

國立臺灣大學電機資訊學院資訊工程學研究所

碩士論文

Department of Computer Science and Information Engineering

College of Electrical Engineering and Computer Science

National Taiwan University

Master Thesis

非中文字母語學習者中文寫作用詞錯誤偵測及更正之研究

Detection and Correction of Chinese Word Usage Errors for

Learning Chinese as a Second Language

薛祐婷

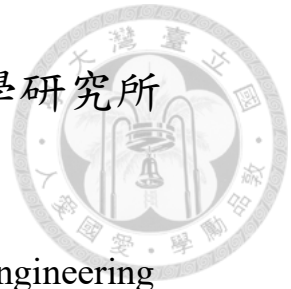
Yow-Ting Shiue

指導教授：陳信希 博士

Advisor: Hsin-Hsi Chen, Ph.D.

中華民國 106 年 7 月

July 2017



國立臺灣大學碩士學位論文
口試委員會審定書

非中文母語學習者中文寫作用詞錯誤偵測及更正之研究

Detection and Correction of Chinese Word Usage Errors
for Learning Chinese as a Second Language

本論文係薛祐婷君（學號 R04922029）在國立臺灣大學資訊工程學系完成之碩士學位論文，於民國 106 年 7 月 20 日承下列考試委員審查通過及口試及格，特此證明

口試委員：

陳信昂

（指導教授）

林川傑

禹良治

蔡宇翰

系主任

趙坤茂



誌謝



能夠完成這篇論文，首先要感謝我的指導教授陳信希老師。我從大學時開始和老師做專題，老師總是非常有耐心地引導我，在我遇到瓶頸時給予有用的建議，往往我自己研究了一個禮拜沒有任何進展，和老師討論後便豁然開朗，可以往自己沒有想過的方向努力看看。此外，也非常感謝博士後黃瀚萱學長，不僅在討論的時候提出許多想法，關於程式實作細節的問題、以及研究生活的壓力適應等等，也不厭其煩地幫助我。感謝學長姐：昂穎、安孜、重吉、衍綺、聖倫、文俊、文立、長睿、葛浩、鈺璋，你們是我的楷模，我從你們身上學到很多。感謝同屆的夥伴宇祥、仕弘、勤和、微涓，這兩年來我們總是相互激勵，不僅在研究上有所交流，也建立了深刻的情誼，多麼幸運能與你們相遇。感謝學弟妹們：傳恩、瑋柔、佳文、博政、子軒、宗翰、斯文，你們新穎的想法和積極的態度也給我許多啟發。感謝口試委員蔡宗翰教授、林川傑教授、禹良治教授，指出我論文的不足之處，給我許多修改的建議，讓這篇論文可以更清楚完整。

感謝我的父母，從小讓我自由探索自己喜愛的事物，在我就讀研究所期間仍然無怨無悔地給予經濟及生活上的支持，讓我可以無後顧之憂地撰寫論文。感謝男友，在服役中仍然花時間陪伴我，在我壓力極大、幾乎要放棄的時候也沒有責備我，反而耐著性子聽我抱怨。感謝我的朋友們，在我感到無助時，適時拉我一把，讓我有繼續前進的動力。只靠我自己一個人，沒有許許多多人的幫助，我是絕對沒辦法通過種種考驗、完成學位論文的。真心謝謝大家。



中文摘要



近年來，世界上越來越多人選擇學習中文，中文文法錯誤偵測及更正工具的需求因而增加。在 HSK 動態作文語料庫中，用詞錯誤是最頻繁的詞層級錯誤。然而，針對用詞錯誤偵測的研究並不多；在更正方面則只有處理特定詞類，如介係詞等。在這篇碩士論文中，我們提出中文用詞錯誤偵測及更正的方法。據我們所知，這是第一篇處理所有詞類之中文用詞錯誤更正之研究。

我們分三個階段處理中文用詞錯誤：(1) 子句層級之偵測、(2) 詞層級之偵測、(3) 更正，使用了中文字、詞、詞性和依存關係等等資訊。在第一階段中，我們訓練二元分類器來判斷一個子句是正確的、還是含有用詞錯誤，最好的模型準確率達 0.84、精確率達 0.95。在第二階段，我們使用雙向長短期記憶神經網路建立序列標記模型，預測每一個詞的錯誤程度。這個模型可以考慮錯誤的詞和其他上下文詞彙的關係，在超過 80% 的測試資料中，可以將標準答案排在前兩名。在第三階段，我們建立神經網路模型，輸入上下文以及需要被更正的詞之特徵，產生一個更正向量，這個向量可以和候選詞彙集合比較以選出適合的更正。由於可能存在不只一種更正，我們對系統的前五名候選更正進行人工標記。根據人工評估的結果，對於超過 91% 的測試資料，前五名中至少有一個是可接受的更正。非母語中文學習者可以使用我們的系統，在沒有語言教師指導的情況下檢查並修正自己所寫的句子。

關鍵字：文法錯誤、中文用詞錯誤、用詞錯誤偵測、用詞錯誤更正、電腦輔助語言教學、HSK 語料庫




ABSTRACT



Recently, more and more people around the world choose to learn Chinese as a second language. The need of Chinese grammatical error detection and correction tools is therefore increasing. In the HSK dynamic composition corpus, word usage error (WUE) is the most common type of errors at the lexical level. However, few studies focus on WUE detection, and for correction only specific types of words such as prepositions are investigated. In this thesis, we propose methods to detect and correct Chinese WUEs. To the best of our knowledge, this is the first research addressing general-type Chinese WUE correction.

We deal with Chinese WUE with three stages: (1) segment-level detection, (2) token-level detection, and (3) correction. Information of character, word, POS and dependency are utilized. In the first stage, we train binary classifiers to tell whether a segment is correct, or contains some WUE. The best model achieves accuracy 0.84 and precision 0.95. In the second stage, we use bidirectional Ling-Short Term Memory to build sequence labeling model that can predict the level of incorrectness of each token. The model can consider the dependency of the erroneous token on context words and rank the ground-truth position within the top two in more than 80% of the cases. In the third stage, we build a neural network model that takes context and target erroneous token features as



input and generates a correction vector, which can be compared against a candidate vocabulary to select suitable corrections. To deal with potential alternative corrections, the top five candidates are judged by human annotators. According to the human evaluation results, for more than 91% of the cases our system can propose at least one acceptable correction within a list of five candidates. With the help of our system, non-native Chinese learners can check and revise their sentences by themselves without the help of language teachers.

Keywords: Grammatical error, Chinese word usage error, Word usage error detection, Word usage error correction, Computer-assisted language learning, HSK corpus

CONTENTS



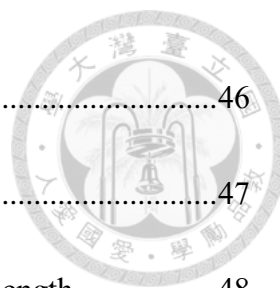
口試委員會審定書	i
誌謝	iii
中文摘要	v
ABSTRACT	vii
CONTENTS	ix
LIST OF FIGURES	xv
LIST OF TABLES	xvi
Chapter 1 Introduction	1
1.1 Motivation.....	1
1.2 Chinese Word Usage Error (WUE)	2
1.3 Overview.....	5
1.4 Thesis Organization	6
Chapter 2 Related Work.....	7
2.1 Grammatical Error Detection and Correction in English	7
2.2 Grammatical Error Detection and Correction in Chinese.....	10
2.3 Distributed Word Representations	11
2.3.1 CBOW/Skip-gram Word Embeddings	12



2.3.2	CWINDOW/Structured Skip-gram Word Embeddings	13
2.3.3	Character-enhanced Word Embedding (CWE)	14
Chapter 3	The HSK Word Usage Error Dataset.....	17
3.1	Data Collection	17
3.1.1	Split Sentences	17
3.1.2	Convert Multiple-error Sentences into Single-error Ones	17
3.2	Linguistic Processing.....	18
3.3	Splitting Sentences into Sentence Segments	19
3.4	Data Filtering	20
Chapter 4	Segment-level Chinese WUE Detection.....	21
4.1	Task Description	21
4.2	Dataset	21
4.3	Features.....	22
4.3.1	Google N-gram Features	22
4.3.2	Dependency Count Features	23
4.3.3	Dependency Bigram Features	26
4.3.4	Single-character Features	26
4.3.5	Word Embedding Features	28
4.4	Machine Learning Classifiers	28



4.5	Results and Discussion	29
4.5.1	Overall Results on the 15000s Dataset.....	29
4.5.2	Results on Different Sub-types of WUEs.....	33
4.6	Conclusion for Segment-level Detection.....	36
Chapter 5	Token-level Chinese WUE Detection.....	37
5.1	Task Description	37
5.2	Dataset	38
5.3	WUE Detection Based on Bidirectional LSTM	38
5.4	Sequence Embedding Features	42
5.4.1	Word Embeddings	42
5.4.2	POS Embeddings	42
5.5	Token Features.....	42
5.5.1	Out-of-Vocabulary Indicator	43
5.5.2	N-gram Probability Features	43
5.6	Evaluation.....	44
5.6.1	Accuracy	44
5.6.2	Mean Reciprocal Rank (MRR)	44
5.6.3	Hit@k Rate.....	44
5.6.4	Hit@r% Rate	45

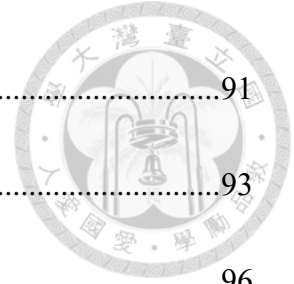


5.7	Results and Analysis	46
5.7.1	Overall Results	47
5.7.2	LSTM v.s. Bi-LSTM on Segments with Different Length	48
5.7.3	Relationship between Top Two Candidates	48
5.7.4	Effectiveness of POS Features	50
5.7.5	Effectiveness of N-gram Features	51
5.7.6	Performance on Commonly Misused Words.....	53
5.8	Conclusion for Token-level Detection	54
Chapter 6	Chinese WUE Correction	55
6.1	Task Description	55
6.2	Dataset	56
6.3	Neural Network-based Correction Generation Model.....	58
6.3.1	Model Overview.....	58
6.3.2	Model Output	59
6.3.3	Candidate Vocabulary.....	60
6.3.4	Model Parameters.....	60
6.4	Features.....	61
6.4.1	Target CWE+P Word Embedding	61
6.4.2	Target CWE Position-Insensitive Character Embedding	62



6.4.3	Context2vec Features	63
6.4.4	Target POS Feature	66
6.5	Language Model Re-ranking	68
6.5.1	Traditional N-gram Language Model (N-gram LM)	68
6.5.2	Recurrent Neural Network Language Model (RNNLM)	69
6.5.3	Re-ranking Method	70
6.6	Automatic Evaluation	71
6.6.1	Overall Results	71
6.6.2	Effect of LM Re-ranking	73
6.7	Human Evaluation	75
6.7.1	Motivation for Human Evaluation	75
6.7.2	Annotation Guideline	76
6.7.3	Annotation Agreement	80
6.7.4	Evaluation with Human Annotation	81
6.8	Error Analysis	83
6.8.1	Performance on Different POS Tags	83
6.8.2	Error Cases	84
6.9	Conclusion for WUE Correction	88
Chapter 7	Conclusion and Future Work.....	91

7.1	Conclusion	91
7.2	Future Work	93
	REFERENCE	96



LIST OF FIGURES



Figure 1-1	Workflow of our Chinese WUE detection and correction system.....	6
Figure 2-1	The architecture of CBOW and Skip-gram (Mikolov et al., 2013a).	13
Figure 2-2	The architecture of CWINDOW and Structured Skip-gram (Ling et al., 2015).	14
Figure 2-3	Comparison between CBOW and CWE (Chen et al., 2015).....	15
Figure 4-1	Accuracy v.s. dataset size.	35
Figure 5-1	LSTM-based WUE detection model.	40
Figure 5-2	Bidirectional LSTM-based WUE detection model.....	41
Figure 5-3	An example of undirected dependency graph.	49
Figure 6-1	A high-level view of our correction generation model.....	59
Figure 6-2	The architecture of Context2vec.....	64
Figure 6-3	The effect of parameter α in LM re-ranking.....	75
Figure 6-4	The annotation guideline presented to the annotators.	79

LIST OF TABLES



Table 1-1	Sub-types of WUE (with error tag CC) defined in the HSK corpus.	3
Table 1-2	Sub-types of WUE defined in this thesis.	4
Table 3-1	Example sentences in our dataset.	17
Table 4-1	The statistics of the balanced dataset “15000s”	21
Table 4-2	Hyperparameters of the CBOW/SG embeddings	28
Table 4-3	Performance of Decision Tree on the 15000s dataset.....	30
Table 4-4	Performance of Random Forest on the 15000s dataset.	30
Table 4-5	Performance of Support Vector Machine on the 15000s dataset.....	31
Table 4-6	Performance of Deep Neural Network on the 15000s dataset.....	31
Table 4-7	Performance of Random Forest on the 4000s_W and 4000s_U dataset.	33
Table 5-1	Parameters of the LSTM-based WUE detection model.....	39
Table 5-2	Performance of the LSTM/Bi-LSTM sequence labeling models with different sets of features.	46
Table 5-3	Hit@20% rates of LSTM and Bi-LSTM on segments with different lengths.	48
Table 5-4	Summary of the analysis of the dependency between the top two candidates.	50

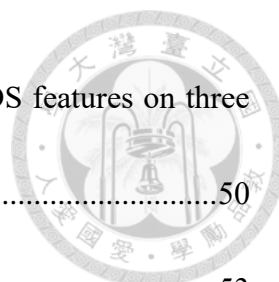


Table 5-5	Hit@20% rates of Bi-LSTM models with or without POS features on three most frequent POS tags of the erroneous token.....	50
Table 5-6	Precision/recall on four most commonly misused words.	53
Table 6-1	Statistics of the dataset used in the WUE correction stage.....	57
Table 6-2	Correction generation model parameters.....	61
Table 6-3	POS transitions that occurs at least 10 times in the validation set.....	66
Table 6-4	Hyperparameters of RNNLM.	70
Table 6-5	Performance of the correction generation model with various target and context features.....	72
Table 6-6	Correction performance with LM re-ranking.	73
Table 6-7	An example of alternative corrections.....	76
Table 6-8	WUE correction annotation instructions.	78
Table 6-9	Average κ of the annotation.....	80
Table 6-10	Proportion of candidates that meet the correctness criterion.....	81
Table 6-11	Correction performance with human evaluation.	83
Table 6-12	System performance on most frequent POS tags of the erroneous token.....	83
Table 6-13	Example in which segmentation error is the source of error.	85
Table 6-14	Comparison of n-gram LM base 10 log probabilities of correct and wrong segmentation results.....	85

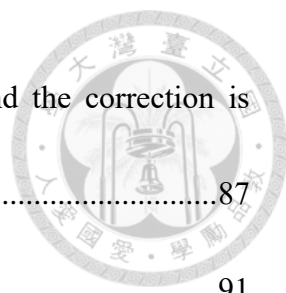


Table 6-15 Example in which the similarity between the target and the correction is context-dependent.....87

Table 7-1 Summary of information used in each stage.....91

Table 7-2 Summary of the best performance of each stage.92

Chapter 1 Introduction



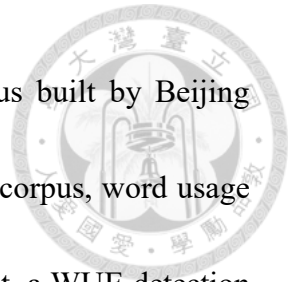
1.1 Motivation

Nowadays, more and more people around the world choose to learn Chinese as their second language. The need for automatic Chinese grammatical error detection and correction (GEC) tools, which facilitate learners in recognizing the errors and revising their sentences, is therefore increasing. Although the techniques for building GEC tools have been studied for decades, most of the studies are based on English learner data. The method of correcting sentences in Chinese, a language which differs substantially from English, has not yet been fully developed.

Leacock et al. (2014) pointed out that the mistakes made by non-native language learners differ from those made by native speakers in types and distribution. For example, native English speakers seldom make a verb tense error, which is one of the most common mistakes made by non-natives. Therefore, learner data should be considered when designing or training a correction system for non-native users. Moreover, such data is required to perform realistic evaluation on GEC systems targeting language learners.

However, since the ground-truth of the correction must be manually annotated by trained annotators, the available amount of data is limited. Compared to English ones, annotated Chinese learner corpora are even fewer. At the time of this study, the largest

available learner corpus was the HSK dynamic composition corpus built by Beijing Language and Culture University¹. According to the analysis of the corpus, word usage error (WUE) is the most frequent lexical-level error². Given this fact, a WUE detection and correction tool is worth developing.



1.2 Chinese Word Usage Error (WUE)

This research deals with the detection and correction of Chinese WUE. In Chinese sentences, a WUE is a grammatically or semantically incorrect token in which either the word itself is written in a wrong form, or the word is a correct existent word but is improper for its context.

On the website of the HSK corpus, WUE, whose error tag is CC, is divided into four sub-types, as shown in Table 1-1.

¹ <http://202.112.195.192:8060/hsk/index.asp>

² <http://202.112.195.192:8060/hsk/tongji2.asp>

Sub-type	Example (<i>correct_word</i> {CC <i>incorrect_word</i> }) ³
(1) Character disorder within a word	首先{CC 先首} (first of all) 眾所周知{CC 眾所知周} (as we all know)
(2) Incorrect selection of a word	雖然現在還沒有實現{CC 實踐}，…… (While it is not yet <u>implemented</u> , ...)
(3) Non-existent word	殘留量{CC 潛留量} (amount of residue) 農產品{CC 農作品} (agricultural product)
(4) Word collocation error	最好的辦法是兩個都保持{CC 走去}平衡。 (The best way is to <u>keep</u> both balanced.)

Table 1-1 Sub-types of WUE (with error tag CC) defined in the HSK corpus.

However, the error annotation does not contain information about which sub-type a WUE belongs to, and the divisions between some sub-types are not very clear. More specifically, CC (1) and (3) are similar in that the misused form is a non-existent word. On the other hand, in both CC (2) and (4) the misused form is an existent word. Therefore, in this thesis, CC (1) along with (3), and CC (2) along with (4) are merged into morphological errors (W) and usage errors (U) respectively.⁴ The WUE sub-types defined by this thesis are summarized in Table 1-2.

