# STATISTICS-BASED SEGMENT PATTERN LEXICON - A NEW DIRECTION FOR CHINESE LANGUAGE MODELING

*Kae-Cherng Yang[1], Tai-Hsuan Ho[2], Lee-Feng Chien[3], Lin-Shan Lee[123]*

[1]Department of Electrical Engineering, National Taiwan University, Taiwan, R.O.C.
[2]Department of Computer Science and Information Engineering, National Taiwan University, Taiwan, R.O.C.
[3]Institute of Information Science, Academia Sinica, Taiwan, R.O.C.

## ABSTRACT

This paper presents a new direction for Chinese language modeling based on a different concept of lexicon. Because every Chinese character has its own meaning and there are no "blanks" in Chinese sentences serving as word boundaries, also because the wording structure in Chinese language is extremely flexible, the "words" in Chinese are actually not well defined, and there does not exist a commonly accepted lexicon. This makes language modeling very sophisticated in Chinese language, and the "out of vocabulary (OOV)" problem specially serious. A new concept for lexicon is thus proposed in this paper. The elements of this lexicon can be words or any other "segment patterns". They should be extracted from the training corpus by statistical approaches with a goal to minimize the overall perplexity. The language models can then be developed based on this new lexicon. Very encouraging experimental results have been obtained.

## 1. INTRODUCTION

The large vocabulary speech recognition technology has matured to a certain extent in recent years. Almost all the techniques are more or less word-based. The acoustic recognition is usually based on a word pronunciation lexicon, i.e., matching between the acoustic events in the utterances and the pronunciation of the words in a lexicon is usually performed. The linguistic decoding, in the other hand, usually operates under a word-based N-gram language model or some similar variants, i.e., the contextual association relations among words provide the most helpful constraints in finding the output sentences. Such word-based approaches make very good sense for western language, in which the words are well defined, and the words in sentences are actually separated by "blanks" serving as word boundaries.

The situation becomes quite different for Chinese language, which is not alphabetic at all. There are huge number of Characters (more than 10,000 are commonly used), almost each of which is an ideographic symbol with its own meaning. A "word" is composed of one to several characters, usually with a different meaning. Some of the words are "compositional" with the characters, i.e., the meaning of the word have to do with the meaning of the component characters, such as the characters "大 (big)" and "學 (learning)" forming a word "大學 (university)", but some other words are actually with the "ideographic" characters. Such as the characters "和 (harmony)" and "尙 (prefer)" forming a word "和尙 (monk)", i.e., the meaning of the word is completely different from the meaning of the component characters. Also, the wording structure in Chinese language are extremely flexible. For example, a long word can be arbitrarily abbreviated, such as "台灣大學 (Taiwan University)" being abbreviated as "台大", and new words can be easily generated everyday, such as the characters "電 (electricity)" and "腦 (brain)" forming a new word "電腦 (computer)". All these have to do with the fact that every character has its own meaning, and thus they can play some linguistic role independently. Furthermore, there are no "blanks" in Chinese sentences serving as word boundaries. As a result, the "word" in Chinese language is actually not well defined, and the segmentation of a sentence into a string of words as definitely not unique.

The above phenomena makes language modeling (which is in general word-based in western languages) much more sophisticated for Chinese language. Since words are not well defined, there does not exist a commonly accepted lexicon. Almost for any input texts there always exist large number of "out of vocabulary (OOV) words" regardless of how large a lexicon is used. Even if a fixed lexicon is given, training a word-based language model requires the whole training corpus being segmented into words at least "consistently", if not necessarily accurately. Some people believe the character-based language models can bypass the above different problem. In fact, character-based models have been found very useful for years. A good reason is that each character has its meaning and therefore character-based N-grams do make sense. However, it is also found that the capabilities of character-based N-grams are in fact limited, since the words constructed by a few characters very often carry different meanings, especially for the large number of the "ideographic words" as mentioned above. Apparently they can't be modeled very well by character-based N-grams unless N is large. In fact, the characters and words provide "double level information" in Chinese languages. Therefore the language modeling is significantly different from those for western languages.

