A Statistical Language Model for Pre-Trained Sequence Labeling: A Case Study on Vietnamese

XIANWEN LIAO, YONGZHONG HUANG, PENG YANG, and LEI CHEN, Guilin University of Electronic Technology

By defining the computable word segmentation unit and studying its probability characteristics, we establish an unsupervised statistical language model (SLM) for a new pre-trained sequence labeling framework in this article. The proposed SLM is an optimization model, and its objective is to maximize the total binding force of all candidate word segmentation units in sentences under the condition of no annotated datasets and vocabularies. To solve SLM, we design a recursive divide-and-conquer dynamic programming algorithm. By integrating SLM with the popular sequence labeling models, Vietnamese word segmentation, part-of-speech tagging and named entity recognition experiments are performed. The experimental results show that our SLM can effectively promote the performance of sequence labeling tasks. Just using less than 10% of training data and without using a dictionary, the performance of our sequence labeling framework is better than the state-of-the-art Vietnamese word segmentation toolkit VnCoreNLP on the cross-dataset test. SLM has no hyper-parameter to be tuned, and it is completely unsupervised and applicable to any other analytic language. Thus, it has good domain adaptability.

CCS Concepts: • **Computing methodologies** → **Natural language processing**; *Phonology / morphology*;

Additional Key Words and Phrases: Unsupervised, statistical language model, sequence labeling

ACM Reference format:

Xianwen Liao, Yongzhong Huang, Peng Yang, and Lei Chen. 2021. A Statistical Language Model for Pre-Trained Sequence Labeling: A Case Study on Vietnamese. *ACM Trans. Asian Low-Resour. Lang. Inf. Process.* 21, 3, Article 50 (December 2021), 21 pages.
<https://doi.org/10.1145/3483524>

1 INTRODUCTION

Sequence labeling is a universal framework for **natural language processing (NLP)**. It has been applied to many NLP tasks including word segmentation performed in analytic language processing, **part-of-speech (POS)** tagging, noun phrase chunking, and **named entity recognition (NER)** [5, 21, 40]. These tasks play very important roles in NLP and fulfill lots of downstream applications, such as relation extraction, syntactic parsing, and entity linking [18, 19]. Due to the

This work was supported by the National Natural Science Foundation of China under grants 61066008 and 61862011, and the Basic and Applied Basic Research Fund of Guangdong Province, China, under grant 2019B1515120085. This work was also supported by Guangxi Science and Technology Plan Project under grant AD18281065.

Authors' address: X. Liao, Y. Huang (corresponding author), P. Yang, and L. Chen, Guilin University of Electronic Technology, 1 Jinji Road, Guilin, Guangxi, China, 541004; emails: {279161795, 2389483289}@qq.com, ypycsy@163.com, 252736019@qq.com.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than ACM must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from permissions@acm.org.

© 2021 Association for Computing Machinery.

2375-4699/2021/12-ART50 \$15.00

<https://doi.org/10.1145/3483524>

S_0 :	Philadelphia sẽ cập nhật kết quả bỏ phiếu qua thư sau vài giờ								
$\{S_i\}$:	Philadelphia/sẽ/cập nhật/kết quả/bỏ phiếu/qua/thư/sau/vài giờ								
h_i :	0	0	2.37	2.63	1.83	0	0	0	1.56
n_i :	n_0	n_1	n_2	n_3	n_4	n_5	n_6	n_7	n_8

Fig. 1. A feasible solution of SLM under a specific input sentence. The objective value of SLM is calculated by $\sum_{i=0}^n n_i \times h_i$. $\{S_i\}$ represents the candidate WSU set obtained by splitting the input sentence. h_i and n_i represent the binding force and the occurrence number of WSU_i in the corpus.

rapid development of deep learning in recent years, deep learning methods have replaced traditional statistical learning methods (e.g., CRF and HMM) as the mainstream methods of sequence labeling, and have achieved much better performance in many tasks.

Training a neural network for sequence labeling requires a large amount of annotated data, and usually the amount of required data increases with the network size. Unfortunately, building large annotated datasets are expensive, and we need well-trained manpower to do the annotation [2]. Even though some labeled datasets are available, most of them are domain related. So the domain adaptability of trained model is greatly limited, especially for low-resource languages [31], such as Vietnamese and Thai. To solve the problems of domain adaptability and insufficiency of labeled data, researchers have exploited many different methods to draw information from other easily available datasets to improve the performance of deep learning models. These methods can be roughly divided into four categories. The first is to train a neural network language model using a large amount of non-labeled text to obtain word embeddings containing rich semantic and grammatical information, such as Word2Vec, GloVe (static word embedding), ELMO, and BERT (dynamic word embedding) [8, 24, 28, 29]. The second is to use additional knowledge to improve the performance of sequence labeling models [10, 31, 40]. The third are transfer learning and meta-learning [2, 38]. The fourth is multi-task learning [19, 34].

In this article, we propose an unsupervised **statistical language model (SLM)** to improve the performance of sequence labeling models in the context of low-resource analytic language. We choose Vietnamese as a study case. Our SLM is an optimization model based on the co-occurrence of n -gram words. It aims at maximizing the total binding force of all candidate **word segmentation units (WSUs)** in sentences. The solution of SLM is to optimally divide the given sentence into relatively independent semantic and grammatical units, namely WSU, which is described in Section 4.1. Figure 1 illustrates a feasible solution of SLM under the specific input. Experiments on Vietnamese datasets show that these pre-divided WSUs can deliver useful information to sequence labeling models and improve their performance. The advantages of our proposed unsupervised SLM are as follows:

- It is completely unsupervised. It does not require any labeled data and other additional knowledge, such as vocabularies and gazetteers.
- It can ignore the differences among different languages and deal with analytic languages indiscriminately, such as Chinese, Vietnamese, Thai, and Burmese.
- It has good domain adaptability, as long as enough task-specific plain texts are collected.
- It has no hyper-parameters.
- Our proposed sequence labeling framework is task independent and loose coupling. SLM can be integrated with a variety of machine learning models, such as SVM, CRF, BiLSTM, and BERT. Our codes and data are available at https://github.com/Liaoxianwen/SLM_code_data.

Fig. 2. The pre-trained sequence labeling framework consisting of an unsupervised SLM and the upper machine learning model.

2 RELATED WORK

How to improve the performance of sequence labeling tasks is a current research hotspot. These sequence labeling tasks typically include word segmentation, POS labeling, and NER [5, 7, 14, 17, 25, 39]. The means of improvement include semi-supervision, knowledge enhancement, multi-tasking, transfer learning, and meta-learning. In the AAAI 2020 conference, a total of 13 sequence labeling papers were accepted, 11 of which used comprehensive models and obtained SOTA on multiple datasets. In these 11 papers, 7 use semi-supervised or knowledge-enhancement models [1, 10, 16, 20, 23, 30, 31], 2 use multi-task learning models [12, 34], and 2 use transfer learning and meta-learning models [2, 38]. The pre-trained sequence labeling framework proposed in this article belongs to the category of semi-supervision or knowledge enhancement.

In those semi-supervised or knowledge enhancement deep learning models, the types of data are different. In the work of Ali et al. [1], a knowledge graph was used to train an edge-weighted attentive graph convolution network to refine noisy mention representations. In the work of Liu et al. [20], distant supervision was generalized by extending dictionaries with headwords. Knowledge graph augmented word representation (KAWR) was introduced for NER by encoding the prior knowledge of entities from an external knowledge base [10]. A domain adaptation method using a Bayesian sequence combination was proposed in the work of Simpson et al. [31] to exploit pre-trained models and unreliable crowdsourced data. In the work of Mayhew et al. [23], a standard BiLSTM-CRF model was integrated with a pretrained truecaser for NER. A semi-supervised learning approach based on multi-view models through consensus promotion was proposed in the work of Lim et al. [16] for multi-task tagging. In the work of Safranchik et al. [30], multiple heuristic rules were used to establish a weak supervision framework for sequence labeling tasks.

At present, SLM has been replaced by the neural language model in NLP. However, SLM still plays its role. A statistical model based on linguistic knowledge was proposed in the work of

Somsap and Seresangtakul [32] to overcome word ambiguity and OOVs in Isarn Dharma word segmentation. The sum of probability values in possible word sequences was used to determine the best-segmented sentence. In the work of Ghodsi and DeNero [9], some key structural properties of language were studied, and a conclusion that RNNs had a structural capacity largely missing from n -gram models was drawn. n -Gram and statistical features were used to identify fake news spreaders on Twitter in the work of Buda and Bolonyai [4]. n -Gram statistics and location-related dictionaries were used to extract location name from target text streams [3]. In the work of Marie et al. [22], a unsupervised neural and statistical machine translation system was implemented. In the work of Varjokallio and Klakow [35], unsupervised morph segmentation and vocabulary expansion were done by the combination of a trigram language model and RNN.

