COMP90049 Project 1 Report
waht wierd spelings! aer peaple carzy?

Anonymous

1 Introduction

The aim of this report is to evaluate the effectiveness of different spelling correction methods, namely Global Edit Distance (GED) and N-gram distance and similarity, on identifying the intended spelling of headwords taken from UrbanDictionary [1].

2 Dataset

The dataset has been taken from UrbanDictionary and curated for use by Saphra and Lopez (2016). The dataset refers to three separate lists of words written in the English language. The first list contains 716 headwords that have been identified as being misspelt. The second list contains the correct spelling and the third list includes around 400,000 tokens from the English language.

3 Evaluation Methodology

Throughout this report, each spelling correction method will be evaluated across four criteria:

- **Accuracy:** Fraction of correct responses for methods that only yield one prediction.
- **Precision:** Fraction of correct responses for methods that yield multiple predictions.
- **Recall:** Proportion of words with a correct response among the predictions yielded.
- **Time:** Seconds taken for the method to complete.

Accuracy is important because the principal concern of any spelling correction method is correctness. If a method were to yield a single predicted result, then that result should be correct. Precision is important when multiple results are equidistant from the misspelt word. Recall is important when the correct spelling can be resolved in the case of multiple predictions i.e. when a human user can confirm the intended spelling of a word. Finally, time is important because certain applications might be time sensitive i.e. instant messaging. Note that time is affected by the size of the dictionary more than method.

[1] http://urbandictionary.com

4 Methodology and Algorithms

This report assesses the effectiveness of 5 spelling correction methods. The first is Levenshtein's GED (L-GED)(Levenshtein, 1966). The second is another variant of GED named Damerau-Levenshtein's GED (DL-GED) (Damerau, 1964). The third is the N-gram Distance method (N-gram) (Kondrak, 2005) and the fourth and fifth are Hybrid methods combining GED and N-gram. All algorithms with the exception of N-gram Distance, use Thibault Debatty and Paul Irwin's stringsimilarity library which is written in java (Debatty and Irwin, 2017). The algorithm for N-gram was inspired by Ralph Rice's Sorenson's dice coefficient algorithm (Rice, 2013).

4.1 Global Edit Distance

The initial GED uses Levenshtein's parameters (m,i,d,r) = (0,1,1,1) to calculate the edit distance between each misspelt word and each entry in the dictionary (Levenshtein, 1966). DL-GED is another variant of GED and distinguishes itself from L-GED by including transpositions in the set of possible operations (Damerau, 1964).

Using **and** as **adn** in the dataset as an example, L-GED(adn, and) = 2 whereas DL-GED(adn, and) = 1. Transposition errors can be considered as 1 edit because they often result from rapid typing and DL-GED was included to prevent the over-penalization of transposition errors.

4.2 N-gram Distance

A shortcoming of GED algorithms is their insensitivity to context (Kondrak, 2005). Because they are concerned with the analysis of unigrams, matching characters that are spread apart are treated identically to matching characters that are grouped together. Take L-GED(natural, counts) = 6 and L-GED(natural, contrary) = 6 as an example. Most people consider contrary to be a closer match to natural than counts (due to the 'ra' subsequence) whereas L-GED considers them to be equidistant. N-gram distance somewhat remedies this problem by comparing subsequences.

However, due to N-gram's attention to substrings, it often loses sight of the bigger picture. It can miss strings that appear alike but lack common n-grams i.e. Verelan/Virilon and find perfect similarity in strings that are not identical i.e. Xanex/Nexan. To increase the sensitivity of N-gram Distance to characters at the beginning and ending of a string (characters that are important to human perception), a special character '#' is added to the beginning and ending of every misspelt word. Only results from the Bigram were included because it was the best performer among N-grams.

4.3 Hybrid Methods

From the discussion thus far, it is clear that GED suffers from weaknesses that are remedied by N-gram and vice versa. As a result, this paper proposes two Hybrid methods that combine GED and N-gram. Note that the tradeoff for adopting hybrid methods is time.

4.4 The Tiered Method

The Tiered method uses the DL-GED method to initially select the best matches. The N-gram method then selects the dictionary entries with the highest score from those matches. Because GED is expected to provide a better filter due to N-grams aggressive elimination, a Tiered method might offer the best of both worlds.

4.5 The Consensus Model

The Consensus method aggregates the scores from DL-GED and N-gram to select the best dictionary matches. If both GED and N-gram provide a suboptimal filter, then it would be better to aggregate the methods.

4.6 Other Methods Considered

A number of other methods were considered but omitted. Since we are interested in comparing words, Local Edit Distance is less relevant because it analyses subsequences. Phonetic methods like Soundex were not considered because the words were typed not spoken (Zobel and Dart, 1996). While it could be argued that difficult to spell words such as drug names can be misspelt because of their phonetics, there is little evidence in the dataset to suggest that this might be the case.

