COMP90049 Project 1

Lexical Normalisation of Twitter Data

Jithya Nanayakkara (342973)

1.0 Introduction

Twitter is a hybrid social networking and microblogging service that enables users to share text messages of up to 140 characters.[1] As of June 2012, the service had 500 million registered users generating over 340 million tweets a day.[2] This vast quantity of data presents opportunities for meaningful analysis of it – for example, there has been research into the viability of twitter data for predicting election results[3], determining the geographic location of users for disease surveillance[4] and predicting financial markets on the basis of online sentiment tracking[5].

However natural language processing (NLP) of such data is problematic as the quality of twitter messages vary greatly, where tweets could be "news-wire like text to meaningless strings"[6]. Due to the restrictive length of tweets, users have created novel syntax to communicate with as much brevity as possible[7]. Tweets often contain URLs, novel words, emoticons and abbreviations [7]. Punctuation and capitalization are rarely used properly, and users often write words with their phonetic spelling (e.g. 'b4' for 'before')[7].

Due to these issues, normalisation of twitter data is of interest as it can allow NLP tools to be more easily applied to it, allowing useful information to be extracted[7].

This report attempts to explore the effectiveness of several simple spelling correction techniques to normalize twitter data and is based on previous research by Han and Baldwin (2011)[6]. An important omission is that context is not used to determine the correctness of words.

2.0 Method

2.1 Data Used

The Twitter data used for the training and testing is a subset of the data produced by Han and Baldwin (2011) [6], which consists of tokenised tweets. Tokens consisting of apostrophes and hyphens are treated as a single token, but not those with whitespace. All tokens are case folded to lower case.

2.2 Token Classification

Not all tokens are candidates for normalisation, therefore a distinction must be made between those that are and those that are not. Tokens can be classified as:

OOV (out of vocabulary)
IV (in vocabulary)
NO (symbol, or Twitter-specific token; not a candidate for normalisation)

An IV token is defined as a token that exists in the dictionary used, in this case, the MySpell US English dictionary. It is important to note that the dictionary used by Han and Baldwin (2011) is Aspell, which would result in some differences in IV classification. For example, tokens such as 'tty' were recognized by MySpell as legitimate words, but not by Aspell.

Punctuation with the exception of " ' " and "-" were classified as NO by Han and Baldwin (2011), but in this report's system, digits were classified NO as well – as it would be extremely difficult to accurately determine if a digit such as '2' was used as a short hand for 'to' or not.

Only tokens classified as OOV were regarded as candidates for normalization.

2.3 Challenges in Token Normalisation

Normalisation of Twitter data is considerably challenging, as despite being similar to spell correction, the words used often contain intentional misspellings, slang, colloquialisms and abbreviations.

To illustrate this, consider the following:

e.g. 1
i jus happened 2 stroll

e.g. 2
smh jaz what was u doin?

e.g. 3
@briannacruz nooooo :(

e.g. 4
i love justin bieber!

In e.g. 1, ‘2’ is used as ‘to’.

E.g. 2 should read as “shaking my head just what was you doing?” “jaz” would be particularly tricky to normalize – how would a program recognize it means “just” as opposed to “jazz”? The spelling “jaz” itself could have been a misspelling of a short form for “just” or simply be a colloquial way of saying it.

In e.g. 3, the token “nooooo” is an example of how Twitter users express emphasis using repeated characters to lengthen the word.

In e.g.4 ‘bieber’, though a legitimate word, would be classified as an OOV token. Thus, an important challenge is to recognize when not to normalize.

2.4 Token Normalisation

The basic outline of the steps used to normalize a token is as follows:

1. Check if token is a known slang term.
2. If it is not, reduce any character repetitions. E.g. ‘nooooo’ becomes ‘noo’.
3. Have MySpell generate a list of suggestions for the token, in order of what it considers ‘best’.
4. Remove any suggestions that are multi-token words.
5. If there are one or more correct suggestions, select the ‘best’ suggestion by using one of the following methods:
    a. The first suggestion given by MySpell.
    b. The lowest Levenshtein distance.
    c. The lowest Editex distance.
    d. Use Soundex codes to find the first suggestion that has the equivalent code to the token’s code. If none exist, the token is not normalised.
6. If there are no suggestions by MySpell, do not normalise the token.

Thus, in automatically trying to determine the best normalization for a token, it can be observed if the system can improve upon MySpell’s first suggestion for correction.

2.4.1 Slang Dictionary

Given the quantity of slang and colloquialisms in Twitter data, a slang dictionary was made based on the listing given at www.noslang.com. The dictionary maps the slang term to its meaning. If a meaning was a multi-token word, the slang term was mapped back to itself (i.e. ‘imma’ => ‘imma’).

The use of the slang dictionary would improve the normalisation process, as the likelihood of correctly normalising a word is higher – as well as avoiding incorrect normalisations. For example, ‘da’ and ‘u’ would be correctly normalised to ‘the’ and ‘you’ respectively. ‘imma’ could be incorrectly normalised to ‘Emma’ if not for the slang dictionary.

2.4.2 Reducing Character Repetitions

The intuition behind reducing consecutively repeated characters in excess, is that it would make it easier for MySpell to generate more relevant suggestions.

2.4.3 Levenshtein Edit Distance

The Levenshtein edit distance is a widely used metric to measure how different two strings are. The difference is defined as the minimum number of operations – i.e. the insertion, deletion or substitution of a single character – are required to transform one string into the other. [8]

The intuition for using this, is that it may be likely the smallest edit distance between the token and its suggested normalisation would be the correct one.