In summary, under the trend of semi-supervision and knowledge enhancement in sequence labeling, we can draw a conclusion that the information extracted from a large corpus by SLM is still useful. The combination of SLM and a neural network can improve sequence labeling tasks' performance, especially in the case of low resource. The experimental results of three Vietnamese sequence labeling tasks show that our pre-trained framework can effectively improve sequence labeling tasks' performance.

3 SLM FOR THE PRE-TRAINED SEQUENCE LABELING FRAMEWORK

The architecture of our proposed pre-trained sequence labeling framework is shown in Figure 2. The proposed framework is composed of SLM and the upper machine learning model. For a sentence $x = (x_1, \dots, x_n)$, its label sequence is $y = (y_1, \dots, y_n)$, where x_i is the i th word¹ in x and y_i is the label of x_i . Before using (x, y) to train a sequence labeling model (e.g., SVM, CRF, and BiLSTM+CRF), we input x into SLM and get a preliminary division—for instance, $x_1/x_2x_3/x_4x_5x_6/\dots/x_{n-2}/x_{n-1}x_n$. This preliminary division marks the boundaries of basic semantic and grammatical units (named *WSUs*) in x . For a *WSU* $w_i = x_i x_{i+1} \dots x_{i+k}$, we use the BIES schema² to mark the position of every x_j in w_i . More specifically, if x_j is at the beginning and the end of w_i , we label x_j with B and E. If x_j is in the middle of w_i , we label it with I. If x_j is a single word in w_i , we label it with S. When we obtain the output of SLM, we convert the position symbols (a tag sequence make up of B, I, E, or S) into position encodings and splice them with the corresponding word embeddings to form the final embeddings. The final embeddings are input into the upper machine learning model.

4 THE PROPOSED UNSUPERVISED SLM

In this section, we define the computable *WSU* and derive two theorems and one corollary from the definition. Based on the derived theorems, we establish an optimization model with the objective of maximizing the total binding force of all candidate *WSUs* in sentences. Finally a recursive divide-and-conquer algorithm is designed to solve SLM.

4.1 The Definition of *WSU* and the Derived Theorems

In ISO 24614-1:2010, *WSU* was defined as a sequence of morphemes or other characters that were treated as a unit, such as “take care of,” “blackboard,” and “Apple pie” in English, or “thi gian” and “Cô gái” in Vietnamese. In Chinese word segmentation specification GB/T 13715-92, *WSU* was defined as basic unit with definite semantic or grammatical function, other qualified words,

¹For languages like English and Vietnamese, x_i represents a word, but for Chinese, x_i represents a Chinese character. Word and character are not distinguished in this article.

²The detail of tagging schema can be seen in the work of Xue [39].

Fig. 3. The relationship between S_{WS_2} and $\bar{S}_{WS_2}$. The operator $\leftarrow$ represents using m bigram word $W_A W_B$ to randomly replace m elements of $\bar{S}_{WS_2}$. $P_{\bar{S}}(W_A)$ and $P_{\bar{S}}(W_B)$ represent the probability of W_A and W_B in the corpus before the replacement. $P(W_A)$ and $P(W_B)$ represent the probability of W_A and W_B in the corpus after the replacement. Since the corpus is large enough, randomly replacing m elements in $\bar{S}_{WS_2}$ with m $W_A W_B$ does not change the probability of W_A and W_B . So we consider that $P_{\bar{S}}(W_A) = P(W_A)$ and $P_{\bar{S}}(W_B) = P(W_B)$.

and phrases. From the atomic characteristics of WSU's definition, it can be inferred that WSU is a morpheme (word) string that combines tightly and is used stably.

In the case of available annotated corpora and vocabularies, WSU is defined by the annotated corpora, vocabularies, and word segmentation specifications [13]. Under the condition of without any annotated corpora, vocabularies, and word segmentation specifications, we define WSU according to its statistical characteristics.

Definition 4.1 (WSU). Let Ω be the word set of an analytic language and W_i be a word, $W_i \in \Omega$, $1 \leq i \leq |\Omega|$. Let $s^{(1,k)} = W_1 W_2 \dots W_k$. Let $P(W_i)$ be the probability of W_i and $P(s^{(1,k)})$ be the probability of $s^{(1,k)}$, $2 \leq k \leq L$, where L is the maximum WSU length under consideration. If $P(s^{(1,k)}) > P(W_1)P(W_2) \dots P(W_k)$, then $s^{(1,k)}$ is called a WSU.

Definition 4.1 is equivalent to the following description. Let S_{WS_k} be a k -gram word set of the corpus. Suppose that when $W_1, W_2, \dots, W_k$ are independent the number of $s^{(1,k)}$ in S_{WS_k} is N_{k0} , and when $W_1, W_2, \dots, W_k$ are not independent the number of $s^{(1,k)}$ in S_{WS_k} is N_{k1} . If $N_{k1} > N_{k0}$, then $s^{(1,k)}$ in S_{WS_k} is a WSU.

THEOREM 4.2 (THE PROBABILITY RELATION OF BIGRAM WSU). Let $W_A, W_B \in \Omega$. Then the necessary and sufficient condition for $W_A W_B$ to be a bigram WSU is $P(W_A W_B) > P(W_A)P(W_B)$.

PROOF. Let us prove the necessity first. In other words, if $W_A W_B$ is a bigram WSU, then $P(W_A W_B) > P(W_A)P(W_B)$.

Suppose there is a bigram word set $\bar{S}_{WS_2}$ ³, and W_A and W_B are independent in $\bar{S}_{WS_2}$. Let $N_2 = |\bar{S}_{WS_2}|$, which can be calculated by Equation (10). Now we use m bigram word $W_A W_B$ to randomly replace m elements of $\bar{S}_{WS_2}$. This is shown in Figure 3. After the replacement, we get another bigram word set S_{WS_2} . In S_{WS_2} , W_A and W_B are not independent and $W_A W_B$ is a bigram WSU (because there are more $W_A W_B$ s after the replacement than before). The probability of $W_A W_B$ in S_{WS_2} is calculated by

$$P(W_A W_B) = \frac{C(W_A W_B)}{N_2}. \quad (1)$$

³ S_{WS_2} is a virtual set, and it is used to represent the bigram word space of the corpus. S_{WS_2} is only used to represent the relationship between two words. Some bigram words in S_{WS_2} are bigram WSUs, and some are just the random combinations of two words. The concept of S_{WS_2} can be generalized to S_{WS_k} . S_{WS_k} is different from k -gram co-occurrence word set S_k that can actually be constructed.

In Equation (1), $C(W_A W_B)$ is the number of $W_A W_B$ in $S_{W_{S_2}}$. It is calculated by (also shown in Figure 3)

$$C(W_A W_B) = P_{\bar{S}}(W_A W_B) (N_2 - m) + m. \quad (2)$$

In Equation (2), $P_{\bar{S}}(W_A W_B)$ represents the probability of $W_A W_B$ in $\bar{S}_{W_{S_2}}$. Because W_A and W_B are independent in $\bar{S}_{W_{S_2}}$, we get

$$P_{\bar{S}}(W_B | W_A) = P_{\bar{S}}(W_B)$$

and

$$P_{\bar{S}}(W_A W_B) = P_{\bar{S}}(W_A) P_{\bar{S}}(W_B | W_A) = P_{\bar{S}}(W_A) P_{\bar{S}}(W_B). \quad (3)$$

By combining Equation (1), Equation (2), and Equation (3), we can obtain

$$\begin{aligned} P(W_A W_B) &= \frac{P_{\bar{S}}(W_A) P_{\bar{S}}(W_B) (N_2 - m) + m}{N_2} \\ &= \frac{P_{\bar{S}}(W_A) P_{\bar{S}}(W_B) N_2}{N_2} - \frac{P_{\bar{S}}(W_A) P_{\bar{S}}(W_B) m}{N_2} + \frac{m}{N_2} \\ &= P_{\bar{S}}(W_A) P_{\bar{S}}(W_B) + \frac{m}{N_2} (1 - P_{\bar{S}}(W_A) P_{\bar{S}}(W_B)). \end{aligned}$$

Since $m/N_2 > 0$ and $1 - P_{\bar{S}}(W_A) P_{\bar{S}}(W_B) > 0$, so we have $P(W_A W_B) > P_{\bar{S}}(W_A) P_{\bar{S}}(W_B)$. As described in Figure 3, we consider that $P_{\bar{S}}(W_A) = P(W_A)$ and $P_{\bar{S}}(W_B) = P(W_B)$. So we obtain $P(W_A W_B) > P(W_A) P(W_B)$.

Next, let us prove the sufficiency. In other words, if $P(W_A W_B) > P(W_A) P(W_B)$, then $W_A W_B$ is a bigram WSU.

