COMP90049 Project1 Report: Waht kinda typoz do poeple mak?

1. Introduction

People often make errors when they are typing the words which known as typographical errors. In this report, firstly, we will make some hypotheses to identify the causes of the typo errors based on the given dataset. Then, some approximate string matching methods such as edit distance algorithms, N-gram algorithms and phonetic algorithms will be implemented to evaluate the dataset. Finally, the results of the experiment will be used to analyse the hypothesis.

In this experiment, the dataset includes three files which are "dict.txt", "wiki_misspell.txt" and "wiki_correct.txt". "dict.txt" is a list of approximately 370k English words. "wiki_mispell.txt" has 4453 words which are commonly misspelled words made by Wikipedia editors. "wiki_correct.txt" has 4453 truly intended words corresponding to the words in the "wiki_mispell.txt"[1].

2. Typographical errors and spelling correction

2.1 Typographical error

A typographical error is a mistake made in the typing process[3]. For example, the phrase "lost waller" will probably be a typographical error, while people may truly intend to type "lost wallet" instead.

2.2 Spelling correction

Many different solutions have been proposed to solve the spelling correction problem, as it is one of the oldest test processing problem. Modelling the causes of spelling errors directly and encode them in an algorithm is the most direct approach.[4] Many researchers have worked out many different spelling correction methods: dictionary look-up, Damerau-Levenshtein edit distance [5], Phonetic indexing algorithms[6], N-gram algorithms, etc.

3. Hypotheses of typo errors

There are many causes that make people make typographical errors in the real world, this may vary a lot with different target authors or environment. However, in this section, three hypotheses are made to identify what are the probable cause to the Wikipedia editors based on the Wiki dataset.

Hypotheses 1: Typo error was made by people that substitute a letter for a letter adjacent on the keyboard. For instance, people may intend to type "e" but instead type "r", as "e" and "r" are adjacent on the keyboard.

Hypotheses 2: Typo error was made because of people made some grammar mistake and used the incorrect format of the stem. For example, people may want to type "waiting" which is the progressive format of the stem "wait". However, a simple past format "waited" was typed. This kind of typo has the correct stem part and the misspell was caused due to the chosen of its wrong inflected variants.

Hypotheses 3: Typo error was made due to the phonetic attempts at spelling or guessing. For instance, people may want to type a word that is very long and rare to use. As the people only know how to read that word, then he/she may guess the spelling of the word based on its phonetic and this may probably cause a typographical error.

4. Approximate matching methods Implementation

In this section, 6 approximate matching methods will be introduced and they are all implemented in the code.

4.1 Edit distance Algorithms

method 1: Global Edit Distance

Transform the string of interest to each dictionary entry using the operation insert, delete, replace and match. Each operation is associated with a score, the best match is the word with best score.[7] The algorithm was shown below:

Algorithm: Global Edit Distance

```
lf = strlen(f); lt = strlen(t);
A[0][0]=0;
for (j=1; j<=lt; j++) A[j][0] = j * i;
for (k=1; k<=lf; k++) A[0][k] = k * d;

for (j=1; j<=lt; j++)
  for (k=1; k<=lf; k++)
   A[j][k] = max3( //Or min3 if m<i,d,r
      A[j][k-1] + d, //Deletion
      A[j-1][k] + i, //Insertion
            A[j-1][k-1] + equal(f[k-1],t[j-1]));
//Replace or match
```

equal() returns *m* if characters match, *r* otherwise Final score is at $A[lt][lf]$

---

method 2: Local Edit Distance

Local Edit Distance is similar to Global Edit Distance, it looks for the best match substring[7].

