Chinese Spell Checking Based on Noisy Channel Model

Hsun-wen Chiu **Jian-cheng Wu** **Jason S. Chang**
Department of Institute of Information Systems and Applications
National Tsing Hua University
chiuhsunwen@gmail.com wujc86@gmail.com jason.jschang@gmail.com

Abstract

Chinese spell checking is an important component of many NLP applications, including word processors, search engines, and automatic essay rating. Compared to English, Chinese has no word boundaries and there are various Chinese input methods that cause different kinds of typos, so it is more difficult to develop spell checkers for Chinese. In this paper, we introduce a novel method for correcting Chinese typographical errors based on sound or shape similarity. In our approach, similar characters are automatically generated using Web corpora, and potential typos in a given sentence are then corrected using a channel model and a character-based language model in the noisy channel model. In the training phase, we estimate the channel probabilities for each character based on ngrams in Web corpus. At run-time, the system generates correction candidates for each character in the given sentence and selects the appropriate correction using the channel model and the language model.

1 Introduction

Spell checking is a necessary task for text processing of every written language, which involves automatically detecting and correcting typographical errors. However, compared to spell checkers for alphabetical languages (e.g., English or French), Chinese spell checkers are more difficult to develop because there are no word boundaries in Chinese writing system and errors may be caused by various Chinese input methods. In this thesis, we define typos as Chinese characters that are misused due to sound or shape similarity. Liu et al. (2011) show that people tend to unintentionally generate typos due to sound similarity (e.g., *索定 (suo ding) instead of 鎖定 (suo ding)) or shape similarity (e.g., *銷 定 (xiao ding) instead of 鎖定 (suo ding)). On the other hand, some typos found on the Web (e.g., forums or blogs) are used deliberately for the purpose of speed typing or just for fun. Therefore, spell checking is an important component for many applications, including computer-aided writing, search engines, and social media text normalization.

Very little work has been done on the task of Chinese spell checking. The methods proposed in the literature can be classified into two types: rule-based methods and statistical methods. Rule-based methods use knowledge resources, for example, dictionaries, confusion sets, and segmentation systems. Simple rule-based methods, however, have their limitations. The following sentence is a snippet collected from students' written essays which is correct .

> 為什麼你要如此地用功呢？如果我不用功，那以後我將趕不上自己所定的目標。(*wei she me ni yao ru ci di yong gong ne ？ru guo wo bu yong gong ， na yi hou wo jiang gan bu shang zi ji suo ding de mu biao 。*)

Unfortunately, based on simple rules the two characters 所 (*suo*) and 定 (*ding*) are likely to be regarded as typos of the dictionary word 鎖定 (*suo ding*) with identical pronunciation.

The data-driven, statistical spell checking approach appears to be more robust and perform better. Statistical methods typically use a large corpus to create a language model to validate the correction hypotheses. Intuitively, by using 自己所定的目標 (*zi ji suo ding de mu biao*), the three characters 所定的 (*suo ding de*) are a trigram with high probability in a monolingual corpus, we may determine the 所定 (*suo ding*) is not a typo after all. Table 1 shows the frequency and probability of 所定的 (*suo ding de*) and 鎖定的 (*suo ding de*).

Proceedings of the Third CIPS-SIGHAN Joint Conference on Chinese Language Processing, pages 202–209,
Wuhan, China, 20-21 October 2014

Trigrams	Freq.	LM prob.(log)
所定的 (*suo ding de*)	5	-0.70
鎖定的 (*suo ding de*)	2	-1.49

Table 1: Example trigrams with corresponding frequency and probability.

In this thesis, we propose a model using statistical approaches and model generates the most appropriate corrections in a given sentence. In the training phase, we automatically generate the channel model (confusion set). We use a Chinese spell checker to correct instances in the training data and estimate the channel probability of a typo condition on a correct character , then re-estimate the probability, and iterate until convergence.

At run-time, the checker corrects typos using a noisy channel model. Consider the following sentence.

