Lexical Normalisation for Twitter Data

Anonymous‡

University of Melbourne COMP90049 Knowledge Technologies

Abstract Lexical normalisation is widely used to preprocess Twitter data that is filled with misspellings and incorrect grammar structures into an appropriate canonical form useful for further analysis in Natural Language Processing (NLP) tools. This paper will aims contrast the performance of various approximate string matching techniques most effective for lexical normalisation.

1 Introduction

The vast amounts of data from Twitter would only be useful for NLP provided it is in it 's canonical form. With Twitter 's 140 character limit, the restriction has caused user 's to innovate shorthand notations, slang and other grammatically incorrect words in order to ' tweet '. The various string matching techniques we will examine the performance on include Levenshtein(L), and phonetic algorithms such as Soundex, Phonex, and New York State Identification and Intelligence System (NYSIIS).

NYSIIS gives better results relatively to the original Soundex as it uses more sophisticated rules in transformation of the final phonetic code. Phonex is also a modification of Soundex and accounts for a few more letter combinations to improve accuracy on some data sets.

The hypothesis of this work is that Levenshtein coupled with a phonetic algorithm will outperform Levenshtein alone.

2 Related Works

Han and Baldwin proposes to improve ill-formedword detection classifier by introducing an Out Of Vocabulary (OOV) word whitelist. Ahmed extends the former 's research by processing tokens with various whitelists, then applying the refined soundex algorithm, Peter Norvig ' s Algorithm and if more than one result is returned in the phonetic match, it will be processed with the 5-Gram Context Matching technique. In the deep-learning space, Rohatgi and Zare proposes ' DeepNorm ' to train a Recurrent Neural Network (RNN) to learn the correct normalisation function (Rohatgi, Zare, 2017) .

3 Data Set

The data set was originally curated from Workshop on Noisy User-generated Text(Han, Baldwin, 2015). A total of three datasets were used in this research; a dictionary containing 370099 words (dict.txt) ; a list of misspelled words (misspell.txt) and ; a list of its respective correct form (correct.txt), both containing 10322 words. All files contain tokens separated line by line.

4 Methodologies

4.1 Data Processing

The misspell.txt and correct.txt is processed to return a list of 3,755 unique misspelled words and 3,516 unique corrected words.

The dictionary file contains all unique words. Due to the heavy computations that result from looping through each token, the reduction to unique words significantly reduced the execution time of the program.

4.2 Global Edit Distance

Global Edit Distance (GED), quantifies the dissimilarity between two strings. It is a measure of the minimum number of edit operations to transform one string into the other (Christopher, Prabhakar, & Hinrich, 2008, p. 58). Edit operations include Insertion, Deletion and Substition of a single symbol in the string and has a positive cost of 1. The lower the GED, the smaller the distance between two words and the more similarity they share.

In the main program, Levenshtein distance is calculated between each token from the unique list of misspelled words and the dictionary and the results are stored in a new list. The new list is then sorted in ascending order and the lowest distance value will be used firstly to compare if it 's within the bounds of two (inclusive) edit distance and secondly, to return a list of predicted words that are within that distance as the best matches to the query token. This project has made the assumption that further than two edit distance returns results that are far too irrelevant and will impact the accuracy of the metrics.

4.3 Phonetic Algorithms

A phonetic algorithm matches two different words with similar pronunciation to the same phonetic code, which allows phonetic similarity comparisons and indexing.

After calculating the Levenshtein distance, a phonetic algorithm such as Soundex, Phonex or NYSIIS will be applied on the returned best match list.

5 Evaluation Metrics

5.1 Precision

Precision expresses the proportion of the data points the model says was relevant actually is of relevance. Measured by the number of correct results divided by the number of all returned results.

5.2 Recall

Recall is a model 's ability to find all the relevant data points in a dataset. More precisely it is defined as the total number of correct results divided by the total number of results that should have been returned.

5.3 F1 Score

The F1 score is the harmonic mean of precision and recall. F1 is perfect when it is 1.

6 Results

Algorithm	Recall	Precision
Levenshtein	0.626365	0.145078
Levenshtein-Phonex (LP)	0.608521	0.732841
Levenshtein-Soundex(LS)	0.601864	0.768446
Levenshtein-NYSIIS (LN)	0.600000	0.816304

Fig.1. Recall and Precision across algorithms

6.1 Analysis

The twitter data contains a lot of noise that originates from misspellings, abbreviations, acronyms, homophones and proper nouns.

In general, on top of L, using phonetic algorithms yielded a significant increase in precision and a slight decrease in recall. Comparing L and LP, recall dropped from 0.626365 to 0.608521 (less than 2% drop) but precision increased from 0.145078 to 0.732841, an equivalent of 58.78% increase (Fig.1).