4.2 N-Gram Algorithms

method 3: Bigram Algorithms

method 4: Trigram Algorithms

The n-gram of a word is all the overlapping n-letter sequences in the word[8]. For instance, the 2-gram of word "crat" are {#c, cr, ra, at, t#}. The n-gram is calculated used the function below[7]:

$$|Gn(s)| + |Gn(t)| - 2 \times |Gn(s) \cap Gn(t)|$$

4.3 Phonetic Algorithms

Phonetic algorithms represent the words by their approximate "sounds like" pronunciation [4].

method 5: Soundex

Soundex is the best-known phonetic matching scheme and was developed by Odell and Russell in 1918[9].

Algorithm: Soundex

Four step process:

1. Except for initial character, translate string characters according to table
2. Remove duplicates (e.g. 4444 → 4)
3. Remove 0s
4. Truncate to four symbols

---

method 6: Metaphone

Metaphone was published by Lawrence Phillips in 1990. It fundamentally improves on the Soundex by using information about variations and inconsistencies in English spelling to produce a more accurate encoding.

5. Evaluation and Results

This section will evaluate the results of the six algorithms in section four based on the evaluation metrics precision and recall.

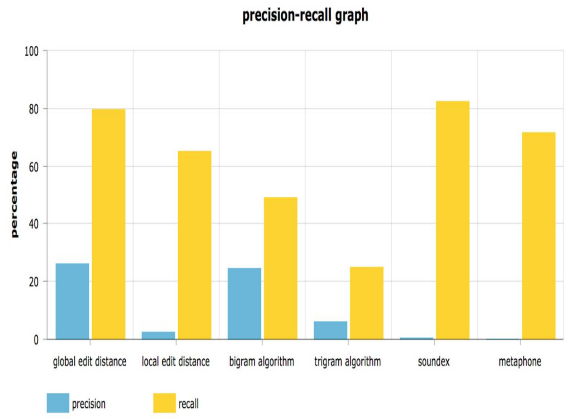
5.1 Evaluation Metrics

The evaluation metrics used in this session are the number of correct words, the number of predicted words, precision and recall. The precision is the fraction of correct responses among attempted responses[7]. The recall is the proportion of words with a correct response[7].

5.2 Results

Method	correct	predicted	total	precision	recall
1. Global Edit Distance	3550	13516	4453	26.27%	79.72%
2. Local Edit Distance	2897	122665	4453	2.36%	65.06%
3. Bigram algorithm	2185	8978	4453	24.34%	49.07%
4. Trigram algorithm	1117	18885	4453	5.91%	25.08%
5. Soundex	3658	1210744	4453	0.3%	82.15%
6. Metaphone	3184	1589251	4453	0.2%	71.5%

Table 1: Results of 6 Methods



Graph1: precision-recall of 6 methods

5.3 Illustrative Example

Method	misspelled word	correct word	predicted words
1. Global Edit Distance	abandonned	abandoned	abandoned
2. Local Edit Distance	abandonned	abandoned	abandoned abandonedly reabandoned unabandoned
3. Bigram algorithm	abandonned	abandoned	abandoned
4. Trigram algorithm	abandonned	abandoned	abandon abandoned
5. Soundex	abandonned	abandoned	abandonned aband abandon abandonable abandoned abandonedly…
6. Metaphone	abandonned	abandoned	abandoned abundant

Table2: Illustrative Example of one word

6. Analysis

This section will discuss how the results of each method provides evidence supporting the hypotheses in section 3.

6.1 Edit distance Analysis

By implementing and evaluating two edit distance algorithms which are global edit distance and local edit distance, the result in the evaluation metrics table which is shown in table 1 provides the evidence of the presence of Hypotheses 1. Hypotheses 1 is typo error was made by people that substitute a letter for a letter adjacent on the keyboard.

The reason that edit distance algorithms can evaluate this hypothesis is that the algorithms select the best match based on four operations: match, insert, delete, replace. If people type a wrong letter adjacent to it, it can just be one replace operation in the edit distance algorithm. The recall of Global Edit Distance is 79.72% which is a pretty high value and it indicates that 79.72% of the misspell dataset can find a correct response by using edit distance algorithms. Hence, it provides the evidence of the presence of hypotheses 1.