³ The example sentences/phrases extracted from the HSK corpus are originally written in simplified Chinese. In this thesis, they are manually converted to traditional Chinese versions by the author.

⁴ Strictly speaking, the result of a CC (1) error might not be a non-existent word. For example, “產生”(generate) can be a misused form of “生產”(produce). However, we define sub-types only based on the “result” of the error. Thus, a WUE is categorized as U-error as long as the misused form is an existent word.

Sub-type	Corresponding HSK CC Sub-types	# instances
Morphological errors (W)	(1) & (3)	4,010
Usage errors (U)	(2) & (4)	13,314

Table 1-2 Sub-types of WUE defined in this thesis.

To determine whether the misused form is an existent word, we query the Beijing University Dictionary (北京大學辭典)⁵. If the misused form cannot be found in the dictionary, the instance is marked as W-error; otherwise it is marked as U-error.

In morphological errors, the Chinese characters within a word are wrongly selected or permuted. Character usage within a word is considered an aspect of word usage, since most of the Chinese characters are able to form meaningful words by themselves. For usage errors, the problem may lie in collocational combination, or a discrepancy between the intended meaning (inferred from the context by annotators) and the literally expressed semantics. According to Table 1-2, the non-native Chinese learners contributing the HSK corpus misuse an existent Chinese word more frequently than writing a word in an incorrect form.

In both cases of WUEs, the error can be corrected by replacing the erroneous token with an appropriate word. However, in morphological errors, a non-existent word is likely to be segmented into a sequence of single characters by a dictionary-based segmentation

⁵ 〈現代漢語語法信息詞典的開發與應用〉，《中文與東方語言信息處理學會通訊》，pp. 81-86

system, which makes the determination of the start and end offset a difficult aspect for automatic correction. On the other hand, to correct usage errors, how to derive the intended semantics is the major issue.



1.3 Overview

This research divides the WUE detection and correction task into three stages.

Stage 1: Segment-level detection

Stage 2: Token-level detection

Stage 3: Correction

In stage 1, the detection task is formulated as a binary classification problem. Given a Chinese segment, typically delimited by punctuation marks like comma (,), we determine whether the segment is erroneous or not. The incorrect segments can then be passed to Stage 2, in which we build a sequence labeling model to estimate the level of incorrectness of each token. The tokens with the highest score of incorrectness are proposed to be potentially erroneous ones. In Stage 3, we assume the error position is known and rank candidate corrections based on both the context and the original (erroneous) token written by the language learner. The workflow of our system is shown in Figure 1-1.

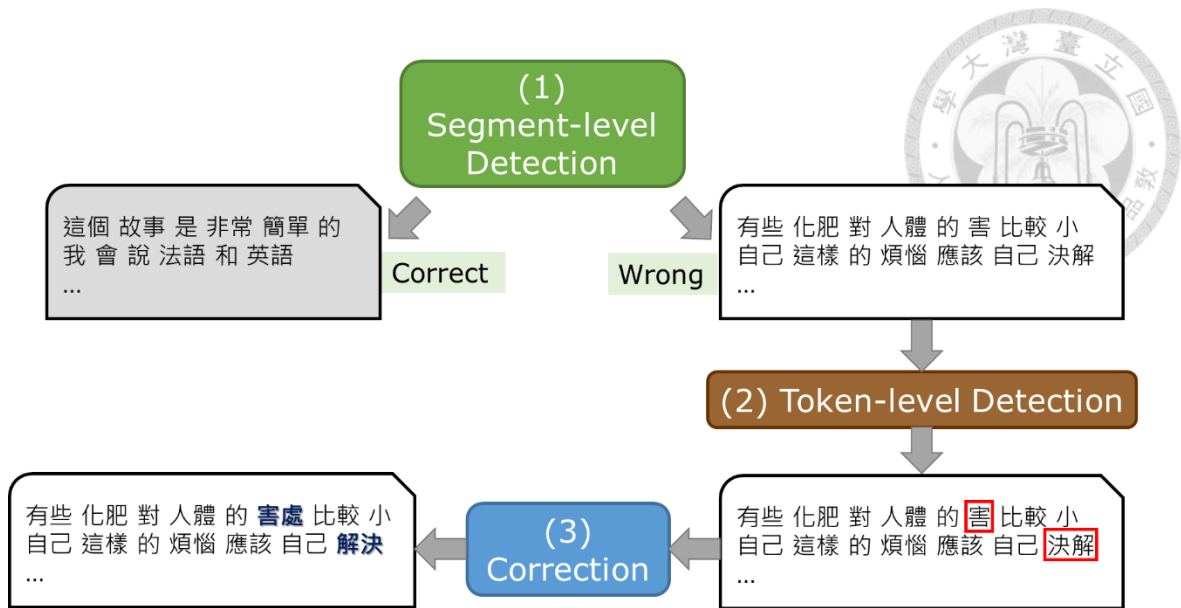


Figure 1-1 Workflow of our Chinese WUE detection and correction system.

1.4 Thesis Organization

The remainder of this thesis is organized as follows. Chapter 2 reviews the related work. Chapter 3 introduces the dataset adopted in this research and describes the pre-processing steps. Chapters 4 and 5 deal with segment-level and token-level WUE detection respectively. Chapter 6 presents our method of WUE correction. Chapter 7 concludes the thesis and discusses possible future directions.

Chapter 2 Related Work

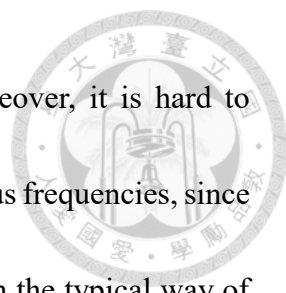


2.1 Grammatical Error Detection and Correction in English

Leacock et al. (2014) give a very comprehensive survey of past studies in automated GEC. They also point out that despite the importance of learner data, the limited amount of annotated learner corpora could make it difficult to build a robust statistical model. Therefore, researchers have also explored to combine statistical models with traditional rule-based approaches. Other methods of overcoming the limitation of the amount of data include constructing artificial error corpora and making use of large “grammatical” text corpora.

A disadvantage of using artificial data for machine learning models is that the distribution of the training data (artificially-made erroneous text) could differ a lot from that of the test data (real text written by language learners). In addition, it is possible that the model ends up learning the way of synthesizing data, instead of language learners’ pattern of making mistakes.

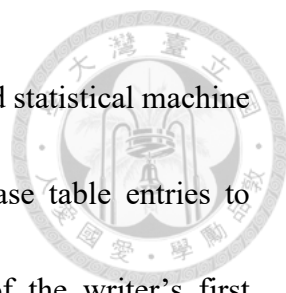
On the other hand, the use of well-formed text may suffer from the difference in domain and style. For example, large corpora which are composed of newspaper or Wikipedia text can be much more formal than the essays written by language learners. Also, the topics can be different, since language learners, especially beginners, are more



likely to write essays about themselves and their daily lives. Moreover, it is hard to determine whether an expression is grammatical based solely on corpus frequencies, since the language learners' way of expressing a meaning might differ from the typical way of native speakers. As a result, it would be difficult to draw a line between rare usage and wrong usage.

Another difficult aspect in GEC research pointed out by Leacock et al. (2014) is evaluation. Different research teams tend to adopt different typology of errors and evaluate on different datasets, so it might be hard to compare various systems and conclude which one is better. To measure the performance of GEC systems in a standardized manner, several shared tasks have been conducted, including Helping Our Own (HOO) 2011 (Dale and Kilgarriff, 2011), HOO 2012 (Dale et al., 2012), CoNLL 2013 (Ng et al., 2013) and CoNLL 2014 (Ng et al., 2014). Different types of grammatical errors were investigated. For example, five error types: article/determiner, preposition, noun number, verb form and subject-verb agreement, are evaluated in the CoNLL 2013 Shared Task on GEC. In CoNLL 2014, it is extended to 28 error types. Language models, machine learning-based classifiers, rule-based classifiers, and machine translation models have been explored.

The machine translation approach to GEC, which aims to model a transformation from incorrect text to its correction, has the major advantage that there is no need to



explicitly formulate the types of the errors. Based on the phrase-based statistical machine translation (SMT) framework, Dahlmeier and Ng (2011) add phrase table entries to handle semantic collocation errors resulting from the influence of the writer's first language (L1). Chollampatt et al. (2016b) explore neural network-based translation models, including Neural Network Global Lexicon Model (NNGLM) and Neural Network Joint Model (NNJM) for GEC. By incorporating features generated by neural network models into the phrased-based SMT model, they achieve substantial improvement over the top systems proposed in the CoNLL 2014 Shared Task. Chollampatt et al. (2016a) adapt a general NNJM with L1-specific text to capture the different regularities of errors made by learners with different L1. A Kullback-Leibler divergence regularization term is introduced to prevent overfitting to the relatively small L1-specific data. Improvements are shown on the data of three L1s: Chinese, Russian and Spanish.

Rei and Yannakoudakis (2016) argue that the evaluation of error correction can be subjective and focus only on error detection. The detection is performed with a sequence labeling framework, where the probability of being correct/incorrect is predicted at token level based on the word representation of each input token. They compare different composition architectures such as Convolutional Neural Network (CNN), Recurrent Neural Network (RNN) and Long-Short Term Memory (LSTM), and conclude that

LSTM is the most suitable model for this task.




2.2 Grammatical Error Detection and Correction in Chinese

For Chinese, spelling check evaluations were held at SIGHAN 2013 Bake-off (Wu et al., 2013) and SIGHAN 2014 Bake-off (Yu et al., 2014). The task is to detect and correct character errors in the given sentence. Similar character sets are provided. Two Chinese characters can be orthographically similar, such as "間" and "門", or phonetically similar, such as "揚" and "悌".

The Shared Task for Chinese Grammatical Error Diagnosis (Yu et al., 2014; Lee et al., 2015, 2016) extends the above task to word errors. Four kinds of errors, including redundant word, missing word, word disorder and word selection, are defined. The performance of the participants are reported on the whole dataset, so it is unclear whether some systems are better at certain error types. Besides, these tasks only deal with the detection but not the correction.

Huang and Wang (2016) use LSTM for Chinese grammatical error diagnosis. Each word in the sentence is represented by a randomly initialized real-valued vector. Their models are trained only on learner data. Without incorporating information derived from external well-formed text, the performance might be limited by the relatively small number of annotated sentences written by non-native learners.




The HSK corpus used in this research has been adopted by Yu and Chen (2012) to study word ordering errors (WOEs) in Chinese. They propose syntactic features, web corpus features and perturbation features for WOE detection. Cheng et al. (2014) identify sentence segments containing WOE, and further recommend the candidates with correct word orderings by using ranking support vector machine (SVM).

Huang et al. (2016) also use the HSK corpus to study the Chinese preposition selection problem. They propose gated recurrent unit (GRU)-based models to select the most suitable one from a closed set of 43 Chinese prepositions given the sentential context. Their approach can be utilized to both detect and correct preposition errors.

Nevertheless, it is still worth investigating how to handle WUEs involving other types of words such as verbs and nouns. Recognizing and correcting errors of such open-set types could be much more difficult since the set of candidates can be huge. To the best of our knowledge, this is the first research dealing with general-type Chinese WUE correction.

2.3 Distributed Word Representations

In the past few years, distributed word representations (word embeddings) derived from neural network models (Mikolov et al., 2013a; Pennington et al., 2014) have become popular among various studies in natural language processing. Based on the assumption



that similar words should share similar context, the representations can be trained on large text corpora in an unsupervised manner. Typically, these representations are in the form of real-valued vectors, whose dimensionality is low compared to the size of the vocabulary. Beyond surface forms, these low-dimensional vector representations can encode syntactic and semantic information implicitly (Mikolov et al., 2013b).

Because WUEs involve syntactic or semantic problems, vector representations could be promising for the detection/correction task. In this research, we experiment with three major types of word embeddings.

2.3.1 CBOW/Skip-gram Word Embeddings

These are the two architectures included in the Word2vec software (Mikolov et al., 2013a). The continuous bag-of-words model (CBOW) uses the words in a context window to predict the target word, while the skip-gram model (SG) uses the target word to predict every word in the context window. The two architectures are shown in Figure 2-1. In these two models, every context word is treated equally, so the information of word order is not preserved.

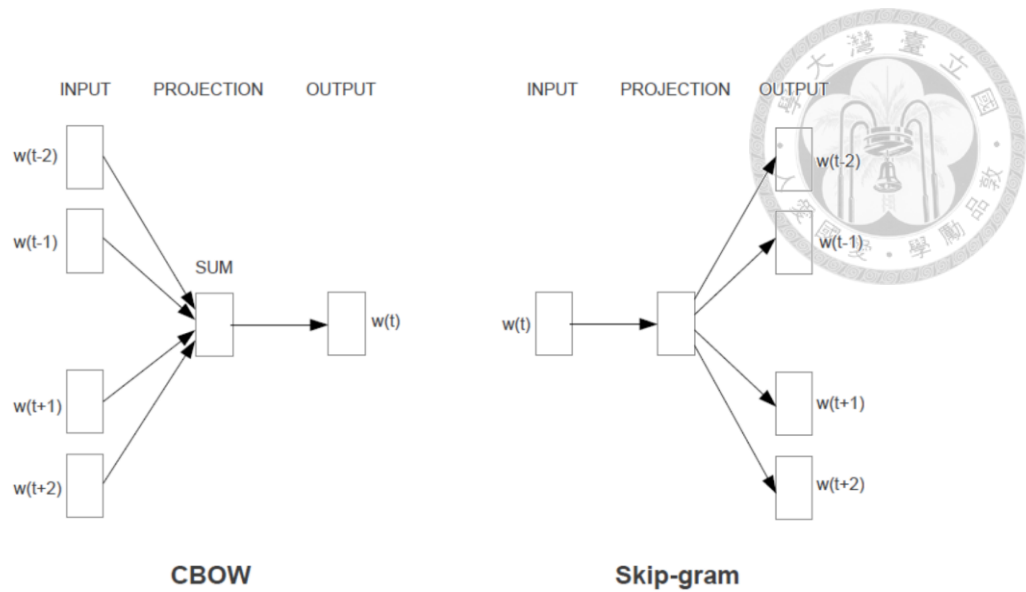


Figure 2-1 The architecture of CBOW and Skip-gram (Mikolov et al., 2013a).

2.3.2 CWINDOW/Structured Skip-gram Word Embeddings

The continuous window model (CWIN) and the structured skip-gram model (Struct-SG) (Ling et al., 2015) are extensions of CBOW and SG respectively, which take the order of context words into consideration. The former replaces the summation of context word vectors in CBOW with a concatenation operation, and the latter applies different projection matrices for predicting context words in different relative position with the target word. Figure 2-2 illustrates the two models.

In their paper, Ling et al. (2015) show that by using CWIN and Struct-SG embeddings, the performance of syntactic tasks such as part-of-speech (POS) tagging and dependency parsing can be enhanced. For Chinese WUE, the incorrect selection of a word might cause syntactic anomaly, so we examine whether these embeddings can help.

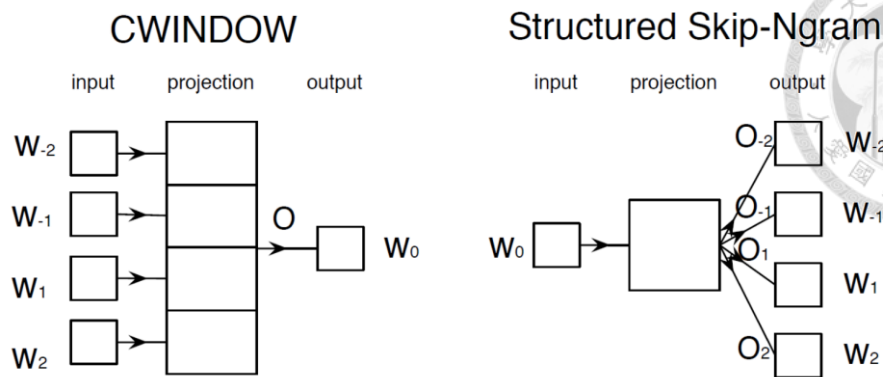
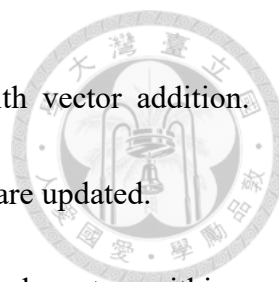


Figure 2-2 The architecture of CWINDOW and Structured Skip-gram (Ling et al., 2015).

2.3.3 Character-enhanced Word Embedding (CWE)

One of the most idiosyncratic aspect of the Chinese writing system is that the components of words, the Chinese characters, usually take on their own meanings. The meanings of individual characters usually contribute to the meaning of the word. For example, one familiar with Chinese can easily infer that “公車”(bus) is very likely to be a kind of “車”(vehicle), even without any context. Also, the character “公”(public) indicates that “公車” is a kind of public transportation.

Based on this characteristic of Chinese, Chen et al. (2015) proposed character-enhanced word embedding model (CWE), in which word vectors and character vectors are learned jointly. The model is based on CBOW. Figure 2-3 compares CWE with CBOW. In CWE, the representation of a context word is a combination of its own vectors and the



vectors of its constituent characters. The combination is done with vector addition. Throughout the process of training, both word and character vectors are updated.

As discussed in Chapter 1, one case of Chinese WUE is that the characters within a word is wrongly chosen or permuted. For instance, “决解” is a misused form of “解决” (solve). Though the misused form is a non-existent word, its character components serve as an important clue for discovering what the writer originally means and help provide more suitable correction. Therefore, we study how to utilize the CWE vectors, which encode both character and word information, to cope with the WUE correction task.