> 為什麼你要如此地用功呢？如果我不用功，那以後我將趕不上自己**鎖**定的目標。(*wei she me ni yao ru ci di yong gong ne？ru guo wo bu yong gong ， na yi hou wo jiang gan bu shang zi ji suo ding de mu biao 。*)

The checker generates correction candidates by the replacements of each character and confusable characters with channel probabilities in a beam search algorithm, then calculates the probability of correction hypotheses according to the language model and the channel model. Three correction candidates are shown in Table 2. Finally, the checker returns the correction with the highest score, e.g., the follow sentence:

> 為什麼你要如此地用功呢？如果我不用功，那以後我將趕不上自己**所**定的目標。(*wei she me ni yao ru ci di yong gong ne？ru guo wo bu yong gong ， na yi hou wo jiang gan bu shang zi ji suo ding de mu biao 。*)

The rest of the paper is organized as follows. We describe the proposed model for automatically correcting the spelling typos in section 2. Section 3 presents the experimental data. We conclude in Section 4.

2 Method

Using fixed rule to correct typos in a given Chinese sentence (e.g., 自己鎖定的目標 *(zi ji suo ding de mu biao)*) does not work very well. Previous work typically corrects typos based on a set of detection rules. Unfortunately, the detection rules depend on a lot of resources, and can be at times unreliable. Typo positions usually are detected using heuristic rules based on Chinese dictionary, word segmentation and the frequency of the ngram. However, Chinese dictionary, and word segmentation have their limitations. For example, the segmentation result of the sentence "自己**鎖**定的目標 " is "自己/**鎖**定/的/目標 ", the two characters 鎖 and 定 may or may not be considered as a word, depend on the segmentation system. To avoid the limitations of rule-based method, a promising approach for Chinese spell checking is to train a noisy channel model based on unannotated data, which containing many information.

Hypotheses
為什麼你要如此地用功呢？如果我不用功，那以後我將趕不上自己**所**定的目標。
為什麼你要如此地用功呢？如果我不用功，那以後我將趕不上自己**瑣**定的目標。
為什麼你要如此地用功呢？如果我不用功，那以後我將趕不上自己**鎖**定的目標。

Table 2: The three correction candidates of the given sentence.

In the rest of this section, we describe our solution to the problem of Chinese spell checking. We describe the process of training the channel model in Section 2.1. More specifically, we describe the method for limiting confusable characters in Section 2.1.1, and the use of ngrams in Section 2.1.2. We will also describe an Expectation-Maximization (*EM*) algorithms for estimating channel probabilities in Section 2.1.3. This algorithm relies on a set of confusable characters and ngrams. Finally in Section 2.2, we describe how to correct typos using the trained noisy channel model at run-time by combining channel model and language model.

2.1 Training Channel Model

We attempt to learn to develop a channel model from the ngrams of Web corpus for correcting Chinese spell typos.

Type	Sound	Shape
Full	所瑣索梭娑嗦縮 唆蓑簑數碩勺鑠 說朔爍帥率妁鎗 鎰鎳鎢鎬鎮鎊鎚 莎蟀鎔鎘	瑣銷鋇鏜鐺鑽貝 賞員賄煩鈔貼敗 狽盼賸賤賊損貽 貞負頁賽贊圓貧 財則
Limited	索瑣鎖所	賸鏜鎖

Table 3: The full confusion set and the limited confusion set of 鎖.

2.1.1 Limiting Confusable Characters

In the first stage of training the channel model, we limit the confusable characters based on the sound and shape similar characters, which containing unlikely confusable characters (as the full confusion set). For example, the full confusion set of 鎖 *(suo)* is shown in Table 3. Liu et al. (2011) analyzed erroneous Chinese character and found that more than 70% of typos were related to the phonologically similar characters, about 50% are visually similar, and almost 30% are both phonologically and visually similar. The goal of this method is to reduce the sizes of the confusion sets and improve the accuracy.

The input to this stage is a set of confusable characters. These confusable characters constitutes the full confusion set. We generate potential confusable characters by reducing some unlikely confusable relations and expanding the confusable characters slightly.

The output of this stage is confusion sets that can be used to correct ngrams for training channel model. Limited confusion set of 鎖 *(suo)*, automatically generated from the full confusion set is shown in Table 3. We can see that the limited confusion set minimize the confusable characters and select more likely characters. The limited confusion set is used to accurately correct ngrams and reduce the computational complexity.

