COMP90049 Project 1:Detect Lexical Blends in Tweets

Anonymous

1 Brief Introduction

Lexical blending is the way of word-formation that selecting a part of two or more words to form a new word (Lepic, 2016). This phenomenon often happens on social media like Twitter and Instagram. This report is to implement a system which can detect blend words in a pre-process list of tokens extracted from Twitter dataset. In this paper, three approximate string matching methods will be used to detect blend words, and the performance of each method will be assessed.

2 Datasets

The datasets needed in this program include a list of lexical blends offered by Deri and Knight (2015), Das and Ghosh (2017), Cook and Stevenson (2010), which is used for evaluating this system. The lexical blends in this list are manually identified and two component words of this lexical blend are included. Besides, the datasets also include an alphabetically-sorted list of tokens that appears on tweets, which is called candidates (Eisenstein, O'Connor, Smith, & Xing, 2010). This program will detect possible lexical blends in this dataset. Another dataset is 370K English entries. This dataset is used as a dictionary to find possible components of blend words and predicted blend words will be returned based on whether components are exist in the dictionary or not.

3 Evaluation Metrics

This System will be evaluated by three different evaluation methods. And the result analysis will mainly base on these three evaluations. Here are the names of these three measures:

• Precision: The proportion of the number of the predicted true lexical blends to the number of predictions the system returned.

• Accuracy: The number of true lexical blends given by the system divide the number of all the tokens in the candidates.txt.

• Recall: The number of predicted true lexical blends in the document divide the number of true lexical blends in the document.

4 Methods

4.1 Global Edit Distance

Global Edit Distance(GED) is a method to measure the difference between two strings by doing edit operations and calculate the score. It means these two strings are more similar to each other if one string transforming to another string need less operations. Insertion, deletion, replacement and match are edit operations (Navarro, 2001).

According to the feature of blend words, blend words can be divided into two parts, each part is a transformation of one component. Each word in the candidate.txt will be divided into two parts, and Global Edit Distance will be used to detect the distance between words in the dictionary and two parts of candidate

words.

Each word in the candidate was divided into two parts, the first part is two letters long and the second was the remaining part of the word. If there are words in the dictionary meeting the distance requirements of the first part and the second part of the word in the candidate respectively, the word in the candidate will be recognized as a blend word.

4.2 Local Edit Distance

Local Edit Distance(LED) can be used to measure the best matching substrings in two words. The blend words and its first component have the best matching substring in their prefix and the blend word has the best matching suffix with its second component.

This program will use Smith-Waterman algorithm to build a matrix for each pair of words, then according to the number in the matrix to find the best matching substrings. The words in the candidtates.txt need to find two best matching substrings with two different words in the dict.txt. If first substring is located in the beginning of the word and another is located in the end of the word, furthermore, then these words will be returned as blend words.

4.3 Jaro-Winkler Similarity

Lexical Blends have a feature that is the component and the blend word have the same prefix, so Jaro-Winkler Similarity is a good way to improve the performance of this system, because Jaro-Winkler Similarity gives high weight to those strings who has the same prefix.

Each word in the candidate.txt will use Jaro-Winkler Similarity method to return the first word in the dictionary which similarity greater than 0.9, then the system will remove the similar prefixes of the word from candidate and the word from dictionary. The candidate word after processing will be used to find if there is a word in the dictionary which the similarity between this word in the dictionary and the processed word is greater than 0.9, if there exist such a word in the dict.txt, then the canditate word will be returned as a blend word.

5 Result Analysis

5.1 Global Edit Distance

5.1.1 Results

The following table is the results of using Global Edit Distance to detect lexical blends in the candidate.

Precision	Accuracy	Recall
1.01%	0.89%	80.9%

Table 1: Results of Global Edit Distance

5.1.2 Analysis

There are three possible reasons for this result.

First, the Global Edit Distance does not focus on some substring of the word, Global Edit Distance is just to see if two strings are similar in general rather than focus on prefix and suffix of a word. For example, "teme" is a blend word composed by "technological" and "meme". The number of Global Edit Distance between "technological" and "teme" is very large, although they have similar prefixes.

Second, the real blend word material is too small. There are only 183 right blend words in the blend.txt, compared to 16684 words in the candidates.txt is too small, so this is one reason of the precision is too small and the recall is about 81%.

Third, the division of the word is not reliable, The system divide the words into two parts, the first part is 2 letters long. however, many real blend words and their first components have more than 2 letters similar prefix. For example, "homeboy" and it's first component "hometown" has 4 similar letters. If not according to the actual situation to divide the word, it will return some unrelated words.