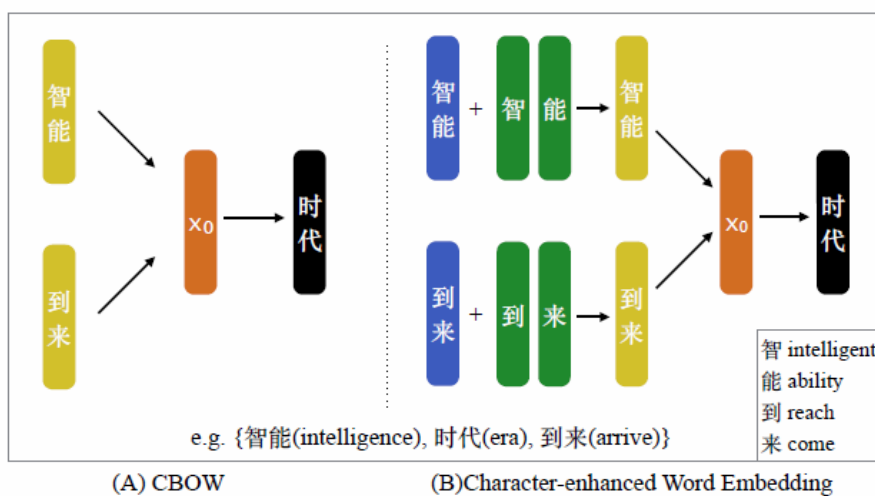


Figure 2-3 Comparison between CBOV and CWE (Chen et al., 2015).



Chapter 3 The HSK Word Usage Error Dataset



3.1 Data Collection

3.1.1 Split Sentences

We extract correct and wrong sentences from the HSK corpus, which contains essays written by non-native Chinese learners. The punctuation marks “。”, “?” and “!” are used to split the sentences. The sentences containing no error annotation are regarded as correct. To study the WUE correction task, we also include the correction of the wrong sentences in our dataset. Example sentences are shown in Table 3-1.

Correct sentence	我曾經到台灣讀書交了很多外國朋友，我們是用漢語說話的。
Wrong sentence	可想而知，他們長大以後會遇到很多的麻煩，甚至不適應生活，造成 不甚 後果。
Correction of the wrong sentence	可想而知，他們長大以後會遇到很多的麻煩，甚至不適應生活，造成 不良 後果。

Table 3-1 Example sentences in our dataset.

3.1.2 Convert Multiple-error Sentences into Single-error Ones

A sentence may contain multiple errors, including errors of types other than WUE. To focus on WUEs and enable systematic analysis, we convert a sentence containing n errors into n sentences, each of which only contains one error. That is, the following sentence with three errors:

○ ○ E1 ○ ○ ○ ○ ○ E2 ○ ○ ○ ○ E3 ○



will be converted into three sentences:

○ ○ E1 ○ ○ ○ ○ ○ ○ ○ ○ ○ ○ ○ ○ ○ ○ ○ ○
○ ○ ○ ○ ○ ○ ○ ○ ○ ○ E2 ○ ○ ○ ○ ○ ○ ○ ○ ○ ○
○ ○ ○ ○ ○ ○ ○ ○ ○ ○ ○ ○ ○ ○ ○ ○ E3 ○ ○

After conversion, only wrong sentences in which the error is a WUE are considered. We

got a total of 33,851 wrong sentences, each of which contains exactly one WUE.

3.2 Linguistic Processing

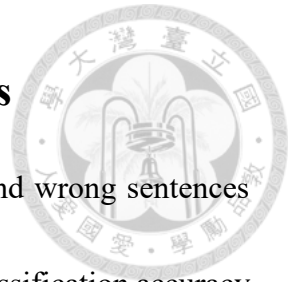
We process the extracted sentences with the Stanford CoreNLP toolkit (Manning et al., 2014). The following annotators are applied.

- Word Segmentation
- POS Tagging
- Dependency Parsing

The length of a sentence is defined to be the number of tokens in its word segmentation result. The POS tagging set of CoreNLP is that of the Chinese Penn Treebank. The tagging guideline can be found in (Xia 2000).

In each stage, we will extract features based on these three levels of information. The details will be provided in subsequent chapters.

3.3 Splitting Sentences into Sentence Segments



In our pilot experiments, we randomly sampled 1,000 correct and wrong sentences respectively and found out that we can achieve nearly 80% binary classification accuracy simply using a threshold on the length of the sentences. The reason is that a Chinese sentence is usually composed of several segments, mostly separated by comma “ , ” .

For example, the following sentence is composed of three segments:

如果我當推銷員的話，為了早點兒習慣，打算盡可能努力。

The longer a sentence is, the more likely a learner would make some grammatical errors somewhere. In fact, among the 2,000 selected sentences, the average length of the correct sentences is 7.8, while that of the wrong sentences is 25.6. If we mark the whole sentence as “wrong” only because one of the segments contains WUE, the benefit to the learner will be limited. Therefore, in the first stage, we consider a sentence segment as a unit of detection. We use the tokens with POS tag “PU” (punctuation mark) to split the segments.

3.4 Data Filtering



The raw essay text contains metadata such as serial number, title, or total number of words. In addition, after splitting with punctuation marks we get short segments such as the ones in “您好！”, “不過，...” and “那時，...”. It is nearly impossible to determine the correctness of these single-word segments individually. To conduct experiments on valid instances, we filter out segments:

- That contain digits or English alphabets
- Whose length is less than five

Finally, our dataset contains 63,612 correct segments and 17,324 segments with WUEs.

In each of the three stages, a subset of data is extracted from this complete dataset. The details will be described in the corresponding chapters.

Chapter 4 Segment-level Chinese WUE Detection



4.1 Task Description

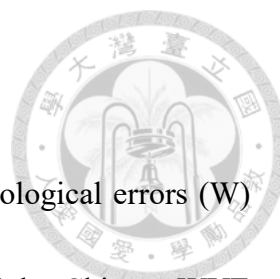
In this stage, we formulate WUE detection as a binary classification problem. Given a Chinese sentence segment, we build models to tell whether there is any WUE in the segment.

4.2 Dataset

According to Section 3.4, the number of the correct segments are much more than that of the wrong segments in our complete dataset. Therefore, a model can reach high performance by always guessing the majority class, yet this kind of system would not be helpful for language learners. To avoid this problem, we build a balanced dataset. We randomly select 15,000 correct and WUE segments respectively, and combine them into a dataset with 30,000 segments in total. This dataset is called “15000s”. The statistics of the dataset is summarized in Table 4-1.

	Correct Segments	Wrong Segments
# segments	15,000	15,000
# tokens	115,597	136,666
# types	8,532	11,187

Table 4-1 The statistics of the balanced dataset “15000s”



4.3 Features

In Section 1.2 we divided WUE into two sub-types, i.e., morphological errors (W) and usage errors (U). Based on this division, several properties of the Chinese WUE detection problem are worth noticing. W-errors can be identified almost at first sight, but for U-errors, even native speakers may have to “think twice”. For example, to determine if “體會” (realize) is a misuse of the word “體驗” (experience) in the sentence “親身體會了一場永遠難忘的電單車意外” (personally realize an accident which was never forgotten), we have to consider its collocation with “意外” (accident). On the other hand, any error with a non-existent word such as “農作品” can be detected solely by its extremely low frequency in a Chinese corpus.

To detect the existence of WUE, we experiment with several sets of features. All the following features are utilized in combination with segment length (s_len), which is defined to be the number of tokens in the segment.

4.3.1 Google N-gram Features

We adopt the Chinese version of Google Web 5-gram (Liu et al., 2010) to generate n-gram features. For every word sequence of length n ($n = 2, 3, 4, 5$) in a segment, we calculate the n-gram probability by Maximum Likelihood Estimation (MLE). For example, the tri-gram probability is defined as follows.

$$p(w_i | w_{i-2}, w_{i-1}) = \frac{c(w_{i-2}, w_{i-1}, w_i)}{c(w_{i-2}, w_{i-1})}$$



, where $c(\cdot)$ is the frequency of the word sequence in the Google Web 5-gram corpus.

For a segment $w_1 w_2 \dots w_L$, all n -gram features are concatenated into a feature vector

$\mathbf{G} = (g_2, g_3, g_4, g_5)$, where

$$g_n = \sum_{i=n}^L p(w_i | w_{i-n+1}, \dots, w_{i-1})$$

The reason for using the summation of probabilities, instead of average probability, is that the relationship between segment length and probability sum might not be linear. Since this set of features is combined with s_len , the model is expected to capture more complex relationship.

To query the n -gram counts efficiently, we utilized `marisa-trie`⁶, a data structure that can save memory usage and enable fast search by grouping keys (n -grams) that share the same prefixes.

4.3.2 Dependency Count Features

Errors in a sentence affect the result of segmentation and parsing. We postulate that there is a certain distribution of dependency counts in normal sentences, and the counts of error sentences deviate from the distribution. An example is shown below.

⁶ <https://github.com/pytries/marisa-trie>

Correct segment	Wrong segment
以下 介紹 一下 我的 簡歷 和 經驗 。	以下 紹 介 一下 我的 簡歷 和 經驗 。
nsubj(介紹-2, 以下-1) root(ROOT-0, 介紹-2) advmod(介紹-2, 一下-3) assmod(經驗-8, 我-4) case(我-4, 的-5) conj(經驗-8, 簡歷-6) cc(經驗-8, 和-7) dobj(介紹-2, 經驗-8)	nsubj(介-3, 以下-1) advmod(介-3, 紹-2) root(ROOT-0, 介-3) advmod(介-3, 一下-4) assmod(經驗-9, 我-5) case(我-5, 的-6) conj(經驗-9, 簡歷-7) cc(經驗-9, 和-8) dobj(介-3, 經驗-9)

By comparing the two dependency parsing results, we can find out that ”紹介”, a misused form of“介紹”(introduce), is (incorrectly) segmented into two words and results in an additional, unreasonable dependency relation *advmod*(介-3, 紹-2).

Therefore, we take the count of each type of dependency as a set of features. Let *dep* be the type of dependency such as *nsubj* and *dobj*. For each dependency, we compute two types of count:

(1) internal count (*dep_int_cnt*): counts the occurrence if the two words are both in the target segment.

(2) external count (*dep_ext_cnt*): counts as long as one of the words is in the target segment.

The total internal (*all_int_cnt*) and external counts (*all_ext_cnt*) are also considered.

An example of the calculation of this set of features is shown below.

聽說貴公司在國內很有名，外國顧客也很多。



root(ROOT-0, 聽說-1)

nn(公司-3, 貴-2)

nsubj(有名-7, 公司-3)

case(國內-5, 在-4)

prep(有名-7, 國內-5)

advmod(有名-7, 很-6)

ccomp(聽說-1, 有名-7)

nn(顧客-10, 外國-9)

nsubj(很多-12, 顧客-10)

advmod(很多-12, 也-11)

conj(有名-7, 很多-12)

The corresponding feature values of the shaded segment are:

nn_int_cnt	1	nn_ext_cnt	1
nsubj_int_cnt	1	nsubj_ext_cnt	1
case_int_cnt	1	case_ext_cnt	1
prep_int_cnt	1	prep_ext_cnt	1
advmod_int_cnt	1	advmod_ext_cnt	1
ccomp_int_cnt	1	ccomp_ext_cnt	1
conj_int_cnt	0	conj_ext_cnt	1
all_dep_int_cnt	6	all_dep_ext_cnt	7

For all other dependency types that do not appear, the feature value is 0. Note that though the unit of detection is a segment, the external count features can introduce information across segments.

There are 45 types of dependencies in our dataset, and we also include total internal and external counts. The result feature vector D has 92 dimensions.



4.3.3 Dependency Bigram Features

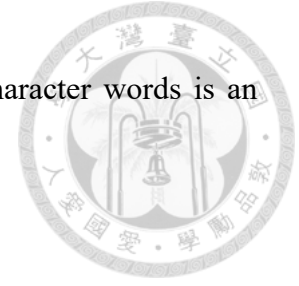
Long distance dependency is common in Chinese sentences. In the example, “親身體會了一場永遠難忘的電單車意外”，“意外” is the object of “體會”，but there are 6 words in-between, falling out of the range of n-gram features. To cope with the problem, we generate dependency bigrams. The above sentence contains dependencies such as nsubj(體會-2, 親身-1) and dobj(體會-2, 意外-9). We compose the two words in each dependency, i.e. (親身, 體會) and (體會, 意外), query the Google n-gram corpus, and calculate the bigram probabilities. Below is an example showing that the frequency of the bigram composed of correct collocation is higher than that composed of wrong collocation.

	Bigram	Frequency
Wrong	體會 意外	0
Correct	經歷 意外	167

Since the collocating behavior may vary with dependency type, we sum the bigram probabilities of each type respectively. Similar to the dependency count features, we calculate both internal sum (*dep_int_sum_prob*) and external sum (*dep_ext_sum_prob*). The probability sums of all dependency types are also included (*all_int_sum_prob*, *all_ext_sum_prob*). This set of features, denoted by feature vector *B*, has 92 dimensions.

4.3.4 Single-character Features

A non-existent Chinese word (W-error) is usually separated into several single-



character words after segmentation, so the occurrence of single-character words is an important indicator of WUEs. We define the following features:

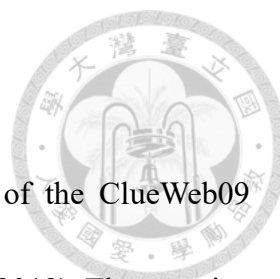
- (1) *seg_cnt*: number of contiguous single-character blocks
- (2) *len2above_seg_cnt*: number of contiguous single-character blocks with length no less than 2
- (3) *max_seg_len*: length of the maximum contiguous single-character block
- (4) *sum_seg_len*: sum of the lengths of all contiguous single-character blocks

Consider the following segment as an example:

而且 我 認為 貴 公司 是 我國 最大 的

(..., and I thought that your company is the biggest in our country.)

The feature values are 4, 1, 3, and 6, respectively. The proposed features are concatenated into a vector *S*.



4.3.5 Word Embedding Features

We train CBOW/SG word embeddings on the Chinese part of the ClueWeb09 dataset⁷. The Chinese part was extracted and segmented by Yu et al. (2012). The negative sampling objective is adopted. The hyperparameter settings are shown in Table 4-2. All other hyperparameters not mentioned are left default.

Embedding size	400
Window size	5
# negative samples	10
Iterations	20

Table 4-2 Hyperparameters of the CBOW/SG embeddings

For each segment, we sum the vectors of all the words in it. We concatenate CBOW and SG embeddings into a feature vector W . The dimensionality of W is therefore 800.

4.4 Machine Learning Classifiers

We experiment with four kinds of machine learning classifiers, including Decision Tree (DT), Random Forest (RF), Support Vector Machine with RBF kernel (SVM), and Feed-forward Neural Network (Deep Neural Network, DNN). For the first three models,

⁷ <http://lemurproject.org/clueweb09.php>



we use the implementation of scikit-learn⁸; for DNN, we use libdnn⁹.

For SVM and DNN, we scale the feature vector to zero mean and unit variance, since the range of the feature values can affect the performance of these two models.

4.5 Results and Discussion

4.5.1 Overall Results on the 15000s Dataset

The performance of each kind of classifiers on different sets of features are shown in Table 4-3 ~ Table 4-6. The *s_len* feature is included in every feature combination. For each combination, we report the performance metrics of the model with best accuracy among various parameter settings. All the reported performance values are the average of 10-fold cross validation. The metrics are computed as follows.

		Model Prediction	
		Wrong (with WUE)	Correct (no WUE)
Ground-truth	Wrong (with WUE)	True Positive (TP)	False Negative (FN)
	Correct (no WUE)	False Positive (FP)	True Negative (TN)

$$\text{accuracy} = \frac{\text{TP} + \text{TN}}{\text{TP} + \text{FN} + \text{FP} + \text{TN}}$$

$$\text{precision} = \frac{\text{TP}}{\text{TP} + \text{FP}} \quad \text{recall} = \frac{\text{TP}}{\text{TP} + \text{FN}}$$

⁸ <http://scikit-learn.org/stable/>

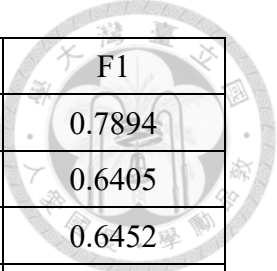
⁹ <https://github.com/botonchou/libdnn>

Feature	Accuracy	Precision	Recall	F1
G	0.8354	0.9268	0.7285	0.8158
D	0.6430	0.6586	0.5955	0.6254
B	0.6437	0.6311	0.6960	0.6620
S	0.5931	0.6009	0.5560	0.5776
W	0.6018	0.6159	0.5446	0.5780
GD	0.8311	0.9557	0.6945	0.8044
GB	0.8311	0.9569	0.6935	0.8042
GS	0.8301	0.9201	0.7230	0.8097
GW	0.8123	0.8687	0.7359	0.7968
All	0.8299	0.9499	0.6968	0.8039

Table 4-3 Performance of Decision Tree on the 15000s dataset

Feature	Accuracy	Precision	Recall	F1
G	0.8398	0.9620	0.7075	0.8154
D	0.6636	0.6767	0.6267	0.6507
B	0.7026	0.7211	0.6611	0.6898
S	0.5970	0.6065	0.5557	0.5800
W	0.6623	0.6779	0.6187	0.6469
GD	0.8376	0.9324	0.7281	0.8177
GB	0.8425	0.9450	0.7274	0.8220
GS	0.8342	0.9571	0.6998	0.8085
GW	0.8281	0.8551	0.7901	0.8213
All	0.8315	0.8405	0.8185	0.8293

Table 4-4 Performance of Random Forest on the 15000s dataset.



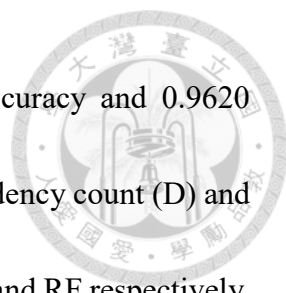
Feature	Accuracy	Precision	Recall	F1
G	0.7965	0.8184	0.7624	0.7894
D	0.6706	0.7050	0.5868	0.6405
B	0.6582	0.6711	0.6213	0.6452
S	0.5982	0.6066	0.5604	0.5826
W	0.6832	0.6783	0.6973	0.6877
GD	0.8017	0.8030	0.7997	0.8013
GB	0.7807	0.7732	0.7945	0.7837
GS	0.7821	0.7900	0.7686	0.7791
GW	0.7222	0.7295	0.7065	0.7178
All	0.7350	0.7218	0.7649	0.7427

Table 4-5 Performance of Support Vector Machine on the 15000s dataset.

Feature	Accuracy	Precision	Recall	F1
G	0.8106	0.8621	0.7399	0.7963
D	0.6398	0.6520	0.6014	0.6257
B	0.6200	0.6342	0.5731	0.6021
S	0.6050	0.6230	0.5403	0.5787
W	0.6800	0.6960	0.6399	0.6668
GD	0.8087	0.8763	0.7203	0.7907
GB	0.7524	0.7683	0.7273	0.7472
GS	0.8123	0.8802	0.7243	0.7947
GW	0.7285	0.7221	0.7469	0.7343
All	0.7955	0.8288	0.7461	0.7853

Table 4-6 Performance of Deep Neural Network on the 15000s dataset.

