NTOU Chinese Spelling Check System in SIGHAN Bake-off 2013

Chuan-Jie Lin and Wei-Cheng Chu
Department of Computer Science and Engineering
National Taiwan Ocean University
No 2, Pei-Ning Road, Keelung 202, Taiwan R.O.C.
{cjlin, wcchu.cse}@ntou.edu.tw

Abstract

This paper describes details of NTOU Chinese spelling check system participating in SIGHAN-7 Bakeoff. The modules in our system include word segmentation, N-gram model probability estimation, similar character replacement, and filtering rules. Three dry runs and three formal runs were submitted, and the best one was created by bigram probability comparison without applying preference and filtering rules.

1 Introduction

Automatic spell checking is a basic and important technique in building NLP systems. It has been studied since 1960s as Blair (1960) and Damerau (1964) made the first attempt to solve the spelling error problem in English. Spelling errors in English can be grouped into two classes: non-word spelling errors and real-word spelling errors.

A non-word spelling error occurs when the written string cannot be found in a dictionary, such as in *fly *fron Paris*. The typical approach is finding a list of candidates from a large dictionary by edit distance or phonetic similarity (Mitten, 1996; Deorowicz and Ciura, 2005; Carlson and Fette, 2007; Chen *et al.*, 2007; Mitten 2008; Whitelaw *et al.*, 2009).

A real-word spelling error occurs when one word is mistakenly used for another word, such as in *fly *form Paris*. Typical approaches include using confusion set (Golding and Roth, 1999; Carlson *et al.*, 2001), contextual information (Verberne, 2002; Islam and Inkpen, 2009), and others (Pirinen and Linden, 2010; Amorim and Zampieri, 2013).

Spelling error problem in Chinese is quite different. Because there is no word delimiter in a Chinese sentence and almost every Chinese character can be considered as a one-syllable word, most of the errors are real-word errors. On the other hand, there can be a *non-character error* where a hand-written character is not legal (thus not collected in a dictionary). Such an error cannot happen in a digital document because all characters in Chinese character sets such as BIG5 or Unicode are legal.

There have been many attempts to solve the spelling error problem in Chinese (Chang, 1994; Zhang *et al.*, 2000; Cucerzan and Brill, 2004; Li *et al.*, 2006; Liu *et al.*, 2008). Among them, lists of visually and phonologically similar characters play an important role in Chinese spelling check (Liu *et al.*, 2011).

This bake-off is the first Chinese spell checking evaluation project. It includes two subtasks: error detection and error correction. The task is organized based on some research works (Wu *et al.*, 2010; Chen *et al.*, 2011; Liu *et al.*, 2011).

2 Architecture

Figure 1 shows the architecture of our Chinese spelling checking system.

A sentence under consideration is first word-segmented. All one-syllable words are replaced by similar characters and the newly created sentences are word segmented again. If a new sentence results in a better word segmentation, spelling error is reported. Details are described in the following subsections. All the examples are selected from the development set.

2.1 Similar character replacement

We only handle the case that a misused character becomes a one-syllable word. In other words, only one-syllable words will be checked whether it is correct or misused. The case of misusing a

Proceedings of the Seventh SIGHAN Workshop on Chinese Language Processing (SIGHAN-7), pages 102–107,
Nagoya, Japan, 14 October 2013.

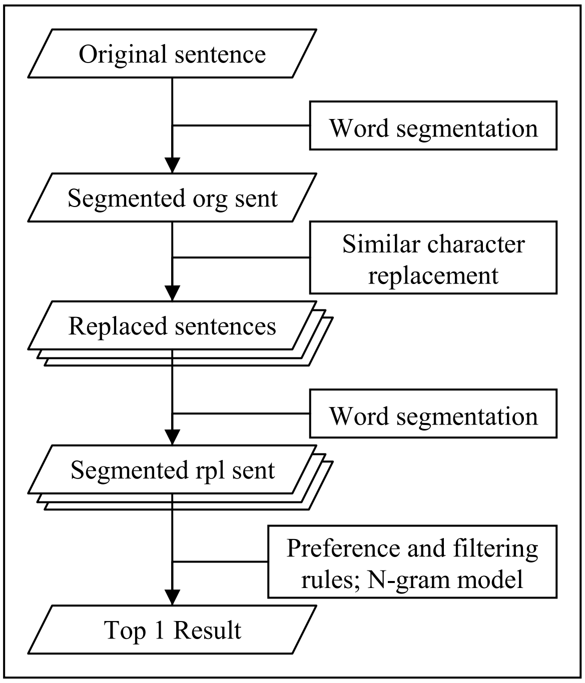

Figure 1. Architecture of NTOU Chinese Spelling Check System

two-syllable word instead of another two-syllable word (or longer words in either side) remains as our future work.

