COMP90049 Project - 1 Report: Waht kinda typoz do poeple mak?

1. Introduction

The aim of this report is to identify the reasons why people make typographical errors by implementing approximate string search methods over several test data. This report will firstly propose some hypotheses based on related literature, then use and adjust different approaches to test those hypotheses.

2. Dataset

The dataset includes a list of common spelling mistakes ("Wikipedia: Lists of common misspellings," n.d.), which involves 4453 misspellings as well as corresponding correct spellings. The dataset also contains a dictionary which is a list of 370,099 English entries. In particular, there are 62 correct spellings are not present in the dictionary and 322 misspellings are present in it (each misspelling is an English word but it is not the intended word).

2. Hypotheses

By analyzing related literature and life experiences, the causes of misspellings probably could be separated to two kinds.

1. The first kind is that the author knows the correct spelling but makes mistakes due to accidentally hitting wrong key on the keyboard, or hitting in wrong orders or skipping some keys when typing fast.

2. The second kind is that the author possibly does not know the correct spellings. The connection between orthography and phonetics in English is complicated and inconsistent (Khansir & Tajeri, 2015). Therefore, words with similar pronunciations could be confused frequently.

3. 1. Levenshtein Distance

3.1.1. Basic

The principle of global edit distance is suitable to correct the first kind of misspelling. Hitting wrong keys could be corrected by DELETE and REPLACE, and missed keys could be added by INSERT. Moreover, edit distance also could be used to correct mistakes caused by phonetic because similar pronunciation words usually have similar orthography (Zobel & Dart, 1996).

The implementation uses the initial Levenshtein parameters: Match (0), Insert/Delete/Replace (+1). For each word in the misspelling list, calculate its distance with every entity in the dictionary, and return all entities which have the same distance as the minimum distance. The code is written in C++ by referring lecture slides (Nicholson, Zobel, & Verspoor, 2018).

3.1.2. Results

The results are presented in the table below:

Precision	0.260
Recall	0.790
Avg. predicted numbers	3.035
Avg. respond time per word	0.459 s

Table 1: The results of the Levenshtein Distance

3.1.3. Evaluation

Levenshtein Distance has a fairly recall rate, but the precision rate seems to be improvable due to it returns multiple predicted words each time.

According to results, typographical errors could be fixed by insert, delete or replace characters. There are some examples:

Misspelling	Prediction	Type
abandonned(×) abandoned(√)	abandoned	extra character
absail(×) abseil(√)	abseil absoil assail	wrong key/similar sounding
inumerable(×) innumerable(√)	enumerable innumerable numerable	missed character
innumerable(×) enumerable(√)	enumerable innumerable numerable	wrong key/similar sounding
councellor(×) councilor(√)	councillor counsellor	similar sounding

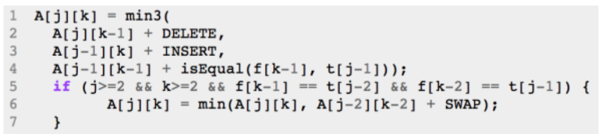
Table 2: The examples of the Levenshtein Distance

Therefore, the two hypotheses about the causes of typographical errors could be reasonable. However, it still has some limits which will discuss next.

4. 1. Improved Global Edit Distance

4.1.1. Basic

By analyzing test data, there is a kind of misspellings appear frequently, while it is hard to handle by Levenshtein Distance. When two letters' order is reserved, such as next and enxt, it would need 2 operations in Levenshtein Distance method. In order to solve it, the program "ImprovedGED.cpp" adds one operation SWAP to the origin algorithm. Instead of only four operations: INSERT, DELETE, REPLACE or MATCH, A[j][k] could be A[j-2][k-2] + SWAP if the last two characters from the two strings are reserved. The parameters are Match (0), Insert/Delete/Replace/Swap (+1).



```
A[j][k] = min3(
  A[j][k-1] + DELETE,
  A[j-1][k] + INSERT,
  A[j-1][k-1] + isEqual(f[k-1], t[j-1]));
  if (j>=2 && k>=2 && f[k-1] == t[j-2] && f[k-2] == t[j-1]) {
        A[j][k] = min(A[j][k], A[j-2][k-2] + SWAP);
  }
```

Figure 1: The key code of the Improved GED

4.1.2. Results

The results are presented in the table below:

Precision	0.334
Recall	0.856
Avg. predicted numbers	2.561
Avg. respond time per word	0.495 s

Table 3: The results of the Improved GED

4.1.3. Evaluation

Both precision and recall improve due to adding SWAP operation, which could prove that hitting in wrong order is a real cause of typographical errors. There are some examples:

Misspelling	Correct	Prediction
addres	adders	adders addles address addrest adores
adn	and	abn ad ada … and …
enxt	next	Ext next pnxt

Table 4: The examples of the Improved GED

5. 1. Soundex

5.1.1. Basic

However, Global Edit Distance is still less sensitive to phonetics. To test the second hypotheses, Soundex will be used in this section. Soundex is a well-known phonetic algorithm which can translate strings according to pronunciations (Zobel & Dart, 1996).