Our method for limiting confusable characters can generate many characters, potentially including a significant number of characters that are not useful in correcting typos. We also remove some loosely similarly relations and expand the confusable characters slightly. For example, we remove all relations based on non-identical phonologically similarity. After that, we add the similarly sounding characters based on nasal consonant in Chinese phonetics (e.g., "ㄣ , ㄥ" *(en, eng)* and "ㄢ , ㄤ" *(an, ang)*), and retroflex consonant (e.g., "ㄙ , ㄕ" *(shi, si)* and "ㄔ , ㄘ" *(chi, chi)*). We also modify the shape similarity by comparing the characters in Cangjie codes *(*倉頡碼*)* to filter out confusable characters with low similarity. We retain character pairs differing from each other by at most one symbol in Cangjie codes that tend to be highly similar in shape. For example, the code of 徵 *(zheng)* and 微 *(wei)* are highly similar in shape, and their corresponding codes "竹人山土大" and "竹人山山大", differ only in one place.

Note that we do not attempt to estimate the channel probabilities of typos of a character at this point. In contrast, we only use sound or shape similarity to limit confusable characters, leading to more effective confusion set as the basis for subsequent probability estimation.

2.1.2 Retrieving Ngrams

In the second stage of the training phase, we retrieve ngrams (e.g., 所定目標 *(suo ding mu biao)*) possibly containing a typo characters (e.g., 所 *(suo)*) that can be corrected using the confusable characters (e.g., 所 *(suo)*, 鎖 *(suo)*, or 索 *(suo)*). Because estimating channel probabilities need a parallel corpus with typos annotated, we use an existing Chinese spell checker *CSC* to correct typos in the ngrams. We use ngrams generated based on collocates of high frequency words containing the confusable character. The procedure for retrieving and correcting ngrams consist of a number of steps, namely, generating collocates for words containing a specific character, filtering these collocates by frequency, producing the ngrams for the remaining collocates, and correcting these ngrams using *CSC*. Each step is described below in detail.

For this stage of the learning process, we use a collection of *(Word, Collocate)* pairs (e.g., (目標, 鎖定) *((mu biao, suo ding))*, (版面, 鎖定) *((ban mian, suo ding))*). We generate the word from the corpus using word frequency and find corresponding collocates using Dice coefficient, which is a statistic association value used for comparing the relation of words and collocates. The collocates of each word are sorted according to the Dice coefficient. We retain at most *K* collocates per word to reduce the computational cost. We compute Dice coefficient using the following equation:

$$Dice(w, c) = \frac{2 \cdot freq_{(\mathrm{w})} \cdot freq_{(\mathrm{c})}}{freq_{(\mathrm{w})} + freq_{(\mathrm{c})}} \qquad (1)$$

where $freq_{(w)}$ is the frequency of the word, and $freq_{(c)}$ is the frequency of the collocate. Take 鎖

Words	Collocates	Dice
鎖定	版面	.025
	單擊	.021
	防偷	.004
	目標	.004
	移動	.004
	已經	.002
	敬請	.001
	解除	.001
Words	Collocates	Dice
封鎖	衝出	.019
	長城	.017
	突破	.015
	嚴密	.007
	網絡	.002
	大陸	.001

Table 4: Two sample collocates of 鎖定 and 封鎖.

Typos	Texts	Count
所	中 所 定 目標	86
	依 所 定 目標	83
	達到 所 定 目標	44
	我們 所 定 的 目標	42
索	索 定 海珠 收	66
	索 定 起息日	93
	索 定 高清	40

Table 5: Sample texts of typo 所 and 索 of 鎖 from the corpus.

(suo) for instance, the words (e.g., 鎖定 *(suo ding)* and 封鎖 *(feng suo)*) and their corresponding collocates of words are shown in Table 4. The word 鎖定 *(suo ding)* has the highest Dice coefficient of 0.025 with the collocate 版面 *(ban mian)*, while 封鎖 *(feng suo)*) has the highest Dice coefficient of 0.019 with the collocate 衝出 *(chong chu)*.