For each one-syllable word, its corresponding character in the original un-segmented sentence is replaced by its similar characters. The organizers of this evaluation project provided two kinds of similar character lists, one for phonologically similar characters and one for visually similar characters. We adopted all these lists except the one consisting of characters written in the same number of strokes with the same radical.

Taking Doc#00076 in the development set as an example. The original sentence is

. . . 不能輕意半途而廢. . .

and it is segmented as

. . . 不能 輕 意 半途而廢. . .

“輕” and “意” are one-syllable words so they are candidates of spelling errors. According to the similar character lists provided by the organizers, the phonologically similar characters of 輕 include 青情傾鯖氫... and its visually similar characters include 逕經涇經徑... Replacing 輕 with similar characters will produce the following new sentences.

. . . 不能青意半途而廢. . .
. . . 不能情意半途而廢. . .
. . . . . .
. . . 不能逕意半途而廢. . .
. . . 不能經意半途而廢. . .
. . . . . .

The newly created sentences are again word segmented and passed to the next steps.

2.2 Preference and filtering rules

Before determining a spelling error, some rules are applied to prefer or discard a similar-character replacement. These rules are defined as follows.

Rule 1: Long word preference

If a replacement results in a word whose length are 3 or more characters, this replacement is ranked first; if there are more than one such replacements, break ties by their N-gram probabilities. Take Doc#00028 as an example:

豐富 的 學識 更 如 海綿 受到 壓迫
而 盪 然 無 存

“蕩” is phonologically similar to “盪”. The newly created sentence is segmented as

豐富 的 學識 更 如 海綿 受到壓迫
而 蕩然無存

where “蕩然無存” is a word with 4-character long. We will prefer such a replacement.

Rule 2: No error at the beginning

If a replacement takes place at the beginning of a sub-sentence, discard it. We assume that a writer seldom makes mistakes at the beginning of a sub-sentence. A sub-sentence is defined as a passage ended by a comma, period, exclamation, question mark, colon, or semicolon.

Take Doc#00001 as an example:

不 怕 措 折 地 奮鬥

Although “不” is a one-syllable word, it occurs at the beginning of a sub-sentence therefore no replacement is performed on this word.

Rule 3: No error in personal names

If a replacement results in a personal name, discard it. Our word segmentation system performs named entity recognition at the same time. If the replacing similar character can be considered as a Chinese family name, the consequent characters might be merged into a personal name. As most of the spelling errors do not occur in personal names, we simply ignore these replacements. Take Doc#00002 as an example:

突然 一 陣 巨 晃

“甄” is phonologically similar to “陣” and is one of the Chinese family names. The newly created sentence is segmented as

突然 一 甄巨晃(PERSON)

where “甄巨晃” is recognized as a personal name. We will discard such a replacement.

Rule 4: Stopword filtering

If the replaced (original) character is a personal anaphora (你 ‘you’ 我 ‘I’ 他 ‘he/she’) or numbers from 1 to 10 (一二三四五六七八九十), discard the replacement. We assume that a writer seldom misspell such words. Take Doc#00002 as an example:

突然 一 陣 巨 晃

Although “一” is a one-syllable word, it is in our stoplist therefore no replacement is performed on this word.

2.3 N-gram probabilities

The newly created sentences are again word segmented. If a new sentence results in a better word segmentation, it is very likely that the replaced character is misused and its similar character is the correct one. But if no replacement is better than the original sentence, it is reported as “no misspelling”.

The possibility of a sequence of words can be measured in its generation probability measured by a language model. We used smoothed unigram and bigram models in our experiments.