Since we use a balance dataset, the baseline accuracy is 50%. As can be seen, all the models with each of the proposed set of features outperform the baseline. Google n-



gram(G) is the most effective set of features, reaching 0.8398 accuracy and 0.9620 precision on its own with the RF model. The best accuracy of Dependency count (D) and Dependency bigram (B) features are about 0.67 and 0.70, with SVM and RF respectively. For single character features (S), the accuracy is only around 0.6. The word embedding features (W) work better with SVM and DNN, with accuracy about 0.68.

Because G is the strongest set of features, we try to combine it with others. Although individual sets of features cooperate better with different classifiers, generally DT and RF handle feature combinations better. Note that RF is a model that ensembles many DTs, so RF is usually better than a single DT. We focus our discussion on RF below. The best accuracy on the 15000s dataset is 0.8425, achieved by the GB feature combination. While the accuracy is better than that of RF only with G features, the precision slightly decreases to 0.945. W features increase the recall by about 9% when used with the G features, but at the expense of precision. D and B features also help increase the recall when used with G. With all sets of features, precision and recall are balanced, which are 0.8405 and 0.8185 respectively, and the best F1 score 0.8293 is achieved.

By utilizing suitable combination of features, we can construct a system that favors precision or recall, according to specific application purposes. Especially, the precision as high as 0.96 we achieved indicates that our system is highly reliable and seldom produces misleading results.



4.5.2 Results on Different Sub-types of WUEs

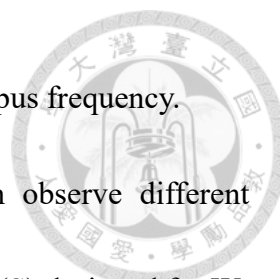
To test the performance of our system on different WUE sub-types, we sample 4,000 segments from each subtype and combine them with 4,000 correct segments respectively.

The generated dataset, called 4000s_W and 4000s_U, contains 8,000 segments respectively. We use RF to conduct the experiments. The experimental results of the two datasets are shown in Table 4-7.

Feature	4000s_W				4000s_U			
	Acc.	P	R	F1	Acc.	P	R	F1
G	0.7969	0.8425	0.7313	0.7829	0.6355	0.6321	0.6515	0.6417
D	0.6420	0.6370	0.6608	0.6486	0.6211	0.6151	0.6528	0.6334
B	0.6210	0.6205	0.6235	0.6220	0.6299	0.6486	0.5678	0.6055
S	0.6330	0.6172	0.7008	0.6563	0.5998	0.5872	0.6725	0.6270
W	0.6095	0.6110	0.6038	0.6074	0.6139	0.6321	0.5453	0.5855
GD	0.8213	0.8630	0.7640	0.8105	0.6813	0.6981	0.6388	0.6671
GB	0.8091	0.8790	0.7188	0.7908	0.6935	0.6906	0.7010	0.6958
GS	0.8226	0.8920	0.7345	0.8056	0.6396	0.6586	0.5805	0.6171
GW	0.7714	0.7722	0.7705	0.7714	0.6529	0.6563	0.6420	0.6491
All	0.8056	0.7757	0.8608	0.8160	0.6881	0.6956	0.6700	0.6826

Table 4-7 Performance of Random Forest on the 4000s_W and 4000s_U dataset.

Detecting U-errors is generally harder than detecting W-errors. The best accuracy of W-errors is above 0.82, but the best of U-errors is less than 0.7. On both sub-types, G is the most effective. However, for U-errors the difference in performance between G and other individual sets is smaller. This is consistent with what we previously discussed, that



compared to W-errors, U-errors cannot be easily detected by low corpus frequency.

Considering the combination of G with the others, we can observe different functionalities of different feature sets. The single-character features (S) designed for W-errors improve the accuracy when used with G on the 4000s_W dataset. S feature is not very effective on its own, probably because the existence of single-character words is not sufficient to conclude that there is something wrong with the segment. For example, the following “correct” sentence contains many single character words due to its grammatical structure:

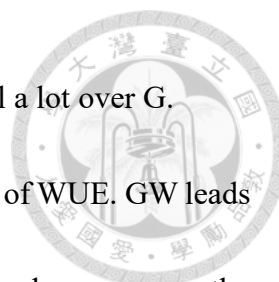
有人對她說

In contrast, the following segment contains a short problematic sequence of single characters “共 敬” whose bigram probability is less than 0.0001. (To correct the error, the two tokens “共 敬” should be replaced by a single word “尊敬”)

他們應該共敬父母

Therefore, G and S features can together provide useful information for the model in such cases.

On the other hand, D and B, the features derived from the result of dependency parsing, are helpful on the 4000s_U dataset. The accuracy both improves when D or B is used with G. This indicates that dependency information can help the models handle collocation errors better, especially those involve long-distance dependency. While GD



results in slightly lower recall compared to G, GB increases the recall a lot over G.

W features also have different effects for two different sub-types of WUE. GW leads to better recall over G on the 4000s_W dataset, and better precision and accuracy on the 4000s_U dataset.

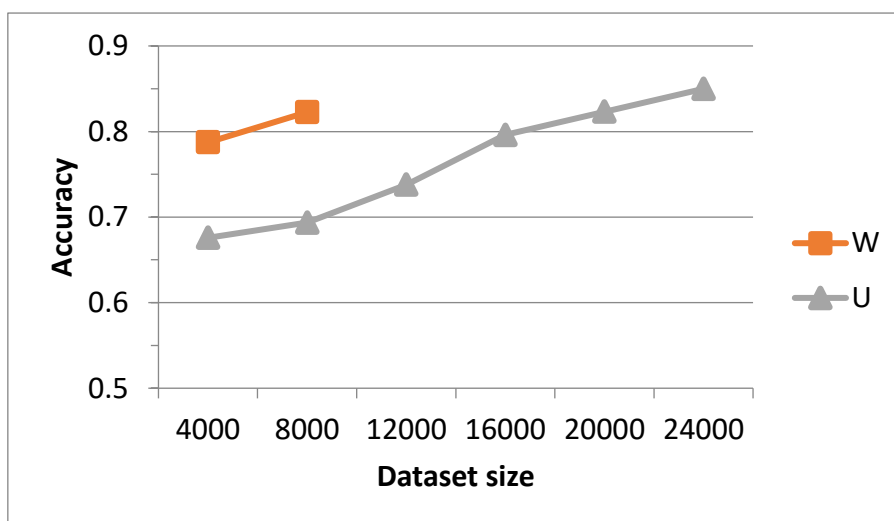


Figure 4-1 Accuracy v.s. dataset size.

Figure 4-1 shows the relationship between the best accuracy and the dataset size in the experiments of RF. Due to the amount of available data for W-errors, only two datasets are generated. The dataset size is the total number of segments, including both wrong and correct ones. Therefore, the datasets with size 8000 are just 4000s_W and 4000s_U we used previously.

With the largest dataset, the accuracy for U-errors reaches 0.8499. The accuracy of two sub-types both increases with the amount of training data. To reach the same level of

accuracy, more training data are needed for U-errors, which also indicates that handling U-errors is more difficult.



4.6 Conclusion for Segment-level Detection

In this stage, we deal with the Chinese WUE detection task with n-gram features, dependency count features, dependency bigram features, single-character features and word embedding features. The best model achieves 0.8425 accuracy, 0.9450 precision, 0.7274 recall, and 0.8220 F1 on the 15000s balanced dataset. Among the tested machine learning classifiers, random forest is the most suitable model for the proposed features.

The single-character features in combination with n-gram features are effective for morphological errors (W), while dependency-derived features help better capture usage errors (U). The detection of usage error is harder and need more training data.

By utilizing suitable model and combination of features, we can also construct a WUE detection system that favors precision, up to 96.2%. If a segment is classified as wrong by our high-precision model, it is very likely that there is indeed some WUE in the segment. Therefore, in our next stage, we consider those segments known to contain WUE and design models to locate the specific erroneous token.

Chapter 5 Token-level Chinese WUE Detection



5.1 Task Description

The goal of this stage is to locate the specific location of WUE given known incorrect segments. We formulate the Chinese WUE detection task as a sequence labeling problem. Each token, the fundamental unit after word segmentation, is labeled either correct (0) or incorrect (1).

As we mentioned in Chapter 1, U-errors are much more than W-errors in the HSK dataset. In fact, many Chinese WUEs result from subtle semantic unsuitability instead of violation of syntactic constraints. In (S5-1), both “權力” (power) and “權利” (right) are existent nouns in Chinese, and both versions are grammatically correct.

(S5-1) 人們有(*權力,權利)吃安全的食品。

(People have the (*power, right) to enjoy safe food.)

To recognize the WUE, we have to understand the meaning of “吃安全的食品”, so that we can determine whether it is a kind of right, or a kind of power. If the context changes, which word is the correct choice is very likely to be different.

The above example can shed light on one of the challenging aspects of this task. The errors usually do not stand on their own, but the problems lie in their relationship with their context. If a model examines the sentence segment token by token, it would not be able to detect this kind of WUEs. Therefore, we need a model that considers the sequence

of words as a whole to determine which position needs correction.



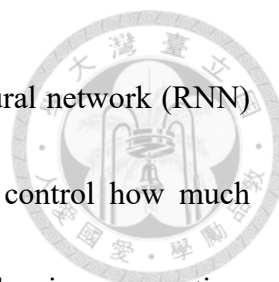
5.2 Dataset

We extract the wrong segments in the 15000s balanced dataset used in the previous stage. Each sentence segment has exactly one token-level position that is erroneous. We filter out any sentence segment whose corrected version differs from it by more than one token due to segmentation issues. Some W-error instances are filtered out since the erroneous token is segmented into several words. We only focus on the cases in which the error can be corrected by replacing one single token.

After filtering, we end up with 10,510 sentence segments. We use 10% data for validation and testing respectively, and the remaining 80% data as the training set.

5.3 WUE Detection Based on Bidirectional LSTM

We previously discuss the challenge of token-level WUE detection and conclude that a model suitable for this task needs to consider the whole sequence words. One possible model is the Long Short-Term Memory (LSTM) model (Hochreiter and Schmidhuber, 1997), which processes sequential data and generates the output based not only on the information of the current time step, but also on the past information stored in the memory layer. Therefore, we utilize LSTM as our sequence labeling model.



LSTM handles long sequences better than simple recurrent neural network (RNN) does, since it is equipped with input, output and forget gates to control how much information is used. The ability of LSTM to capture longer dependencies among time steps makes it suitable for modeling the complex dependencies of the erroneous token on the other parts of the sentence.

We train the LSTM model with the Adam optimizer (Kingma and Ba, 2014) implemented in Keras¹⁰. The parameters are shown in Table 5-1. The training process is stopped when the validation accuracy does not increase for two consecutive epochs. The model with the highest validation accuracy is selected as the final model.

LSTM layer size	400
Cost function	binary_crossentropy
Optimizer	Adam
Batch size	32
Initial learning rate	0.001

Table 5-1 Parameters of the LSTM-based WUE detection model.

We apply a sigmoid activation function before the output layer, so the output score of each token, which is between 0 and 1, can be interpreted as the predicted level of incorrectness. With these scores, our system can output a ranked list of candidate error

¹⁰ <https://keras.io/>



positions. The positions with the highest incorrectness scores will be marked as incorrect.

We show an example labeling result of our system with segment (S5-2). The tokens “差” (bad) and “知識”(knowledge), which are given the highest scores, are most likely to be incorrect. We also use (S5-2) to illustrate the LSTM-based WUE detection model in Figure 5-1. The darkness of the blocks in the bottom indicates the level of incorrectness predicted by the model.

(S5-2)	學習	的	知識	也	很	差
Incorrectness score	0.056	0.035	0.153	0.039	0.030	0.429

(The knowledge learned is also very bad.)

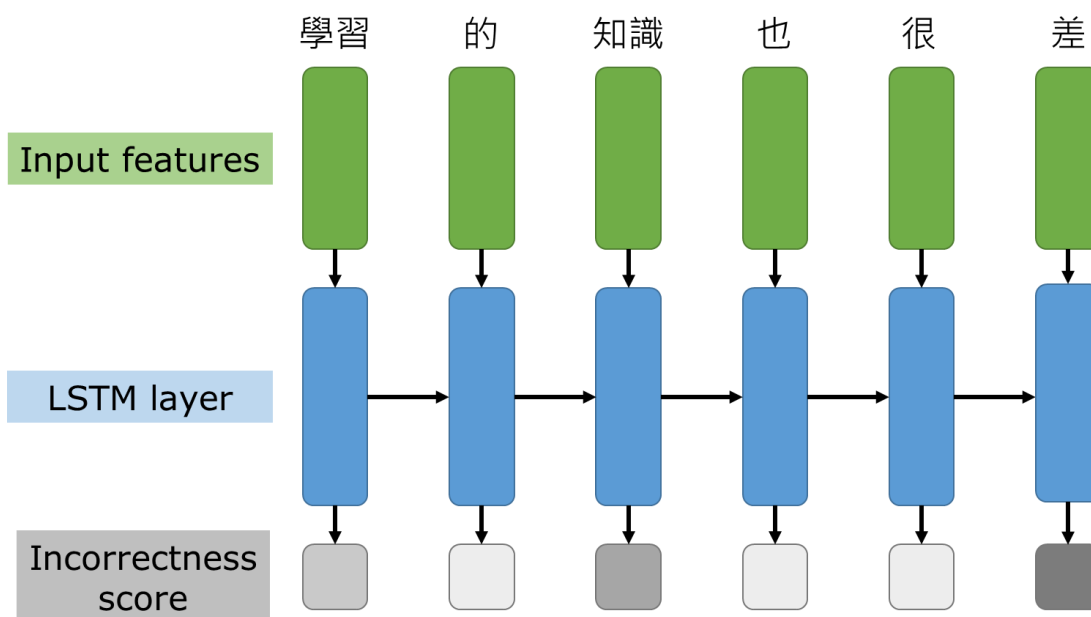
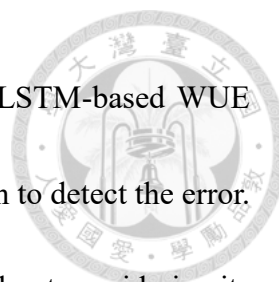


Figure 5-1 LSTM-based WUE detection model.

Bidirectional LSTM (Schuster and Paliwal, 1997) is an extension of LSTM which includes a backward LSTM layer. Both information before and after the current time step



are taken into consideration. Figure 5-2 illustrates a bidirectional LSTM-based WUE detection model. In segment (S5-3), we need the “future” information to detect the error.

The incorrectness of the token 留在(left at) cannot be determined without considering its object 我們(us).

(S5-3) 店是爸爸(*留在,留給) 我們的。

(The store is our father left (*at,to) us.)

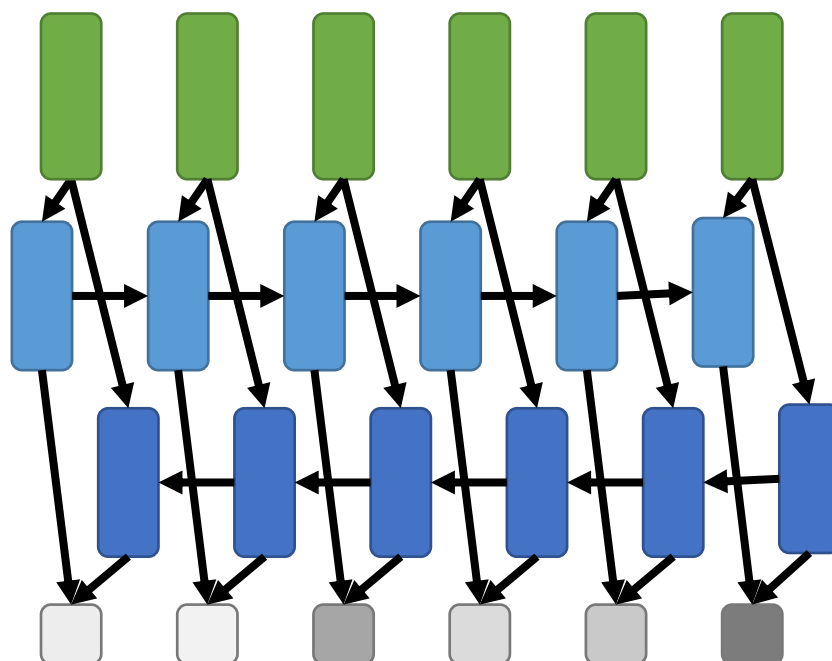


Figure 5-2 Bidirectional LSTM-based WUE detection model.



5.4 Sequence Embedding Features

We consider the word sequence in a sentence segment and the corresponding POS tag sequence. They are mapped to sequences of real-valued vectors through an embedding layer. These vectors are also updated during the training process.

5.4.1 Word Embeddings

We set the word embedding size to 400. Besides randomly initialized embedding, we also try several types of pre-trained word vectors.

1. CBOW/Skip-gram Word Embeddings: two basic architectures of Word2vec
2. CWINDOW/Structured Skip-gram Word Embeddings: take the order of the context words into consideration

The training corpus and the hyperparameter settings are the same as those used for obtaining W features in the previous stage.

5.4.2 POS Embeddings

The POS embeddings are randomly initialized. We set the embedding size to 20, which is slightly smaller than the number of different POS tags (30) in our dataset.

5.5 Token Features

In addition to representing each token as a real-valued vector, we also incorporate some abstract features. These features are derived from the Google Chinese Web 5-gram



corpus and will be referred to as “n-gram features”.

5.5.1 Out-of-Vocabulary Indicator

This feature is simply a bit indicating whether a word is an out-of-vocabulary (OOV) word or not. If a token never appears in the Web 5-gram corpus, the bit is set to 1; otherwise it is set to 0. A *W*-error token which is not segmented into several words is expected to be captured with the help of this binary indicator.

5.5.2 N-gram Probability Features

We compute the *n*-gram probability of each token using the occurrence count in the Web 5-gram corpus. We consider only up to trigrams since the probabilities are mostly zero when $n > 3$. For boundary cases such as the first token for bigram, and the first and second token for trigram, a special value -1 is given.