For each *(Word, Collocate)* pair, we generate all possible potential ngrams *N* containing *Word* and *Collocate*. This stage of the learning process operates over a corpus of ngram words. The sample texts of the typos (所, 索, and 瑣) of 鎖 found in a corpus is shown in Table 5. We find the ngrams in the corpus with identical collocates and *Word* containing confusable characters (e.g., (所定, 目標)). Sample instances of character 鎖 is shown in Table 6. In this sample, we can find that 鎖 may be misused as confusable characters (e.g., 所, 索, 瑣) in the corpus with such information in the ngrams, we can generate typo pairs (e.g., [所, 鎖], [索, 鎖], [瑣, 鎖]). Finally, we correct the typos in these ngrams by using existing Chinese spell checker (In Section 2.1.3). With the typos and corrections, we can estimate the channel probabilities.

Words	Collocates	Characters	Instances
鎖定	目標	所	目標所定
封鎖	突破	索	突破封索
深鎖	眉頭	瑣	眉頭深瑣

Table 6: A sample of instances containing character 鎖 and potentially confusable characters.

2.1.3 Correcting Ngrams and Training Channel Model

In the third and final stage of training, we correct the ngrams and train the channel model for supporting correction candidates. Figure **??** shows the algorithm for correcting ngrams using a Chinese spell checker and estimating the channel probabilities related to typo pairs. The procedure is repeated for all ngrams obtained in the previous stage until the channel probabilities converge.

We are given a set of ngrams as training data (described in Section 2.1.2). Recall that our goal is to estimate the channel model for each character, in the form of [original, correction, log channel probability] (e.g., [所, 鎖, -4.284] and [索, 鎖, -5.264]). In order to generate a parallel corpus, we need to provide representative ngrams to the training algorithm. The training set is created by retrieving the ngrams from *Word*s of each character and the corresponding *Collocate*s in the corpus.

We apply a previously developed Chinese spell checker(*CSC*) to correct ngrams. In this checker, we adopt the confusion set limited in Stage (1) to reduce the unlikely confusable characters and improve the accuracy for generating typo pairs. We combine the global error probability (GP) and local error probability (LP) to reliably estimate the channel probabilities (CP) using following equation:

$$CP(O,C) = W_{\text{GL}} \cdot GP + (1 - W_{\text{GL}}) \cdot LP(O,C) \quad (2)$$

where *O* is original character, *C* is corrected character, and W_{GL} is a weight for probability. The global error probability is a prior probability calculated from a development data set, which can instead the detection and avoid data sparse. The global error probability calculated by the follow-

Ngrams	Corrections	Typo Pairs
目標所定	目標所定	[目, 目], [標, 標] [所, 所], [定, 定]
突破封索	突破封鎖	[突, 突], [破, 破] [封, 封], [索, 鎖]
眉頭深瑣	眉頭深鎖	[眉, 眉], [頭, 頭] [深, 深], [瑣, 鎖]

Table 7: A sample of the typo pairs for 鎖.

ing equation.

$$GP(Devedata) = \left\{ \begin{array}{c} \frac{count(nochange)}{count(char)} \\ \frac{count(typos)}{count(char)} \end{array} \right\} \quad (3)$$

where *count(nochange)* is the count of corrected characters, *count(typos)* is the count of typos, and *count(char)* is the count of characters. The *Devedata* is the development data.

We use the Expectation-maximization algorithm to estimate the local error probabilities related to the confusion set. We initialize the confusion set with uniformed probability in the E-step and re-estimate the probability of each character in M-step until the local error probability converge. For each of the potentially confused ngram (e.g., 所定目標 *(suo ding mu biao*), we attempt to find typos and corrections using *CSC* (Step (1)) and produce the typo pairs (Step (2)). The typo pairs are in the form of [Original, Correction]. The frequency is the count of how many times of the ngram occurs in the corpus. We estimate the local error probability based on nochange pair (e.g., [所, 所] *([suo, suo])*), and correction pair (e.g., [所, 鎖] *([suo, suo])*). In Table 7, we show a sample of the typo pairs in the ngrams of the character 鎖 *(suo)*.