Definition 4.1 clearly makes the sufficiency true. $\square$

THEOREM 4.3 (THE PROBABILITY RELATION OF TRIGRAM WSU). *Let W_A, W_B , and $W_C \in \Omega$, $W_A W_B$ be a bigram word. The necessary and sufficient condition for $W_A W_B W_C$ to be a trigram WSU is $P(W_A W_B W_C) > P(W_A W_B) P(W_C)$.*

PROOF. The proof of Theorem 4.3 is the same as the proof of Theorem 4.2. Let us prove the necessity first. In other words, if $W_A W_B W_C$ is a trigram WSU, then $P(W_A W_B W_C) > P(W_A W_B) P(W_C)$.

Suppose there is a trigram word set $\bar{S}_{W_{S_3}}$, and $W_A W_B$ and W_C are independent in $\bar{S}_{W_{S_3}}$. Let $N_3 = |\bar{S}_{W_{S_3}}|$, which can be calculated by Equation (9). Now we use m trigram word $W_A W_B W_C$ s to randomly replace m elements of $\bar{S}_{W_{S_3}}$. After the replacement, we get another trigram word set $S_{W_{S_3}}$. In $S_{W_{S_3}}$, $W_A W_B W_C$ is a trigram WSU (because there are more $W_A W_B W_C$ s after the replacement than before). The probability of $W_A W_B W_C$ in $S_{W_{S_3}}$ is calculated by

$$P(W_A W_B W_C) = \frac{C(W_A W_B W_C)}{N_3}. \quad (4)$$

In Equation (4), $C(W_A W_B W_C)$ is the number of $W_A W_B W_C$ in $S_{W_{S_3}}$. It is calculated by (similar to Figure 3)

$$C(W_A W_B W_C) = P_{\bar{S}}(W_A W_B W_C) (N_3 - m) + m. \quad (5)$$

In Equation (5), $P_{\bar{S}}(W_A W_B W_C)$ represents the probability of $W_A W_B W_C$ in $\bar{S}_{W_{S_3}}$. Because $W_A W_B$ and W_C are independent in $\bar{S}_{W_{S_3}}$, we get

$$P_{\bar{S}}(W_C) = P_{\bar{S}}(W_C | W_A W_B)$$

and

$$P_{\bar{S}}(W_A W_B W_C) = P_{\bar{S}}(W_A W_B) P_{\bar{S}}(W_C | W_A W_B) = P_{\bar{S}}(W_A W_B) P_{\bar{S}}(W_C). \quad (6)$$

By combining Equation (4), Equation (5), and Equation (6), we can obtain

$$\begin{aligned}
 P(W_A W_B W_C) &= \frac{P_S(W_A W_B) P_S(W_C) (N_3 - m) + m}{N_3} \\
 &= \frac{P_S(W_A W_B) P_S(W_C) N_3}{N_3} - \frac{P_S(W_A W_B) P_S(W_C) m}{N_3} + \frac{m}{N_3} \\
 &= P_S(W_A W_B) P_S(W_C) + \frac{m}{N_3} (1 - P_S(W_A W_B) P_S(W_C)).
 \end{aligned}$$

Since $m/N_3 > 0$ and $1 - P_S(W_A W_B) P_S(W_C) > 0$, we have $P(W_A W_B W_C) > P_S(W_A W_B) P_S(W_C)$. As described in Figure 3, we consider that $P_S(W_A W_B) = P(W_A W_B)$ and $P_S(W_C) = P(W_C)$. So we obtain $P(W_A W_B W_C) > P(W_A W_B) P(W_C)$.

Next, let us prove the sufficiency. In other words, if $P(W_A W_B W_C) > P(W_A W_B) P(W_C)$, then $W_A W_B W_C$ is a trigram WSU.

We treat $W_A W_B$ as a unit, and Definition 4.1 clearly makes the sufficiency true. $\square$

According to the proofs of Theorem 4.2 and Theorem 4.3, we can easily draw a conclusion that the necessary and sufficient conditions for W_A , W_B , and W_C to be a trigram WSU is $P(W_A W_B W_C) > P(W_A) P(W_B) P(W_C)$, and for W_A and $W_B W_C$ to be a trigram WSU $W_A W_B W_C$ is $P(W_A W_B W_C) > P(W_A) P(W_B W_C)$. We can generalize the case to n -gram.

COROLLARY 4.4. *If $s^{(1,k)} = W_1 W_2 \dots W_k$ is a k -gram WSU, then $P(s^{(1,k)}) > \prod_i P(s_i)$, where $s_i \in S_{s^{(1,k)}}$ and $s_i \neq \emptyset$, $S_{s^{(1,k)}}$ is an arbitrary end-to-end segment of $s^{(1,k)}$ —that is, $\sum_{i=1}^{n_k} s_i = s^{(1,k)}$, $n_k = |S_{s^{(1,k)}}|$.*

Corollary 4.4 is a generalization of Theorem 4.2 and Theorem 4.3. According to the proofs of the two theorems, if we treat s_i as a unit, Corollary 4.4 can also be proved. The function of Corollary 4.4 is to determine whether a k -gram word constitutes a WSU. It is used to construct one of SLM's constraints.

4.2 SLM

In this section, we first define the binding force of k -gram word represented by $W_1 W_2 \dots W_k$, and describe how to construct k -gram co-occurrence word set that is used to estimate the size of k -gram word set of the corpus. Next, we define $P(W_1 W_2 \dots W_k)$ and construct SLM with the objective of maximizing the total binding force of all candidate WSUs in sentences.

Definition 4.5 (Binding Force). For a word string $s^{(1,k)} = W_1 W_2 \dots W_k$, $W_i \in \Omega$, its binding force is denoted by $H(s^{(1,k)})$, which is defined as

$$H(s^{(1,k)}) = \lg \frac{P(s^{(1,k)})}{\prod_i P(W_i)}. \quad (7)$$

The binding force is used to indicate how tightly a word string combines. Generally, if the number of a word string in the corpus is larger, its binding force may be larger. Definition 4.5 is the generalized definition of binding force defined in the work of Sproat and Shih [33], and we use 10 as the base of the logarithm in Equation (7).

Since it is difficult to obtain all k -gram words of the corpus, it is almost impossible for us to estimate the exact size of k -gram word set. As an alternative, we construct the k -gram co-occurrence word set of the corpus to solve this problem. The bigram and trigram co-occurrence word sets are constructed by intercepting two and three consecutive words from sentences. For a Vietnamese sentence "Xem Danh sách các trường đại học ở Ohio," its derived bigram co-occurrence word

Fig. 4. The distribution of trigram co-occurrence words in the sentence.

set is {"Xem Danh," "Danh sách," "sách các," "các trường," "trường đại," "đại học," "học ở," "ở Ohio"}, and its derived trigram co-occurrence word set is {"Xem Danh sách," "Danh sách các," "sách các trường," "các trường đại," "trường đại học," "đại học ở," "học ở Ohio"}. The k -gram co-occurrence word set can be constructed like this, although k cannot exceed sentence length. For languages whose words are separated by spaces naturally, such as Vietnamese and English, the order of co-occurrence word is in unit of word. In some languages, such as Chinese and Thai, there are no natural separators between syllables, and the order of the co-occurrence word is in unit of character. In other words, one character is considered as one word.

Definition 4.6 (The Probability of a k -Gram Word). Let S_k be a k -gram co-occurrence word set, $s^{(1,k)} = W_1 W_2 \dots W_k$ be a k -gram word, and $s^{(1,k)} \in S_k$. The probability of $s^{(1,k)}$ is defined as

$$P(s^{(1,k)}) = \frac{C(s^{(1,k)})}{N_k}, \quad (8)$$

where $C(s^{(1,k)})$ is the number of $s^{(1,k)}$ in S_k , and N_k is the number of all k -gram words in the corpus.

Since N_k cannot be accurately calculated, and the estimation of N_k is relatively complicated, we first derive the estimate formula of N_3 and next N_k .

Figure 4 shows the distribution of trigram co-occurrence words in the sentence of length n , W_j is a word. Suppose s_j is a trigram word, then s_{j-2} , s_{j-1} , s_{j+1} and s_{j+2} are no longer trigram words. Therefore, only one-fifth of all trigram co-occurrence words are treated as trigram words. However, the blue marked s_0 , s_1 , s_n , and s_{n+1} are not real trigram co-occurrence words, so we need to add these four nonexistent trigram co-occurrence words before dividing 5. Therefore, N_3 is estimated by

$$N_3 \approx \frac{|S_3| + 4 \times C(\text{sentences})}{5}, \quad (9)$$

where $|S_3|$ and $C(\text{sentences})$ represent the size of trigram co-occurrence word set and the number of sentences in the corpus. Suppose there is only one sentence of length n in the corpus; we have $|S_3| = n - 2$.