In fact, this is similar to the Google *n*-gram features we used in the previous segment-level detection task. The difference is that we preserve the sequence of probabilities. Given the limited amount of available learner data, these probabilities may serve as useful features indicating how likely an expression is valid in Chinese.



5.6 Evaluation

5.6.1 Accuracy

We use the detection accuracy as our main evaluation metric. A test instance is regarded as correct only if our system gives the highest score of incorrectness for the ground-truth position. This metric is relatively strict as the average length of the sentence segments in our dataset is 9.24. The McNemar’s test (McNemar, 1947) is adopted to perform statistical significance test.

5.6.2 Mean Reciprocal Rank (MRR)

The mean reciprocal rank rewards the test instances for which the model ranks the ground-truth near the top of the candidate list. The definition of MRR is:

$$MRR = \sum_{i=1}^N \frac{1}{rank(i)}$$

, where N is the total number of test instances and $rank(i)$ is the rank of the ground-truth position of test instance i .

5.6.3 Hit@k Rate

The Hit@k rate regards a test instance as correct if the answer is ranked within the top k places. In the experiments, k is set to 2. We report this metric since one of the most common types of WUEs is collocation error. In example (S5-2), the problem involves a pair of words, i.e., the adjective “差” (bad) is not a suitable modifier of the noun “知識”



(knowledge). (S5-4) and (S5-5) are both acceptable.

(S5-4) 學習的知識也很不足

(The knowledge learned is also insufficient.)

(S5-5) 學習的態度也很差

(The attitude of learning is also very bad.)

Which correction is better highly depends on the context or even the intended meaning in the writer's mind. If the model proposes two potentially erroneous tokens which are closely related to each other, it can be useful for Chinese learners.

5.6.4 Hit@r% Rate

Finding the exact position of the error could be more challenging in a longer sentence segment. Therefore, we propose another hit rate measure which takes the segment length (s_len) into account. Specifically, we regard one test instance as correct if the answer is ranked within the top $\max(1, \lfloor s_len * r\% \rfloor)$ candidates. We report hit@20%. That is, for segments shorter than 10 tokens, the system is allowed to propose one candidate; for those whose length is between 10 and 14, the system is allowed to propose two, and so on. Equivalently, this measure judges whether our system can rank the ground-truth error position within the top 20% of the candidate list. This metric compromises Accuracy and Hit@k.

5.7 Results and Analysis



Model	Features	Acc.	MRR	Hit@2	Hit@20%
Rand. baseline	-	0.1239	0.3312	0.2478	0.1611
LSTM	Rand. Init. Word Emb.	0.4186	0.6010	0.7222	0.6565
	CBOW	0.4072	0.5923	0.7155	0.6432
	CBOW + POS	0.4263	0.6150	0.7564	0.6908
	CBOW + POS + n-gram	0.4386	0.6204	0.7526	0.6755
	SG	0.4072	0.5910	0.7146	0.6365
	SG + POS	0.4301	0.6170	0.7593	0.6965
	SG + POS + n-gram	0.4386	0.6205	0.7507	0.6755
	CWIN	0.4853	0.6537	0.7774	0.7031
	CWIN + POS	0.4681	0.6435	0.7783	0.7022
	CWIN + POS + n-gram	0.4700	0.6502	0.7945	0.7269
	Struct-SG	0.4710	0.6412	0.7650	0.6889
	Struct-SG + POS	0.4757	0.6441	0.7593	0.6822
Struct-SG + POS + n-gram	0.4881	0.6577	0.7840	0.7184	
Bi-LSTM	CWIN	0.4795	0.6547	0.7840	0.7174
	CWIN + POS	0.5138	0.6789	0.8097	0.7479
	CWIN + POS + n-gram	0.4948	0.6719	0.8173	0.7507
	Struct-SG	0.4786	0.6544	0.7945	0.7212
	Struct-SG + POS	0.4653	0.6470	0.7850	0.7146
	Struct-SG + POS + n-gram	0.4948	0.6658	0.8040	0.7374

Table 5-2 Performance of the LSTM/Bi-LSTM sequence labeling models with different sets of features.



5.7.1 Overall Results

Table 5-2 shows the performance of our WUE detection models with different input features. The random baseline is a system randomly choosing one token as the incorrect position. The LSTM model using only randomly initialized word embeddings largely outperforms the random baseline. The pre-trained CBOW/SG word embeddings seem not very useful, leading to detection performance slightly lower than the model with random initial word embeddings. For both CBOW and SG, introducing the POS sequence improves the detection accuracy by about 2% and also improves all other measurements. The n-gram features further increase the accuracy by about 1%.

On the other hand, the CWIN and Struct-SG embeddings themselves are very powerful. Incorporating the POS and n-gram features leads to only slight improvements in terms of accuracy. Despite the small impact on accuracy, the n-gram features bring obvious improvements on hit@2 and hit@20% rates, indicating that they do facilitate the model in promoting the rank of the ground-truth position. Under the same set of features, all models with CWIN/Struct-SG significantly outperform their CBOW/SG counterparts ($p < 0.05$).

Bidirectional LSTM (Bi-LSTM) further enhances the performance of LSTM. Bi-LSTM with CWIN+POS features achieves the best accuracy and MRR, and significantly outperforms its LSTM counterpart ($p < 0.005$). Bi-LSTM with CWIN+POS+n-gram



features achieves the best Hit@2 and Hit@20%.

5.7.2 LSTM v.s. Bi-LSTM on Segments with Different Length

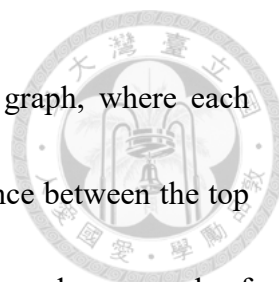
Length	(# tests)	# proposed	LSTM	Bi-LSTM
< 10	(645)	1	0.7426	0.7659
10 ~ 14	(317)	2	0.6908	0.7319
15+	(89)	3+	0.7416	0.7079

Table 5-3 Hit@20% rates of LSTM and Bi-LSTM on segments with different lengths.

To take a closer look at what is gained by moving from LSTM to bidirectional LSTM, we analyze the performance of the two types of models on different length of segments in Table 5-3. We use the versions with all set of features and report hit@20% rates. Using Bi-LSTM leads to some improvement on short (< 10 tokens) segments, and larger improvement on mid-length (10 ~ 14 tokens) ones. Even longer (15 tokens up) segments are relatively rare since foreign learners seldom construct complex sentences.

5.7.3 Relationship between Top Two Candidates

We previously justify the use of the hit@2 metric by pointing out that a WUE usually involves a pair of words dependent on each other. We can verify whether the top two candidates proposed by our model are closely related by examining the dependency distance. We take the output of the Bi-LSTM model with CWIN+POS+ngram features and analyze the error cases where the model ranks the ground-truth error position second.



We use the dependency parsing result to construct an undirected graph, where each dependency corresponds to an edge, and calculate the shortest distance between the top two candidates in these cases. Figure 5-3 shows the undirected dependency graph of segment (S5-2). The shortest path distance between token “知識” and “差” is 1.

Dependency Parsing Result

- relcl(知識, 學習)
- mark(學習, 的)
- nsubj(差, 知識)
- advmod(差, 也)
- advmod(差, 很)

Undirected Dependency Graph

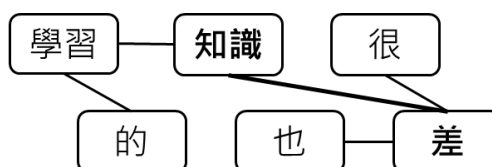


Figure 5-3 An example of undirected dependency graph.

The dependency analysis results are summarized in Table 5-4. a denotes the ground-truth error position. c_1 and c_2 denote the first and the second candidate positions proposed by the model. $dis(c_1, c_2)$ is the distance between c_1 and c_2 on the dependency graph. The average distance (2.07) is small compared to the average length of the segments (9.24), indicating that our model can consider the dependencies among words when ranking the candidate positions. Moreover, among the $c_2 = a$ cases, more than one third of them have $dis(c_1, c_2) = 1$. This means that the top two candidates are closely related as in the case of “知識” and “差” in (S5-2).

# correct ($c_1 = a$)	520 (49.48%)
# tests where $c_2 = a$	339 (32.25%)
Average $dis(c_1, c_2)$ when $c_2 = a$	2.07
# tests where $c_2 = a$ and $dis(c_1, c_2) = 1$	129 (12.27%)

Table 5-4 Summary of the analysis of the dependency between the top two candidates.

5.7.4 Effectiveness of POS Features

POS (# tests)	CWIN	CWIN+POS
VV (325)	0.8123	0.8185
NN (282)	0.6879	0.7447
AD (134)	0.6194	0.7015

Table 5-5 Hit@20% rates of Bi-LSTM models with or without POS features on three most frequent POS tags of the erroneous token.

A factor that might limit the effectiveness of POS features is that the POS tagger trained on well-formed text may not perform well on noisy learner data. In fact, for 26.7% of the test data, the POS tag of the original erroneous token differs from that of its corrected version. We compare the performance of Bi-LSTM with or without POS features on three most frequent POS tags in Table 5-5. As can be seen, the POS information of the erroneous segment, which potentially contains errors, can still be helpful for detecting anomaly of the segment. In example (S5-6) we show the scores of incorrectness predicted by models with or without POS features. The "DEC + AD" construction is invalid in Chinese, so in this case the error can be detected more easily if



POS information is available.

(S5-6)	應該	有	別人	的	*盡力
POS tag	VV	VE	NN	DEC	AD
w/o POS	0.048	0.226	0.030	0.016	0.042
w/ POS	0.010	0.066	0.031	0.071	0.077

(There should be someone else's *utmost.)

5.7.5 Effectiveness of N-gram Features

Though being very powerful in the segment-level detection task, the Google n-gram features only lead to small improvement in accuracy and MRR according to Table 5-2. In the case of Bi-LSTM with CWIN+POS features, including n-gram features even results in slightly lower accuracy and MRR.

We find out that a segmentation error somewhere in the segment can affect the model's decision. For example, in (S5-7) the token “就在這時候” is an incorrect merge of several words. The Bi-LSTM model with CWIN+POS selects the token “起來”, which is the ground-truth error position, but the model with CWIN+POS+n-gram selects the OOV token “就在這時候”. Note that “起來” is still given second highest incorrectness score, which may explain the improvement on the hit rates after incorporating n-gram features.

(S5-7)	他	就在這時候	想	*起來	一	個	辦法
OOV	0	1	0	0	0	0	0
CWIN+POS	0.0228	0.0860	0.0466	0.5440	0.0105	0.0712	0.0885
+ n-gram	0.0488	0.2677	0.1427	0.2540	0.0150	0.1158	0.0319

(At this moment, he thought *of a way.)



Even when there is no segmentation error, making decision based on n-gram probabilities is still rather complex. We use (S5-8) as an example.

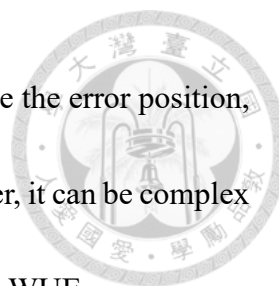
(S5-8)	哪個	城市	都	有	特色	的	氣氛
2gram	-1	0.0139	0.0037	0.0544	0.0013	0.1020	0.0002
3gram	-1	-1	0.0317	0.2462	0.0002	0.3981	0.000039

(Every city has *specialty atmosphere.)

Regarding bigram probabilities, the erroneous token “特色” is not the one given lowest probability. Instead, the bigram “的 氣氛” has the lowest probability, since many words can be the succeeding word of “的”. Thus, the probability of a certain possibility, “氣氛” in this case, would be low. A low probability does not indicate a wrong usage when involving such a “general” word “的”, so making decision simply based on probability threshold would not work. It is expected that the WUE in (S5-8) can be recognized if we consider trigrams, since “特色的 氣氛” is a local wrong usage. However, the trigram probability does not drop until the last token “氣氛”, which makes it difficult to trace the problem to the inappropriate usage of “特色”.

Although the model has access to word and POS information, which can be used for determining different thresholds for different situations, it seems that the model is better at matching word and POS patterns where the Chinese learners are more likely to make mistakes.

In sum, if a segment contains some WUEs, it is more likely to have low overall n-gram probability sums. Thus, the Google n-gram features can be useful for the segment-



level detection task. Nevertheless, when the model needs to determine the error position, the noise from segmentation error may lead to wrong results. Moreover, it can be complex to interpret the n-gram probabilities and trace which token causes the WUE.

5.7.6 Performance on Commonly Misused Words

Word	Error rate	Precision	Recall
產生 (generate)	0.571 (8/14)	0.700 (7/10)	0.875 (7/8)
經驗 (experience)	0.500 (5/10)	0.667 (4/6)	0.800 (4/5)
發生 (happen)	0.455 (5/11)	0.571 (4/7)	0.800 (4/5)
而 (so)	0.417 (20/48)	0.550 (11/20)	0.550 (11/20)

Table 5-6 Precision/recall on four most commonly misused words.

In Table 5-6 we show the precision/recall of the Bi-LSTM model with CWIN+POS features on four most commonly misused (error rate above 0.4) words. The error rate of a word w is calculated on the test set by

$$\text{err_rate}(w) = \frac{\# \text{ segments in which } w \text{ is misused}}{\# \text{ segments containing } w}$$

We exclude words that occur in less than 10 segments regardless of their error rates. In general, our model achieves high recall and fair precision. Discriminating correct and wrong usage of the conjunction 而(so), which often connects more than one segment, seems to be the most difficult. For example, in (S5-9) the inappropriateness of 而 cannot be recognized unless we consider the wider context of this segment.

(S5-9) (*而,並) 當成此生做人的道理

(..., (*so,and) take it as a lifelong way to behave around others.)



5.8 Conclusion for Token-level Detection

In this stage, we propose an LSTM-based sequence labeling model for detecting WUEs in sentences written by non-native Chinese learners. The experimental results suggest that the CWIN/Struct-SG embeddings, which consider word orders, are better word features for Chinese WUE detection. The POS information can help detect incorrect grammatical constructions. For the model, Bi-LSTM is more preferred than LSTM since the context after the erroneous token can be taken into account.

While a wrong usage often involves more than one token, making it difficult to determine which one should be corrected, the best Bi-LSTM model can rank the ground-truth error position within the top two in 80.97% of the cases. In addition, analysis on dependency distances shows that the top two candidates are usually closely related to each other.

Chapter 6 Chinese WUE Correction



6.1 Task Description

In this stage, given a token in a segment that is known to be erroneous, we aim to generate a suitable correction for it. The criteria for a suitable correction are:

- **Correctness:** After substituting the erroneous token with the correction token, the result must be a syntactically and semantically correct Chinese sentence segment.
- **Similarity:** The meaning of the correction must be as close to the writer's intended meaning as possible.

We discuss the criteria with the following examples.

(S6-1) *生活方式已經**猛烈地**改變了

(S6-2) *生活方式已經**強烈地**改變了

(S6-3) 生活方式已經**緩慢地**改變了

(S6-4) 生活方式已經**劇烈地**改變了

For the wrong segment (S6-1), (S6-2) is not a correction since it is incorrect itself. The adverb “強烈地” does not collocate with the verb “改變”. (S6-3) is grammatical, but its meaning differs from the original meaning of (S6-1), so neither is a suitable correction. (S6-4) is a good correction that meets both criteria.

Nevertheless, there are some cases in which the similarity criterion is hard to meet.

For example, the intended meaning of (S6-5) is very difficult to recognize. (S6-6) is the ground-truth correction, but the association between the original erroneous token “情緒” and the correction “因素” is unclear without wider context information.

(S6-5) 發生這種情況的**情緒**很多

(S6-6) 發生這種情況的**因素**很多

When either criterion cannot be met, the correctness criteria should have higher priority, since an incorrect sentence can confuse the language learner and have bad impact on learning.

In this chapter, “target” refers to the original erroneous token written by the language learner, and “context” refers to other words in the sentence segment. A pair consisting of the target and its corresponding correction is called a “correction pair”. In the above examples, “猛烈” and “情緒” are targets, and (猛烈, 劇烈) and (情緒, 因素) are correction pairs.

6.2 Dataset

We follow the train/validation/test split of the previous stage. For each split, we filter the instances where the correction is not within the top 50,000 frequent words in the Chinese ClueWeb dataset we used to train the word embeddings. This decision is made based on the fact that the vocabulary used by the non-native language learners is limited.



When human annotators correct the sentence, they are also unlikely to replace the erroneous token with a rare, hard word, since it will be too difficult for the non-native learner to understand and acquire the usage.

Out of the 10,510 segments used in the token-level detection task, 954 (less than 10%) are filtered out since the correction token is out of the vocabulary of top 50,000 frequent words. In fact, most of the filtered segments involve segmentation error, or inconsistency of segmentation. Below is an example of wrong segmentation.

(Wrong segmentation result) 我的大姐七歲時因患上破傷風症死去

(Expected segmentation) 我的大姐七歲時因患上破傷風症死去

For the inconsistency problem, one example is the phrase “越來越大”, which is usually segmented into two tokens “越來越 大”, but is occasionally segmented into one single word.

Our final dataset of this stage contains erroneous segments and their corresponding corrections. The statistics of the datasets are shown in Table 6-1.

	# segments
Train	8,205
Validation	1,026
Test	1,025

Table 6-1 Statistics of the dataset used in the WUE correction stage.



6.3 Neural Network-based Correction Generation Model

6.3.1 Model Overview

Our correction generation model is a deep neural network that takes features as input and generates the embedding, or word vector, of the correction token. Features are derived from the target and the context.

Let $\mathbf{f}_{\text{target}}$ and $\mathbf{f}_{\text{context}}$ be the feature vector for target and context respectively. Given the original erroneous sentence segment, which includes both the target and the context, our model selects the most suitable correction word c^* over a set C of all possible corrections c :

$$c^* = \underset{c \in C}{\operatorname{argmax}} \cos(\operatorname{DNN}(\mathbf{f}_{\text{target}}, \mathbf{f}_{\text{context}}), \operatorname{vec}(c))$$

, where $\operatorname{vec}(c)$ denotes the embedding of the candidate word c . $\operatorname{DNN}(\mathbf{f}_{\text{target}}, \mathbf{f}_{\text{context}})$ is referred to as the correction vector. In fact, this model can propose several candidates, ranked by their cosine similarities to the correction vector. The goal is to rank the ground-truth correction toward the top of the list. Figure 6-1 illustrates the whole model.

