Twitter information extraction system

Yao Chen Torsten Suel
Department of Computer Science
NYU Polytechnic School of Engineering
6 MetroTech Center
Brooklyn, NY 11201
sunnychen@nyu.edu

Abstract:the function, technique

Keywords:

1. Introduction

In twitter message analysis area, there are many work about the sentimental analysis. But there is little software which is used to extract information from twitter message. The reason is twitter messages are not complete English sentence. The existing technique is hard to analysis those information. However, the twitter messages are considerably valuable in many area, such as economic, political field. To solve this problem, this paper filter the twitter message and study on special topic.

In this paper, the “new year resolution” related twitter message is chosen as data source. The system extracts the users' resolution from twitter message and get the top 3 resolution. Techniques in information retrieval and machine learning area are evolved in this system.

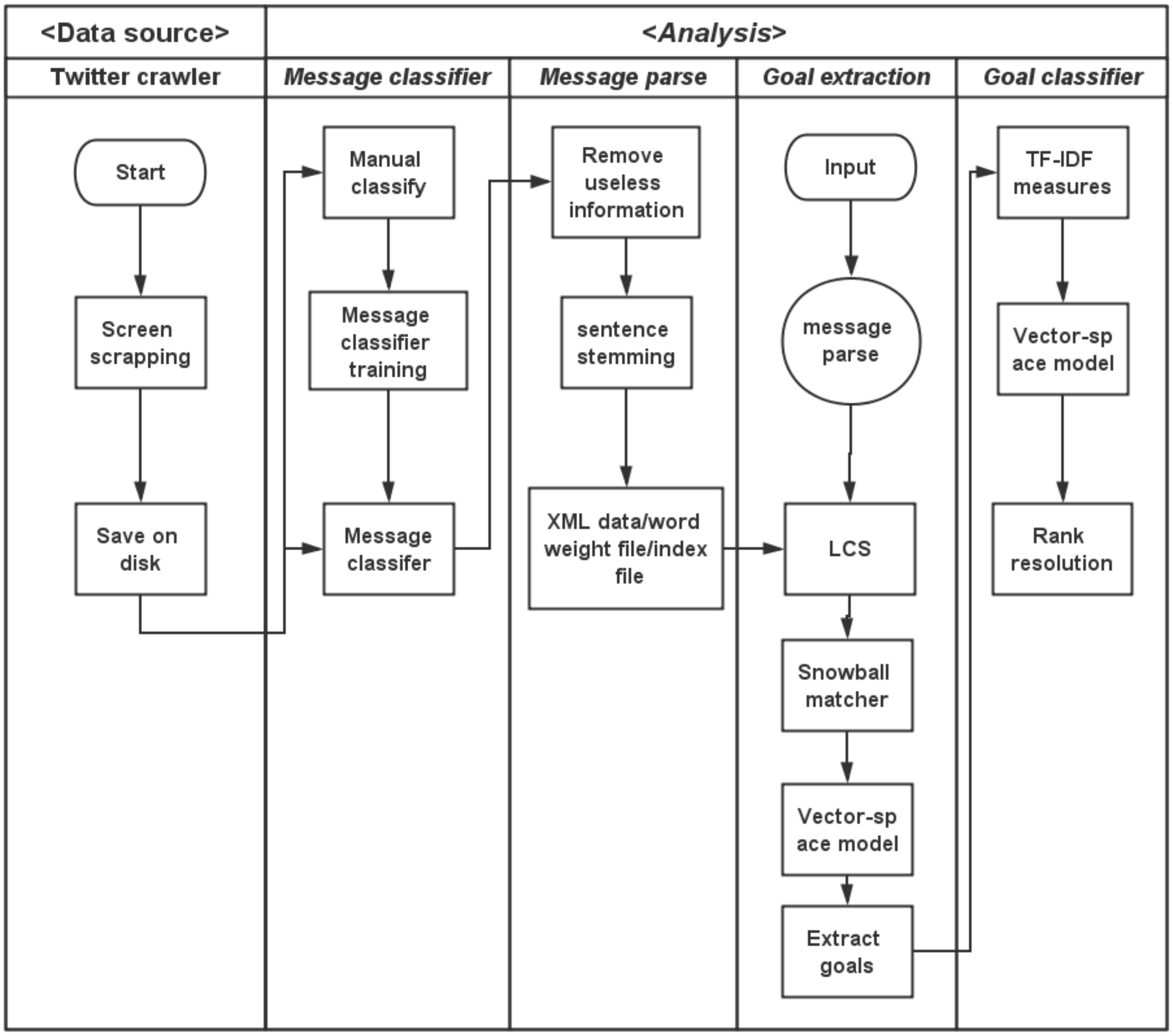
Related work “Snowball: Extracting Relations from Large Plain-Text Collections” studies on the texture material and extract the “organization” and “location” information. This study is based on Brin's DIPRE method which is used to extract the “book” and “authority” information. Those two system are partly similar with this paper. However, those study are about information extracted on nouns. Under this background, the work is much more easier for the named-entity tagger algorithm. But, the named-entity tagger algorithm can not be used in this paper, because the users' goal are unusually complex components include verbs, nouns and Adjective.