5.2 Local Edit Distance

5.2.1 Results

The Local Edit Distance method returns the most correct blend words, but the number of predictions returned is much higher due to the precise matching. Here are the result table of LED.

Precision	Accuracy	Recall
1.03%	0.89%	80.9%

Table 2: Results of Local Edit Distance

5.2.2 Analysis

The precision is improved compared to global edit distance. Local Edit Distance almost return every possible blend words. So this algorithm returns the most of real blend words in the blends.txt.

However, the Local Edit Distance also will return those words like "youtube" and "youmail", because the prefix and suffix of those words exist in dict.txt. And "you" are the best matching prefix between word "you" and word "youtube". "tube" is also the best matching suffix of word "tube" and word "youtube". However, those words are not exist in blends.txt. So those conditions will reduce the precision.

5.3 Jaro-Winkler Similarity

5.3.1 Results

The results of using Jaro-Winkler Similarity to detect the lexical blends are showing in the following table.

Precision	Accuracy	Recall
1.12%	0.73%	66.1%

Table 3: Results of Jaro-Winkler Similarity

5.3.2 Analysis

By using the Jaro-Winkler Similarity method, the precision of finding real blend words is improved to 1.12%. The Precision is still very low, the effect is not obvious. Here are some possible reasons for this result.

Firstly, there are some blend words existing in the blends.txt but not existing in candidates.txt, so the system will not return those words. The maximum number of predicted real blend words is 151. So the recall is no more than 82.51%. Here are some examples on the table.

Number	blend words
1	beefaroni
2	bootylicious
3	brainiac
4	fucktard
5	meggings

Table 4: Examples of blends word not in candidate.txt

Secondly, the disadvantage of this system is it just pick the first word in the order of the dictionary that large than the similarity parameter. However the first word may not be the

component of this possible blend word. After removing the prefix of the candidate word, it will result the suffix of the candidate word cannot find a suitable matching word, so this will result many real blend words cannot be returned.

6 Improvement

Obviously, according to the results and analyses above, this system can be improved in many aspects. Here is some advices for improving this system.

1. There are many words which are not blend words obviously in the candidate.txt. For example,"aaaaa" is not a blend word, but this word is likely to be considered as a blend word in this system. This is because there are many words in the dictionary having the same prefix and suffix with "aaaaa". So the system should pre-process this candidate.txt to delete those words which has four or more consecutive identical letters. This will improve the system.

2. Some predicted words returned by the system are not blend words in the blend.txt. These words are just two words joined together without any changes. For example, "afterdark" and "anythin", the system can delete those plausible blend words before giving the output. The performance and the precision of this system will be improved by reducing the this kind of words.

7 Conclusion

In this project, three kinds of approximate string matching methods are used in detecting lexical blends. The precision is always maintain at a small number. The Global Edit Distance method is not good at detecting prefix and suffix of the word. Although it can return more blend words in the blends.txt, the total number of predictions are also very large. The precision of GED is lower than other methods. Jaro-Winkler Similarity method is also not good. When setting the similarity parameter of Jaro-Winkler method, we can reference the similarity of real blend word and its components in the blend.txt rather than setting a very high similarity randomly. The system might has a higher precision by doing this.

References

Cook, P., & Stevenson, S. (2010). Automatically identifying the source words of lexical blends in English. *Computational Linguistics*, *36*(1), 129–149.

Das, K., & Ghosh, S. (2017). Neuramanteau: A neural network ensemble model for lexical blends. In *Proceedings of the the 8th international joint conference on natural language processing* (p. 576-583). Taipei, Taiwan.

Deri, A., & Knight, K. (2015). How to make a frenemy: Multitape FSTs for portmanteau generation. In *Human language technologies: The 2015 annual conference of the north american chapter of the acl* (p. 206-210). Denver, USA.

Eisenstein, J., O'Connor, B., Smith, N. A., & Xing, E. P. (2010). A latent variable model for geographic lexical variation. In *Proceedings of the 2010 conference on empirical methods in natural language processing (emnlp 2010)* (p. 1277-1287). Cambridge, USA.

Lepic, R. (2016). Lexical blends and lexical patterns in english and in american sign language. In *Mediterranean morphology meetings* (Vol. 10, pp. 98–111).

Navarro, G. (2001). A guided tour to approximate string matching. *ACM computing surveys (CSUR)*, *33*(1), 31–88.