COMP90049 Project 1 Report:

Waht kinda typoz do poeple mak?

# 1 Introduction

Approximate string search algorithms are broadly applied as spelling correction methods to resolve typographical errors in languages which are omnipresent. This project works on lists of misspelt words to get them automatically corrected using approximate string search approaches.

The report discusses several implementations of these methods along with the evaluation towards them according to related experiments. The main datasets involved in the project include:

*wiki\_misspell.txt* and *wiki\_correct.txt*: Originally from Wikipedia: Lists of common misspellings 1, both consisting of 4453 words in identical alphabetical sequences. (Hereinafter referred to as *Wiki Data*)

*birkbeck\_misspell.txt* and *birkbeck\_correct.txt*: Originally from Birkbeck spelling error corpus in University of Oxford Text Archive 2, containing 34683 entries. (Hereinafter referred to as *Birkbeck Data*)

*dict.txt*: A dictionary of the study comprising 370099 English words in alphabetical order, each existing only once. (Hereinafter referred to as Dictionary)

# 2 Methodologies

There are numerous existing researches regarding to approximate string search. Edit distance (EDIT) counts the minimum number of edits required to transform one string into the other. It can be further classified as global (GED) and Local (LED) 3 The calculation of N-Gram Distance also aims to choose among the similarities between strings for the optimal result of correction. 4 There are also Phonetic String Matching algorithms (e.g. Soundex) which can detect phonetic spelling mistakes due to similar pronunciations between two words. 5

In this project we are implementing spelling correction discussed below.

## 2.1 Global edit distance

In this project we choose Levenshtein distance as the implementation, which is a most known global edit distance. The standard scores specified for Levenshtein distance are: *Match* (0), *Insert*/*Delete*/*Replace* (+1). For each misspelt character in the list**,** we calculate its Levenshtein distances with every word in the Dictionary and choose the entry with minimum distance as the optimal match.

## 2.2 N-gram distance

Bigram distance is applied in this project, which means substrings are of length 2. Same as Section 2.1, we calculate bigram distances between a misspelt word with every entry in Dictionary, and finally pick the candidate(s) with minimum bigram distance.

## 2.3 Soundex

The Soundex algorithm is articularly implemented in this project to test *Birkbeck Data*, which contains phonetic typographical errors. Firstly, we translate each entry in misspelt list and dictionary respectively into the 4-digit code defined by Soundex. Then for each word in misspelt list, pick entries as a list from the translated dictionary with the identical code. Therefore, further select words with the minimal global edit distance to the misspelt word to narrow the range of candidates.

# 3 Hypotheses

According to the initial investigations over the typos in *Wiki Data*, combined with B. John Oommen’s research 6, the typographical errors in this study can be classified into basically four types, which are:

* Insertion, with unnecessary character(s) inserted or duplicated. E.g. abandonned (abandoned)
* Omission, with necessary character(s) omitted. E.g. abilties (abilities)
* Transposition, with two characters swapped unexpectedly. E.g. abritrary (arbitrary)
* Substitution, with some character(s) wrongly replaced by other character(s). E.g. acadamy (academy)

The observation to Birkbeck Data indicates that it contains a considerable portion of phonetic error apart from the four types above.

* Phonetic errors, usually with words whose pronunciations are quite similar. E.g. board (bored)

Certainly, different combinations of these errors are found in the two datasets and will be revealed and discussed later.

The experiments are designed based on the hypothetic errors above.

# 4 Experiments and results

Based on the characteristics of the test data, I’ve applied Levenshtein distance and N-Gram distance to *Wiki Data* because of the absence of phonetic errors, whereas we bring Soundex to *Birkbeck Data* to evaluate the performance of phonetic matching techniques, using Levenshtein distance for comparison.

Occasionally there are situations where multiple predictions with same distances. As there are no extra information provided by Dictionary (frequency, context, etc.), here we just keep them as a list of candidates to be evaluated by the precision metrics. Therefore, this study uses **precision** and **recall** in terms of evaluation metrics of the approximate string search algorithms. Precision is used to represent the proportion of correct predictions among all the candidates generated. Recall stands for the proportion of correct predictions out of the correct words list.

The results of experiments are illustrated in Tab.1 and Tab.2.

Tab.1 Evaluation of *Wiki Data*

|  |  |  |
| --- | --- | --- |
|  | Precision | Recall |
| Levenshtein Distance | 0.3984 | 0.8739 |
| N-Gram Distance | 0.5733 | 0.7748 |

Tab.2 Evaluation Birkneck Data

|  |  |  |
| --- | --- | --- |
|  | Precision | Recall |
| Soundex | 0.1571 | 0.4296 |
| N-Gram | 0.2142 | 0.3584 |
| Levenshtein Distance | 0.0902 | 0.4591 |

As Tab.1 implies, for *Wiki Data*, N-Gram distance outperforms Levenshtein distance in precision but has a relatively lower recall than Levenshtein distance. Nevertheless, the results in Tab.2 given by Soundex algorithm and Levenshtein distance on the Birkneck dataset are both not satisfactory, probably as a result of the much larger portion of phonetic/multiple errors. However, it indicates a better precision from Soundex than Levenshtein distance (with a slightly lower recall). Thus, the phonetic algorithm does seem to have a positive impact on the spelling correction of phonetic errors.