In this paper, an initial study toward a new direction to handle the Chinese language modeling problem is presented. The language model developed here is neither word-based, nor character-based. A new lexicon is certainly needed. The elements in this new lexicon can be either words, or phrases, proper nouns, compound words, commonly accepted templates, etc., many of which are "out of vocabulary (OOV)" for most conventional lexicons. Such elements in this new lexicon are called "segment patterns" of characters in this paper, and should be extracted form the training corpus from the training corpus by statistical approaches, with a goal to find a best set of elements for N-gram models which can minimize the overall perplexity. The language models can then be developed based on these "segment patterns". All the detailed algorithms to extract the "segment patterns" and estimation of the N-gram parameters are presented in the sections below. Preliminary results on some initial experiments are also given. It was found that many "out of vocabulary (OOV) patterns" can be obtained in this way, and


0-7803-4428-6/98 $10.00 © 1998 IEEE

very good model performance can be achieved at a much smaller parameter size, since many of the infrequently used words can be actually deleted from a conventional lexicon. This approach also leads to a "live lexicon", since the most updated new words or new "segment patterns" can automatically be extracted from most current test corpus.

## 2. SEGMENT PATTERN EXTRACTION

This segment pattern extraction approach proposed here is based on the average Kullback-Leibler distance [1] to extract the most appropriate segment patterns from the training corpus. Both prefix and suffix trees are constructed from the training corpus, such that all character strings occurring in the training corpus can be stored and retrieved very efficiently. In order not to obtain an undesired local optimal solution, a two-phase approach is used, in which $V+R$ segment patterns are extracted in the first phase, and then the $R$ redundant or noisy segment patterns are deleted in the second phase, such that a finite lexicon with $V$ extracted entries will be obtained, where $V$ is the desired vocabulary size.

The initial lexicon for the extracted segment patterns is denoted as $\boldsymbol{L}(0) = \{ u_1=ch_1, u_2=ch_2, \ldots, u_C=ch_C \}$ where $u_i$ is the $i^{th}$ unit (or segment pattern) in the desired lexicon, $ch_i$ is the $i^{th}$ character in Chinese language, and $C$ is the total number of commonly used Chinese character been considered. More segment patterns are then extracted from the corpus and added to this lexicon. After $i$ iterations, total $i$ segment patterns have been extracted, and the lexicon becomes $\boldsymbol{L}(i) = \{u_1=ch_1, u_2=ch_2, \ldots, u_C=ch_C, u_{C+1}, \ldots, u_{C+i}\}$, where $u_{C+i}$ is $i^{th}$ segment pattern we extracted at the $i^{th}$ iteration. The $i^{th}$ segment pattern $u_{C+i}$ obtained in the $i^{th}$ iteration is the concatenation of two segment patterns in $\boldsymbol{L}(i-1)$ which can minimize the overall perplexity. This is achieved as follows. Let $\boldsymbol{T}(i)$ denote the set of all possible candidates for the $(i+1)^{th}$ segment pattern to be added to $\boldsymbol{L}(i)$ to construct $\boldsymbol{L}(i+1)$ in the $(i+1)^{th}$ iteration. Thus $\boldsymbol{T}(i)$ contain all possible concatenations of any two segment patterns in $\boldsymbol{L}(i)$ except for those concatenations which didn't occur at all in the training corpus. Let $t_j = (u_{j,1}, u_{j,2})$ be the $j^{th}$ concatenation (or candidate) in $\boldsymbol{T}(i)$ where $u_{j,1}, u_{j,2}$ are the two component segment patterns, the average Kullback-Leibler distance for $t_j$ can be evaluated as follows, where $(s_1, s_2, \ldots s_N)$ is a string of segment patterns occurring in the training corpus, $s_k$, $k=1,\ldots,N$, is any segment pattern in $\boldsymbol{L}(i)$, with one of them to be replaced by $t_j$.

$$d(t_j) = \sum_{k=1}^{N} d_k(t_j) \qquad (1)$$

where

$$d_1(t_j) = \sum_{(s_2, s_3, \ldots, s_N)} P(t_j, s_2, s_3, \ldots, s_N) \log \frac{P(s_N | t_j, s_2, s_3, \ldots, s_{N-1})}{P(s_N | u_{j,2}, s_2, s_3, \ldots, s_{N-1})} \qquad (2)$$

$$d_k(t_j) = \sum_{\substack{(s_1, s_2, \ldots, s_N) \\ s_k = t_j}} P(s_1, s_2, s_3, \ldots, s_N) \log \frac{P(s_N | s_1, s_2, s_3, \ldots, s_{N-1})}{P(s_N | s_2, s_3, \ldots, s_{N-1})} \quad \text{for } 1 < k < N \qquad (3)$$