2.4 Error detection

The detail of our error detection algorithm is delivered here. Given a sentence,

1. Perform word segmentation on the original sentence
2. For each one-syllable word not violating the filtering rules (leading words or stop words), for each of its similar characters:
   (1) Its corresponding character in the original un-segmented sentence is replaced by the similar character
   (2) Perform word segmentation on the new sentence
   (3) If the new word sequence matches a preference rule (long words), rank this replacement to the top.
   (4) If the new word sequence matches a filtering rule (personal names), discard this replacement.
   (5) Otherwise, measure the N-gram probability (ungiram and bigram in this paper) of the new word sequence. Assign the rank of this replacement according to its N-gram probability.
3. If the top one segmentation is of the original sentence, report “no error” (either in error detection or correction subtasks).
4. If the top one segmentation is of a new sentence:
   - ♦ For error detection subtask, report “with error”
   - ♦ For error correction subtask, report the location of the replaced character and its similar character as the correction

Some examples of successful and wrong corrections by unigram and bigram probabilities are given in Table 1 to Table 4. All the values in “prob” columns are the logarithms of the probabilities.

Table 1 shows an example of correctly detecting an error by unigram probabilities. In Doc#00076, although replacing “輕” by “情” or “經” can form longer words, their unigram probabilities are less than the segmentation produced by replacing “意” by “易”.

However, Table 2 gives an example of incorrectly detecting an error by unigram probabilities. In Doc#00002, the segmentation produced by replacing “晃” by “星” has a higher unigram probability than the correct replacement of “貴” by “櫃”.

Table 3 shows an example of correctly detecting an error by bigram probabilities. In Doc#00001, although replacing “怕” by “必” or “地” by “抵” can form longer words, their bigram probabilities are less than the segmentation produced by replacing “措” by “挫”.

However, Table 4 gives an example of incorrectly detecting an error by bigram probabilities. In Doc#00046, the segmentation produced by replacing “每” by “個” has a higher bigram probability than the correct replacement of “蹟” by “跡”.

3 Performance

There are two sub-tasks in this bake-off: error detection and error correction.

Original sub-sentence in Doc#00076			Unigram prob	Bigram prob
不能 輕 意 半途而廢			-237.12	-360.48
Org	Rpl	Segmentation	Unigram prob	Bigram prob
輕	青	不能 青 意 半途而廢	-238.49	-362.78
輕	情	不能 情意 半途而廢	-230.79	-360.48
:	:	……		
輕	逕	不能 逕 意 半途而廢	-239.45	-360.48
輕	經	不能 經意 半途而廢	-234.12	-360.48
:	:	……		
意	易	不能 輕易 半途而廢	☺ -229.78	-357.08

Table 1. Success example of finding errors by unigram probability

Original sub-sentence in Doc#00002			Unigram prob	Bigram prob
突然 一 陣 巨 晃 ， 我們 家書 貴 倒 了			-262.14	-385.31
Org	Rpl	Segmentation	Unigram prob	Bigram prob
晃	星	突然 一 陣 巨星 ， 我們 家書 貴 倒 了	☹-256.13	-385.31
貴	櫃	突然 一 陣 巨 晃 ， 我們 家 書櫃 倒 了	-257.76	-385.31

Table 2. Failure example of finding errors by unigram probability

Original sub-sentence in Doc#00001			Unigram prob	Bigram prob
不 怕 措 折 地 奮鬥			-201.12	-308.93
Org	Rpl	Segmentation	Unigram prob	Bigram prob
怕	必	不必 措 折 地 奮鬥	-192.34	-308.93
措	挫	不 怕 挫折 地 奮鬥	-193.31	☺ -305.53
地	抵	不 怕 措 折抵 奮鬥	-198.82	-308.93

Table 3. Success example of finding errors by bigram probability

Original sub-sentence in Doc#00046			Unigram prob	Bigram prob
…都 有 它 的 蹤 蹟 ， 可以 算是 每 個人 長大 的 東西			-280.11	-405.72
Org	Rpl	Segmentation	Unigram prob	Bigram prob
蹟	跡	… 公 車站 等 都 有 它 的 蹤跡	-273.85	-399.71
每	個	可以 算是 個 個人 長大 的 東西	-268.56	☹ -399.06