Then we calculate the global error probability using the development data (Step (3)). In Step (4), the typo pairs are sorted according to the *Original*. For each [*Original, Correction*] pair, we calculate the local error probability of the *Original* conditioned on *Correction* (Step (5a)). The probability is calculated as follows:

$$LP(C_{\mathrm{O}}, O) = \frac{count(O, C_{\mathrm{O}})}{count(O)} \quad (4)$$

As shown in Table 8, the total *count* of 所 *(suo)* is 6799532 + 529 + 235 = 6800296, the *count* of (索, 所) is 235, and the *LocalErrorProbability(索—所)* is calculated as follows:

Original	Correction	Frequency
所	所	6,799,532
所	匠	529
所	索	235
Total Frequency		6,800,296

Table 8: Sample of the typo pairs with frequency.

Original	Correction	Freq.	LP_{log}
所	所	6799532	-0.0001
所	匠	529	-9.4614
所	索	235	-10.2728
所	瑣	1	-15.7324
所	鎖	1	-15.7324

Table 9: The result of the local error probability with smoothing.

LocalErrorProbability(索—所) = Count(索, 所)) / Count(所) = 235/6800296 =0.0000346

However, we can not estimate that 所 *(suo)* as a typo of 瑣 *(suo)*, if *CSC* does not find [所, 瑣] *([suo, suo])*. In that case, we use smoothing algorithm to solve this problem. If a confusable character does not has a certain typo pair, we use add-one smoothing algorithm to deal with the unseen problem. For example, confusable characters (e.g., 瑣, 鎖) of 所 *(suo)* are not found in the corpus, so we add count one for them. Table 9 shows a confusion set of 所 *(suo)* and the corresponding smoothed local error probability.

We combine the global error probability and the local error probability to estimate the channel probabilities in Step (5b), and save the *Original*, *Correction*, and their channel probability in the channel model in Step (5c). Steps (1) through (5) are repeated to re-estimate the local error probability until the probabilities converge. The output of this stage of training is a channel model with reliable probabilities, automatically estimated using the confusable characters and ngrams based on collocates. A samples of the channel model for 所 *(suo)* is shown in Table 10.

2.2 Run-time Typo Correction

Once the channel model is automatically trained for each character, we store the model as a confusion set. We then correct a given sentence using the procedure shown in Figure **??** with the character-based language model and the channel model.

For each character in the given sentence of *n*

Original	Correction	Freq.	CP_{log}
所	所	6799532	-0.1416
所	匠	529	-2.2111
所	索	235	-4.4357
所	瑣	1	-10.4947
所	鎖	1	-10.4947

Table 10: A sample of the channel model for 所 *(suo)*.

Originals	Corrections	Ngrams	Score
自	自	()	0.0
己	己	(,自)	-2.6049
鎖	所	(自,己)	-2.6756
定	定	(己,所)	-5.1145
的	的	(所,定)	-6.3698
目	目	(定,的)	-5.1627
標	標	(的,目)	-5.7875
		(目,標)	-10.2282

Table 11: A sample of the hypotheses.

characters (e.g., 自己**鎖**定的目標 *(zi ji suo ding de mu biao)*), we correct typos as follows. In Step (1), the system initializes *n* stacks for the channel model, [*Character, Ngram, Score*]. In Step (2), the system replaces each character with the confusable characters (e.g., 所,索,瑣,鎖 *(suo, suo, suo, suo)*) in the channel model as the correction candidates. For each confusable characters, we create new hypotheses with a score, character ngram state, character, and correction candidates. In order to reduce computational complexity, we use beam search algorithm to replace each and calculate the score of sentences. The score in a hypothesis is calculated based on the channel model and the language model as follows.

$$S(hypothesis) = log(LP^{W_{\text{LC}}} \cdot CP^{(1-W_{\text{LC}})}) \quad (5)$$

$$= W_{\text{LC}} \cdot log(LP) + (1 - W_{\text{LC}}) \cdot log(CP) \quad (6)$$

where *LP* is language model probability, *CP* is channel probability, and W_{LC} is a weight parameter in channel model and language model. A sample hypothesis is shown in Table 11. In Step (3), the new hypothesis are stored in the stack and combined with the existing hypothesis in Step (4). If the stack has too many hypotheses, we prune the stack down to a fixed size in Step (5).

Finally in Step (6), we compare the score of all the hypotheses in the last stack, and output the correction candidate with the highest score as output.