$$d_N(t_j) = \sum_{\substack{(s_1, s_2, \ldots, s_N) \\ s_N = t_j = (u_{j,1}, u_{j,2})}} P(s_1, s_2, s_3, \ldots, s_N) * \log \frac{P(s_N | s_1, s_2, s_3, \ldots, s_{N-1})}{P(u_{j,2} | s_2, s_3, \ldots, s_{N-1}, u_{j-1}) P(u_{j,1} | s_1, s_2, \ldots, s_{N-1})} \qquad (4)$$

The numerator of the log term in eq (2) is the N-gram probability for $s_N$ given history $(s_1, s_2, \ldots s_{N-1})$ if $t_j$ is in the lexicon and at the $s_1$ position, and the denominator is the N-gram probability if $t_j$ is not in the lexicon and therefore only $u_{j,2}$ (the second part of $t_j$) is considered. So eq (2) gives the difference caused when $t_j$ is either inside or outside of the lexicon. Similarly, eq (4) gives the distance if $t_j$ is at the $s_N$ position, while eq (3) is for $t_j$ at $s_k$ position, where $1 < k < N$. Therefore in the $(i+1)^{th}$ iteration, the distances $d(t_j)$ in eq (1) are evaluated for all possible candidates $t_j$ in $\boldsymbol{T}(i)$. Assume $t_o$ is the candidate in $\boldsymbol{T}(i)$ which gives the maximal average Kullback-Leibler distance, the $(i+1)^{th}$ segment pattern to be added to the lexicon $\boldsymbol{L}(i+1)$ is then $u_{C+i+1}=t_o$. This loop will be continued until $R+V$ patterns have been extracted.

In the implementation of this first phase, prefix and suffix trees were used to make the computation more efficient. One can traverse across the prefix and suffix trees to find all possible strings $(s_1, s_2, \ldots s_N)$ and estimate the necessary probabilities using the suffix tree only. After a new segment pattern $t_o = (u_{o,1}, u_{o,2})$ was extracted and added to the lexicon, these trees have to be reset so that all connected nodes of $(u_{o,1}, u_{o,2})$ must be merged into $t_o$ in both trees. Besides, some heuristic rules can also be used to reduce the size of $\boldsymbol{T}(i)$. For example, if the occurrence count a segment candidate $t_j$ in the training corpus is smaller than a threshold, it can be ignored and never considered.

The second phase is performed after all the sentences in the training corpus have been segmented based on the segment patterns obtained in the first phase and the N-gram parameters are evaluated accordingly. All these processes will be performed using a forward-backward algorithm to be presented in section 3 below. After that the new parameters thus obtained are needed to construct a new set of prefix and suffix trees. Now under the new environment based on the extracted segment patterns, the average Kullback-Leibler distances as in eqs (1-4) are evaluated again for all extracted segment patterns. $r$ out of the $V+R$ segment patterns with lowest distances are then found and deleted. After this procedure, the sentences in the corpus are re-segmented and the N-gram parameters are re-estimated. This loop will continue until all the $R$ redundant or noisy segment patterns are deleted. Finally, the lexicon with desired exactly $V$ segment patterns is obtained.

## 3. FORWARD-BACKWARD ALGORITHM FOR SENTENCE SEGMENTATION AND N-GRAM PARAMETER ESTIMATION

After a lexicon of segment patterns is obtained as described in the above session, all sentences in the training corpus should be properly segmented based on this lexicon and N-gram language model parameters evaluated accordingly. For this purpose, a forward-backward algorithm that has been used for other models [2][4] was developed here to integrate the sentence segmentation and parameter estimation processes together for segment pattern N-gram model.