The sentimental classifier algorithm is good tool to classify information, such as Naive Bayes. The drawback of those algorithm is only constant classification of the data source. In this paper, the “new year resolution” message has strong diversity which should be classified into variable number of classifications. “The Evaluation of Sentence Similarity Measures” study is used to solve this problem.

This system can be used in many area, such as advertisement, market analysis. For advertisement, the company will sell their product to their real customers who's new year resolution is related to the products. The users also will can use the analysis result to find friends who may already achiever their goal and get help.

2. Twitter message analysis system

The system is can be divide into two parts, data source and Analysis. The data source part will get the data source from web site. The Analysis part will extract information and get the final rank information. It also divided to five phases: Twitter crawler, Message classifier, Message parse, Goal extraction and Goal classifier. The details of each part will explain as following.



Twitter crawler

In this phase, the system will crawler twitter message from twitter site. The search key word will be set to "new year resolution" and time will from 12-25-2013 to 01-05-2014. The screen scrapping technique will be used to get information, since the twitter API has limitation on the messages. By using twitter API, only recently one week message can be crawled. The "new year resolution" messages are more dense around new year. So twitter API will can not be used for the message time limitation. I use the " Scraper" to crawl message from twitter website, which is an Google chrome extension.

Message classifier

Message classifier is used to classify message into two sets: resolution and not resolution. The machine learning and sentence sentimental algorithm is evolved in this part. For the training set, the data source has been divided into two sets manually. Then lingpipe lib will used to train the classifier, which has implement sentence sentimental algorithm and training process algorithm.