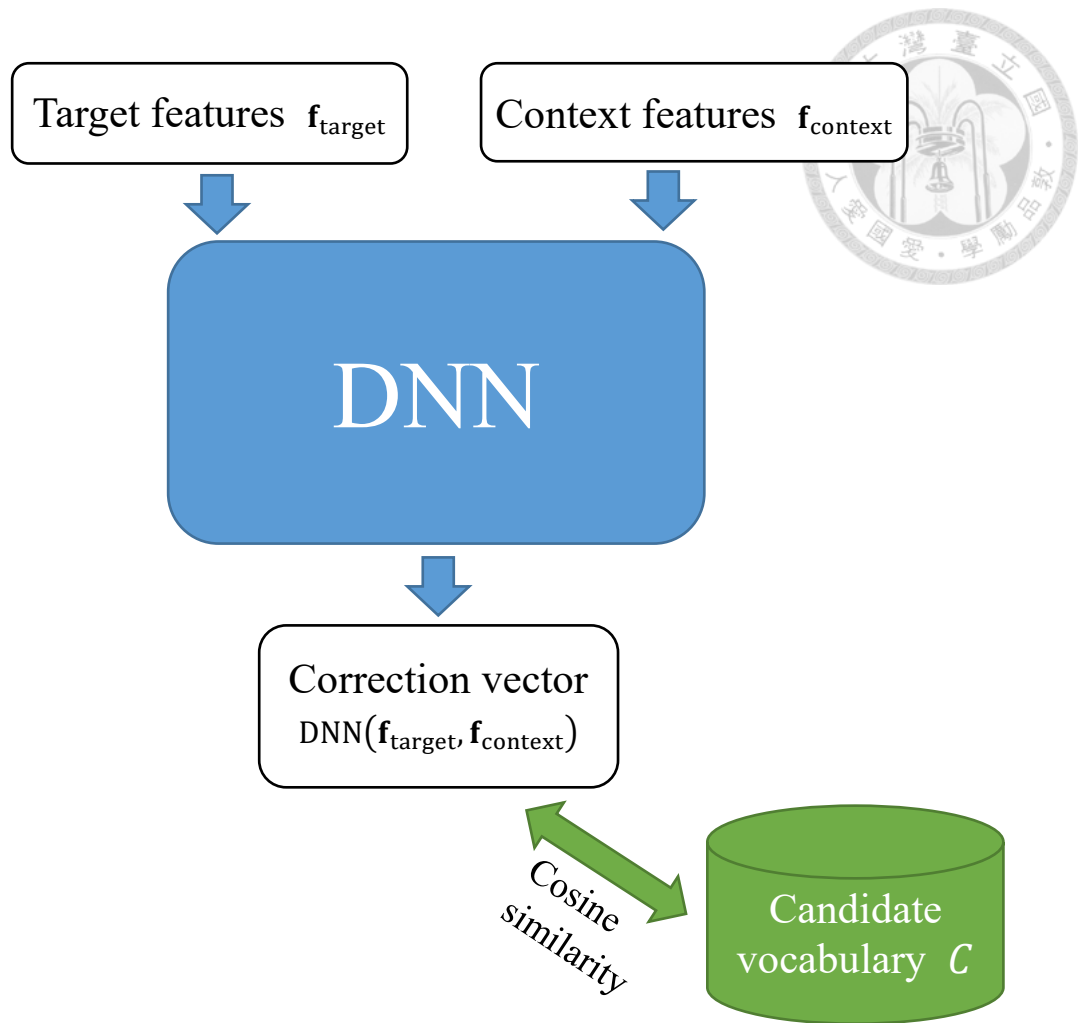


Figure 6-1 A high-level view of our correction generation model.

6.3.2 Model Output

For the embedding of the correction token, we use the CWE word vectors trained on the Chinese ClueWeb dataset. We adopt the position-based variant (CWE+P), which keeps three embeddings for each character according to the character's position in the word. The three embeddings are labeled **s** (start), **m** (middle) and **e** (end). This variant is designed to capture the different morphological functions of a Chinese character when it is at different position in a word. According to Chen et. al. (2015), the representation of



the word “農產品”, for example, should be

$$\text{vec}(\text{農產品}) + \frac{1}{3} [\text{vec}(\text{農 s}) + \text{vec}(\text{產 m}) + \text{vec}(\text{品 e})]$$

, where $\text{vec}(\cdot)$ is the vector of a word or a character.

If we ignore the internal structure of the word, a misused form “農作品” is simply an OOV word and has no association with the correction “農產品”. With the use of CWE vectors, it is possible for the model to learn a transformation from the incorrect token to its correction.

We use the publicly released implementation of CWE¹¹ to train the CWE+P model on the Chinese ClueWeb corpus. We set the embedding size to 400 and train for 20 iterations.

All other hyperparameters are left default.

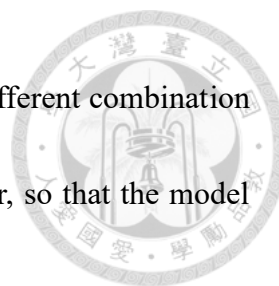
6.3.3 Candidate Vocabulary

According to the filtering criterion described in Section 6.2, any correction must be one of the top 50,000 frequent words. Nevertheless, we found that within these “frequent” words, some such as punctuation marks or English words are not possible correction. We eliminate these invalid candidates and the result candidate vocabulary size, $|C|$, is 48,394.

6.3.4 Model Parameters

We implement the model with Keras. Table 6-2 shows the parameters. Models with

¹¹ <https://github.com/Leonard-Xu/CWE>



this setting generally perform the best on the validation set across different combination of features. The activation function is not applied at the output layer, so that the model output can fit better to the vector of the correction.

When the validation accuracy does not increase for two consecutive epochs, the training process is terminated. We choose the model with the highest validation accuracy for each feature combination and report the results on the test set.

Hidden layer size	4096
# hidden layers	2
Activation function	ReLU
Cost function	cosine_proximity
Optimizer	Adagrad
Batch size	32
Initial learning rate	0.01

Table 6-2 Correction generation model parameters.

6.4 Features

To meet the two criteria for suitable WUE correction, we adopt several target and context features as input to the correction generation model. The division of target and context features is only for indicating the source of information. The model does not distinguish between these two kinds of features.

6.4.1 Target CWE+P Word Embedding

We use the same CWE+P model as with we use for the model output. Let $w =$



$c_1 c_2 \dots c_k$ be the erroneous token, where c_1, c_2, \dots, c_k are its character components. We

compute the target CWE+P word embedding feature vector, CWE_w , by:

$$CWE_w(w) = \begin{cases} \text{vec}(w) + \frac{1}{k} \left[\text{vec}(c_1, s) + \sum_{i=2}^{k-1} \text{vec}(c_i, m) + \text{vec}(c_k, e) \right] & \text{if } w \in V \\ \frac{1}{k} \left[\text{vec}(c_1, s) + \sum_{i=2}^{k-1} \text{vec}(c_i, m) + \text{vec}(c_k, e) \right] & \text{otherwise} \end{cases}$$

, where V is the word vocabulary of the CWE+P model. For example, the target feature

vector for the erroneous token “農作品” is:

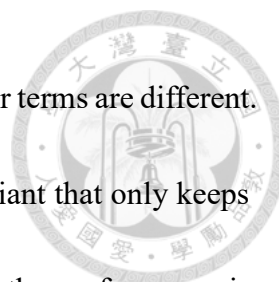
$$\frac{1}{3} [\text{vec}(\text{農}, s) + \text{vec}(\text{作}, m) + \text{vec}(\text{品}, e)]$$

As can be seen, this vector shares terms $\text{vec}(\text{農}, s)$ and $\text{vec}(\text{品}, e)$ with the vector of its correction “農產品”。

The word and character vectors used to calculate this set of features are in the same space with the model output, enabling the model to directly learn a transformation between a correction pair.

6.4.2 Target CWE Position-Insensitive Character Embedding

Although a character’s position in a word could reflect its morphological function, non-native Chinese learners might not be so familiar with Chinese morphology. One common type of morphological error (W-error), as we defined in Chapter 1, is incorrect ordering of characters within a word. For example, (*決解, 解決) is the tenth frequent correction pair in our dataset. With the use of CWE+P embeddings, the similarity between



these two tokens might be underestimated since all the character vector terms are different.

To cope with this problem, we experimented with the CWE variant that only keeps one vector for each Chinese character regardless of its position, but the performance is not as good as the model with CWE+P features. Alternatively, we design a separate set of character embedding features CWE_c that include character embedding of all positions.

That is,

$$CWE_c(w) = \frac{1}{k} \sum_{i=1}^k [\text{vec}(c_i, s) + \text{vec}(c_i, m) + \text{vec}(c_i, e)]$$

By doing so, $CWE_c(\text{決解})$ will contain $\text{vec}(\text{解}, s)$ and $\text{vec}(\text{決}, e)$, which are the terms of $CWE_w(\text{解決})$.

6.4.3 Context2vec Features

Context2vec (Melamud et al. 2016) is a bidirectional LSTM-based model that can encode a “context” into a real-valued vector. A context is a sequence of words with a certain position blanked out. For instance, below is a context:

可是每個人的 [] 都千差萬別

In general, a context can be represented by $w_1 \dots w_{p-1} [] w_{p+1} \dots w_L$, where each w_i is a token, L is the number of tokens, and p is the index of the blank. The vector encoding of the context is a combination of the sequence of words before and after the blank:



$$C2V_{ctx}(w_1 \dots w_{p-1} [\quad] w_{p+1} \dots w_L) = LSTM(w_1 \dots w_{p-1}) \oplus LSTM(w_L \dots w_{p+1})$$

, where \oplus is the vector concatenation operation.

Context2vec also keeps the embeddings of individual words, which are called target embeddings¹² by Melamud et al. (2016). We use $C2V_{trg}(w)$ to denote the vector of target word w . Both target embeddings and the parameters in the LSTM layers are updated during training. The objective of the model is to predict the target word that actually occurs in the training sentence, given the encoded context vector. Figure 6-2 illustrates the architecture of Context2vec.

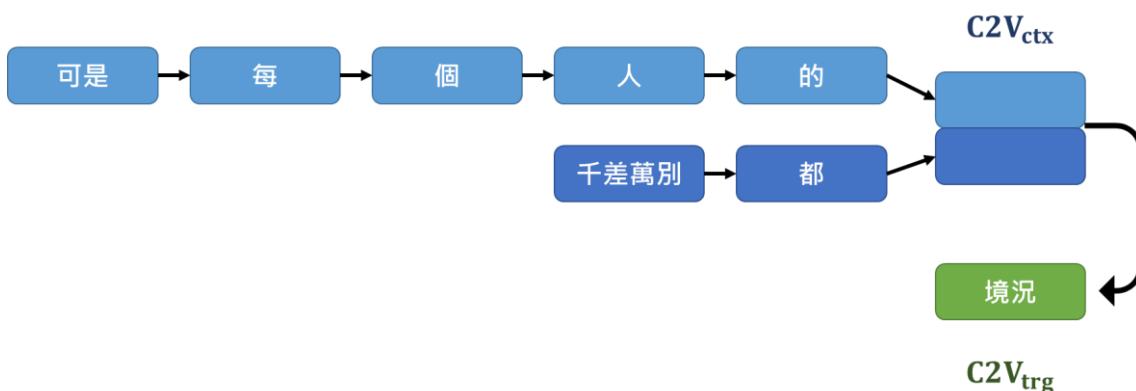


Figure 6-2 The architecture of Context2vec.

We train Context2vec on the Chinese ClueWeb corpus. We set the embedding size to

¹² Note that the definition of “target” in the Context2vec paper is slightly different from ours. In our definition, target only refers to the original erroneous token, while for Context2vec, target can refer to any word to be put into the blank, regardless of whether the result is a correct sentence.



300 and train for 5 epochs.

The formulation of context makes Context2vec suitable for the sentence completion task. A candidate c to fill the blank can be selected according to how similar $C2V_{\text{trg}}(c)$ is to the context vector in terms of cosine similarity. That is, given a context where the p -th position is the blank, the best candidate c^* would be:

$$c^* = \underset{c \in C}{\operatorname{argmax}} \cos(C2V_{\text{ctx}}(w_1 \dots w_{p-1} [] w_{p+1} \dots w_L), C2V_{\text{trg}}(c))$$

In their paper, Melamud et al. (2016) have shown promising results of context2vec in several sentence completion benchmarks.

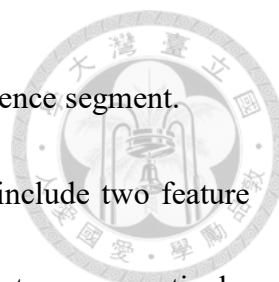
For the example context we mentioned above, the best candidate selected in this way using our trained model is “境況”, which can be put into the blank and the result is a correct sentence segment.

可是每個人的 [境況] 都千差萬別

In fact, this context is extracted from a wrong segment in our dataset. The original erroneous segment and the corresponding correction are shown below.

可是每個人的 (*對應, 反應) 都千差萬別

Given the original segment, one can conclude that the candidate “境況” selected by Context2vec is less suitable compared to the ground-truth “反應”, according to the similarity criteria. Therefore, WUE correction is different from sentence completion in that if the model ignores the word originally written by the language learner, it is likely



to generate a correction that changes the meaning of the original sentence segment.

To take both context and target information into account, we include two feature vectors $C2V_{tgr}$ and $C2V_{ctx}$, which belong to target and context features respectively.

$C2V_{tgr}$ refers to $C2V_{tgr}(w)$, which is the embedding of the original erroneous token w .

w can reveal important information about the writer's intended meaning as we discussed

above. We use the above-mentioned candidate selection method based solely on the

cosine similarity of context and candidate vectors, without any training on the Chinese

WUE dataset, as one of our baselines.

6.4.4 Target POS Feature

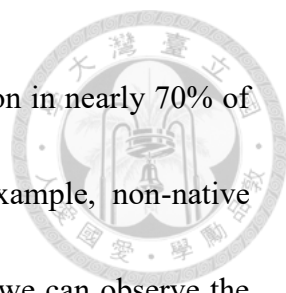
If we regard the process of replacing the erroneous token with a correction token as

a kind of transition, we can analyze the transition of POS tags. The most frequent POS

transition in the validation set are shown in Table 6-3.

Original POS	Correction POS	# instances (%)
(unchanged)		722 (68.70%)
VV	NN	27 (2.57%)
NN	VV	21 (2.00%)
P	VV	17 (1.62%)
DEC	DEV	15 (1.43%)
VV	P	13 (1.24%)
AD	VV	10 (0.95%)
VV	VA	10 (0.95%)

Table 6-3 POS transitions that occurs at least 10 times in the validation set.



As can be seen, the POS tag does not change after the correction in nearly 70% of the cases. Besides, there are some systematic transitions. For example, non-native Chinese learners seem to confuse some nouns with some verbs, so we can observe the interchanging phenomenon of the VV and NN tag. It is similar for the case of VV and P. DEC and DEV are special tags for Chinese particles “的” and “地” respectively (Xia 2000). The former is a nominalizer such as in “這件事的發生”, while the latter is an adverb marker, such as in “清楚地記得”. In fact, (的, 地) is the most frequent correction pair in our dataset.

The systematic transitions of POS tags indicate that it is possible to reduce the candidate vocabulary. We tried to limit the candidate to the POS transitions observed in the training and validation set. However, this results in slightly lower accuracy and MRR, because the correction of some test instances is out of the candidate set and this impact is larger than the gain from eliminating less likely candidates.

Therefore, instead of modifying the candidate set directly, we encode the POS of the erroneous token in a one-hot vector and feed this feature to the correction generation model. This allows the model to learn different transformation function for different source POS, that is, the POS of the erroneous token.



6.5 Language Model Re-ranking

One drawback of our DNN correction generation model is that the correctness criterion does not explicitly take priority over the similarity criterion. In our experiments, we found that our model sometimes generates segments that seriously violate the correctness criterion, since the model can bias toward the similarity criterion. Below is an example.

Wrong segment: 到山頂之間路走得不容易
Model prediction: 到山頂期間路走得不容易
Ground-truth correction: 到山頂的路走得不容易

The candidate “期間” is selected since it is similar to the target “之間”, but the result sentence segment is incorrect. It is necessary to deal with this problem since the correctness criterion is more important, as we previously discussed in the beginning of this chapter.

It is expected that this kind of unsuitable candidates can be eliminated by a language model (LM), if we assume that LM probability reflects the level of correctness of a sentence segment. Therefore, we train two kinds of LM on the Chinese ClueWeb corpus, and propose a method to re-rank the correction candidates.

6.5.1 Traditional N-gram Language Model (N-gram LM)

The traditional N-gram LM estimates n-gram probabilities with MLE based on the



observed counts in the training corpus. We train a 5-gram language model with KenLM¹³.

The modified Kneser-Ney smoothing (Heafield et al., 2013) is applied to handle unseen n-grams.

6.5.2 Recurrent Neural Network Language Model (RNNLM)

RNNLM is first developed by Mikolov et al. (2011). It is a prediction-based LM that can evaluate the probability of a sequence of n words conditioned on previous $n - 1$ words. When processing the word sequence, the recurrent layer keeps holding the information of previous time steps. Therefore, different from traditional n-gram model, there is no limitation on n . As a result, RNNLM is capable of capturing longer dependency.

We use the Faster RNNLM toolkit¹⁴, which speeds up the training process by using the Hierarchical Softmax (HS) or Noise Contrastive Estimation (NCE). We choose NCE because it gives better performance on our WUE validation set.

We process the ClueWeb corpus before training. The words whose frequency is less than 10 are replaced with a “<unk>” token. When testing, an OOV is treated as “<unk>”.

We split 10% of the corpus for validation. The toolkit automatically adjusts the learning

¹³ <https://kheafield.com/code/kenlm/>

¹⁴ <https://github.com/yandex/faster-rnnlm>



rate and early-stops the training process based on validation entropy. The hyperparameter settings are shown in Table 6-4.

Layer type	GRU
Layer size	128
# negative samples	20

Table 6-4 Hyperparameters of RNNLM.