With the assistance of Needleman-Wunsch algorithm 7, it is possible to record the operations that transform a spelling to another spelling. And it can provide evidence of the way people misspell a word. Therefore, an additional experiment is designed to aggregate the types of typos people make in their spelling in terms of *Wiki Data* and *Birkbeck Data*. The results are shown in Tab.3 and Tab.4 respectively.

Tab.3 Types of typos in *Wiki Data*

|  |  |  |
| --- | --- | --- |
| Type | Count | Percentage |
| Insertion | 809 | 18.17% |
| Deletion | 1509 | 33.89% |
| Disposition | 723 | 16.24% |
| Substitution | 894 | 20.08% |
| Multiple | 518 | 11.63% |

Tab.4 Types of typos in *Birkbeck Data*

|  |  |  |
| --- | --- | --- |
| Type | Count | Percentage |
| Insertion | 2528 | 7.29% |
| Deletion | 3836 | 11.06% |
| Disposition | 2989 | 8.62% |
| Substitution | 4347 | 12.53% |
| Phonetic/Multiple | 20983 | 60.5% |

As implied by Tab.3, the most common type of typos that people make is deletion of words – making up 33.89% of all typos provided by *Wiki Data*. Additionally, there are situations people make more than one mistake in their spelling, whose proportion has reached 11.63%. In contrast, the phonetic/multiple errors make up a much bigger portion in *Birkbeck Data* (60.5%), and the most frequent mistake is substitution (12.53%) though not largely prior to other types of typographical errors.

The different levels of precision and recall between the two datasets indicate that the proportion of multiple/phonetic errors can have a significant negative effect on the performance of approximate string search algorithms.

# 5 Critical analysis

According to the experiments, the types of typographical errors people usually make are in line with the hypotheses proposed previously – that is, *insertion*, *deletion*, *disposition* and *substitution*, plus *phonetic errors* particularly in *Birkbeck Dataset*.

Levenstein distance and N-Gram distance have achieved relatively high performance in non-phonetic spelling correction. However, they don’t work well with phonetic misspellings, whereas Soundex can improve the precision of the operations by checking the pronunciation of words.

Phonetic errors are comparably hard to identify, mainly because some misspelt words might be legal and can be sought from the dictionary, which means that word will definitely be picked as the optimal candidate. For example, the misspelt word list of *Birkbeck Data* begins with more than ten “a”s, each originated from different correct words (above, all, an, and, answer, any, as, at, etc), which are extremely hard to source. Moreover, as the testing data are merely single words without contextual implication, the entire prediction is based on the similarities between two words with no additional information. Therefore, the precision and recall of the result are limited and presumably not as satisfactory as those in terms of paragraphs or articles.

The algorithms implemented in this study are quite fundamental ones. There are variances and improvements of these methods which are not included in this analysis task.

# 6 Conclusion

This report argues that the typos people make in the given datasets are classified into *insertion*, *deletion*, *disposition*, *substitution* and *phonetic errors*. Then it uses a series of experiments to correct the spellings with the approximate string search methods, as well as investigate the correctness of the hypotheses of typos. As a result, the results of the experiments are in line with the hypotheses proposed in Section 3.

Critically speaking, the experiments can be improved by implementing more advanced algorithms and using more informative datasets.

# 7 References

1. Wikipedia contributors (n.d.) Wikipedia:Lists of common misspellings. In Wikipedia: The

Free Encyclopedia, https://en.wikipedia.org/w/index.php?title=Wikipedia:

Lists\_of\_common\_misspellings&oldid=813410985.

2. Mitton, Roger (1980) Birkbeck spelling error corpus. In University of Oxford Text Archive,

http://ota.ox.ac.uk/headers/0643.xml.

3. Levenshtein, Vladimir I. (1966). “Binary codes capable of correcting deletions, insertions, and reversals”. Soviet Physics Doklady 10 (8): 707710.

4. Kondrak, Grzegorz (2005). “N-Gram Similarity and Distance”. In Proceedings of the 12th international conference on String Processing and Information Retrieval (SPIRE’05), pp. 115-126, Buenos Aires, Argentina.

5. Zobel, Justin and Dart, Philip (1996). “Phonetic String Matching: Lessons from Information Retrieval”. In Proceedings of the 19th annual international ACM SIGIR conference on Research and development in information retrieval (SIGIR’96), pp. 166-172, New York, USA.

6. B. John Oommen (1995). “String alignment with substitution, insertion, deletion, squashing, and expansion operations”. Information Sciences, vol. 83, Issues 1–2, pp.89-107.

7. S. B. Needleman and C. D. Wunsch (1970). “A general method applicable to the search for similarities in the amino acid sequence of two proteins”. Journal of molecular biology, vol. 48, no. 3, pp. 443-453.