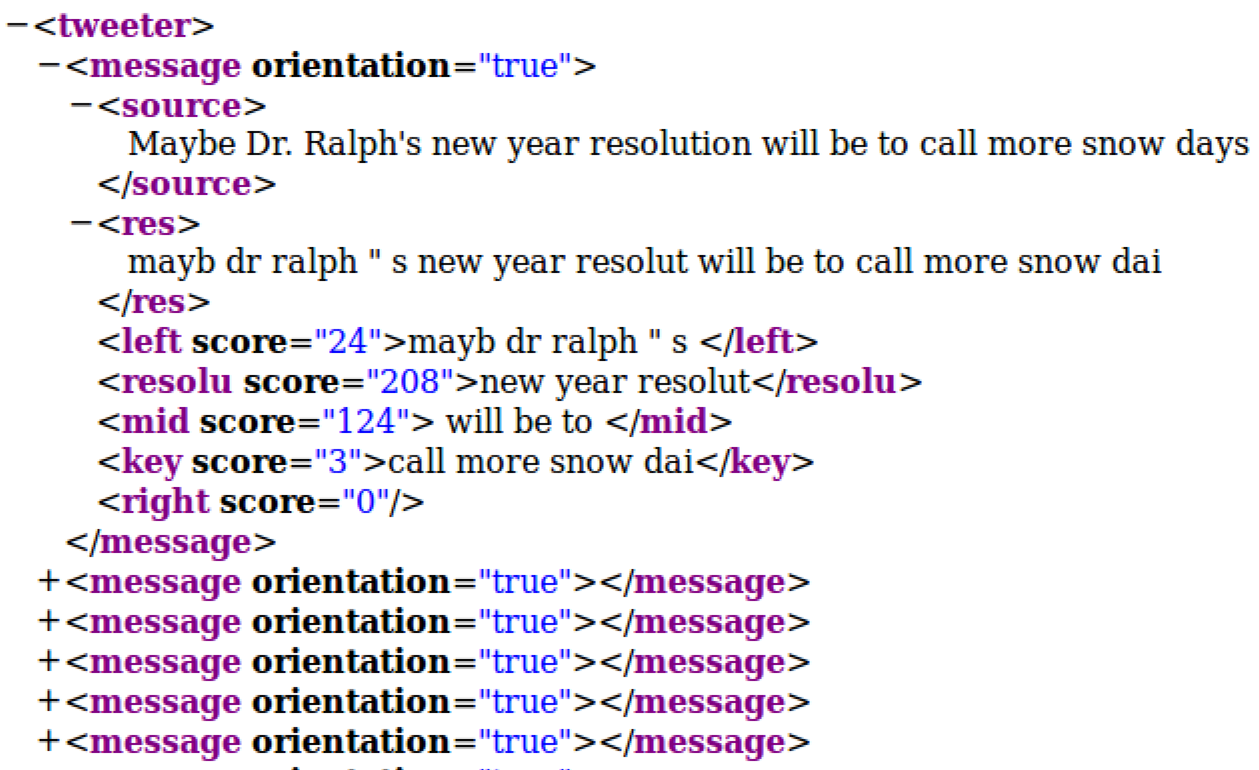
Message parse

This parse is used to filter sentence and get the useful part of each sentence. According to the study on the data source, the useful information for the analysis includes web link and punctuation, such as period, comma. But other punctuations will be retained since it may be a mark for the resolution such as colon. Those punctuations will be add space to split them from words. For example, “My new year resolution: quit smoke”. The colon here is sign for user's goal. The message will be transfer to “My new year resolution : quit smoke”. The regular expression will be used here to remove or modify information.

Then the message will be stemmed. The stemming process will increase the information matching accuracy. “The Porter Stemming algorithm” and implement is used to deal with this problem.

In this parse, the each message will be divide into five part for the latter phase “Goal extraction” manually. The divided information is used to training data set. The related algorithm will be used in “Goal extraction” phase. The word frequency will record, which will be used in “snowball matcher” algorithm of “Goal extraction” phase. The account of sentence also should be record, which will be used in “Goal classifier” phase.

The output file includes source XML data, word frequency and index file. The source XML date will include source sentence, dealed sentence and their five parts in XML format. The formation should be as following.



```
-<tweeter>
  -<message orientation="true">
    -<source>
       Maybe Dr. Ralph's new year resolution will be to call more snow days
     </source>
    -<res>
       mayb dr ralph " s new year resolut will be to call more snow dai
     </res>
     <left score="24">mayb dr ralph " s </left>
     <resolu score="208">new year resolut</resolu>
     <mid score="124"> will be to </mid>
     <key score="3">call more snow dai</key>
     <right score="0"/>
   </message>
  +<message orientation="true"></message>
  +<message orientation="true"></message>
  +<message orientation="true"></message>
  +<message orientation="true"></message>
  +<message orientation="true"></message>
```

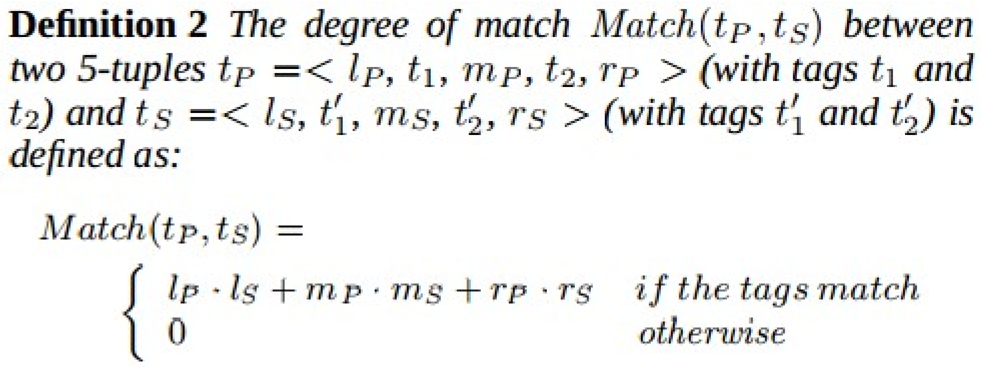
The root node is “tweeter” node, which contains many children node “message”. Each message node represents one twitter message. The message node contains seven children node, source, res, left, resolu, mid, key, right. The source node is contain the raw twitter message. The res node is the result after information filter. The sentence should be divide into five part by “resolu” and “key” part. The “resolu” node contain the flag word, which has same meaning with “new year resolution”. The “key” node is user specific goals. If the “resolu” part appears earlier than the “key” part in the source sentence, the orientation attribution of message node will be set as “true”; otherwise, it will be set as “false”.