6.5.3 Re-ranking Method

We aim to emphasize the correctness criterion by incorporate the LM scores into the candidate selection process. One possible approach is to apply a probability cut-off and discard candidates that result in segments with low probability. Nevertheless, the “acceptable” LM probability varies from sentence to sentence since it can be affected by, for instance, length and lexical complexity of the sentence. Though we can let the cutoff be a function taking the above factors into consideration, it is difficult to design such a function explicitly.

Therefore, instead of performing combination of scores, we combine the rank proposed by the LM with that based on our correction generation model. One advantage of this approach is that the range of rank is the same for all instances, so the ranks can be evaluated with the same standard across different instances.

For a candidate correction, let r_{LM} be its rank based on the LM probability, and



r_{DNN} be the rank based on its cosine similarity to the correction vector generated by our DNN model. We adopt the following weighted harmonic mean to obtain a new “rank” for the candidate.

$$r_{\text{com}} = \frac{1}{\frac{\alpha}{r_{\text{LM}}} + \frac{1 - \alpha}{r_{\text{DNN}}}}$$

, where α is a parameter that can be tuned with the validation set. Preliminary experiments show that harmonic mean performs better than arithmetic mean and geometric mean. Though r_{com} may not be an integer, it can be interpreted as rank. That is, the correction with smaller r_{com} is considered better.

6.6 Automatic Evaluation

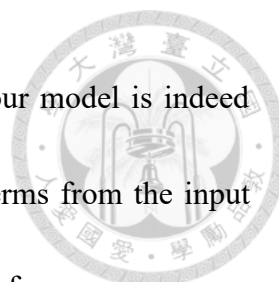
6.6.1 Overall Results

We first evaluate our correction generation model with the ground-truth correction provided in the HSK dataset. The evaluation is based on the rank of the ground-truth correction. We report accuracy, MRR and hit rates. The results are shown in Table 6-5. The baselines include the two LMs and the Context2vec sentence completion method, which select a candidate most similar to the context but ignore the original erroneous token. N-gram LM is the strongest baseline for the WUE correction task.

Target features	Context features	Acc.	MRR	Hit@5	Hit@10	Hit@50
Baselines (No training on the WUE dataset)						
-	N-gram LM	0.1659	0.2438	0.3268	0.4029	0.5951
-	RNNLM	0.1468	0.2208	0.2847	0.3611	0.5793
-	C2V _{ctx}	0.0714	0.1170	0.1575	0.2114	0.3611
Correction Generation Model – Context2vec Features						
C2V _{trg}	-	0.2507	0.3030	0.3561	0.3932	0.5024
-	C2V _{ctx}	0.1249	0.1746	0.2273	0.2741	0.4010
C2V _{trg}	C2V _{ctx}	0.3249	0.3891	0.4566	0.4976	0.6185
Correction Generation Model – CWE + Other Features						
CWE _w		0.2898	0.3545	0.4195	0.4693	0.5971
+ CWE _c		0.2946	0.3570	0.4234	0.4722	0.6078
+ C2V _{trg}	+ C2V _{ctx}	0.3512	0.4250	0.5024	0.5571	0.6800
+ POS		0.3717	0.4378	0.5063	0.5688	0.6956

Table 6-5 Performance of the correction generation model with various target and context features.

The second part of Table 6-5 shows the result of a set of experiments with Context2vec features. The DNN model with only C2V_{ctx} features differs from the Context2vec baseline in that it is trained with the WUE dataset. Learning a transformation from the erroneous token to the correction seems to be easier than guessing a correction only from context, probably because some common correction pairs can be learned. The model using only C2V_{trg} achieves performance substantially higher than that using only C2V_{ctx}. Note that we use CWE+P embeddings for the model output, which does not lay



in the same space as the Context2vec vectors. This indicates that our model is indeed capable of learning a transformation, not just copying the vector terms from the input features. Combining $C2V_{trg}$ and $C2V_{ctx}$ can further enhance the performance.

The third part of Table 6-5 shows another set of experiments starting from the CWE+P target features. The model with CWE_w features performs better than that with $C2V_{trg}$, since CWE+P composes a vector representation for OOV targets such as “農作品”, giving the model more clues for generating the correction. The position-insensitive character feature CWE_c slightly improves the performance over CWE_w . After including Context2vec context and target feature, the model can consider the context and reaches accuracy 0.3512. Finally, incorporating POS information further improves the performance. The best result of our DNN correction generation model is accuracy 0.3717 and MRR 0.4378.

6.6.2 Effect of LM Re-ranking

Model	Acc.	MRR	Hit@5	Hit@10	Hit@50	Hit@100
Best DNN	0.3717	0.4378	0.5063	0.5688	0.6956	0.7415
+ N-gram LM ($\alpha = 0.355$)	0.3727	0.4605	0.5561	0.6439	0.8039	0.8488
+ RNNLM ($\alpha = 0.255$)	0.3727	0.4527	0.5278	0.6205	0.7808	0.8302

Table 6-6 Correction performance with LM re-ranking.



We apply LM re-ranking to the candidate ranks generated by the best correction generation model. The results of two kinds of LM are shown in Table 6-6. We include the hit@100 rate to better illustrate the change of the ranks. Although there is only slight improvement on the accuracy, the MRR and hit rates increase substantially after LM re-ranking is applied. N-gram LM gives slightly better MRR than RNNLM. The optimal α for N-gram LM is also larger than that of RNNLM, showing that N-gram LM is more suitable for this task.

The following is an example in which LM re-ranking helps promote the rank of the answer. Though “一起” and “一直” share a common Chinese character, the meaning of the two words are not quite similar. Thus, the DNN rank is very low. In contrast, the LM rank is high, since “就...都不...” is a suitable context for “一直”. The high LM rank makes the final combined rank higher, so the MRR can be enhanced.

我從上小學起成績就(*一起,一直)都不理想

LM rank: 7 / DNN rank: 1284

Combined rank: 19

We plot validation MRR with respect to different α values in Figure 6-3. As can be seen, LM re-ranking works like regularization and the two LMs have similar trend. The MRR gradually increases with α for small α values, but when α increases above 0.5, the performance drops sharply. This also supports the point that the information about the target should have certain impact in the process of decision. If LM score is emphasized

too much, the result correction could change the original meaning of the sentence segment.

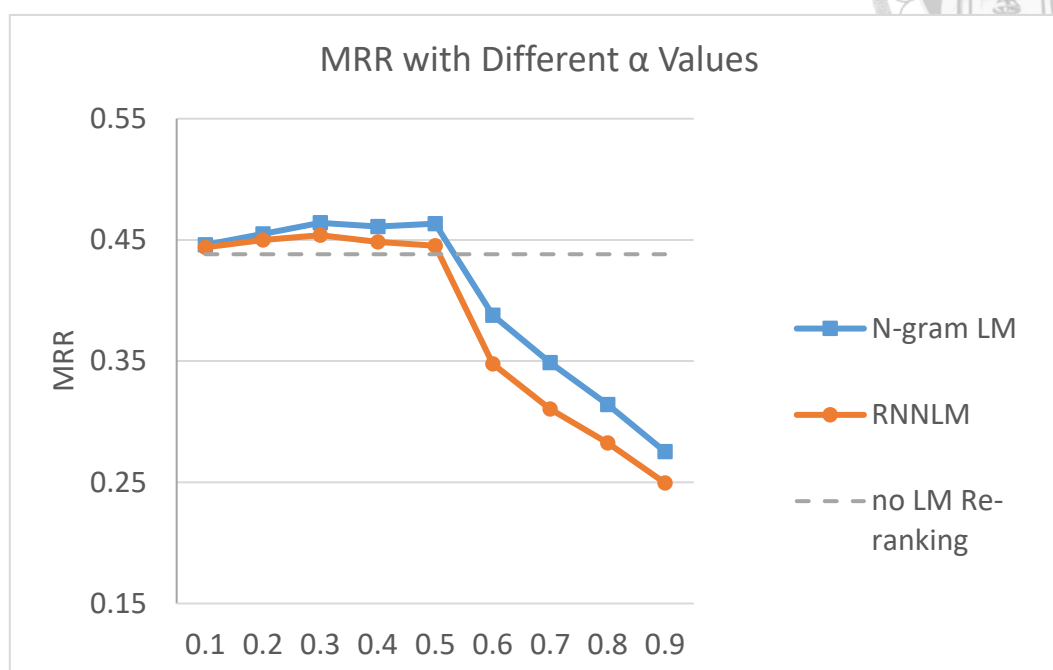


Figure 6-3 The effect of parameter α in LM re-ranking.

6.7 Human Evaluation

6.7.1 Motivation for Human Evaluation

In the automatic evaluation, there is only one answer for each test instance. However, correction can be subjective and alternatives may exist. Moreover, since we divide the essays into sentences and further into segments, the model has no access to the context outside of the segment. This results in difficulties in making the choice among several candidate corrections that are with different meanings but all seem to be acceptable in given the segment. Table 6-7 shows an example.

Wrong segment	不過 我們 要以 堅定的 定心 與 病 對抗
System rank 1 correction	不過 我們 要以 堅定的 自信 與 病 對抗
System rank 2 correction	不過 我們 要以 堅定的 信念 與 病 對抗
System rank 3 correction	不過 我們 要以 堅定的 理智 與 病 對抗
System rank 4 correction	不過 我們 要以 堅定的 自信心 與 病 對抗
System rank 5 correction	不過 我們 要以 堅定的 毅力 與 病 對抗
Ground-truth correction	不過 我們 要以 堅定的 決心 與 病 對抗

Table 6-7 An example of alternative corrections.

In Table 6-7, the top five candidates proposed by our system are all differ from the ground-truth correction. However, except for the rank 3 candidate, all other four candidates result in acceptable corrections. This example shows that the single-answer automatic evaluation can underestimate the performance of our system. Therefore, we perform human evaluation, in which the top candidates proposed by our model are judged by annotators.

6.7.2 Annotation Guideline

Each instance of annotation consists of two sentence segments:

(S0) The original wrong segment written by non-native Chinese learners

(S1) A correction segment, which is either the ground-truth or one of the top k candidates proposed by our system.

We set $k = 5$; however, only the candidates ranked before the ground-truth need annotation. For example, if our system gives rank 3 to the ground-truth correction, only



the ground-truth, the first candidate and the second candidate need to be judged by human.

For the 1,025 test segments, a total of 3,692 annotation instances are generated.

An annotator is asked to answer (at most) two questions for each annotation instance.

Q1 (*is_g*): Is (S1) syntactically and semantically correct?

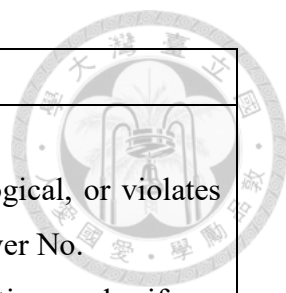
Q2 (*is_c*): Is (S1) a correction of (S0)?

The answers to both questions are either Yes (1) or No (0). If the answer to Q1 is No, the answer to Q2 must be No, since (S1) violates the correctness criterion; we will skip Q2 in such cases.

There are 10 annotators. All of them are native speakers of Chinese in Taiwan. We use OpenCC¹⁵ to convert the original simplified Chinese HSK data into traditional Chinese. Given that the conversion is not perfect, we also provide a button which can be clicked and then the simplified Chinese version will be shown.

The instructions in Table 6-8 are given to the annotators.

¹⁵ <https://github.com/BYVoid/OpenCC>



Question	Instructions
Q1	<ol style="list-style-type: none"> 1. If (S1) is ungrammatical, please answer No. 2. If (S1) is grammatical but its semantics is not logical, or violates some common-sense knowledge, please also answer No. 3. Since we split sentences into segments by punctuation marks, if you encounter an “incomplete” sentence, please answer Yes to it, if it is itself correct and can be completed in some reasonable way.
Q2	<ol style="list-style-type: none"> 1. This question will be presented only if you answered Yes to Q1. 2. If the meaning of (S1) is not the intended meaning of (S0), please answer No. 3. If you cannot understand the meaning of (S0), please answer Yes. 4. If you think that both (S0) and (S1) are correct, and their meanings are similar, please answer Yes.

Table 6-8 WUE correction annotation instructions.

The third instruction of Q2 corresponds to our previous claim that the correctness criterion is more important, especially when the similarity criterion is difficult to meet. The annotation guideline is presented to the annotators in traditional Chinese. The content is shown in Figure 4-1. The example segments are selected from the validation set to prevent guiding the annotator’s response to specific test instances.



1. (S1)是否正確

- 不合文法的句子請標「錯誤」
 - E.g. 已經到了晚輩的看法錯了
- 雖然合文法但語意違反常識或邏輯的請標「錯誤」
 - E.g. 自己的生活自己吃
- 因為句子是用標點符號切開的，所以如果句子看起來沒寫完，但本身正確且可以合理接下去的話，請標成「正確」
 - E.g. 雖然我跟她說話

2. (S1)是不是(S0)的更正

- (S1)選擇「正確」才會出現這個問題
- (S1)雖然正確，但如果不是(S0)想表達的意思，請選擇「否」
- 如果你沒辦法理解(S0)想表達的意思，請選擇「是」
- 如果你覺得(S0)和(S1)都是正確的句子，而且意思差不多，請選擇「是」

Figure 6-4 The annotation guideline presented to the annotators.

Each annotation instance is randomly assigned to two annotators. If the two annotators disagree on either question, a third annotator is introduced to break the tie. We use majority voting to determine the final answer.



6.7.3 Annotation Agreement

We evaluate the inter-annotator agreement of the first two annotators with Cohen's

Kappa:

$$\kappa = \frac{p_o - p_e}{1 - p_e}$$

, where p_o is the observed rate of agreement and p_e is the expected rate of agreement when the annotators label the data randomly. Since the labels for both questions are binary, we have

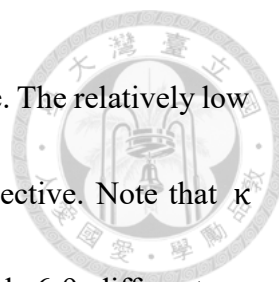
$$p_e = \frac{1}{N^2}(n_{Y1}n_{Y2} + n_{N1}n_{N2})$$

, where n_{Yi} and n_{Ni} are the number of times annotator i ($i = 1,2$) answers Yes and No respectively.

Since the annotators are assigned randomly, we can assume that the difficulty of the data assigned to each annotator is the same. Therefore, we report average κ of all pairs of annotators in Table 6-9.

	Average κ	
	<i>is_g</i>	<i>is_c</i>
Ground-truth	0.2778	0.3086
System candidates ranked before ground-truth	0.4025	0.3567
All annotation instances	0.4205	0.4070

Table 6-9 Average κ of the annotation.



As can be seen, the levels of agreement are from fair to moderate. The relatively low agreement on the ground-truth indicates that correction can be subjective. Note that κ evaluates relative level of agreement. In each of the three rows of Table 6-9, different p_e is used for calculating κ . For the ground-truth part, since it is expected that the ground-truth corrections are suitable in most of the cases, all annotators have high probability to answer Yes, that is, both n_{Y1} and n_{Y2} are large. Therefore, the value of p_e is larger and the agreement is evaluated more strictly. In contrast, when ground-truth and system candidates are mixed, p_e would not be that large.

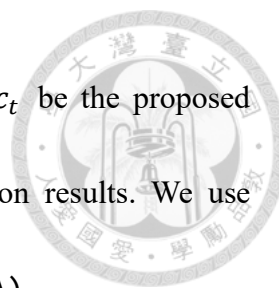
6.7.4 Evaluation with Human Annotation

We first examine the correctness criterion and show the result in Table 6-10. Given that more than 95% of the ground-truth corrections are judged as correct by the annotators, the quality of the ground-truth is not an issue. About 82% of the system top candidates are correct, which is lower than that of the ground-truth but still acceptable.

	% Correct (<i>is_g</i>)
Ground-truth	95.60%
System rank 1 candidate	82.73%

Table 6-10 Proportion of candidates that meet the correctness criterion.

We can then use the annotation result to update the rank of each test instance, and use the new ranks to re-calculate the evaluation metrics. For each test instance, let r be



the original rank calculated with the single-answer ground-truth, c_t be the proposed candidate with rank t , and A be a table containing the annotation results. We use Algorithm 6-1 to obtain the updated rank $\bar{r} = \text{update_rank}(r, A)$.

Algorithm 6-1: update ranks according to annotation results
<pre>update_rank(r, A) for $t = 1$ to k if $A.is_g(c_t)$ and $A.is_c(c_t)$ return t return r</pre>

With the updated ranks, we can re-evaluate our best model. The performance metrics before and after applying annotation results are shown in Table 6-11. As can be seen, there is large performance increase in all metrics, verifying the existence of alternative corrections. Both accuracy and MRR increase by more than 30%. Moreover, the hit@5 rate is above 91%, which means that for most of the test data, at least one of the top five candidates is an acceptable correction. A language learner can therefore choose the one that is close to his or her intended meaning from the list of five candidates. In fact, only the writer knows the exact “intended meaning” and it is nearly impossible for a system to guess the right meaning all the time. Therefore, we argue that a fairly short list of candidates can be helpful for learning foreign languages.

Evaluation	Acc.	MRR	Hit@5	Hit@10	Hit@50	Hit@100
Ground-truth	0.3727	0.4605	0.5561	0.6439	0.8039	0.8488
+ Annotation	0.6829	0.7784	0.9122	0.9171	0.9502	0.9600

Table 6-11 Correction performance with human evaluation.

6.8 Error Analysis

The human evaluation result is used to perform further analysis.

6.8.1 Performance on Different POS Tags

POS (# instances)	Accuracy	MRR	Mean rank
VV (316)	0.67	0.77	26.12
NN (277)	0.64	0.73	73.97
AD (130)	0.65	0.75	96.16
P (62)	0.81	0.88	3.10
VA (45)	0.60	0.76	1.98
DEV (23)	1.00	1.00	1.00
PN (21)	0.71	0.80	2.33

Table 6-12 System performance on most frequent POS tags of the erroneous token.

We analyze the system performance on different POS tags of the erroneous token.

The results of POS tags that occur more than 20 times are shown in Table 6-12. The most frequent POS tags, VV and NN, which are open-set word types, contribute the most difficult cases. The accuracy is less than 70% and MRR is less than 80%. Similarly, the set of Chinese adverbs (AD) is also rather open, and the performance is similar to that of verbs and nouns. On the other hand, for closed-set word type such as prepositions (P),



our system performs very well, reaching accuracy 0.81 and MRR 0.88. DEV, the POS tag of adverb marker “地”, inherits very regular usage, and regular pattern of learner errors.