Table 4. Failure example of finding errors by bigram probability

Error detection is evaluated by the following metrics:

False-Alarm Rate = # of sentences with false positive error detection results / # of testing sentences without errors

Detection Accuracy = # of sentences with correctly detected results / # of all testing sentences

Detection Precision = # of sentences with correctly error detected results / # of sentences the system return as with errors

Detection Recall = # of sentences with correctly error detected results / # of testing sentences with errors

Detection F-Score= (2 * Detection Precision * Detection Recall) / (Detection Precision + Detection Recall)

Error Location Accuracy = # of sentences with correct location detection / # of all testing sentences

Error Location Precision = # of sentences with correct error locations / # of sentences the system returns as with errors

Error Location Recall = # of sentences with correct error locations / # of testing sentences with errors

Error Location F-Score= (2 * Error Location Precision * Error Location Recall) / (Error Location Precision + Error Location Recall)

Error correction is evaluated by the following metrics:

Location Accuracy = # of sentences correctly detected the error location / # of all testing sentences

Correction Accuracy = # of sentences correctly corrected the error / # of all testing sentences

Correction Precision = # of sentences correctly corrected the error / # of sentences the system returns corrections

We submitted 3 dry runs and 3 formal runs based on different system settings. The settings and evaluation results are described as follows.

3.1 Dry run evaluation

We submitted 3 dry runs in this Bake-off. The first run used only visually similar characters. The second run used only phonologically similar characters. And the third run used both kinds of similar characters. All three runs used bigram probability to detect errors.

Table 5 and 6 illustrate the evaluation results of dry runs in Subtask 1 and Subtask 2. (Evaluation results of Dryrun3_NTOU in Subtask 2 will be provided in the camera-ready version.) As we can see, using only phonologically similar characters achieve better F-scores than other strategies.

3.2 Formal run evaluation

We submitted 3 formal runs in this Bake-off. The first run used unigram probability while the other runs used bigram probability to detect errors. Besides, preference and filtering rules were applied only on the first run and the third run. All three runs used all similar characters to do the replacement.

Table 7 and 8 illustrate the evaluation results of formal runs in Subtask 1 and Subtask 2. As we can see, using bigram probability without preference and filtering rules achieve the best performance.

4 Conclusion

We submitted 3 dry runs and 3 formal runs based on different system settings. The evaluation results show that using bigram probability without preference and filtering rules achieve the best performance. Besides, phonologically similar characters are more useful than visually similar characters.

In the future, more features should be investigated. Errors of misusing one word into

References

R.C. de Amorim and M. Zampieri. 2013. “Effective Spell Checking Methods Using Clustering Algorithms,” *Recent Advances in Natural Language Processing*, 7-13.

C. Blair. 1960. “A program for correcting spelling errors,” *Information and Control*, 3:60-67.

A. Carlson, J. Rosen, and D. Roth. 2001. “Scaling up context-sensitive text correction,” *Proceedings of the 13th Innovative Applications of Artificial Intelligence Conference*, 45-50.

A. Carlson and I. Fette. 2007. “Memory-Based Context-Sensitive Spelling Correction at Web Scale,” *Proceedings of the 6th International Conference on Machine Learning and Applications*, 166-171.

C.H. Chang. 1994. “A pilot study on automatic chinese spelling error correction,” *Journal of Chinese Language and Computing*, 4:143-149.

Q. Chen, M. Li, and M. Zhou. 2007. “Improving

Run	FAlarm	DetcAcc	DetcP	DetcR	DetcF	LocAcc	LocP	LocR	LocF
Dryrun1_NTOU	**0.475**	**0.600**	0.321	0.900	0.474	**0.440**	0.036	0.010	0.053
Dryrun2_NTOU	0.525	0.580	0.323	1.000	**0.488**	**0.440**	0.097	0.300	**0.146**
Dryrun3_NTOU	0.700	0.440	0.263	1.000	0417	0.280	0.053	0.200	0.083

Table 5. Dry run performance in Subtask 1

Run	LocAcc	CorrAcc	CorrP
Dryrun1_NTOU	0.320	0.220	0.225
Dryrun2_NTOU	0.500	0.380	0.388
Dryrun3_NTOU	---	---	---