The index file records each word and the account of documents which contain the word. The word frequency record each word and the account of each word in the whole data set.

Goal extraction

In this part, the user specific goals are extract from input twitter message. The system will compare the test twitter message with the data source. The data source is used as pattern to test the test message. The Longest common subsequence (LCS) algorithm will be used here to solve this problem. Every message in data source has been divided into five part as shown in last phase, left, resolu, mid, key, right part. The “resolu” parts have highest similarity in each sentence because the search key is “new year resolution”. Firstly, we will compare the test twitter message with the “resolu” part of one sentence in the data source using LCS algorithm. We will use the result subsequence to divide the test twitter message into thire part, left, resolu, right part. Then according to the orientation of the data source sentence, we will pick the left or right part to do the next step. If the orientation is true, the mid and key part is latter than the “resolu” part. So we will compare the right part of tested message with the mid part of the data source sentence using LCS algorithm; otherwise, the left part of tested message should be used to compare. We chose the “mid” part to test because the mid part has stronger similarity than the rest part. It is designed according to the “Snowball algorithm”.

Snowball algorithm this system is used to test the organization and location of sentence. For each tested sentence, it will be divided into five part, left, organization, mid, location and right part. The named-entity tagger algorithm is used here to find the organization and location part. Then the sentence will be divided into five part. Then the single pass clustering technique will be used to compare the similarity of two sentence. It is defined as following:



Definition 2 *The degree of match* $Match(t_P, t_S)$ *between two 5-tuples* $t_P = < l_P, t_1, m_P, t_2, r_P >$ *(with tags* t_1 *and* t_2*) and* $t_S = < l_S, t'_1, m_S, t'_2, r_S >$ *(with tags* t'_1 *and* t'_2*) is defined as:*

$$Match(t_P, t_S) = \begin{cases} l_P \cdot l_S + m_P \cdot m_S + r_P \cdot r_S & if\ the\ tags\ match \\ 0 & otherwise \end{cases}$$

According to the similarity between tested sentence and data source, snowball system will generate new pattern and add it into the data source. The pattern included in left, mid and right part of tested sentence.

In my system, uses' specific goals can not be tested by the named-entity tagger algorithm. So I use the “resolu” and mid part to divide sentence. The result of this phase is four part sentence, the left, resolut, mid, rest part. Compare to five part structure, the rest part include the users' specific goals and right part. Because it is hard to distinguish them, which has little influence on the final rank result.

To compare the similarity between two sentence, I use the single pass clustering technique. I use the frequency of each word on the data set as the weight of each word. If “smoke” appears 60 times on the source data set, the weight of “smoke” will be 60. If the word appear on the middle part it will be two times of word weight. If the word appear on the left or middle part, the value will be 0.2 * (word weight). The mid part always more important than other part for the users' goal identify. Secondly, the word weight should be add together to get the weight of each part. The “mid” and “left” part will be add together. Then there are four parts with weight on both tested sentence and sentence from data source set. The single pass clustering technique will be used to computer the final similarity score of two sentence.

The tested sentence will be compared with every sentence of the data set. The highest score will be show the highest similarity. Then the most similarity sentence structure will be used to divide the tested sentence. The tested sentence should use the “resolu” and “mid” part to divided into four part. The rest part contain the users' specific goals.

Goal classifier

After we get the users' specific goals, we should classifier those goal and get the rank of each class. The sentence sentimental algorithm is good tools to classify, but the number of classes should be set before training and testing. The user's specific goals are different from different data set. So the algorithm is not suitable. We will test the goal similarity to classify by using sentence similarity algorithm. There are three classes of measures for sentence similarity, word overlap measures, TF-IDF measures and Linguistic Measures. The word overlap measure is mainly compare the common subsequence word between two sentence. In our system, we should compare the rest parts which contain many other words besides the users' specific goals. So the word overlap is not suitable. The Linguistic measures should compare the syntactic structure between two sentence. But the users' goals may have great different syntactic structure with the same meaning. Therefore, the Linguistic measures is not suitable.