Figure 5 shows the distribution of k -gram co-occurrence words in the sentence of length n . According to the estimation of N_3 , if the red marked s_j is a k -gram word, the black marked s_{j-k+1} , s_{j-k+2} , $\dots$, s_{j-1} , s_{j+1} , $\dots$, s_{j+k-1} are no longer k -gram words. Their number is $2(k - 1)$. Therefore, only $1/(2(k - 1) + 1)$ of all k -gram co-occurrence words are treated as k -gram words. Because the blue marked co-occurrence words are nonexistent, we need to add these $2(k - 1)$ co-occurrence words before dividing $2(k - 1) + 1$. So N_k is estimated by

$$N_k \approx \frac{|S_k| + 2(k - 1) \times C(\text{sentences})}{2(k - 1) + 1}, \quad (10)$$

Fig. 5. The distribution of k -gram co-occurrence words in the sentence.

where $|S_k|$ and $C(\text{sentences})$ represent the size of k -gram co-occurrence word set and the number of sentences in the corpus.

Usually there are several different sizes of WSU in sentences. How to determine the boundaries of these WSUs is the key problem to be solved by SLM. The divisions of candidate WSUs in sentences affect each other. So both the locality and globality should be taken into account. In the feasible solution space of a sentence, we consider the division with maximum total binding force as the optimal WSU division—that is,

$$sp^* = \arg \max_{sp_j \in SP} \sum_i^{|sp_j|} H(W_i \cdots W_{i+k_i}) \cdot n_i. \quad (11)$$

In Equation (11), SP represents the feasible solution space of the sentence. sp_j represents a feasible WSU division. $|sp_j|$ represents the number of candidate WSUs in sp_j . $k_i \geq 0$ and $i + k_i \leq n$, n is the length of the sentence. n_i is the number of co-occurrence word $W_i \cdots W_{i+k_i}$ in the corpus. So we establish the following optimization model:

$$\begin{aligned} \max \quad & \sum_{i+k_i \leq n} H(s^{(i, k_i)}) \cdot n_i \\ \text{s.t.} \quad & \begin{cases} P(s^{(i, k_i)}) > \prod_m P(s_m^{(i, k_i)}), \\ H(W_{i+k_i} W_j) < \min[H_L, H_R], \\ H_L = H(s^{(i, k_i-1)} W_{i+k_i}), \\ H_R = H(W_j s^{(j+1, k_j)}), \\ s^{(i, k_i)} = W_i \cdots W_{i+k_i}, \\ s^{(j, k_j)} = W_j \cdots W_{j+k_j}, \\ W_{i+k_i+1} = W_j. \end{cases} \end{aligned} \quad (12)$$

The constraint $P(s^{(i, k_i)}) > \prod_m P(s_m^{(i, k_i)})$ is explained by Corollary 4.4, and $\{s_m^{(i, k_i)}\}$ is an arbitrary end-to-end segment of $W_i \cdots W_{i+k_i}$. The concept of $s_m^{(i, k_i)}$ is the same as the concept of s_i in Corollary 4.4. The interpretation of $H(W_{i+k_i} W_j) < \min[H_L, H_R]$ is shown in Figure 6.

In Figure 6, if $s^{(i, k_i)} = W_i W_{i+1} \cdots W_{i+k_i-1} W_{i+k_i}$ and $s^{(j, k_j)} = W_j W_{j+1} \cdots W_{j+k_j-1} W_{j+k_j}$ are adjacent WSUs, the binding force of W_{i+k_i} and W_j (denoted by $H(W_{i+k_i} W_j)$) must be smaller than the minimum of $H(s^{(i, k_i-1)} W_{i+k_i})$ and $H(W_j s^{(j+1, k_j)})$. Otherwise, W_{i+k_i} and W_j will form another candidate WSU. We only consider this simple situation. The actual situation is more complicated. $H(s^{(i, k_i-1)} W_{i+k_i})$ and $H(W_j s^{(j+1, k_j)})$ are the internal binding forces in $s^{(i, k_i)}$ and $s^{(j, k_j)}$. We agree that $H(W_i) = 0$.

Fig. 6. The relationship of binding force between two adjacent WSUs.

Fig. 7. The maximum length WSU in the sentence.

4.3 The Recursive Divide-and-Conquer Algorithm for Solving SLM

The solution of SLM is equivalent to inserting several separators into the sentence input into SLM. Word strings between adjacent separators are treated as candidate WSUs. The optimal WSU division of the input sentence is that whose total binding force of all candidate WSUs is largest. Judging the optimal WSU division of a sentence is as difficult as finding the optimal WSU division of it. So this optimization problem is *NP-hard*. We can solve SLM by enumeration whose time complexity is $O(2^n)$, which is unbearable. So we need a more efficient method. In this section, we prove the additivity of the optimal WSU division and design a recursive divide-and-conquer algorithm for solving SLM based on the dynamic programming principle.

THEOREM 4.7 (THE ADDITIVITY OF THE OPTIMAL WSU DIVISION). *Suppose S is a sentence of length n , $S = W_1 \dots W_{k-1} W_k \dots W_{n-1} W_n$. S 's optimal WSU division is $U = U_1 U_2 \dots U_{k-1} U_k \dots U_{n-1} U_n$. If we divide U into $U^{(1)} = U_1 U_2 \dots U_{k-1}$ and $U^{(2)} = U_k \dots U_{n-1} U_n$, then $U^{(1)}$ and $U^{(2)}$ are the optimal WSU division of $S^{(1)}$ and $S^{(2)}$. $S^{(1)} = W_1 W_2 \dots W_{k-1}$ and $S^{(2)} = W_k \dots W_{n-1} W_n$.*

PROOF. This theorem can be easily proved by reduction to absurdity. Suppose $U^{(1)}$ is not the optimal WSU division of $S^{(1)}$, and the optimal WSU division of $S^{(1)}$ is $U^{(1*)}$. Then $U^{(1*)} \oplus U^{(2)}$ is the optimal WSU division of S , and $\oplus$ indicates concatenation. This contradicts the premise that the optimal WSU division of S is U . $\square$

Generally there are very few long words in the corpus. So we only consider WSUs with length $\leq \text{MaxWsuLen}$. In this article, we set $\text{MaxWsuLen} = 8$. For a word string with the length of $\text{MaxWsuLen} + 1$, there must be at least one WSU boundary in it. But we do not know where this WSU boundary is. In the actual calculation, we need to assume that each candidate position is a WSU boundary. It is assumed that the candidate position j is a WSU boundary. So j divides sentence S into two sub-sentences $S^{(1)} = W_1 W_2 \dots W_j$ and $S^{(2)} = W_{j+1} \dots W_{n-1} W_n$. According to Theorem 4.7, we only need to find the optimal WSU divisions of $S^{(1)}$ and $S^{(2)}$, and concatenate them together to make the optimal WSU division of S . So we design a recursive divide-and-conquer algorithm to solve SLM (see Algorithm 1).

In Algorithm 1, Function $\text{Len}(\text{sentence})$ gets the length of input. Function $\text{Call_This_Algorithm}(\cdot)$ calls Algorithm 1 recursively. Function $\text{getOptimalWSUDivision_In_Set}(D)$ selects

ALGORITHM 1: The Recursive Divide-and-Conquer Algorithm**Input:** *sentence*, *MaxWsuLen***Output:** The optimal WSU division of *sentence*

```

1: if Len(sentence) < MaxWsuLen then
2:   optWSUdiv = getOptimalWSUDivision_By_Enumeration(sentence)
3:   return optWSUdiv
4: else
5:   D =  $\emptyset$ 
6:   for i = 0 to MaxWsuLen : do
7:     LeftSubSentence, RightSubSentence = getSubSentenceAt(i, sentence, MaxWsuLen)
8:     LeftOptimalWSUDivision = Call_This_Algorithm(LeftSubSentence, MaxWsuLen)
9:     RightOptimalWSUDivision = Call_This_Algorithm(RightSubSentence, MaxWsuLen)
10:    D  $\leftarrow$  LeftOptimalWSUDivision  $\oplus$  RightOptimalWSUDivision
11:   end for
12: end if
13: optWSUdiv = getOptimalWSUDivision_In_Set(D)
14: return optWSUdiv

```

the optimal WSU division from set *D* by judging the total binding force of each WSU division. This is easy to achieve by enumeration. Function *getSubSentenceAt*(*i*, *sentence*, *MaxWsuLen*) gets left and right sub-sentences separated by position *i*. It is implemented by Algorithm 2.

