COMP90049 Project1 Report: Word Blending in Twitter

Anonymous

1 Introduction

The development of times is beyond people's imagination. Apparently, it will lead to a fact that existing native English words are difficult to express new phenomena or inventions. Thanks to users of Twitter with their creativity, blend words were born, formed by two word fragments. For instance, a meal between breakfast and lunch is called "brunch".

The purpose of this research is to filter items which is possible to be blend words from a given set of strings through leverage distance. To start with, this paper will present a brief overview of the dataset used, including its source and content. Secondly, it will put forward the hypothesis concerning the word formation method for blend word. Then, it will introduce the basic approach applied in the research. Eventually, it will evaluate the results and draw a conclusion.

2 Dataset

The experimental datasets consist of five parts, three of which are the main observation in this report including "dict", "candidates", "blends". Each of these is described as follows.

a. dict.txt: It contains about 370, 000 entries, adapted from Dwyl's (2014) data, and it will be used as a dictionary for approximate string search method in the report.

b. candidates.txt: There are 16925 tokens in the candidates.txt, which is a result of a series of pretreatments for the dataset "tweets" (Jacob et al., 2010).

c. blends.txt: It consists of 183 entries, which is edited according to the publications written by Deri et al. (2015), Das et al. (2017) and Cook et al. (2010).

3 Related literature and Hypothesis

Generally, there are three classification of word formation for blend words.

First, combine the beginning of one word with the end of another. For example, spork is a blend of **sp**oon and f**ork**. It seems that most blend words are composed in this way.

Second, extract the beginning of two words respectively to from a new blend word. Take Botox for instance, it is blended by **bo**tulin and **tox**in.

Third, extract the end of one source word, and then directly add it to another original word to contrast a new blend word. For example, gaydar is formed by **gay** and ra**dar**.

According to Gries (2004), although blend words present numerous different type of structural characteristics, their structure is ultimately determined by the desire to identify the original word. This means that people can always find the characteristics of the source words in a blend word. Based on this, a hypothesis can be proposed: In most cases, a blend word is similar to its source word, that is, the score for edit distance between them can be high. Additionally, two operations, deletion and substitution, instead of insertion, are used frequently when converting a blend word into one of the source word with higher score of edit distance.

4 Basic Methods

Three methods, regular expression, Levenshtein Distance and Jaro-Winkler (JW) Similarity are mainly used in the report. The dataset "candidates.txt" is preprocessed through regular expression first. It is then filtered on the basis of edit distance or JW similarity.

4.1 Regular Expression

Regular expression can be generally defined as a sequence of characters which represents different search patterns. In the dataset "candidates.txt", there are numerous meaningless fragments, including modal particle and repetition. Regular expression is used to match these items and delete them. To be specific, the pattern ". *(\w)\1{2,}.*" is used to remove all the entries which have a large amount of duplicate letters, like "aaaaaaahhhhhh" and "aaaand". In a similar way, the pattern ".*(\w\w)\1{1,}" is used to

delete several modal particles, like “ahahaha”. These preprocesses can reduce the amount of detection data in order to reduce the burden on the program.

4.2 Levenshtein Distance

Edit distance, known as Levenshtein Distance, is a quantitative measure of the degree of difference between two strings, especially for two English words. Edit distance has four parameters corresponding four operations. They are, respectively, “m” for matching, “i” for insertion, “r” for “replacement” and d for “deletion”. Generally, parameters are [0, 1, 1, 1], but they will be modified slightly in the report.

4.3 Jaro-Winkler Similarity

For the two strings s1 and s2, their Jaro similarity is given by the following formula:

$$sim_j = \begin{cases} 0 & \text{if } m = 0 \\ \frac{1}{3}\left(\frac{m}{|s_1|} + \frac{m}{|s_2|} + \frac{m-t}{m}\right) & \text{otherwise} \end{cases}$$

|s1| and |s2| indicate the length of the strings "s1" and "s2";

“m” represents the number of matching characters for two strings;

“t” is for half the number of transpositions

Jaro-winkler similarity is further modified on the basis of Jaro similarity. In this algorithm, the importance of the same prefix is highlighted more. That is, if two strings are identical in the first few characters, they will have a higher similarity. The formula of this algorithm is as follows:

$$sim_w = sim_j + \ell p(1 - sim_j),$$

“l” represents the number of common prefix characters for two strings, generally up to four;

"p" has the default value 0.1

5 Result and Evaluation

5.1 Evaluation Metrics

In this report, precision and recall rate are the main evaluation metrics, which contributes to judging the advantages and disadvantages of the algorithms.

5.2 Results

This experiment adopts different methods or different parameters of the same method to obtain the final results. These results are shown as followed table.

