COMP30018: Knowledge Technologies Project 1 Diana Ruth

Introduction

Objective

The purpose of this assignment was to identify misspelled locations in tweets. Given a text file containing acceptable location names and a text file containing tweets from Twitter, the software system employed the global edit distance technique for string matching to attempt to identify misspelled locations.

Matching Method: Global Edit Distance

Overview

This software system used Levenshtein distance for string comparisons, which is a type of global edit distance that adds one to the edit distance for every insertion, deletion, or replacement and adds zero for a character match.

Application of Global Edit Distance to the Problem

Removal of Unnecessary Data

In the tweet file, only the tweet text was analyzed because the other information is unnecessary in this context. In a tweet, words were discarded if they contained special characters or numbers because most locations contain only letters.

Handling Multi-Word Locations

To handle multi-word locations, the location was compared to chunks of the tweet that had the same number of words. So if a location had only one word, it would be compared to each word in the tweet separately. If a location had two words, it would be compared to the first and second word, then the second and third word, and so on until the end of the tweet is reached. This way, multi-word locations were much more likely to be caught even if they are misspelled, because they are being compared to strings of similar length and word count.

Capitalization

Since all locations are meant to be capitalized, the system does not ignore capitalization when comparing strings. A word or phrase in a tweet is much more likely to match a location if the first word was capitalized, which cut out many occurrences of normal English words being matched with locations.

Effectiveness of Global Edit Distance

Locations That Are Common English Words

Often a word that is identified as a location is simply an English word used in a context other than location. The following examples illustrate this issue:

Original Tweet	Assumed Location in Tweet	Matching Location
Classic Home Furniture Center	Home	Home
Up to a73 downloads.... but yo, its FREE... so cop it and enjoy! DJ ROC Old School Mix @ http://tinyurl.com/yhz4pfs	Mix	Mix
On Mad Money: Dow +127, get Jim's pin action plays. Plus can JCG have you dressed for success? And JAH's CEO. CNBC@11p	Money	Money

Although these words in the tweet match the location exactly, the location is used more often in a general context rather than as a location. Clearly the user did not intend for any of these words to be used as a location in these tweets based on the context, however the system has no way of deciding this because the strings are exact matches.

Multiple Locations with Similar Spellings

Another common case the software system discovered was when a location was present in a tweet and was discovered by the system, but several other similar locations were discovered as well. In this case, it may difficult to decide which location the user intended to use. The following example illustrates this problem:

Original Tweet	Assumed Location in Tweet	Matching Location
BREAKING NEWS: Fire at Bowler in Fargo:	Bowler	Bosler
	Bowler	Bowlder
	Bowler	Bowler
	Bowler	Bowser
	Bowler	Bowyer
	Bowler	Fowler

The user meant to use a location in this tweet, but there are several close locations that match said location. Since so many other valid locations match the intended location, most of the reported matching locations are incorrect.

Non-Locations Being Treated as Locations

Another issue found with the algorithm is that many common words that were not used as locations were identified as locations.

Original Tweet	Assumed Location in Tweet	Matching Location
HighDensity Garden To Provide Your Family With Fresh Wholesome And TastyVegetables.	Garden	Barden
	Garden	Carden
	Garden	Darden
	Garden	Gardena
	Garden	Garten
	Garden	Rarden
	Garden	Varden
	Garden	Warden

Although all these locations are very similar to the word “garden,” the word “garden” was not meant to be used as a location so these identifications are incorrect. This shows that many normal English words that are not used as locations can very closely match valid locations.

Precision

While the software system matches words in the tweets with locations based on allowing a global edit distance proportional to the length of the strings, a majority of the words assumed to be locations are in fact not used as locations in the tweets. Precision is negatively impacted by the large number of locations in the locations file that can also be used as regular English words. Additionally, even when a location is identified correctly, several other similar but incorrect locations may be identified as well. On the other hand, normal words in the tweet may match with locations that are spelled very similarly, and those words end up being seen as a location. Due to these false positives in attempting to identify misspelled locations, the overall precision of this system is low and could be improved upon by modifying the algorithm to filter out more of these false positives.

Conclusions

Global edit distance provides an decent quantitative measure of similarity between two strings. One of the most difficult challenges of solving this problem was in choosing appropriate thresholds for accepting a string as a misspelled location. When testing the system with low thresholds, that is, accepting a higher global edit distance based on the length of the string, the system yielded a large amount of results for each tweet and most were incorrect. When the thresholds were changed to allow only small global edit distances based on the length of the string, far less results were yielded and many were correctly identified as locations. However, the possibility of missing a misspelled location in a given tweet was heightened when the thresholds were changed.

Another major finding of this project was that even though a string may have a low global edit distance, the string in the tweet is meant to be a normal English word and not a location. This pattern proved to negatively impact precision, as it is impossible for the system to discern whether the string was meant to be used as a location or as a word. This distinction can only be made in the context of the tweet which is incredibly difficult for a software system to detect.

While global edit distance provides an accurate measure of similarity between strings, each comparison takes O(nm) time (where m and n are the lengths of the two strings), which is costly. Because each tweet must be compared to the list of locations several times and there are millions of words in the tweet file and over a million locations, the running time for this system can become excessive. Running the software system on a small collection of 13 tweets took 16 minutes, which means that to run the system on the large tweet file would take about 341 days.

Both running time and accuracy could be improved with further research and development of the software system, but investigating global edit distance as a method for comparing string similarity provided much insight on the challenges of working with large amounts of user data. Many problems involving user data are messy, and the results of working with this data may not always be perfect.