LN revealed the best performance in phonetic algorithms in terms of precision, whilst LP was the most performant in terms of recall. LN 's precision is high because the resultant phonetic code it generates is more specific than the one generated by LS and is able to produce often 1 2 potential matches that often includes the correct form.

In general, for mispellings L performs effectively, but for some mispellings with intended repeating vowels, " soooo ", because it costs much more operations to transfer these words into their correct form, the right match is often missed. This will have an effect of decreasing the precision and recall metric for all tokens with repeating vowels.

Algorithm	"emoji"	"selfies"
L	16 matches	38 matches
LP	None	None
LS	None	None
LN	None	None

Fig.2. Total potential matches for slang words

One of the anomalies were found in words such as " emoji ", and " selfies ", an urban slang word that yields no correct match using L and since it sounds unique, it produces no potentials matches across all phonetic algorithms as seen in Fig.2.

Due to the high number of matches yet no actual correct pairing, this impacts the precision and recall of L 's performance.

Acronyms such as " 2K " whose canonical form is itself yielded many results in L with no correct match, whilst across the phonetic algorithms, it yielded no results.

Legitimate words such as " lumpur " which originates from Malaysia 's' Kuala Lumpur 'is incorrectly matched with " lumper " across all five methods. This indicates the flaw of these algorithms that work distinctively with American words only.

LN has the highest F1 score from Fig.3 which indicates it has low false positives and low false negatives. Hence, it is correctly identifying matches and are not disturbed by false matches.

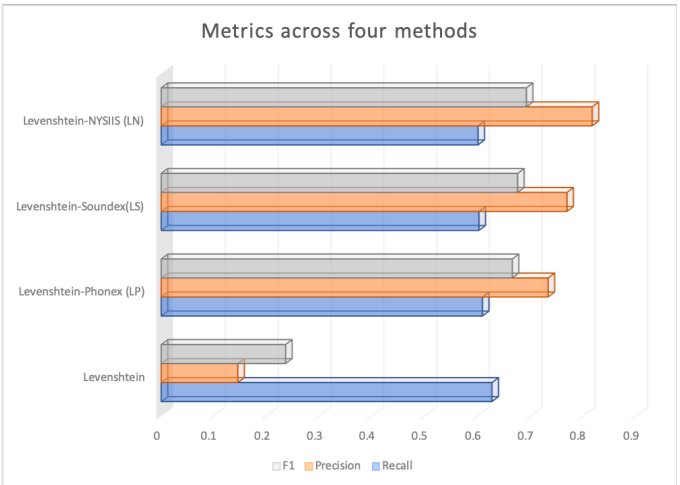

Fig.3. Evaluation Metrics across four methods

7 Improvements

1. Compliment the dictionary dataset with a slang dictionary as well as a foreign dictionary of english words. This will improve the performance when matching acronyms, abbreviations as well as foreign english words.

2. Implement a recurrent neural network to train data through a context-aware classification model before being processed by a normalizer.

3. Repeat experiment to increase the accuracy of the mean across metrics.

8 Conclusion

In summary, this report assesses the effectiveness of different methods for lexical normalisation of Twitter Data. The nature of“ Tweets ”, includes abbreviations, acronyms and homophones, all of which render the traditional spelling correction methods limited. Although the results has proved phonetic algorithms (NYSIIS was the best performer) with a string matching algorithm significantly outperformed the performance of using no phonetic algorithm, there are also many flaws in this system.

To tackle the flaws of the lexical normalisation system, it is important to compliment the dataset with an up-to-date slang and foreign english words dictionary, and to implement new deep learning technologies that are able to be context-aware.

References

Baldwin, Timothy, Marie Catherine de Marneffe, Bo Han, Young-Bum Kim, Alan Ritter,and Wei Xu (2015) Shared Tasks of the 2015 Workshop on Noisy User-generated Text:Twitter Lexical Normalization and Named Entity Recognition. In *Proceedings of the ACL2015 Workshop on Noisy User-generated Text*, Beijing, China, pp. 126135.

Christopher, D. M., Prabhakar, R., Hinrich, S. C. H. U. T. Z. E. 2008. Introduction to information retrieval. *An Introduction To Information Retrieval,* p.58.

Han, B., Baldwin, T. 2011. Lexical normalisation of short text messages: Makn sens a twitter. *In Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies.* Association for Computational Linguistics. *Volume 1,* pp. 368-378.

Zare M, Rohatgi S. 2017. *DeepNorm—a deep learning approach to text normalization.* arXiv : 1712.06994 (arXiv preprint)