Let $S_i = (u_{i,1}, u_{i,2}, \ldots, u_{i,N-1})$ denote the $i^{th}$ state in N-gram Markov chain model $\lambda$, which is defined as a string of segment patterns occurring in the training corpus, where $u_{i,k}$, $k = 1,\ldots,N-1$, is a segment pattern obtained in the lexicon above, and $l(S_i)$, $l(u_{i,k})$ denote the character length (number of character) of $S_i$ and $u_{i,k}$

respectively. Let $C_{m,n}$ denote a sequence of Chinese characters, $C_{m,n}=(C_m, C_{m+1}, \ldots, C_n)$. For a given sentence in the corpus $C_{1,T} = (c_1, c_2, \ldots, c_T)$ with $T$ characters, consider the forward variable $\alpha_t(S_i)$ defined as

$$\alpha_t(S_i) = P(C_{1,t}; C_{t-l(S_i)+1,t} = S_i | \lambda), \quad 0 \le t \le T \quad (5)$$

given the Markov chain model $\lambda$, $\alpha_t(S_i)$ is the probability for a character sequence $C_{1,t}$,whose last $l(S_i)$ characters are exactly $S_i$, $C_{t-l(Si)+1,t}=S_i$. We can solve for $\alpha_t(S_i)$ inductively as follows, where $S_j$ is any possible and legal state ( the first N-2 segment patterns of $S_j$ is the same as the last N-2 segment patterns of $S_i$) that one can go from $S_i$ to $S_j$ in the Markov chain model $\lambda$ for the given sentence $C_{1,T}$,

$$\alpha_0(\cdot) = 1 \quad (6)$$

$$\alpha_{t+l(u_{j,N-1})}(S_j) = \sum_{S_i} \alpha_t(S_i) P(S_j; C_{t+l(u_{j,N-1})-l(S_j),t+l(u_{j,N-1})} = S_j | S_i, \lambda) \quad (7)$$

and finally

$$P(C_{1,T}|\lambda) = \sum_{S_k} \alpha_T(S_k) \quad (8)$$

Sinilarly, consider the backward variable $\beta_t(S_i)$ defined as

$$\beta_t(S_i) = P(C_{t-l(S_i)+1,T} | C_{t-l(S_i)+1,t} = S_i, \lambda) \quad (9)$$

In other words, given the Markov model $\lambda$ $\beta_t(S_i)$ is the probability for a character sequence $C_{t-l(Si)+1,T}$, where first $l(S_i)$ character are exactly $S_i$, $C_{t-l(Si)+1,t} = S_i$. Again we can solve for $\beta_t(S_i)$ inductively as follows, where $S_j$ is any possible and legal state in the model for the given sentence,

$$\beta_0(\cdot) = 1 \quad (10)$$

$$\beta_t(S_i) = \sum_{S_j} \beta_{t+l(u_{j,N-1})}(S_j) P(S_j | S_i, \lambda) \quad (11)$$

Moreover, we can now define $\gamma_t(S_i)$ as the probability with $C_{t-l(Si)+1,t} = S_i$ given whole sentence $C_{1,T}$ and the model $\lambda$,

$$\gamma_t(S_i) = P(C_{t-l(S_i)+1,t} = S_i | C_{1,T}, \lambda) \quad (12)$$

We can write that

$$\gamma_t(S_i) = \frac{\alpha_t(S_i)\beta_t(S_i)}{P(C_{1,T}|\lambda)} \quad (13)$$

Furthermore, we can define $\xi_t(S_i,S_j)$ as the transition probability from $S_i$ to $S_j$ at the $t^{th}$ and $(t+1)^{th}$ characters given the whole sentence $C_{1,T}$ and the model $\lambda$,

$$\xi_t(S_i, S_j) = P(C_{t-l(S_i)+1,t+l(u_{j,N-1})} = S_i, S_j | C_{1,T}, \lambda) \quad (14)$$

and it can be written as

$$\xi_t(S_i, S_j) = \frac{\alpha_t(S_i) P(S_j | S_i, \lambda) \beta_{t+l(u_{j,N-1})}(S_j)}{P(C_{1,T}|\lambda)} \quad (15)$$

With all the above derivations, for the training corpus with sentences set $\boldsymbol{C}=\{C^k\}$, where $C^k$ is the $k^{th}$ sentence in $\boldsymbol{C}$. The probability of $P(S_j | S_i, \lambda)$ at the $m^{th}$ iteration is estimated as follows

$$P^{(m)}(S_j | S_i, \lambda_{m-1}) = \frac{\sum_{C^k}\sum_t \xi_t^{(m-1)}(S_i, S_j)}{\sum_{C^k}\sum_t \gamma_t^{(m-1)}(S_i)} \quad (16)$$