ALGORITHM 2: Get Sub-Sentences Before and After Position *i***Input:** *i*, *sentence*, *MaxWsuLen***Output:** Left and right sub-sentences separated by position *i*

```

1: sentence_length = Len(sentence)
2: middle_position =  $\lfloor (\text{sentence\_length} + 1) / 2 \rfloor$ 
3: Left_Subsentence = sentence[1 : middle_position -  $\lfloor \text{MaxWsuLen} / 2 \rfloor$  + i]
4: Right_Subsentence = sentence[middle_position -  $\lfloor \text{MaxWsuLen} / 2 \rfloor$  + i + 1 : sentence_length]
5: return Left_Subsentence, Right_Subsentence

```

Figure 8 (the abscissa represents x in $y = 2^x$, the ordinate represents the sentence length) shows the compression effect of Algorithm 1 on search space. We can easily see that the growth rate of Algorithm 1's search space is much smaller than the growth rate of enumeration. The shorter the maximum WSU length is, the smaller the growth rate of Algorithm 1's search space is. For example, when *MaxWsuLen* = 8 and the sentence length are 60, 80, and 100, the search spaces of Algorithm 1 are 2^{21} , $2^{22.67}$, and $2^{23.97}$, respectively, whereas the search spaces of enumeration are 2^{60} , 2^{80} , and 2^{100} , respectively.

5 EXPERIMENT

In this section, we first introduce the storage of *k*-gram co-occurrence word set. Next, we show the performance promotion of SLM on three Vietnamese sequence labeling tasks: word segmentation, POS tagging, and NER.

Fig. 8. The compression effect of Algorithm 1 on search space.

Table 1. Parts of an 8-Gram Co-Occurrence Word Set

W_0	W_1	W_2	W_3	W_4	W_5	W_6	W_7	<i>coe</i>
0	1	4	23	9	190	196	101	662
0	1	4	28	20	111	285	133	57
0	1	4	23	9	190	196	40	53
0	1	4	28	20	167	129	69	46
0	1	4	9	190	196	101	48	44
0	1	4	28	89	30	92	292	37
0	1	4	28	12	285	133	43	30
...	...	...	...	...	...	...	...	...

5.1 The Storage of Co-Occurrence Words

The co-occurrence word matrix⁴ has a huge sparsity. When the 863-MB Vietnamese raw text is downloaded from Wikidata, it costs 195 MB in *.txt* format to store the bigram co-occurrence word matrix (only considers the first 10,000 words sorted by frequency in descending order). However, the storage of trigram co-occurrence word matrix requires 1.95 TB. So it is impossible to store all k -gram co-occurrence word matrices in this manner. In the bigram co-occurrence word matrix, about 97.5% of elements do not appear in the corpus. The higher order the co-occurrence word matrix is, the sparser it is. Considering various technologies currently used to store sparse matrices, we believe that storing all the k -gram co-occurrence word matrices in the database is a good choice. Database makes co-occurrence word matrix management and probability calculation more convenient and efficient, especially when the order of co-occurrence word matrices are high.

Table 1 shows the parts of an 8-gram co-occurrence word matrix. The first eight columns of the table store the word indexes. The last *coe* column stores the number of corresponding 8-gram co-occurrence words in the corpus. Co-occurrence words that do not appear in the corpus are not stored.

We only consider the first 10,000 Vietnamese words sorted by frequency in descending order and build 1- to 8-gram co-occurrence word matrices from the raw text with the size of 863 MB

⁴The k -gram co-occurrence word matrix is a set of arbitrary ordered combinations of k words.

Table 2. The Size, Number of Sentences, and N_k of a k -Gram Co-Occurrence Word Set

	<i>Unary</i>	<i>Bigram</i>	<i>Trigram</i>	4-Gram	5-Gram	6-Gram	7-Gram	8-Gram
<i>Size</i>	75,017,451	69,357,253	63,697,055	58,036,857	52,376,659	46,716,461	41,056,263	35,396,065
<i>Sentences</i>	—	14,169,993	11,980,799	10,311,927	9,005,608	7,841,471	6,827,199	5,660,224
N_k	75,017,451	26,892,549	17,267,569	13,142,577	10,850,915	9,392,585	8,382,972	7,642,589

Table 3. Detail of Vietnamese Word Segmentation Datasets

	Sentences		Tokens	
	Train Set	Test Set	Train Set	Test Set
Vnwordseg	6,245	1,560	164k	42k
Conll2017	1400	800	26k	15k

downloaded from Wikidata. The size, number of sentences, and N_k of each gram are shown in Table 2. N_k is calculated by Equation (10).

5.2 Vietnamese Word Segmentation Experiment

5.2.1 Dataset. We use Vnwordseg⁵ and Conll2017⁶ datasets for the Vietnamese word segmentation experiment. The detail of the two datasets is shown in Table 3. All sentences are pre-processed to be lowercase.

5.2.2 Baselines and Parameter Settings. We choose the most popular deep learning models and the official Vietnamese word segmentation tool as baselines for this test. The most popular deep learning models for sequence labeling include BERT⁷ [8], LSTM+CRF [11], and BiLSTM+CRF and its various variants [7, 14, 21].

vnTkoenizer [15] is the official word segmentation tool of the Vietnam Language and Speech Processing project⁸ [36].

VnCoreNLP is a fast and accurate Vietnamese word segmenter and obtains state-of-the-art results [26, 37].

The parameter settings of these baselines are shown in Table 4. We use Word2Vec [24] and GloVe [28] to pre-train Vietnamese word embeddings.

5.2.3 Vietnamese Word Segmentation Experimental Results. As shown in Figure 2, we first use SLM to determine the position symbol of each token in sentences and convert position symbols into one-hot position encodings. Next, we concatenate all tokens' position encodings and their word embeddings together to form the final input of the upper model. By comparing whether to add these position embeddings to the input of the upper word segmentation model, we can evaluate SLM's performance promotion.

Table 5 shows the cross-dataset experimental results of word segmentation on the Conll2017 dataset. "+SLM" means adding position encoding to the input of corresponding model. "(GloVe)" represents that word embeddings are pre-trained by GloVe [28]. "(Word2Vec)" represents that word embeddings are pre-trained by Word2Vec [24]. "Vnwordseg Test" means test on the Vnwordseg test set. It can be seen from Table 5 that "+SLM" can improve the word segmentation performance. The train set of Conll2017 has only 1,400 sentences. The best F value of cross-dataset test on

⁵<http://www.jaist.ac.jp/~hieuxuan/vnwordseg/data/>.

⁶Train set: <https://lindat.mff.cuni.cz/repository/xmlui/handle/11234/1-1983>, Test set: <https://lindat.mff.cuni.cz/repository/xmlui/handle/11234/1-2184>.

⁷We add BiLSTM+CRF on the top of BERT for sequence labeling tasks.

⁸<https://vlspl.org.vn/>.

Table 4. Parameter Settings of Vietnamese Word Segmentation Baselines

Model	Pre-Trained Word Embedding	LSTM Hidden Units	Word's Character Sequence Embedding	Word's Character Sequence Convolution Filters	Epoch	Dropout
LSTM+CRF	100	100	No	No	50	0.3
BiLSTM+CRF	100	50	No	No	50	0.3
Lample. [14]	100	50	15	No	50	0.3
Ma. [21]	100	50	No	30	50	0.3
Chen. [7]	100	150	No	No	50	0.3
BERT [8]	768	200	No	No	50	0.3

Vnwordseg we obtain is 0.8833, which is +0.0113 more than the best F value of VnTokenizer. The test result of VnTokenizer on the Conll2017 test set is $P = R = F = 1.0$. We rerun VnCoreNLP on the Conll2017 test set and get a good F value ($F = 0.9753$). However, the train sets of VnTokenizer and VnCoreNLP are almost the same⁹ [26]. We regard VnTokenizer and VnCoreNLP testing on Vnwordseg as the cross-dataset test.

Table 6 shows the cross-dataset experimental results of word segmentation on the Vnwordseg dataset. We can draw the same conclusion from Table 6 as Table 5. The Vnwordseg dataset is also a small dataset containing 6,245 sentences (train set). The best F value of the cross-dataset test on Conll2017 we obtain is 0.9061, which are +0.0341 and +0.0108 more than the F value of VnTokenizer and VnCoreNLP (we rerun VnCoreNLP on Vnwordseg, and the cross-dataset test F value we get is 0.8953). However, VnTokenizer's train set includes a Vietnamese dictionary and a 70,000-sentence Vietnamese treebank [36], and VnCoreNLP used a 75,000-sentence Vietnamese treebank to train. When we conduct the experiment on the Vnwordseg dataset, the hidden units of LSTM and BiLSTM are 150 and 75. The words' character embedding has 40 dimensions, and the words' character sequence convolution filters are 40. Other parameters are shown in Table 4.

GloVe [28] and Word2Vec [24] are both static word embedding models. The experimental results show that there is no obvious difference in performance between them. Unsurprisingly, BERT performs better than all of the other models. In the first cross-dataset test, BERT+SLM surpasses the performance of VnTokenizer using only a training set of 1,400 sentences (only 2% of VnTokenizer's train set). In the second cross-dataset test, BERT+SLM uses less than 10% of the training set of VnCoreNLP (we also do not use a dictionary), and its best F (0.9061) value exceeds state-of-the-art VnCoreNLP's F (0.8953) value of 1.08%.