Table 6. Dry run performance in Subtask 2

Run	FAlarm	DetcAcc	DetcP	DetcR	DetcF	LocAcc	LocP	LocR	LocF
Formalrun1_NTOU	**0.980**	**0.314**	0.304	**1.000**	0.467	0.109	0.096	0.317	0.148
Formalrun2_NTOU	0.943	0.338	0.311	0.993	0.474	**0.149**	**0.114**	**0.363**	**0.173**
Formalrun3_NTOU	0.926	0.350	**0.315**	0.993	**0.478**	0.135	0.088	0.277	0.133

Table 7. Formal run performance in Subtask 1

Run	LocAcc	CorrAcc	CorrP
Formalrun1_NTOU	0.324	0.279	0.279
Formalrun2_NTOU	**0.371**	**0.311**	**0.312**
Formalrun3_NTOU	0.318	0.268	0.269

Table 8. Formal run performance in Subtask 2

Query Spelling Correction Using Web Search Results", *Proceedings of the 2007 Conference on Empirical Methods in Natural Language* (*EMNLP-2007*), 181-189.

Y.Z. Chen, S.H. Wu, P.C. Yang, T. Ku, and G.D. Chen. 2011. "Improve the detection of improperly used Chinese characters in students' essays with error model," *Int. J. Cont. Engineering Education and Life-Long Learning*, 21(1):103-116.

S. Cucerzan and E. Brill. 2004. "Spelling correction as an iterative process that exploits the collective knowledge of web users," *Proceedings of EMNLP*, 293-300.

F. Damerau. 1964. "A technique for computer detection and correction of spelling errors." *Communications of the ACM*, 7:171-176.

S. Deorowicz and M.G. Ciura. 2005. "Correcting Spelling Errors by Modelling Their Causes," *International Journal of Applied Mathematics and Computer Science*, 15(2):275-285.

A. Golding and D. Roth. 1999. "A winnow-based approach to context-sensitive spelling correction," *Machine Learning*, 34(1-3):107-130.

A. Islam and D. Inkpen. 2009. "Real-word spelling correction using googleweb 1t 3-grams," *Proceedings of Empirical Methods in Natural Language Processing* (*EMNLP-2009*), 1241-1249.

M. Li, Y. Zhang, M.H. Zhu, and M. Zhou. 2006. "Exploring distributional similarity based models for query spelling correction," *Proceedings of the 21st International Conference on Computational Linguistics and 44th Annual Meeting of the ACL*, 1025-1032.

W. Liu, B. Allison, and L. Guthrie. 2008. "Professor or screaming beast? Detecting words misuse in Chinese," *The 6th edition of the Language Resources and Evaluation Conference*.

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*, 10(2), 10:1-39.

R. Mitton. 1996. *English Spelling and the Computer*, Harlow, Essex: Longman Group.

R. Mitton. 2008. "Ordering the Suggestions of a Spellchecker Without Using Context," *Natural Language Engineering*, 15(2):173-192.

T. Pirinen and K. Linden. 2010. "Creating and weighting hunspell dictionaries as finite-state automata," *Investigationes Linguisticae*, 21.

S. Verberne. 2002. *Context-sensitive spell checking based on word trigram probabilities*, Master thesis, University of Nijmegen.

C. Whitelaw, B. Hutchinson, G.Y. Chung, and G. Ellis. 2009. "Using the Web for Language Independent Spellchecking and Autocorrection," *Proceedings Of Conference On Empirical Methods In Natural Language Processing* (*EMNLP-2009*), 890-899.

S.H. Wu, Y.Z. Chen, P.C. Yang, T. Ku, and C.L. Liu. 2010. "Reducing the False Alarm Rate of Chinese Character Error Detection and Correction," *Proceedings of CIPS-SIGHAN Joint Conference on Chinese Language Processing* (*CLP 2010*), 54-61.

L. Zhang, M. Zhou, C.N. Huang, and H.H. Pan. 2000. "Automatic detecting/correcting errors in Chinese text by an approximate word-matching algorithm," *Proceedings of the 38th Annual Meeting on Association for Computational Linguistics*, 248-254.