TF-IDF this algorithm compare the overlap word between two sentence, considering the word importance for specific topic. For exampel, *“Suppose we have a set of English text documents and wish to determine which document is most relevant to the query "the brown cow". A simple way to start out is by eliminating documents that do not contain all three words "the", "brown", and "cow", but this still leaves many documents. To further distinguish them, we might count the number of times each term occurs in each document and sum them all together; the number of times a term occurs in a document is called its' term frequency.*

However, because the term "the" is so common, this will tend to incorrectly emphasize documents which happen to use the word "the" more frequently, without giving enough weight to the more meaningful terms "brown" and "cow". The term "the" is not a good keyword to distinguish relevant and non-relevant documents and terms, unlike the less common words "brown" and "cow". Hence an inverse document frequency factor is incorporated which diminishes the weight of terms that occur very frequently in the document set and increases the weight of terms that occur rarely.”

TF-IDF algorithm defination:

The **inverse document frequency** is a measure of how much information the word provides, that is, whether the term is common or rare across all documents. It is the logarithmically scaled fraction of the documents that contain the word, obtained by dividing the total number of documents by the number of documents containing the term, and then taking the logarithm of that quotient.

$$\mathrm{idf}(t, D) = \log \frac{N}{|\{d \in D : t \in d\}|}$$

with

- N: total number of documents in the corpus
- $|\{d \in D : t \in d\}|$: number of documents where the term t appears (i.e., $\mathrm{tf}(t, d) \neq 0$). If the term is not in the corpus, this will lead to a division-by-zero. It is therefore common to adjust the denominator to $1 + |\{d \in D : t \in d\}|$.

Mathematically the base of the log function does not matter and constitutes a constant multiplicative factor towards the overall result.

Then tf–idf is calculated as

$$\mathrm{tfidf}(t, d, D) = \mathrm{tf}(t, d) \times \mathrm{idf}(t, D)$$

A high weight in tf–idf is reached by a high term frequency (in the given document) and a low document frequency of the term in the whole collection of documents; the weights hence tend to filter out common terms. Since the ratio inside the idf's log function is always greater than or equal to 1, the value of idf (and tf-idf) is greater than or equal to 0. As a term appears in more documents, the ratio inside the logarithm approaches 1, bringing the idf and tf-idf closer to 0.

Before computing, we should get the common words of two rest part by using the hashMap, because the sequence of word is not importance. To compute the TFIDF value of the common words, we will use the document account of each word, which we get from the "message parse" phase. After we get the tfidf value of each word in each users' specific goal, we will use the vector-space model to compute the similarity of two part. If the result of vector-spece is bigger than 0.05, the two goals belong to the same class.

The classify process starts from the first sentence in the date set. The first sentence will be used to compare with the rest sentence and get the similar goals. Then remove the first sentence and other sentence with the similar goals with the first sentence. After that, the second sentence will used to test the rest sentence and pick the second class out. This process will be iterate until the source data set is empty.

goals classifier trainning

Rank resolution

3. Experimental setting

screen scrapping

4. Experimental results

5. Conclusion

This paper present twitter message extraction system for special topic “new year resolution”. We introduced novel method to analysis the users' new year resolution and get the rank and distribution of those goals. In the future, the system can track how the users archive their goals and analysis those information to get statistic data. The user suggestion is also good direction. The system will analysis the similarity between the achievement process of two users to get suggestion when the users run into trouble of their new year resolution. It will help people to become better themselves.

drawback

6. References

1. Snowball: Extracting Relations from Large Plain-Text Collections. Eugene Agichtein, Luis Gravano, 2000.
2. Sergey Brin. Extracting patterns and relations from the World-Wide Web. In Proceedings of the 1998 International Work-shop on the Web and Databases(WebDB'98), March 1998.
3. The Evaluation of Sentence Similarity Measures . Palakorn Achananuparp, Xiaohua Hu1 , and Shen Xiajiong
4. Scraper: https://chrome.google.com/webstore/detail/scraper/mbigbapnjcgaffohmbkdlecaccepngjd
5. lingpipe lib: http://alias-i.com/lingpipe/
6. The Porter Stemming algorithm: http://tartarus.org/martin/PorterStemmer/
7. Single Pass Clustering Technique: http://facweb.cs.depaul.edu/mobasher/classes/csc575/assignments/single-pass.html

7. tf–idf: http://en.wikipedia.org/wiki/Tf%E2%80%93idf