If we denote $[S_i,S_j]$ as a string of N segment patterns, $[S_i,S_j]= (u_{ij,1}, u_{ij,2}, \ldots, u_{ij,N})$, whose first N-1 segment patterns are exactly $S_i$, i.e., $S_i = (u_{i,1}, u_{i,2}, \ldots, u_{i,N-1}) = (u_{ij,1}, u_{ij,2}, \ldots, u_{ij,N-1})$, while whose last N-1 segment patterns are exactly $S_j$, i.e., $S_j = (u_{j,1}, u_{j,2}, \ldots, u_{j,N-1}) = (u_{ij,2}, u_{ij,3}, \ldots, u_{ij,N})$, then the numerator of (16) can be regarded as the frequency count for the N-gram parameter $[S_i,S_j] = (u_{ij,1}, u_{ij,2}, \ldots, u_{ij,N})$ occurring in the training corpus, and the denominator of (16) is the count for $S_i = (u_{i,1}, u_{i,1}, \ldots, u_{i,N-1}) = (u_{ij,1}, u_{ij,2}, \ldots, u_{ij,N-1})$ occurring in the training corpus.

## 4. EXPERIMENTAL RESULTS

In the preliminary experiments, we wish to evaluate the performance of language models based on the new concept of segment pattern lexicon developed here, and compare the results with those using traditional Chinese character set and word lexicon designed by Chinese linguists [3]. The training data used include 6 millions of Chinese characters. Three families of language models are developed based on extracted segment patterns obtained here, characters, and words provided by CKIP [3] respectively. The test data include 500K Chinese characters. Both the training and test data are taken from local newspapers provided by CKIP.

Language model perplexity is used as the performance measure. A character-based perplexity [4] measure specially defined for Chinese language is used here in order to achieve a meaningful comparison among different families of language models. The definition of the character-based perplexity is straightforward,

$$PP_C = \exp\{-\frac{1}{N_C}\sum_{C^k} \log P(C^k|\lambda)\}$$

where $N_C$ is the total number of characters in the testing data, and the language model probability, $P(C^k|\lambda)$, of a given sentence $C^k$ is calculated by (8) for the segment pattern models developed here. Sparseness problem is inevitable when estimating the language model probabilities for the test data. The discounting model [5] was adopted to estimate the probabilities for unseen events.

In order to reduce the computation complexity in segment pattern extraction, we confined the length the extracted segment patterns within 4 Chinese characters, since it was observed that most of the longer patterns can be decomposed into two or more smaller patterns in Chinese language. In the processes of segment pattern extraction, 163K segment patterns (including roughly 13K characters, each of which is taken as a mono-character segment patttern) were first extracted from the training corpus in the first phase. The second phase of the processes then removed all the unpromising patterns using the average Kullback-Leibler distance criterion as discussed above, and a series of segment pattern lexicons of size ranging from 33K to 113K for every 10K patterns were generated.

The CKIP word lexicon [3], a Chinese lexicon specially designed by Chinese linguists, is taken as the baseline for comparisons in our experiments. This CKIP lexicon contains 94,188 word entries. It was found that among our 100K extracted multi-character segment patterns, only 33,111 patterns are also present in the CKIP word lexicon, and the other roughly 67K of segment pattern are meaningful phrases, proper nouns, compound words, commonly used templates, and many new words not collected in the CKIP lexicon. A nice feature of the approach is that the most updated wording can almost be extracted automatically, as long as existing

in the training corpus. For example, a new word "長紅 (red for long, which means high in stock market for long)" which has been frequently used only recently but not included in the CKIP word lexicon, was extracted with very high priority.

In the first experiment, we compare the bigram and trigram perplexities based on the segment pattern lexicon obtained here and the words in CKIP lexicon. The result is shown in table 1. For bigram language models, it can be found that the for the extracted segment patterns perplexity decreases as the lexicon size is increased from 33K to 83K, and then increases slightly when we continue to increase the lexicon size. Similar situation happens in the trigram case, and the lowest perplexity appears at the lexicon size of 63K. The larger lexicon size does not bring more benefits from higher coverage rates probably due to the sparseness of the training data. Besides, for the bigram models, the segment pattern based model is always better than the CKIP based model, even with a much smaller vocabulary size, which implies much smaller memory and computation requirements as well. However, for the trigram models, performance difference seems to be very limited, probably due to the same reason of the sparseness of the training corpus.