From the word segmentation experimental results, we can draw a conclusion that the pre-divided WSUs output by SLM can deliver useful information to word segmentation models and improve their performance.

5.3 Vietnamese POS Labeling Experiment

5.3.1 Dataset. We use Conll2017 for the Vietnamese POS labeling experiment. Its label distribution is shown in Table 7. All sentences are pre-processed to be lowercase.

5.3.2 Baselines and Parameter Settings. We also choose BERT, LSTM+CRF, BiLSTM+CRF, and its various variants, the state-of-the-art Vietnamese POS tagging, and the NER system (PhoNLP) [27] as baselines. These baselines' parameter settings are also shown in Table 4.

⁹<https://vlsphda.vn/demo/?page=resources>.

Table 5. Vietnamese Word Segmentation Experimental Results on the Conll2017 Dataset

	Vnwordseg Test (GloVe)				Vnwordseg Test (Word2Vec)			
	<i>P</i>	<i>R</i>	<i>F</i>	ΔF	<i>P</i>	<i>R</i>	<i>F</i>	ΔF
LSTM_CRF	0.7811	0.8465	0.8125	–	0.7776	0.8455	0.8102	–
LSTM_CRF +SLM	0.8236	0.874	0.848	+0.0356	0.8197	0.868	0.8432	+0.033
biLSTM+CRF	0.79	0.866	0.8263	–	0.7872	0.8544	0.8194	–
biLSTM+CRF +SLM	0.8285	0.8797	0.8533	+0.027	0.8243	0.8728	0.8479	+0.0285
Chen. [7]	0.7823	0.8316	0.8062	–	0.7828	0.8289	0.8052	–
Chen. [7]+SLM	0.802	0.8464	0.8236	+0.0174	0.8017	0.8451	0.8228	+0.0176
Lample. [14]	0.7913	0.8551	0.822	–	0.7924	0.8548	0.8224	–
Lample. [14]+SLM	0.831	0.8514	0.8411	+0.019	0.8219	0.8734	0.8469	+0.0245
Ma. [21]	0.785	0.8533	0.8177	–	0.7848	0.8504	0.8163	–
Ma. [21]+SLM	0.8237	0.8739	0.8481	+0.0304	0.8256	0.8734	0.8488	+0.0325
BERT [8]	–	–	–	–	0.8386	0.8723	0.8551	–
BERT [8]+SLM	–	–	–	–	0.8752	0.8915	0.8833	+0.0282
VnTokenizer [15, 36]	–	–	–	–	0.928	0.822	0.872	–

Table 6. Vietnamese Word Segmentation Experimental Results on the Vnwordseg Dataset

	Conll2017 Test (GloVe)				Conll2017 Test (Word2Vec)			
	<i>P</i>	<i>R</i>	<i>F</i>	ΔF	<i>P</i>	<i>R</i>	<i>F</i>	ΔF
LSTM_CRF	0.84	0.8465	0.8432	–	0.8304	0.8386	0.8344	–
LSTM_CRF +SLM	0.8628	0.8579	0.86	+0.017	0.8695	0.8461	0.8576	+0.0232
biLSTM+CRF	0.857	0.8378	0.8473	–	0.8583	0.831	0.8445	–
biLSTM+CRF +SLM	0.869	0.8536	0.8612	+0.014	0.8751	0.8418	0.8581	+0.0136
Chen. [7]	0.8491	0.8325	0.8407	–	0.8547	0.8238	0.839	–
Chen. [7]+SLM	0.8593	0.8464	0.8528	+0.012	0.8656	0.8248	0.8447	+0.0057
Lample. [14]	0.837	0.8255	0.8312	–	0.8393	0.8217	0.8393	–
Lample. [14]+SLM	0.876	0.849	0.8623	+0.031	0.8753	0.8456	0.8602	+0.0209
Ma. [21]	0.86	0.837	0.8483	–	0.8584	0.8300	0.844	–
Ma. [21]+SLM	0.8637	0.8584	0.861	+0.0127	0.8688	0.8546	0.8616	+0.0176
BERT [8]	0.8842	0.8583	0.871	–	–	–	–	–
BERT [8]+SLM	0.913	0.8994	0.9061	+0.0351	–	–	–	–

Table 7. POS Labels' Distribution of the Conll2017 Vietnamese Dataset

	NOUN	ADP	X	VERB	ADJ	PUNCT	SCONJ
Train Set	6,495	1,189	1,595	4,123	1,104	2,960	245
Test Set	3,838	688	970	2,178	738	1,722	122
	NUM	DET	CCONJ	PROPN	AUX	PART	INTJ
Train Set	617	310	530	823	182	109	3
Test Set	412	232	335	494	132	87	7

Table 8. POS Labeling Experiment Results on the Conll2017 Vietnamese Dataset

	Embedding Trained by GloVe				Embedding Trained by Word2Vec			
	<i>P</i>	<i>R</i>	<i>F</i>	ΔF	<i>P</i>	<i>R</i>	<i>F</i>	ΔF
LSTM_CRF	0.7362	0.7693	0.7524		0.7321	0.7663	0.7488	
LSTM_CRF +SLM	0.7805	0.8	0.7901	+0.038	0.772	0.794	0.7838	+0.035
biLSTM+CRF	0.76	0.799	0.779		0.7653	0.7926	0.7787	
biLSTM+CRF +SLM	0.806	0.8074	0.8067	+0.028	0.7904	0.8084	0.7993	+0.021
Lample. [14]	0.759	0.783	0.7708		0.7707	0.798	0.7841	
Lample. [14]+SLM	0.798	0.7969	0.7974	+0.027	0.7945	0.8109	0.8026	+0.019
Ma. [21]	0.783	0.815	0.7987		0.7867	0.8045	0.7955	
Ma. [21]+SLM	0.7975	0.816	0.8066	+0.008	0.7955	0.8116	0.8035	+0.008
BERT [8]	0.7923	0.8045	0.7984		–	–	–	
BERT [8]+SLM	0.8084	0.8172	0.8128	+0.0144	–	–	–	–

Table 9. Detail of Our Vietnamese NER Dataset

	Sentences	Tokens	PER	LOC	ORG
Train Set	1,060	39k	845	834	834
Test Set	441	17k	351	372	378

5.3.3 Vietnamese POS Labeling Experimental Results. The Vietnamese POS labeling experimental results are shown in Table 8. “Model-A+SLM” means adding position encodings output by SLM to the input of Model-A. We choose five popular sequence labeling models for this test. From the experimental results, it can be seen that SLM can improve the performance of existing sequence labeling models. The minimum improvement of absolute *F* value is 0.8%. Models [14] and [21] both extract the features of character sequence within a word. The former uses a BiLSTM, and the latter uses a convolution network. BERT [8]+SLM has the best performance, and LSTM has the worst. Model Ma. [21]+SLM has the least improvement, but LSTM+SLM has the most. The accuracies of BERT+SLM and PhoNLP are 82.33% and 93.88% (on the Vietnam Language and Speech Processing 2013 POS tagging dataset) [27], respectively. PhoNLP uses joint training, and its training set includes 30,000 sentences. The training set of BERT+SLM only contains 1,400 sentences.

From the POS labeling experiment, we can draw a conclusion that the pre-divided WSUs can also deliver useful information to the POS labeling models and improve their performance. The performance of BERT is better than all of the other models.

5.4 Vietnamese NER Experiment

5.4.1 Dataset. Since there is no publicly available Vietnamese NER dataset, we hired two Vietnamese native speakers to annotate a small Vietnamese NER dataset.¹⁰ Its detail is shown in Table 9. PER, LOC, and ORG stand for the tags of people, places, and organizations, respectively. We use the BIES labeling schema described in Section 3 to annotate the dataset. All sentences are pre-processed to be lowercase.

¹⁰https://github.com/Liaoxianwen/SLM_code_data.

Table 10. Parameter Settings of Vietnamese NER Baselines

Model	Pre-Trained Word Embedding	LSTM Hidden Units	Word's Character Sequence Embedding	Word's Character Sequence Convolution Filters	Epoch	Dropout
LSTM+CRF	100	100	No	No	150	0.3
LSTM+CRF +SLM	100	104	No	No	150	0.3
BiLSTM+CRF	100	50	No	No	150	0.3
BiLSTM+CRF +SLM	100	54	No	No	150	0.3
Lample. [14]	100	50	15	No	150	0.3
Lample. [14]+SLM	100	54	15	No	150	0.3
Ma. [21]	100	50	No	30	150	0.3
Ma. [21]+SLM	100	54	No	30	150	0.3
BERT [8]	768	204	No	No	100	0.3

5.4.2 Baselines and Parameter Settings. Baselines and their parameter settings are shown in Table 10. “Model-A+SLM” means adding position encodings output by SLM to the input of Model-A. In “Model-A+SLM,” since the dimension of input vector is increased by 4, the hidden units numbers of BiLSTM and LSTM increase to 54 and 104.

