy of character recognition. However, the improvement is limited to some extent since the language model is still an auxiliary tool in the recognition task. Our system of language model contextual processing has finally achieved high recognition accuracy with smaller storage space and lower time complexity.

In the future, we may go into the following ways aimed to further improve the performance of the language model and design a more efficient algorithm to implement it in contextual processing:

1. Further analyze the property of smoothing methods and design a perplexity reduction oriented smoothing method. The language model may be extended to be even higher order to get multi-transition probabilities, which describes the corpora more accurately.
2. The new tree storage structure proposed in this paper lays off 2 bytes in a word bound to indicate the following words whose ids are beyond 65535. We may perfect the structure concerned with the beyond-boundary-ids, which can also step beyond the bounds of 2 bytes when higher order language model is proposed.
3. A more accurate way of confidence measure is needed to take into account both the merits of statistical method and the problem of high complexity.
4. Combination of the word based language model and the character based language model is worth trying, since this may achieve the high accuracy of WB model and also avoid its high time complexity. Besides, a more efficient Viterbi algorithm may be designed to better utilize the information of recognition system and the language model.

## ACKNOWLEDGMENTS

The authors would like to thank the University of Pennsylvania for providing us with the linguistic database in the experiments, and the reviewers for their comments on the previous version of this paper.

## REFERENCES

[1] Huang, F.-L., Yu, M.-S., "Analyzing statistical properties of smoothing methods for language models," Proc. IEEESMC 1, IEEE System, Man and Cybernetics Conference, 512-517(2001).

[2] Peter A.Heeman, "Driving phrase-based language models," Proc. IEEE Workshop, (1997).

[3] X. Lin, X. Ding, M. Chen, R. Zhang, and Y. Wu, "Adaptive confidence transform based classifier combination for Chinese character recognition," Pattern Recognition Letters 19, 975-988(1998).

[4] Y. Li, X. Ding, C. L. Tan, "Combining character-based bigrams with word-based bigrams in contextual postprocessing for Chinese script recognition," ACM Trans. Asian Lang. Inf. Process. 1, 297-309(2002).

[5] Y. Li, C. Liu, X. Ding, "An Adaptive post processing method using proofreading information for Chinese character recognition," Journal of Chinese Information Processing, 15(1), 46-52(2001).

[6] Chen, S.F., Goodman, J., "An empirical study of smoothing techniques for language modeling," Computer Speech and Language 13, 359-394(1999).

[7] David Graff, "http://www.ldc.upenn.edu/Catalog/CatalogEntry.jsp?catalogId=LDC2007T38"

[8] Patrick Nguyen, Jianfeng Gao, and Milind Mahajan., "a scalable language modeling toolkit," Microsoft Research Technical Report, MSR-TR-2007-144(2007).

[9] Whittaker E, Raj B., "Quantization-based language model compression," Proc. EuroSpeech 41, 33-36(2001).

[10] Andreas Stolcke, "SRILM- an extensible language modeling toolkit," Speech Technology and Research Laboratory SRI International, Menlo Park, CA, U.S.A., http://www.speech.sri.com/ (2002).

[11] Fang-Hui Chu, "Introduction to SRILM Toolkit," Department of Computer Science & Information Engineering National Taiwan normal University, (2005).

[12] Lee, Yue-Shi and Hsin-Hsi Chen, "Analysis of Error Count Distribution for Improving the Postprocessing Performance of OCCR," Communication of Chinese and Oriental Languages Information Processing Society 6(2), 81-86(1996).

[13] C. M. Takenga, K. R. Anne, K. Kyamakya, and J. C. Chedjou, "Comparison of gradient descent method, Kalman filtering and decoupled Kalman in training neural networks used for fingerprint-based positioning," Proc. IEEE VTC 6, 4146-4150(2004).