6.2 N-Gram Analysis

By implementing and evaluating the Bigram algorithm and trigram algorithm, the result in the evaluation metrics table can not provide the evidence of the presence of Hypotheses 2. Hypotheses 2 is typo error was made because of people has some grammar mistake to use the correct format of the stem.

As N-Gram only involves the comparison of the letters, it will select the word that has most parts in common. If hypotheses 2 type of error happens, for instance, using bigram algorithm to evaluate the misspelled word is “abandon” and the truly intended word is “abandoned”. Then as they both have {#a, ab, ba, an, nd, do, on}, I assume it will predict the word “abandoned”.

However, as both words “abandon” and “abandoned” are in the dictionary, bigram and trigram will just predict the word “abandon” as they are just exactly the same. Hence, they can not provide the evidence of hypotheses 2. In addition, bigram and trigram have a lower recall compared to other methods which indicate n-gram is not a good choice to correct the misspelled in this dataset case. In conclude, hypotheses 2 is hard to be evaluated in this dataset with these algorithms.

6.3 Phonetic Analysis

By implementing and evaluating two phonetic analysis algorithms which are Soundex and Metaphone, the result in the evaluation metrics table the evidence of the presence of Hypotheses 3. Hypotheses 3 is typo error was

made due to the phonetic attempts at spelling or guessing.

As the data shown in table 1, Soundex's recall is 82.15% and Metaphone has a 71.5% recall. Compared to other methods, these two phonetic methods have better performance on the recall. These two methods also generate more predicted words which make the precision lower. The good result of recall is a strong evidence that there exists the type of typo error due to the phonetic attempts at spelling.

7. Conclusion

To conclude, three hypotheses are made in this report and after evaluating by 6 approximate string matching algorithms, the results show the evidence supporting the presence of two hypotheses and absence of one hypotheses. It indicates that two causes of typographical errors made by wiki editors are: substitute a letter for a letter adjacent on the keyboard and phonetic attempts at spelling or guessing.

Different approximate string match methods have trade-offs among different evaluation metrics factor and efficiency. According to the table 1, the Soundex method has the best recall which is 82.15%. Correcting typographical errors is a knowledge task and it is very difficult to get a 100% recall. However, there are still improvements can be implemented if more time is given, such as by applying two algorithms together and they may generate better recall.

Reference

[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 [online, accessed 9-02-2018]

[2] M. Webster, "Typo-Definition", Free Merriam-Webster Dictionary, [online, accessed 9-02-2018]

[3] K. Kukich. 1992. Techniques for automatically cor- recting words in texts. *ACM Computing Surveys 24*, pages 377–439.

[4] C. Whitelaw, B. Hutchinson, G. Y. Chung, G. Ellis, "Using the web for Language Independent Spellchecking and Autocorrection" in *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing*, 2009, pp. 890-899.

[5] F.J. Damerau. 1964. A technique for computer detection and correction of spelling errors. *Communications of the ACM 7*, pages 171–176.

[6] K. Atkinson. 2009. Gnu aspell. In *Available at* http://aspell.net.

[7] J. Nicholson, J.Zobel, K. Verspoor, COMP90049. Class Lecture, Topic: "Approximate String Matching" Department of Computing and Information System, The University of Melbourne, Melbourne, VIC, Aug. 7, 2018.

[8] M. Du. 2005. "Approximate Name Matching". [online] Available: https://www.nada.kth.se/utbildning/grukth/exjobb/rapportlistor/2005/rapporter05/du_mengmeng_05137.pdf [Accessed: 9-02-2018].

[9] Z. Justin and P. Dart. (1996). Phonetic String Matching: Lessons from Information Retrieval. In *Proceedings of the Eighteenth International ACM SIGIR Conference on Research and Development in Information Retrieval*. Zu ¨rich, Switzerland. pp. 166–173.