5.4.3 Vietnamese NER Experimental Results. Table 11 shows Vietnamese NER experimental results. P , R , F , and ΔF in the “All Tags” column represent the statistics considering all tags in the test. F_{per} and ΔF_{per} represent the statistics considering only PER, and so on for LOC and ORG. The subscripts WV and GLO represent using Word2Vec [24] and GloVe [28] to pre-train Vietnamese word embeddings.

From the NER experimental results, it can be seen that our SLM can improve the performance of different popular deep learning models. The pre-divided WSUs in sentences also help the labeling of named entities. The absolute improvement of F value is up to 5%. Overall, BiLSTM performs better than LSTM. Models [14] and [21] both extract features of character sequence within word. One of them uses a convolution network, and the other uses a BiLSTM. However, their performances are not much different. The performance of BERT+SLM is the best. The difference in NER performance between Word2Vec [24] and GloVe [28] is not obvious. BERT+SLM and PhoNLP use their own test sets, so we can only roughly compare the performance between the two. The training sets of PhoNLP and BERT+SLM contain 14,861 and 1,060 sentences, respectively [27]. The sentences number of the former is 14 times that of the latter.

5.5 The Control of Average WSU Length in the Optimal WSU Division

In Equation (12), if we change the objective function to

$$\text{Max} \sum_{0 \leq i+k_i \leq n} H(s^{(i,k_i)}) \cdot \max(1, \log_b n_i), \quad (13)$$

the average WSU length in the optimal WSU division will vary with b ($b > 0, b \neq 1$). $\max(1, \log_b n_i)$ in Equation (13) is a penalty for the binding force. It can reduce the differences between candidate WSUs' binding force. The influences of b on the average WSU length and NER performance is shown in Figure 9. We use Model [21]+SLM to perform the experiment, and the epoch is set to 100.

Table 11. Vietnamese NER Experimental Results

	All Tags				PER		LOC		ORG	
	P	R	F	ΔF	F_{per}	ΔF_{per}	F_{loc}	ΔF_{loc}	F_{org}	ΔF_{org}
LSTM+CRF _{WV}	0.9199	0.9521	0.9357	–	0.7204	–	0.7355	–	0.6071	–
LSTM+CRF _{WV} +SLM	0.9252	0.9514	0.9381	+0.002	0.7436	+0.023	0.7637	+0.028	0.6202	+0.013
LSTM+CRF _{GLO}	0.918	0.9492	0.933	–	0.72	–	0.7328	–	0.606	–
LSTM+CRF _{GLO} +SLM	0.9308	0.9615	0.946	+0.012	0.7511	+0.031	0.766	+0.033	0.6289	+0.023
biLSTM+CRF _{WV}	0.9387	0.9587	0.9486	–	0.7872	–	0.7643	–	0.649	–
biLSTM+CRF _{WV} +SLM	0.9419	0.9627	0.9522	+0.004	0.806	+0.019	0.7898	+0.026	0.6985	+0.05
biLSTM+CRF _{GLO}	0.9313	0.9594	0.945	–	0.7851	–	0.762	–	0.648	–
biLSTM+CRF _{GLO} +SLM	0.945	0.9691	0.9569	+0.012	0.81	+0.025	0.806	+0.044	0.689	+0.041
Lample _{WV}	0.9373	0.961	0.949	–	0.757	–	0.7746	–	0.6496	–
Lample _{WV} [14] +SLM	0.9435	0.9641	0.9537	+0.005	0.7976	+0.041	0.8068	+0.032	0.6733	+0.024
Lample _{GLO} [14]	0.9387	0.9561	0.947	–	0.753	–	0.779	–	0.631	–
Lample _{GLO} [14] +SLM	0.9489	0.9677	0.958	+0.011	0.79	+0.037	0.815	+0.036	0.674	+0.043
Ma _{WV} [21]	0.9369	0.9614	0.949	–	0.779	–	0.7667	–	0.643	–
Ma _{WV} [21]+SLM	0.9443	0.9641	0.9541	+0.005	0.8157	+0.037	0.7957	+0.029	0.6889	+0.046
Ma _{GLO} [21]	0.941	0.9606	0.9507	–	0.78	–	0.762	–	0.659	–
Ma _{GLO} [21]+SLM	0.946	0.9674	0.957	+0.006	0.82	+0.04	0.798	+0.036	0.697	+0.038
BERT [8]	0.952	0.973	0.962	–	0.9264	–	0.7867	–	0.7197	–
BERT [8]+SLM	0.9539	0.979	0.9663	+0.004	0.9387	+0.012	0.8012	+0.015	0.745	+0.025
PhoNLP [27]	–	–	0.9451	–	–	–	–	–	–	–

In Figure 9, “Avg. Length” stands for “the average WSU length in the optimal WSU division.” F_{all} represents the overall F value. F_{PER} , F_{LOC} , and F_{ORG} represent the F value of PER, LOC, and ORG, respectively. From Figure 9, we can see that as b increases, Avg. Length also increases, but F_{all} decreases slightly. When we carefully compare F_{PER} , F_{LOC} , and F_{ORG} under different Avg. Length, we find that the increase of Avg. Length reduces the performance of NER slightly.

In summary, we do not need to add $\max(1, \log_b n_i)$ to SLM’s objective function.

6 CONCLUSION

We first defined the computable WSU and derive two theorems and one corollary at the beginning of this article. Next, we constructed the unsupervised SLM with the objective of maximizing the total binding force of all candidate WSUs in sentences. The solution of SLM is *NP-hard*. We proved the additivity of the optimal WSU division and designed a recursive divide-and-conquer algorithm to solve SLM based on the dynamic programming principle. Our SLM is completely unsupervised and has no hyper-parameter, and it can deal with analytic languages indiscriminately. So it has good domain adaptability.

Deep learning models have achieved great success in NLP; however, these models require a large amount of high-quality annotated data. Extracting information from raw text to reduce the dependency of deep learning models on annotated data and improving deep learning models’ performance have been research hotspots in NLP recently. In this article, we established the pre-trained sequence labeling framework shown in Figure 2. Based on the proposed framework, we integrated our SLM model with several popular sequence labeling models and conducted three

Fig. 9. The influences of b on the Avg. Length in the optimal WSU division and NER performance.

types of Vietnamese sequence labeling experiments. Whether it is word segmentation, POS labeling, or an NER test, SLM can efficiently improve the performances of baselines. In particular, without using a dictionary, we only used 10% of the training data to make the word segmentation performance of BERT+SLM exceed VnCoreNLP [37]. This shows that SLM can also effectively improve the performance of the dynamic word embedding model.

Our SLM can be improved in the following aspects to further promote the performance of sequence labeling tasks:

- Use vocabulary in SLM. Vocabulary includes dictionaries, names of persons, locations, organizations, and proper nouns. First, we can use vocabulary to determine those unambiguous boundaries of WSUs to narrow the search space. Next, we can use SLM to deal with the ambiguities [15].
- Add morphological information to position encodings before they concatenate with word embeddings. Experiments in the work of Mayhew et al. [23] show that the case-sensitive information can promote the performance of NER.
- Extract information from other knowledge bases and add it to word embedding [1, 10, 31].
- Use regular expressions to pre-process the input. Experiments have shown that this may improve the performance of WSU division [5–7].

ACKNOWLEDGMENTS

We thank for insightful comments of anonymous reviewers.