	Precision(%)	Recall(%)
ED [0, 1, 1, 1]	1.28	76.82
JW similarity	1.16	81.19
ED [0, 3, 1, 1]	1.45	72.95

Table 1- The results of three methods (ED means Edit Distance).

5.3 Analysis and Comparison

5.3.1 Different Methods

Through running results of the Java program, Edit Distance method with parameters [0, 1, 1, 1] (Respectively represent the weight of matching, insertion, replacement and deletion operation), returns 9052 possible entries. Among them, 116 entries are detected to be correct. It can be figured out that the precision is 1.28% while the recall is 76.82%. In terms of JW similarity, it returns 10568 possible results, of which 122 are correct. The precision and recall for this method are 1.16% and 81.19% respectively.

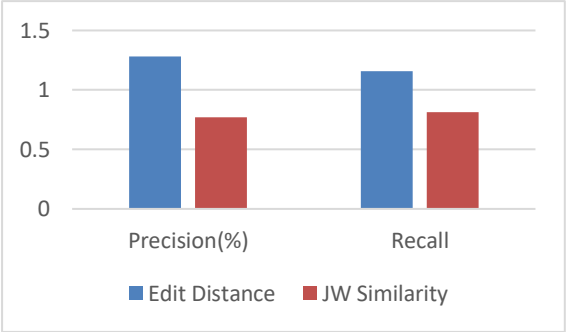

Figure 1- Comparison chart of precision and recall of Edit Distance and JW Similarity.

As is shown in Figure 1, compared with JW Similarity, Edit Distance achieves higher accuracy on the premise of only a slight loss recall rate. Therefore, in this report, Edit Distance is superior.

5.3.2 Different Parameters

According to the hypothesis mentioned above, the parameters of Edit Distance are adjusted to [0, 3, 1, 1] at this step. There is a decrease in predicting outcomes from 9052 to 7586. However, there are still 110 correct outcomes survived. The precision of this method reaches 1.45% and the recall drop slightly to 72.95%

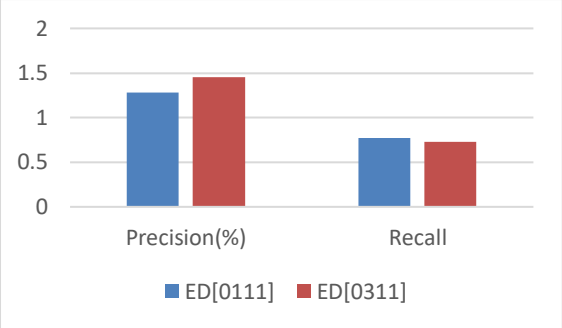

Figure 2- Comparison chart of precision and recall of Edit Distance with parameters [0, 1, 1, 1] and with parameters [0, 3, 1, 1].

According to Figure 2, Edit Distance with parameters [0, 3, 1, 1] has higher precision, and its recall rate is still over 70%, reaching 72.95. The adjustment of parameters achieves better result. Additionally, this also confirms the hypothesis proposed before, that is, the operation of insertion is rarely used when transferring a blend word to one of its source words.

6 Conclusions

Overall, after a series of improvements to the algorithm, including several pretreatment and adjustment of parameters, it finally achieves 1.45% precision with 72.95% recall rate for searching blend words from candidates. Compared with previous data, it seems to obtain a relatively big improvement. However, this precision is far from enough when solving practical problems.

In this report, each time just a single method is used for processing data. Hence it can be believed that a possibility that combining two various methods can gain a high precision exists. In addition, another way to enhance the precision is machine learning. The machine is trained with a large amount of samples to improve accuracy.

References

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

K. Das and S. Ghosh. Neuramanteau: A neural network ensemble model for lexical blends. In *Proceedings of the The 8th International Joint Conference on Natural Language Processing*, pages 576–583. Taipei, Taiwan, 2017.

A. Deri and K. Knight. How to make a frenemy: Multitape FSTs for portmanteau generation. In *Human Language Technologies: The2015 Annual Conference of the North American Conference of the North American Chapter of the ACL*, pages 206–210. Denver, USA, 2015.

Dwyl. 2014. https://github.com/dwyl/english-words.

J. Eisenstein, B. O'Connor, N. A. Smith, and E. P. Xing. A latent variable model for geographic lexical variation. *In Proceedings of the 2010 Conference on Empirical Meth- ods in Natural Language Processing (EMNLP 2010)*, pages 1277–1287. Cambridge, USA, 2010.

S. T. Gries. "Shouldn't it be breakfunch? A quantitative analysis of blend structure in English". In *Linguistics.* 42 (3): 639–667, 2004.