	Sentences
Given	遇到逆竟時，我們必須勇於面對。
Corrected	遇到逆境時，我們必須勇於面對。
Given	人生難免會碰到的一些錯折。
Corrected	人生難免會碰到的一些挫折。

Table 12: A sample of the given sentences and corrections.

When there is no correction candidates with the highest score (e.g., score(自己所定的目標) = -10.2282), we output the given sentence. Table 12 shows three input sentences and the corresponding corrected sentences output. For example, 竟 *(jing)* is corrected as 境 *(jing)*, because 境 *(jing)* is the most appropriate for the context of 遇到逆 * 時 *(yu dao ni jing shi)*.

3 Experiment Setting

Our systems were designed to provide wide coverage spell checking for Chinese texts. As such, we trained our systems using the confusion set, a compiled corpus, Web-scale ngrams, and an existing Chinese spell checker. These resources are used for different purposes: the confusion sets provide the correction candidates; the compiled corpus provide the training data for the language model; Web-scale ngrams and the existing Chinese spell checker are used for training the channel model. We evaluate our systems on the sentence level. In this section, we present the details of data sources used in training(Section 3.1 to Section 3.4).

3.1 Confusion Set

The confusion sets we used are the same as in Liu et al. (2011) and provided for SIGHAN 7 Bake-off 2013: Chinese Spelling Check Shared Task. The confusion sets represent sound similarity and shape similarity between a typo and potential corrections.

There four categories of phonological similarity between two characters: identical sound and tone (*II*), identical sound but different tone (*ID*), similar sound and identical tone (*SI*), similar sound and different tone (*SD*), and identical radical and number of strokes (*RS*). A sample of sound-related confusion sets from SIGHAN 7 Bake-off 2013. In this sample, the confusion sets of 已 *(yi)*, 勇 *(yong)*, and 胡 *(hu)* contain a lot of unlikely con-

N-gram Types	Google Chinese 5-gram
Unigram	1,616,150
Bigram	281,107,315
Trigram	1,024,642,142
Fourgram	1,348,990,533
Fivegram	1,256,043,325

Table 13: The information of n-grams in Google Chinese 5-gram.

N-gram Types	Traditional Chinese 5-gram
Unigram	527,694
Bigram	102,092,428
Trigram	237,599,483
Fourgram	201,500,549
Fivegram	126,959,922

Table 14: The information of n-grams in Traditional Chinese 5-gram.

fusable characters. Examples of unlikely pairs include 已 *(yi)* and 肄 *(yi)* in *ID*, 勇 *(yong)* and 穩 *(wen)* in *SI*, 胡 *(hu)* and 馥 *(fu)* in *SD*. The shape-related confusion sets of 已 *(yi)*, 勇 *(yong)*, and 胡 *(hu)*. The confusion sets also contain loosely similarly relations, for instance, 已 *(yi)* and 圈 *(quan)* are not very similar visually.

In our work, we expand the sets slightly and also remove some unlikely confusable characters in order to improve the performance. We modify the confusion set using the pronunciation and Cangjie codes (倉頡碼). The process is described in detail in Section 2.1.1.

3.2 Google Chinese Web 5-gram

In 2010, *Google* published a Chinese Web 5-gram dataset based on public webpages through Linguistics Data Consortium (LDC).[1] Chinese Web 5-gram consists of Chinese word n-grams and their observed frequency counts generated from approximately 883 billion word tokens of text in publicly accessible Web pages. The Google Chinese Web 5-gram contains 30 GB (gzip compressed) of text files with n-grams ranging from unigrams (single words) to fivegrams. In this work, we used only the traditional Chinese 5-grams. Table 13 and Table 14 show the information of 5-grams in Google Chinese Web 5-gram and traditional Chinese Web 5-gram. We use the traditional Chinese Web 5-gram to retrieve ngrams (at most ten *Word*s) in the training phase for estimate channel model probabilities. The advantage of using the Web ngram is that unlike a compiled corpus, it contains many typos.