REFERENCES

- [1] Muhammad Asif Ali, Yifang Sun, Bing Li, and Wei Wang. 2020. Fine-grained named entity typing over distantly supervised data based on refined representations. In *Proceedings of AAAI 2020*, Vol. 34. 7391–7398. <https://doi.org/10.1609/aaai.v34i05.6234>
- [2] M. Saiful Bari, Shafiq Joty, and Prathyusha Jwalapuram. 2020. Zero-resource cross-lingual named entity recognition. In *Proceedings of AAAI 2020*, Vol. 34. 7415–7423. <https://doi.org/10.1609/aaai.v34i05.6237>
- [3] Jakab Buda and Flora Bolonyai. 2018. Location name extraction from targeted text streams using gazetteer-based statistical language models. In *Proceedings of the 27th International Conference on Computational Linguistics*. 1986–1997.
- [4] Jakab Buda and Flora Bolonyai. 2020. An ensemble model using n -grams and statistical features to identify fake news spreaders on Twitter. In *Proceedings of CLEF*.
- [5] Deng Cai and Hai Zhao. 2016. Neural word segmentation learning for Chinese. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics*. 409–420. <https://doi.org/10.18653/v1/P16-1039>

- [6] Deng Cai, Hai Zhao, Zhisong Zhang, Yuan Xin, Yongjian Wu, and Feiyue Huang. 2017. Fast and accurate neural word segmentation for chinese. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics*. 608–615.
- [7] Xinchu Chen, Xipeng Qiu, Chenxi Zhu, Pengfei Liu, and Xuanjing Huang. 2015. Long short-term memory neural networks for chinese word segmentation. In *Proceedings of EMNLP*. 1385–1394.
- [8] Jacob Devlin, Mingwei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of NAACL 2019*. 4171–4186. <https://doi.org/10.18653/v1/N19-1423>
- [9] Aneiss Ghodsi and John DeNero. 2016. An analysis of the ability of statistical language models to capture the structural properties of language. In *Proceedings of the 9th International Natural Language Generation Conference*. 227–231.
- [10] Qizhen He, Liang Wu, Yida Yin, and Heming Cai. 2020. Knowledge-graph augmented word representations for named entity recognition. In *Proceedings of AAAI 2020*, Vol. 34. 7919–7926. <https://doi.org/10.1609/aaai.v34i05.6299>
- [11] Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short term memory. *Neural Computation* 9, 8 (Aug. 1997), 1735–1780.
- [12] Anwen Hu, Zhicheng Dou, Jirong Wen, and Jianyun Nie. 2020. Leveraging multi-token entities in document-level named entity recognition. In *Proceedings of AAAI 2020*, Vol. 34. 7961–7968. <https://doi.org/10.1609/aaai.v34i05.6304>
- [13] Changning Huang and Hai Zhao. 2007. Chinese word segmentation: A decade review. *Journal of Chinese Information Processing* 21, 3 (May 2007), 8–19.
- [14] Guillaume Lample, Miguel Ballesteros, Sandeep Subramanian, Kazuya Kawakami, and Chris Dyer. 2016. Neural architectures for named entity recognition. In *Proceedings of NAACL-HLT 2016*. 260–270.
- [15] Hong Phuong Le, Thi Minh Huyen Nguyen, Azim Roussanaly, and Tuong Vinh Ho. 2008. A hybrid approach to word segmentation of Vietnamese texts. In *Proceedings of the 2nd International Conference on Language and Automata Theory and Applications*. 240–249.
- [16] KyungTae Lim, Jay Yoon Lee, Jaime Carbonell, and Thierry Poibeau. 2020. Semi-supervised learning on meta structure: Multi-task tagging and parsing in low-resource scenarios. In *Proceedings of AAAI 2020*, Vol. 34. 8344–8351. <https://doi.org/10.1609/aaai.v34i05.6351>
- [17] Junxin Liu, Fangzhao Wu, Chuhan Wu, Yongfeng Huang, and Xing Xie. 2019. Neural Chinese word segmentation with dictionary. *Neurocomputing* 338 (April 2019), 46–54.
- [18] Liyuan Liu, Xiang Ren, Qi Zhu, Huan Gui, Shi Zhi, Heng Ji, and Jiawei Han. 2017. Heterogeneous supervision for relation extraction: A representation learning approach. In *Proceedings of EMNLP*. 46–56.
- [19] Liyuan Liu, Jingbo Shang, Xiang Ren, Frank F. Xu, Huan Gui, Jian Peng, and Jiawei Han. 2018. Empower sequence labeling with task-aware neural language model. In *Proceedings of the 32nd AAAI Conference on Artificial Intelligence*. 5253–5260.
- [20] Shifeng Liu, Yifang Sun, Bing Li, Wei Wang, and Xiang Zhao. 2020. HAMNER: Headword amplified multi-span distantly supervised method for domain specific named entity recognition. In *Proceedings of AAAI 2020*, Vol. 34. 8401–8408. <https://doi.org/10.1609/aaai.v34i05.6358>
- [21] Xuezhe Ma and Eduard Hovy. 2016. End-to-end sequence labeling via bi-directional LSTM-CNNs-CRF. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics*. 1064–1074.
- [22] Benjamin Marie, Haipeng Sun, Rui Wang, Kehai Chen, Atsushi Fujita, Masao Utiyama, and Eiichiro Sumita. 2019. NICT’s unsupervised neural and statistical machine translation systems for the WMT19 news translation task. In *Proceedings of the 4th Conference on Machine Translation (WMT’19)*. 294–301.
- [23] Stephen Mayhew, Nitish Gupta, and Dan Roth. 2020. Robust named entity recognition with truecasing pretraining. In *Proceedings of AAAI 2020*, Vol. 34. 8480–8487. <https://doi.org/10.1609/aaai.v34i05.6368>
- [24] Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Efficient estimation of word representations in vector space. In *Proceedings of the 1st International Conference on Learning Representations (ICLR’13)*. <http://arxiv.org/abs/1301.3781>
- [25] A. Nguyen, K. Nguyen, and V. Ngo. 2019. Neural sequence labeling for Vietnamese POS tagging and NER. In *Proceedings of the 2019 IEEE-RIVF International Conference on Computing and Communication Technologies (RIVF’19)*. 1–5. <https://doi.org/10.1109/RIVF.2019.8713710>
- [26] Dat Quoc Nguyen, Dai Quoc Nguyen, Thanh Vu, Mark Dras, and Mark Johnson. 2018. A fast and accurate Vietnamese word segmenter. In *Proceedings of the 11th International Conference on Language Resources and Evaluation (LREC’18)*. 2582–2587.
- [27] Linh The Nguyen and Dat Quoc Nguyen. 2021. PhoNLP: A joint multi-task learning model for Vietnamese part-of-speech tagging, named entity recognition and dependency parsing. In *Proceedings of NAACL-HLT 2021*. 1–7.
- [28] Jeffrey Pennington, Richard Socher, and Christopher Manning. 2014. GloVe: Global vectors for word representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing*. 1532–1543. <https://doi.org/10.3115/v1/D14-1162>

- [29] Matthew E. Petersy, Mark Neumann, Mohit Iyyery, Matt Gardnery, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 2018. Deep contextualized word representations. In *Proceedings of NAACL 2018*. 1532–1543.
- [30] Esteban Safranchik, Shiyang Luo, and Stephen H. Bach. 2020. Weakly supervised sequence tagging from noisy rules. In *Proceedings of AAAI 2020*, Vol. 34. 5570–5578. <https://doi.org/10.1609/aaai.v34i04.6009>
- [31] Edwin Simpson, Jonas Pfeiffer, and Iryna Gurevych. 2020. Low resource sequence tagging with weak labels. In *Proceedings of AAAI 2020*, Vol. 34. 8862–8869. <https://doi.org/10.1609/aaai.v34i05.6415>
- [32] Sittichai Somsap and Pusadee Seresangtakul. 2020. Isarn Dharma word segmentation using a statistical approach with named entity recognition. *ACM Transactions on Asian and Low-Resource Information Processing* 19, 2 (Feb. 2020), Article 27, 16 pages. <https://doi.org/10.1145/3359990>
- [33] Richard Sproat and Chilin Shih. 1990. A statistical method for finding word boundaries in Chinese text. *Computer Processing of Chinese and Oriental Languages* 4, 4 (March 1990), 336–351.
- [34] Chuanqi Tan, Wei Qiu, Mosha Chen, Rui Wang, and Fei Huang. 2020. Boundary enhanced neural span classification for nested named entity recognition. In *Proceedings of AAAI 2020*, Vol. 34. 9016–9023. <https://doi.org/10.1609/aaai.v34i05.6434>
- [35] Matti Varjokallio and Dietrich Klakow. 2016. Unsupervised morph segmentation and statistical language models for vocabulary expansion. In *Proceedings of ACL*. 175–180.
- [36] VLSP Project 2010. Vietnamese Language Resources. Retrieved August 5, 2020 from <https://vlsp.hpdata.vn/demo/?page=resources&lang=en>.
- [37] Thanh Vu, Dat Quoc Nguyen, Dai Quoc Nguyen, Mark Dras, and Mark Johnson. 2018. VnCoreNLP: A Vietnamese natural language processing toolkit. In *Proceedings of NAACL-HLT 2018*. 56–60.
- [38] Qianhui Wu, Zijia Lin, Guoxin Wang, Hui Chen, Borje F. Karlsson, Biqing Huang, and Chinyew Lin. 2020. Enhanced meta-learning for cross-lingual named entity recognition with minimal resources. In *Proceedings of AAAI 2020*, Vol. 34. 9274–9281. <https://doi.org/10.1609/aaai.v34i05.6466>
- [39] Nianwen Xue. 2003. Chinese word segmentation as character tagging. *International Journal of Computational Linguistics and Chinese Language Processing* 8, 6 (March 2003), 29–48.
- [40] Jie Yang, Yue Zhang, and Fei Dong. 2017. Neural word segmentation with rich pretraining. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics*. 839–849.

Received June 2019; revised December 2020; accepted August 2021