2.4.4 Soundex

Soundex is a widely used phonetic matching technique that converts a string into a four-character code that represents its sound [9]. Strings that have the same codes are considered similar.

The intuition for using this is that since Twitter users may use phonetic spelling, Soundex would be able to find a more accurate match. For example, the codes for ‘lyf’ and ‘life’ are L100.

2.4.5 Editex

Editex is a phonetic distance measure that combines the properties of the phonetic grouping of characters like in Soundex with that of edit distances[9]. While standard edit distances have fixed costs for operations, Editex has two penalties: high for letters that never sound similar and low for those that do.

3.0 Results on Test Data

The following tables show the results of the different algorithms used for normalization, relative to the baseline, which is the Han and Baldwin (2011) results.

Precision is defined as the proportion of correctly normalized tokens.

Recall is defined as the number of tokens correctly normalized by the algorithm divided by the number of tokens normalized by the baseline, which is 572.

Under and over normalized tokens are relative to the baseline as well.

MySpells First Suggested Word	
Under-normalised tokens	116
Over-normalised tokens	198
Precision	325/654 = 0.50
Recall	0.57

Table 1.1

Levenshtein Distance	
Under-normalised tokens	116
Over-normalised tokens	198
Precision	328/654 = 0.50
Recall	0.57

Table 1.2

Soundex	
Under-normalised tokens	150
Over-normalised tokens	128
Precision	314/550 = 0.57
Recall	0.55

Table 1.3

Editex	
Under-normalised tokens	116
Over-normalised tokens	198
Precision	328/654 = 0.50
Recall	0.57

Table 1.4

4.0 Discussion

It can be seen that all the algorithms perform much worse than the baseline. While the use of different dictionaries and the inclusion of digits to be classified as NO would affect the results, it's unlikely to be significant.

The immediate observation is that the three algorithms used to select the 'best' normalisation from MySpell's list of suggestions, were not a significant improvement over simply using the first suggestion.

A likely problem is too much over-normalisation. The Soundex approach was able to obtain a slightly higher precision than the rest was because it did not normalise as many tokens – however it did have a relatively higher number of under-normalisations.

In order to address the issue of over-normalisation, the most effective approach would be to use a context based approach. This would aid the program to correctly identify which OOV tokens should be left alone. It would also allow digits to be included as OOV tokens, as the probability of correctly determining if '2' means 'to' or not is greater. This means it would also aid in reducing incorrect normalisations.

A possible source of incorrect normalisation would be the slang dictionary. Consider:

he looks like he comes from one of those cultures that will marry off a 10 y o to a 30 y o

'y' would be normalised to 'why' using the slang dictionary – yet we can see from context that 'y' and 'o' actually mean 'year old'. Even if a context based approach was used, it would be quite difficult to arrive at that normalisation – however, it would be more likely to identify that the tokens 'y' and 'o' should be left alone rather than incorrectly normalising them.

A possible improvement would be to consider not only words suggested by MySpell for normalising, but generating a list of potential slang words based on the slang dictionary – as it may be possible that some tokens are slang words that are incorrectly spelled. This would allow them to be mapped to the slang word dictionary to correctly normalise them.

Further processing can be done on OOV tokens, such as expanding phonetic abbreviations for tokens of small length. For example 'b4' could be expanded to 'before' by having mappings of b => be; 4 => fo, fore, foe. However this approach has a strong potential of adding a lot of noise into tokens, decreasing the possibility of finding relevant normalisations.

5.0 Conclusion

The results of this report show that given the nature of Twitter data, simple approaches that maybe effective in spell checking will not be nearly as effective for normalising Twitter data. The high rate of both deliberate and accidental spelling errors, together with the regular use of novel words and proper nouns, means that far more sophisticated approaches would be required to make sense of the data.

References

[1] Wikipedia, *Twitter* (April 22, 2013), <http://en.wikipedia.org/wiki/Twitter>

[2] Lunden, I. *Analyst: Twitter Passed 500M Users in June 2012, 140M in US; Jakarta 'Biggest Tweeting City'* (July 30, 2012). TechCrunch <http://techcrunch.com/2012/07/30/analyst-twitter-passed-500m-users-in-june-2012-140m-of-them-in-us-jakarta-biggest-tweeting-city/>

[3] Gayo-Avello, D. *A meta-analysis of state-of-the-art electoral prediction from Twitter data* (June 25, 2012). <http://arxiv.org/ftp/arxiv/papers/1206/1206.5851.pdf>

[4] Burton, S.H., Tanner, K.W., Girraud-Carrier, C.G., West, J.H. and Barnes, M.D. 'Right Time, Right Place" Health Communication on Twitter: Value and Accuracy of Location Information' (2012): 14(6) *Journal of Medical Internet Research* 366-376

[5] Mao, H., Counts, S., Bollen, J. *Predicting Financial Markets: Comparing Survey,News, Twitter and Search Engine Data* (December 5, 2011). <http://arxiv.org/pdf/1112.1051v1.pdf>

[6] Han, B., Baldwin, T. *Lexical Normalisation of Short Text Messages: Makn Sens a #twitter* <http://www.aclweb.org/anthology-new/P/P11/P11-1038.pdf>

[7] Kaufmann, M. *Syntactic Normalization of Twitter Messages* (2010) <http://cs.uccs.edu/~jkalita/work/reu/REUFinalPapers2010/Kaufmann.pdf>

[8] Sasu, L., 'A Probabilistic Model for Spelling Correction' (2011) 4(53) *Bulletin of the Transilvania University of Brasov* 141-146