3.3 Existing Chinese Spell Checker

We use an existing Chinese spell checker (*CSC*) we previously developed in 2013 (Chiu et al., 2013) with the training data described in (Wu et al., 2013). This *CSC* is based on a novel method for detecting and correcting Chinese typographical typos. The approach involves word segmentation, detection rules, and phrase-based machine translation. The error detection module detects typos by segmenting words and checking word and phrase frequency based on compiled and Web corpora. The phonological or morphological typographical typos found then are corrected by running a decoder based on the statistical machine translation model. The language model is trained using the word-based corpus using the SRILM (Stolcke et al., 2011) toolkit. The translation model is trained using the frequency of the word containing typos and the corrected word. The results show that the proposed system achieves high accuracy in error detecting and correcting. We use this Chinese spell checker to train the channel model and as a system to compared with the proposed method.

3.4 Sinica Corpus

”Academia Sinica Balanced Corpus of Modern Chinese”, or ”Sinica Corpus”, is the first balanced Chinese corpus with part-of-speech tags. The size of the corpus we used is about 5 million words. The corpus is segmented according to the word segmentation standard proposed by the ROC Computational Linguistic Society. Each segmented word is manually tagged with a part of speech. Texts in the corpus are collected from different areas: Literature, Life, Society, Science, Philosophy, and Art. Table 15 shows the information about numbers of word, character, article, and percentage by area. We use Sinica Corpus (ignoring word segmentation) to train a character-based n-gram language model running the SRILM (Stolcke et al., 2011) toolkit. The sizes of the ngrams of the character-based language model is shown in Table 16

[1] https://catalog.ldc.upenn.edu/LDC2010T06

Areas	Word Token	Character	Article
Literature	777,050	1,169,801	1,385
Life	858,750	1,398,791	2,301
Society	1,610,997	2,711,720	3,246
Science	629,838	1,054,738	994
Philosophy	439,955	673,080	695
Art	474,340	781,415	518
Others	101,394	160,306	89
Total Count	4,892,324	7,949,851	9228

Table 15: The information of the word, character, article, and percentage in the area of sinica corpus.

Ngram Types	Ngram Count
Unigram	17,201
Bigram	741,739
Trigram	859,442
Fourgram	791,846
Fivegram	588,200

Table 16: The information of n-grams in character-based language model.

4 Conclusions and Future Work

Many avenues exist for future research and improvement of our system. For example, confusion sets can be automatically generated using Web-based character n-grams to improve correction performance. Part of speech tagging can be performed to provide more information for the noisy channel model. Named entities can be recognized in order to avoid false alarms. A supervised statistical classifier can be used to model channel probability more accurately. Additionally, an interesting direction to explore is using Web corpus in addition to a compiled corpus for correcting typos. Yet another direction of research would be to consider errors related to a missing or redundant character, or collect data from user to update channel probabilities dynamically.

In summary, we have introduced a novel method for Chinese spell checking. In our approach, the channel model is trained based the sound and shape similarity using Web corpus, and the potential typos in a given sentence is corrected using a noisy channel model. In the training phase, we limit the confusable characters, retrieve the ngrams from the Web corpus, and correct ngrams and estimate the channel probability. At run-time, our system generate the correction candidates and calculate their probabilities using the language model and channel model from a given sentence. The results prove that the channel probability we estimate for the noisy channel model are useful in Chinese spell checking.

References

[Chiu et al.2013] Hsun-wen Chiu, Jian-cheng Wu, and Jason S. Chang. 2013. Chinese spelling checker based on statistical machine translation. In *Sixth International Joint Conference on Natural Language Processing*, page 49.

[Liu et al.2011] C-L Liu, M-H Lai, K-W Tien, Y-H Chuang, S-H Wu, and C-Y Lee. 2011. Visually and phonologically similar characters in incorrect chinese words: Analyses, identification, and applications. *ACM Transactions on Asian Language Information Processing (TALIP)*, 10(2):10.

[Stolcke et al.2011] Andreas Stolcke, Jing Zheng, Wen Wang, and Victor Abrash. 2011. Srilm at sixteen: Update and outlook. In *Proceedings of IEEE Automatic Speech Recognition and Understanding Workshop*, page 5.

[Wu et al.2013] Shih-Hung Wu, Chao-Lin Wu, and Lung-Hao Lee. 2013. Chinese spelling check evaluation at sighan bake-off 2013. In *Proceedings of the 7th SIGHAN Workshop on Chinese Language Processing*, pages 35–42.