COMP90049 Knowledge Technologies Project 1 Report

Anonymous

1 Introduction

Lexical normalisation is the problem of finding a canonical form for each token within a document. The task of quantifying the similarity between two strings is required in numerous applications. The most well-known string approximation algorithms are based on simple measures such as edit-distance and n-gram similarity. In contrast, other forms of normalisation are based on phonetic similarity, rather than lexical similarity.

This report analyses different spelling correction methods for each token within a document. The document will correspond to short messages, i.e. tweets from the social media platform Twitter. The focus of this report will be to contrast two approximate string matching techniques, based on edit-distance. The main intent is to reveal whether Damerau-Levenshtein distance performs better than Levenshtein distance, with respect to computational efficiency and numerous evaluation methods.

2 Dataset

The data given is a collection of words composed of English alphabetical symbols, some standard English words, while many are non-standard lexical items, including non-alphanumeric characters. More specifically, the dataset, curated by [1] et. al, comprises of 10,322 misspelled token and corrected token pairs, lowercased, and a reference dictionary containing 370,099 possible matches, sorted alphabetically. For each misspelled token in the dataset, lexical normalisation was attempted by predicting the best possible matches with respect to the reference collection and a similarity metric.

3 String Similarity Metrics

Two variations of edit-distance and one variation of n-gram similarity have been used for lexical normalisation.

3.1 Edit Distance

In this section, we discuss the notion edit distance in the context of its applicability as a string similarity metric used for the aforementioned task.

3.1.1 Levenshtein Distance

Levenshtein distance [2] [6] (LD), with parameters (m,i,d,r) = (0,1,1,1) is a form of global edit distance. Informally, the Levenshtein distance between two words is the minimum number of single character edits (insertions, deletions, or replacements) required to change one word into another.

3.1.2 Damerau-Levenshtein Distance

An extension thereof is the Dameru-Levenshtein distance [3] (DLD). This differs from LD by including transposition of two adjacent characters among its allowable operations, in addition to the three classical single-character edit operations. For both the standard and extended variations, a lower value indicates greater similarity between two strings.

3.2 N-gram Similarity

N-gram similarity (n=2) was used to break ties where more than one match occurred. The reason for doing so was an attempt to increase precision, and decrease the number of incorrect matches returned.

4 Evaluation Metrics

For the purpose of this task, two different evaluation metrics have been utilised to analyse the performance of the system, including:

- Precision: the proportion of correct results returned, with respect to the total number of results returned
- Recall: the proportion of correct results returned, with respect to the total number of possible correct results

Here, the notion of accuracy is encapsulated in recall. Moreover, the efficiency of both systems was also evaluated, but little emphasis has been placed on this due to the variability of externally sourced implementations, and heterogeneity of the machines the implementation ran on.

5 Methodology & Implementation

In order to efficiently process the data, the algorithms were applied to each unique misspelled word. The LD and DLD [4] algorithms were compiled in C and imported into Python for increased efficiency. N-gram similarity was computed via the Python ngram library [5]. To capture the effect of the same misspelled words potentially being used in widely varying contexts, a decision was made to evaluate both edit distance normalisation techniques on the full set of 10,322 word pairs.

6 Results

The two tables below summarise the final results:

Precision	0.1962
Recall	0.7738
Avg. Matches	3.9447
Tokens/min	142.96

Table 1: LD results for 10,322 word pairs

Precision	0.1957
Recall	0.7741
Avg. Matches	3.9557
Tokens/min	23.46

Table 2: DLD results for 10,322 word pairs

7 Analysis

7.1 Overall Summary

From the results, it can be seen that LD and DLD are moderately accurate in returning the canonical form for a given misspelled word, with both having a recall of approximately 78%. However, both metrics exhibit low precision, with each returning an approximate average of 4 matches per misspelled token.

7.2 Similarities

Both edit distance metrics perform very similarly. With the algorithms essentially sharing three of four same allowable operations, and the denseness of the dictionary, these similarities in performance are not unprecedented. One key indicator that the algorithms perform very similarly is the fact that out of 3,755 unique misspelled words, 3,550 (94.5%) of them have the exact same set of matches for both LD and DLD.

As mentioned previously, both algorithms exhibit a fairly low precision. This could be due to the density of the dictionary, and the fact that many dictionary words share the same lowest edit-distance for a given misspelled word. For example, the misspelled word '2b806641' returns 7,823 potential matches when LD is used. Although this is an extreme example, it was common for many misspelled words to be returned a plethora of potential matches.