In the second experiment, perplexities are calculated for character based N-grams for N=2,3,4. The results and the number of parameters required are listed in table 2. From this table, it can be found that character based trigram model is significantly better than the bigram model with a perplexity reduction on the order of 46%. Since the average length for Chinese words is about 1.8 characters as measured from CKIP data, the capabilities of character bigram is certainly limited. By comparing tables 1 and 2, it is also observed that the perplexity of character based trigram is roughly comparable with that of the word based bigram models, but with at least 80% more parameters. Similarly, the situation for character based 4-gram can also be found in table 2, which does not performs as well as the word trigram, but with much more parameters. This confirms the concept mentioned earlier that language models based on characters alone will be limited in capabilities in any case. Since many of the words or segment patterns really carries more information which is beyond the character level. As a result, selecting a good set of words or including properly chosen segment patterns in a lexicon will be a very critical task in Chinese language modeling.

Finally, all models mentioned above are trained by the forward-backward algorithm iteratively. The initial model at iteration 0 is bootstrapped by segmenting the sentences into words or patterns using a simple longest-match heuristic algorithm. Table 3 show the perplexity versus number of iterations using the 83K segment pattern lexicon as an example. It can be observed that the perplexity begins to saturate at the second iteration. All the models mentioned above have gone through such training process until the test data perplexity is converged.

## 5. CONCLUSION

In this paper, an effective algorithm is presented to extract segment patterns automatically from the training data to construct a lexicon specially useful for large vocabulary speech recognition, with a goal to minimize the overall perplexity. A forward-Backward training algorithm is also proposed to integrate the sentence segmentation and parameter estimation processes. In the experiments, very encouraging results were obtained. It is thus believed that such segment lexicon can preserve more important information to model the Chinese language better, and at the same time much of the redundant information can be deleted to reduce the parameter size. Besides, such a lexicon can also be "live" to include the most updated segment patterns extracted from most current text corpus. This is only an very initial study. Much more further studies regarding various issues of language models are certainly needed in the future.

## 6. REFERENCES


[1] R. Kneser, "Statistical Language Modeling Using a Variable Context Length", Proc. ICSLP, 1996, vol I, pp 494-497

[2] S. Deligne, F. Bimbot, "Language Modeling by Variable Length Sequences: Theoretical Formulation and Evaluation of Multigrams", Proc. ICASSP, 1995, vol I, pp. 169-172

[3] Chinese Knowledge Information Group, *Technical Report No. 93-02.* Institute of Information Science, Academia Sinica, Taiwan.

[4] H. Law, C. Chan, "Ergodic Multigram HMM Integrating Word Segmentation And Class Tagging For Chinese Language Modeling", Proc. ICASSP, 1996, vol I, pp. 196-199

[5] H. Ney, U. Essen, R. Kneser, "On Structuring Probabilistic Dependences In Stochastic Language Modeling", Computer Speech and Language, 1994, vol.8, pp. 1-38


| Iteration | 0 | 1 | 2 | 3 | 4 | 5 |
|---|---|---|---|---|---|---|
| PP | 52.74 | 51.48 | 51.33 | 51.32 | 51.31 | 51.31 |

Table 3. Perplexity of pattern bigram for 0-5 iterations using extracted lexicon with 70K patterns, where longest-match rule is used at iteration 0.

| Lexicon | CKIP | Extracted Segment Pattern Lexicon | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| | 94K | 33K | 43K | 53K | 63K | 73K | 83K | 93K | 103K | 113K |
| Bigram | 54.88 | 54.86 | 53.07 | 52.13 | 51.73 | 51.42 | 51.31 | 51.36 | 51.43 | 51.57 |
| Trigram | 43.22 | 43.54 | 43.32 | 43.19 | 43.11 | 43.24 | 43.68 | 44.25 | 44.97 | 46.02 |

Table 1. Character-based perplexity values of bigram and triigram Using CKIP word lexicon and the extracted segment pattern lexicons with different sizes

| Model Order | Character Based N-gram | | | CKIP Word Based N-gram | | |
|---|---|---|---|---|---|---|
| | Bigram | Trigram | 4-gram | Unigram | Bigram | Trigram |
| PP | 97.99 | 52.74 | 47.81 | 150.36 | 54.88 | 43.22 |
| Parameter No. | 0.5M | 1.8M | 4.3M | 43K | 1.0M | 2.8M |

Table 2. Character-based perplexity of character N-gram and word N-gram using CKIP lexicon