5 Results

Method	Accuracy	Time(s)
L-GED	0.147	51.4
DL-GED	0.161	359.2
Bigram	0.0879	194.7
Tiered	0.1844	590.6
Consensus	0.176	615.7

Table 1: Evaluation Criteria For Single Prediction Methods

Method	Precision	Recall	Time(s)
L-GED	0.0458	0.3534	85.4
DL-GED	0.0548	0.4120	589.0
Bigram	0.0984	0.2123	347.8
Tiered	0.1154	0.2318	588.9
Consensus	0.1112	0.2277	962.9

Table 2: Evaluation Criteria For Multi-Prediction Methods

Method	Avg Predictions
L-GED	7.72
DL-GED	7.52
Bigram	2.16
Tiered	2.01
Consensus	2.05

Table 3: Average Number of Predictions for Multi-Prediction Methods

6 Discussion

6.1 GED methods outperform N-gram in recall but not accuracy and precision

GED methods such as L-GED and DL-GED outperform N-gram in terms of recall accuracy but suffer in terms of precision. This is likely the result of N-gram's 'short sightedness' discussed

earlier; As a result of focusing on subsequences (of size n), N-gram aggressively eliminates other options whose characters may match but are further apart. The significantly lower number of average predictions (7.52 for DL-GED v.s. 2.00 for Bigram) is indicative of this occurring.

6.2 Padding penalises N-gram for shorter strings

Interestingly, padding penalises N-gram when it comes to comparing shorter strings. The following example is taken from the output from running N-gram distance. The misspelt word is **blar** and the correct word is **blah** which is present in the dictionary.

Predicted	Correct?
bar	✗
belar	✗
blair	✗
blare	✗
blart	✗
blear	✗
bolar	✗
lar	✗

Table 4: Predicted words from Bigram

Clearly, the padding is biasing the results in favour of words that begin with 'b' and end in 'r'. This problem is less significant for longer strings with more N-grams. It is also not present in GED methods where DL-GED and L-GED correctly predict **blah**.

6.3 Hybrid methods outperform in accuracy and precision but not recall

Hybrid models appear to improve in terms of precision and accuracy but are worse than GED methods in terms of recall. This is again likely the result of N-gram's aggressive elimination of alternative options as evidenced by the lower average number of predictions exhibited by the hybrid models.

6.4 Spelling correction methods are limited by the dictionary

There is little spelling correction methods can do when the correct word is absent from the referenced dictionary. In the dataset, there is a misspelt string **ofmg** where the correct string is **omfg**. All the methods discussed failed to identify the correct word because it is absent from the dictionary.

6.5 Spelling correction methods fail when the misspelt word is contained in the dictionary

Spelling correction methods are guaranteed to fail when the the misspelt word is contained in the dictionary. In the dataset, there is a misspelt string **oaky** where the correct string is **okay** but all the methods concluded that **oaky** was the correct spelling. A better method would take into account the commonality of a word.

7 Potential Improvements

Potential improvements include:

1. The methods can be quickened by reducing the search space with neighbourhood search (NS). Since misspelt strings are unlikely to deviate more than 2 edits from the correct spelling, NS might reduce the time taken dramatically. This might make Hybrid methods more attractive.
2. Dictionary entries can be further sorted based on their commonality. Recycling the earlier example, okay might be suggested as a replacement for oaky because it is more common.
3. Spelling correction methods can further take into account the position of keys on the keyboard. Methods could prioritise strings whose differing characters occupy neighbouring keys on the keyboard.

8 Conclusions

This report implements 5 spelling correction methods. The results show that GED methods are superior to N-gram and Hybrid methods in terms of recall. However, in situations where conflicts in predictions cannot be resolved, Hybrid methods provide a superior result due to their high precision. Ultimately, the results suffered because of deficiencies in the dictionary and it would be interesting to see what would have resulted with a better dictionary.

References

Fred J Damerau. 1964. A technique for computer detection and correction of spelling errors. *Communications of the ACM*, 7(3):171–176.

Thibault Debatty and Paul Irwin. 2017. java-string-similarity. `https://github.com/tdebatty/java-string-similarity/tree/master/src/main/java/info/debatty/java/stringsimilarity`.

Grzegorz Kondrak. 2005. N-gram similarity and distance. In *International symposium on string processing and information retrieval*, pages 115–126. Springer.

Vladimir I Levenshtein. 1966. Binary codes capable of correcting deletions, insertions, and reversals. In *Soviet physics doklady*, volume 10, pages 707–710.

Saul B Needleman and Christian D Wunsch. 1970. A general method applicable to the search for similarities in the amino acid sequence of two proteins. *Journal of molecular biology*, 48(3):443–453.

Ralph Rice. 2013. java-string-similarity. `https://github.com/rrice/java-string-similarity/blob/master/src/main/java/net/ricecode/similarity/DiceCoefficientStrategy.java`.

Naomi Saphra and Adam Lopez. 2016. Evaluating informal-domain word representations with urbandictionary. *Proceedings of the 1st Workshop on Evaluating Vector-Sace Representations for NLP*, pages 94–98.

Justin Zobel and Philip Dart. 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*, pages 166–172. ACM.