As a result, the system can achieve perfect performance.

6.8.2 Error Cases

We analyze the test segments where our system fails to propose an acceptable correction within the top five candidates and discuss two main sources of system errors below.

1. Segmentation error

Segmentation error could have large impact on correction generation, since some crucial grammatical structures cannot be recognized due to improper word boundary. Table 6-13 shows an example. The ground-truth correction is actually a “把 + noun + verb + 得 + complement” construction. In fact, this example is not a case in which the error can be corrected by replacing a single token. However, the verb “弄” and “得” are incorrectly merged into one word, so we did not filter out this instance in the dataset preparation step. This segmentation error makes correction extremely hard under our framework, since “弄得” is not so similar to the target “造成”. Neither is it “frequent” enough to result in sufficiently high LM score. Moreover, “倒把” is also a problematic merge of two words “倒” and “把”, making the important component “把” of the construction invisible to both the language model and Context2vec context encoder,



which adds another level of difficulty. The low DNN and LM rank reflect this fact. The proposed candidates are by and large those similar to the erroneous token “造成”.

Wrong segment	倒把事造成更糟	
System rank 1 correction	倒把事不更糟	
System rank 2 correction	倒把事導致更糟	
System rank 3 correction	倒把事引發更糟	
System rank 4 correction	倒把事本身更糟	
System rank 5 correction	倒把事引起更糟	
Ground-truth correction	倒把事弄得更糟	
	DNN rank: 593	LM rank: 1250

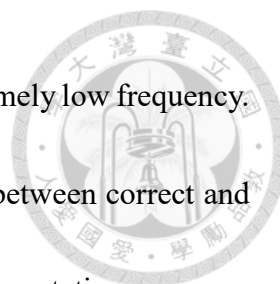
Table 6-13 Example in which segmentation error is the source of error.

To take a closer look at the influence of segmentation error on LM, we show the n-gram LM scores of the correct and wrong segmentation in Table 6-14. The values are log probabilities with base 10.

Correct segmentation	LM log prob.	Wrong segmentation	LM log prob.
弄得	-10.92	弄得	-10.95
弄得更	-13.44	弄得 更	-14.32
弄得 更糟	-14.96	弄得 更糟	-17.83

Table 6-14 Comparison of n-gram LM base 10 log probabilities of correct and wrong segmentation results.

In normal cases, longer sequence of words would have lower log probabilities than shorter ones. However, the score of the single token “弄得” is less than that of two tokens “弄



得”, indicating “弄得” is a result of incorrect segmentation with extremely low frequency.

When we incrementally add words to the sequences, the difference between correct and wrong segmentation in LM scores becomes even larger. Thus, a segmentation error can have serious impact on evaluating the suitability of correction candidates.

Similar cases include:

幾個月後阿姨瘦了 (*又瘦, 很多)

亦需負上一定的法律 (*制裁, 責任)

In the former segment, both the target “又瘦” and the correction “很多” are incorrect merge of several words into single tokens; in the latter case, the collocational combination “負” and “責任” cannot be composed since “負” is not segmented correctly.

2. Context-dependent Similarity

In some cases, the correction is similar to the target erroneous token in restricted types of scenario or context. That is, the similarity may not apply in all the cases. Table 6-15 shows an example illustrating this kind of restricted similarity.

Wrong segment	可是最近的年輕人的自己主張很強	
System rank 1 correction	可是最近的年輕人的教育主張很強	
System rank 2 correction	可是最近的年輕人的他們主張很強	
System rank 3 correction	可是最近的年輕人的他主張很強	
System rank 4 correction	可是最近的年輕人的政治主張很強	
System rank 5 correction	可是最近的年輕人的別人主張很強	
Ground-truth correction	可是最近的年輕人的自我主張很強	
	DNN rank: 18	LM rank: 16

Table 6-15 Example in which the similarity between the target and the correction is context-dependent.

From a certain point of view, the correction “自我” is similar to the target “自己”. However, “自己” has another usage where it works as a pronoun and “別人” is its counterpart. In fact, in this wrong segment, “自己” is given POS tag “PN” in our linguistic processing step, which can guide the model through the wrong direction. The rank 2, 3, 5 candidates are probably selected based on this kind of usage, while the rank 1 and 4 candidates are promoted because of strong collocations such as “年輕人的教育” or “政治主張”. Both DNN and LM rank of the ground-truth are not too low, but combined rank does not outperform the proposed top candidates. Therefore, the system misses the ground-truth correction.

Below is another example. “沒有” and “癒合” do not sound “similar” in general cases. The association would not be considered valid without the given topic “傷口”. In this case, neither does the n-gram LM rank the ground-truth within top 100, probably

because the dependency between “傷口” and “癒合” is a rather long one. It turns out that our system proposes candidates such as “不”, which is more similar to “沒有” in general.

我的傷口也 不知不覺地 (*沒有, 癒合) 了



6.9 Conclusion for WUE Correction

In this stage, given a wrong segment with a known error position, we aim to generate correction candidates that not only result in a correct Chinese sentence segment, but also preserve the original meaning of the writer. Our DNN correction generation model takes target and context features as input and output a correction vector, which can be compared against the vectors of words in the candidate set. We apply language model re-ranking to put emphasis on correctness of the candidates, avoiding generating corrections that mislead the language learners.

In the single-ground-truth automatic evaluation, we achieve accuracy 0.3727 and MRR 0.4605. Since alternative corrections may exist, we perform human evaluation. The top five candidates proposed by our system are judged by human annotators. With the annotation results, a second evaluation is performed, and the accuracy increases to 0.6829 and MRR to 0.7784. Moreover, the hit@5 rate reaches 0.9122, which means that at least one of the top five proposed candidates is an acceptable correction in more than 91% of the cases. Since a list of five candidates is rather short, a language learner can choose a

suitable one from the candidate list and revise his or her sentences even without the help of a language teacher.





Chapter 7 Conclusion and Future Work



7.1 Conclusion

In this thesis, we deal with Chinese word usage errors with three stages, namely, the segment-level detection stage, the token-level detection stage, and the correction stage.

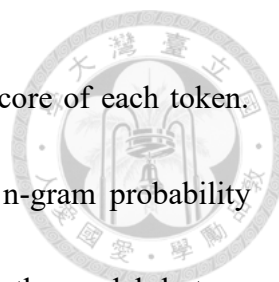
The information used in each stage is summarized in Table 7-1.

	Segment-level detection	Token-level detection	Correction	
Character	<ul style="list-style-type: none"> • Single character features 		<ul style="list-style-type: none"> • CWE word & character embedding 	
Word	<ul style="list-style-type: none"> • N-gram probability • CBOW/SG 	<ul style="list-style-type: none"> • CWIN/ Struct-SG • N-gram probability 		<ul style="list-style-type: none"> • Context2vec • N-gram LM
POS		<ul style="list-style-type: none"> • POS embedding 	<ul style="list-style-type: none"> • POS one-hot encoding 	
Dependency	<ul style="list-style-type: none"> • Dep. count • Dep. bigram 	<ul style="list-style-type: none"> * <i>Evaluation</i> 		

Table 7-1 Summary of information used in each stage.

In the first stage, we formulate segment-level detection as a binary classification task and explore various features to train machine learning classifiers. The features cover character, word and dependency information.

In the second stage, we formulate token-level detection as a sequence labeling task



and use (bidirectional) LSTM models to predict the incorrectness score of each token.

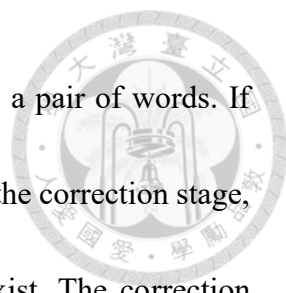
Words and POS tags are represented by real-valued vectors, and n-gram probability features are also adopted. We do not use dependency information in the model, but we use dependency distance to verify that the top two proposed candidates are closely-related words.

In the third stage, we use a DNN that takes context and target features as input and generates a correction vector. We use CWE, which is a joint character and word embedding model, to represent the target erroneous token, and use Context2vec to represent the context. POS information is also provided for the model to learn different transformations for target with different POS tags. In order to enhance the correctness, we further use n-gram LM to re-rank the candidates.

Segment-level detection	Token-level detection	Correction
Accuracy: 0.8425	Accuracy: 0.5138	Accuracy: 0.6829
Precision: 0.9450	MRR: 0.6789	MRR: 0.7784
Recall: 0.7274	Hit@2: 0.8097	Hit@5: 0.9122
F1: 0.8220	Hit@20%: 0.7479	Hit@50: 0.9502

Table 7-2 Summary of the best performance of each stage.

The best performance of each stage is summarized in Table 7-2. Our segment-level WUE detection model has high precision, which means it seldom marks a correct segment as wrong. The token-level detection accuracy is just above 51%. The major challenge is



to determine which token needs correction when the WUE involves a pair of words. If the model proposes two candidates, the hit rate can exceed 80%. For the correction stage, we perform human evaluation since alternative corrections may exist. The correction model can suggest at least one acceptable correction in a list of five candidates for more than 91% of the cases.

7.2 Future Work

In this thesis, we start at the segment level to detect and correct Chinese WUEs. One future direction is to exploit the information of wider context such as sentence and paragraph. For example, in the following cases, the WUE cannot be recognized without looking at the context outside of the segment.

- Conjunction

(*終於, 所以)我只好放棄自己的希望

(*還是, 並且)努力要理解媽媽時代的思想和看法

In the above two segments, the ground-truth corrections are replacing a conjunction with another. To determine which conjunction is suitable, the relationship of current segment with other segments must be considered.

- Discourse dependent

如果我是(*我, 她)的話



To determine that “我” should be replaced with “她”, there must be a reference to someone, who is a female, in the wider context. Without additional contextual information, “你” can also be an acceptable correction.

- Meaning changed

(*理解, 解決)各種的問題

The meaning of “理解” and “解決” are different. The ground-truth correction seems to violate the similarity criterion. However, there might be some clues in the context, so that the human annotator inferred that the intended meaning is that of “解決” instead of “理解”.

In our system, segment-level detection and token-level detection are two stages. We may also consider constructing a model that jointly predicts segment-level and token-level incorrectness. In Chapter 5 we discussed the case in which it is difficult to make a decision between a pair of closely related words. In such cases, even if the model does not rank the ground-truth error position first, the token-level information can still be helpful for determining the segment-level incorrectness.

For the correction task, we argue that besides the correctness criterion, the similarity criterion is also important, since a correction whose meaning is close to the original intended meaning is considered better. We have dealt with the similarity in overlapping Chinese characters, through character embedding terms in common, and semantic



similarity, through the information encoded in word and character embeddings.

Nevertheless, we have not handled phonetical similarity. The following are some examples.

最深刻的(*影響, 印象)是島上的小學運動會

就會(*揮服, 恢復)到以前的穩定的經濟情況了

In the above two examples, the similar pronunciation of the erroneous token and corresponding correction is the source of the mistakes. Since homophones are very common in Chinese, this kind of error-making regularities is worth considering. If pronunciation information is provided, the model can get more clues, and it is more likely that the suitable correction can be selected.

REFERENCE



Chen, X., Xu, L., Liu, Z., Sun, M., & Luan, H. (2015). Joint Learning of Character and Word Embeddings. In *Proceedings of the 24th International Joint Conference on Artificial Intelligence (IJCAI2015)*, pp. 1236-1242.

Cheng, S. M., Yu, C. H., and Chen, H. H. (2014). Chinese Word Ordering Errors Detection and Correction for Non-Native Chinese Language Learners. In *Proceedings of the 25th International Conference on Computational Linguistics (COLING 2014)*, pp. 279–289.

Chollampatt, S., Hoang, D. T., and Ng, H. T. (2016a). Adapting Grammatical Error Correction Based on the Native Language of Writers with Neural Network Joint Models. In *Proceedings of the 2016 Conference on Empirical Methods on Natural Language Processing*, pp. 1901–1911.

Chollampatt, S., Taghipour, K., and Ng, H. T. (2016b). Neural Network Translation Models for Grammatical Error Correction. In *Proceedings of the 25th International Joint Conference on Artificial Intelligence (IJCAI 2016)*. pp. 2768–2774.

Dahlmeier, D., and Ng, H. T. (2011). Correcting Semantic Collocation Errors with L1-induced Paraphrases. In *Proceedings of the 2011 Conference on Empirical Methods on Natural Language Processing*, pp. 107-117.

Dale, R., Anisimoff, Il. and Narroway, G. (2012). HOO 2012: A Report on the Preposition and Determiner Error Correction Shared Task. In *Proceedings of the 7th Workshop on the*

Innovative Use of NLP for Building Educational Applications, pp. 54–62.



Dale, R., and Kilgarriff, A. (2011). Helping Our Own: The HOO 2011 Pilot Shared Task. In *Proceedings of the 13th European Workshop on Natural Language Generation*, pp. 242-249.

Hochreiter, S., and Schmidhuber, J. (1997). Long Short-Term Memory. *Neural computation*, 9(8), pp. 1735-1780.

Huang, S. and Wang, H. (2016). Bi-LSTM Neural Networks for Chinese Grammatical Error Diagnosis. In *Proceedings of the 3rd Workshop on Natural Language Processing Techniques for Educational Applications (NLPTEA 2016)*, pp. 148–154.

Huang, H. H., Shao, Y. C., and Chen, H. H. (2016). Chinese Preposition Selection for Grammatical Error Diagnosis. In *Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers*, pp. 888–899.

Kingma, D., and Ba, J. (2014). Adam: A Method for Stochastic Optimization. *arXiv preprint arXiv:1412.6980*.

Leacock, C., Chodorow, M., Gamon, M. and Tetreault, J. (2014). Automated Grammatical Error Detection for Language Learners. 2nd Edition. Morgan and Claypool Publishers.

Lee, L.H., Rao, G., Yu, L.C., Xun, E., Zhang, B., and Chang, L.P. (2016). Overview of NLP-TEA 2016 Shared Task for Chinese Grammatical Error Diagnosis. In *Proceedings*

of the 3rd Workshop on Natural Language Processing Techniques for Educational Applications (NLPTEA 2016), pp. 40–48.



Lee, L. H., Yu, L. C., and Chang, L. P. (2015). Overview of the NLP-TEA 2015 Shared Task for Chinese Grammatical Error Diagnosis. In *Proceedings of the 2nd Workshop on Natural Language Processing Techniques for Educational Applications (NLPTEA2015)*, pp. 1-6.

Liu, F., Yang, M. and Lin, D. (2010). Chinese Web 5-gram Version 1. Linguistic Data Consortium, Philadelphia.

Ling, W., Dyer, C., Black, A. W., and Trancoso, I. (2015). Two/Too Simple Adaptations of Word2vec for Syntax Problems. In *Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (HLT-NAACL)*, pp. 1299–1304.

McNemar, Q. (1947). Note on the Sampling Error of the Difference Between Correlated Proportions or Percentages. *Psychometrika*, 12(2), pp. 153-157.

Melamud, O., Goldberger, J., and Dagan, I. (2016). Context2vec: Learning Generic Context Embedding with Bidirectional LSTM. In *Proceedings of the 20th SIGNLL Conference on Computational Natural Language Learning (CoNLL 2016)*, pp. 51-61.

Mikolov, T., Chen, K., Corrado, G., and Dean, J. (2013a). Efficient Estimation of Word Representations in Vector Space. *arXiv preprint arXiv:1301.3781*.



Mikolov, T., Kombrink, S., Deoras, A., Burget, L., & Cernocky, J. (2011). RNNLM - Recurrent Neural Network Language Modeling Toolkit. In *Proceedings of the 2011 Automatic Speech Recognition and Understanding Workshop (ASRU)*, pp. 196-201.

Mikolov, T., Yih, W., and Zweig, G. (2013b). Linguistic Regularities in Continuous Space Word Representations. In *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (HLT-NAACL)*, pp. 746–751.

Ng, H. T., Wu, S. M., Wu, Y., Hadiwinoto, C., and Tetreault, J. (2013). The CoNLL-2013 Shared Task on Grammatical Error Correction. In *Proceedings of the 17th Conference on Computational Natural Language Learning: Shared Task*, pp. 1–12.

Ng, H. T., Wu, S. M., Briscoe, T., Hadiwinoto, C., Susanto, R. H., and Bryant, C. (2014). The CoNLL-2014 Shared Task on Grammatical Error Correction. In *Proceedings of the 18th Conference on Computational Natural Language Learning: Shared Task*, pp. 1–14.

Rei, M. and Yannakoudakis, H. (2016). Compositional Sequence Labeling Models for Error Detection in Learner Writing. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 1181–1191.

Schuster, M., & Paliwal, K. K. (1997). Bidirectional Recurrent Neural Networks. *IEEE Transactions on Signal Processing*, 45(11), pp. 2673-2681.



Wu, S. H., Liu, C. L., and Lee, L. H. (2013). Chinese Spelling Check Evaluation at SIGHAN Bake-off 2013. In *Proceedings of the 7th SIGHAN Workshop on Chinese Language Processing*, pp. 35-42.

Xia, Fei. 2000. The Part-of-Speech Tagging Guidelines for the Chinese Treebank (3.0). University of Pennsylvania.

Yu, C. H., and Chen, H. H. (2012). Detecting Word Ordering Errors in Chinese Sentences for Learning Chinese as a Foreign Language. In *Proceedings of COLING 2012: Technical Papers*, pp. 3003–3018.

Yu, C. H., Tang, Y. J., and Chen, H. H. (2012). Development of a Web-Scale Chinese Word N-gram Corpus with Parts of Speech Information. In *Proceedings of the Eighth International Conference on Language Resources and Evaluation (LREC 2012)*, pp. 320–324.

Yu, L. C., Lee, L. H., and Chang, L. P. (2014). Overview of Grammatical Error Diagnosis for Learning Chinese as a Foreign Language. In *Proceedings of the 1st Workshop on Natural Language Processing Techniques for Educational Applications (NLPTEA 2014)*, pp. 42-47.

Yu, L. C., Lee, L. H., Tseng, Y. H., and Chen, H. H. (2014). Overview of SIGHAN 2014 Bake-off for Chinese Spelling Check. In *Proceedings of the Third CIPS-SIGHAN Joint Conference on Chinese Language Processing*, pp. 126–132.