7.3 Differences

LD narrowily outperforms DLD with respect to precision, but DLD performs better with respect to recall. One reason for this could be the fact that DLD allows for adjacent character transpositions. Due to the nature of the misspelled words being derived from tweets, it can be argued that character transpositions are very likely to occur in some misspellings, and hence, allow the system to capture this property of tweets and reverse it during lexical normalisation. Examples of this phenomenon can be seen in Table 4.

A reason for DLD having lower precision than LD could be the fact that the allowable edit operations of LD are a subset of the operations of DLD. Hence, when DLD does not find matches with a lower shared edit distance than LD, DLD matches are generally a superset of the LD matches. Out of the 205 unique misspelled words that did not have the exact same set of matches for both LD and DLD, 200 (97.6%) of the LD matches were subsets of the corresponding DLD matches. An example of this is shown in Table 3 for the misspelled token 'katniss'. Due to additional matches for DLD, this slightly decreased its overall precision.

7.3.1 Behavioural Differences

An example showing the different behaviours of the two edit-distance algorithms is seen via the misspelled word 'lebron' and the dictionary term 'reborn'. The token 'reborn' was not included in the LD matches, since LD(lebron,

LD Matches	DLD Matches
catnips	catnips
fatness	fatness
kainits	kainits
patness	patness
warniss	warniss
	kantism
	kantist

Table 3: LD and DLD matches for misspelled token 'katniss' (minimum LD = minimum DLD = 2)

Misspelled Word	Canonical Form
liek	like
betetr	better
hsit	shit

Table 4: Correct predictions for DLD only

reborn) = 3, and the minimum LD = 2. However, 'reborn' is included in DLD matches, due to character transposition between the adjacent 'r' and 'o', allowing LDL to equal 2.

7.3.2 Illustrative Examples

There were zero instances where LD predicted the correct canonical form but DLD did not, and three unique misspelled and corrected word pairs, where DLD correctly predicted the canonical form, but LD failed. These are outlined in Table 4. In all three cases, the minimum LD was 1, whereas the LD between the misspelled and canonical words was 2. Hence, the canonical words were not included in the matches for LD. The DLD between each pair was 1, due to a single transposition.

8 Improvements

As both LD and DLD are relatively imprecise metrics with respect to the problem, an attempt was made to increase precision by using bigram similarity to break ties amongst multiple matches. Since DLD has higher recall but lower precision, this was only applied to DLD. The results from the attempt are summarised in the Table 5.

Hence, with bi-gram similarity breaking ties, precision has increased by 338.28%, whereas recall has only decreased by 0.54%. As a result of these observations, it can be seen that system performance can be improved tremendously via further modification and extension.

Precision	0.6637
Recall	0.7699
Avg. Matches	1.1599

Table 5: DLD & 2-gram similarity results for 10,322 word pairs

9 Conclusions

The use of approximate string matching methods certainly has a place in the task of lexical normalisation of short messages from social media. It is clear, however, that such rudimentary methods such as LD and DLD would not be the most suitable for approaching such tasks in real word applications. Instead, these methods may be used as baselines and extended to reduce the number of arbitary ties between matches, in order to greatly narrow down matching results to the most likely correct candidate tokens.

References

[1] Baldwin, Timothy, Marie Catherine de Marneffe, Bo Han, Young-Bum Kim, Alan Ritter, and Wei Xu, "Shared Tasks of the 2015 Workshop on Noisy User-generated Text: Twitter Lexical Normalization and Named Entity Recognition", *Proceedings of the ACL 2015 Workshop on Noisy User-generated Text*, Beijing, China, pp. 126–135, 2015.

[2] Yianilos N. Peter, and Ristad Sven Eric, "Learning String Edit Distance", *IEEE Transactions on Pattern Analysis and Machine Intelligence*, pp. 522-532, 1998.

[3] Muhammad Marwan Muhammad Fuad, Pierre-François Marteau, "The Extended Edit Distance Metric", *Content-Based Multimedia Indexing, CBMI 2008*, Jun 2008, London, United Kingdom. pp.242 - 248, ff10.1109/CBMI.2008.4564953ff.

[4] II Jensen M. James, Portable C99 Implementation of Damerau–Levenshtein Distance Algorithm for Strings, `https://github.com/badocelot`

[5] Poulter Graham, "Python N-Gram: Set that supports searching by ngram similarity", `https://github.com/gpoulter/python-ngram`

[6] Kondrak Grzegorz, "N-Gram Similarity and Distance", *Consens M., Navarro G. (eds) String Processing and Information Retrieval. SPIRE 2005. Lecture Notes in Computer Science, vol 3772. Springer*, Berlin, Heidelberg, 2005.