The implementation of Soundex refers lecture slides (Nicholson, Zobel, & Verspoor, 2018) and written in C++. Firstly, find the Soundex code for every word in the dictionary and the misspelling list, then find the Levenshtein Distance between each misspelling's Soundex code and each dictionary entity's Soundex code. Finally, return all entities which have the shortest distance.

5.1.2. Results

The results are presented in the table below:

Precision	0.003
Recall	0.811
Avg. predicted numbers	271.722
Avg. respond time per word	0.085 s

Table 5: The results of the Soundex

5.1.3. Evaluation

Soundex has a satisfied recall rate, but its precision rate is unacceptable. The method returns almost 272 predicted words on average for each misspelling. As Soundex only preserve at most four characters, there are a large number of words which have the same Soundex codes.

Misspelling	Correct	Prediction
abandonned	abandoned	aband abandon abandonable abandoned abandonedly …
absail	abseil	abacli abaculi

		abaculus … <u>abseil</u> …
inumerable	enumerable	iminourea immemorable immemorial immemorially … (╳)
councellor	councilor	cahenslyism camachile camisole … <u>councilor</u> …

Table 6: The examples of the Soundex

The data shows that Soundex has possibilities to cover the phonetic shortages of Levenshtein Distance. In contrast, Soundex is less sensitive to lapses made during typing.

6. 1. Hybrid Method

6.1.1. Basic

According to the analysis above, Improved GED method and Soundex are respectively suitable to hypotheses one and two. It could be helpful if we combine the two methods together.

The program “GEDSoundex.cpp” is written in C++. It firstly uses Improved GED to find the predicted collection which have the minimum distance. To reduce the number of predicted words, the system ranks the collection by applying Soundex algorithm, then return all ties.

6.1.2. Results

The results are presented in the table below:

Precision	0.530
Recall	0.805
Avg. predicted numbers	1.519
Avg. respond time per word	0.492 s

Table 7: The results of the Hybrid Method

6.1.3. Evaluation

The precision of the Hybrid Method shows a significantly improvement, though the recall rate goes down slightly due to the limited predictions.

In most cases, Improved GED and Soundex have a good cooperation. There are some examples:

Word	ImprovedGED Prediction	After Soundex
aberation (╳) aberration(√)	aberration aeration	<u>aberration</u>

abondon(╳)
abandon(√)
	abandon
	bondon
		<u>abandon</u>

emmited(╳)
emitted(√)
	ammites
	edited
	emailed
	emerited
	emitted
	emmies
	… (total: 18)
		<u>emitted</u>
		emoted
		endited

Table 8: The examples of the Hybrid Method

However, as Soundex is less applicable to the first hypotheses, it may cause problems in some cases. In addition, the algorithm of Soundex has some limitations (Zobel & Dart, 1996). For example, if the first letter is different, such as knight and night (Nicholson, Zobel, & Verspoor, 2018), it is almost impossible to match two strings by Soundex. Therefore, Soundex could lead to a wrong space:

Word	ImprovedGED Prediction	After Soundex
enlish (╳) english(√)	<u>english</u> enlist	enlist (╳)
envolutionary (╳) evolutionary (√)	<u>evolutionary</u> involutionary	involutionary (╳)

Table 9: Some wrong examples of the Improved Method

7. Analysis

The final results of each method are presented in table above:

	Precision	Recall
Levenshtein Distance	0.260	0.790
Improved GED	0.334	0.865
Soundex	0.003	0.811
Hybrid Method	0.530	0.805

Table 10: Results summary

Both pure Levenshtein Distance and Improved GED have fairly precision and recall. Although Soundex has some algorithm shortages, it still has a high recall rate which could prove that misspellings could be caused by similar pronunciations. As the Hybrid Method has the best performance, both hypotheses could be confirmed.

8. Conclusions

In conclusion, this report proposed two hypotheses. The first one is that misspelling is caused by typing lapses, and it could be corrected as well as proved by Levenshtein Distance and Improved GED. The second hypotheses is that phonetics is another reason why typographical errors happen. The performance of Soundex and the Hybrid Method could help to confirm its exist.

References

Nicholson, J., Zobel, J., & Verspoor, K. (2018) COMP90049: Approximate Matching, week 3 notes [PowerPoint slides]. Retrieved from https://app.lms.unimelb.edu.au/bbcswebdav/pid-6595057-dt-content-rid-31507421_2/courses/COMP90049_2018_SM2/lectures/05-06-approx.pdf

Khansir, A. A., & Tajeri, M. (2015). The relationship between spelling and pronunciation in English language. *Language In India*, (12). 57.

Wikipedia contributors. (n.d.). Wikipedia: Lists of common misspellings. *Wikipedia: The Free Encyclopedia*. Retrieved from https://en.wikipedia.org/w/index.php?title=Wikipedia:Lists_of_common_misspellings&oldid=813410985

Zobel, J., & Philip, D. (1996). Phonetic String Matching: Lessons from Information Retrieval. *Proceedings of the Eighteenth International ACM SIGIR Conference on Research and Development in Information Retrieval*. Zu¨rich, Switzerland. pp. 166 – 173.