Identifying Similarities between Tweets

Iuliana Minea
Faculty of Computer Science,
"Alexandru Ion Cuza"
University of Iasi
General Berthelot, No. 16
iuliana.minea@info.uaic.ro

Adrian Iftene
Faculty of Computer Science,
"Alexandru Ion Cuza"
University of Iasi
General Berthelot, No. 16
adiftene@info.uaic.ro

ABSTRACT
In this paper, an application built by us with aim to provide users the possibility to explore topics from Twitter and find out people's opinion even if it is a positive one or a negative one is presented. The result of searched topic was divided into groups, based on the named entity from tweets. Using algorithms that calculate distance between two strings, similar tweets was removed. More than that, users have the opportunity to see a sentiment analysis of tweets.

Author Keywords
Twitter; information retrieval; strings similarity.

ACM Classification Keywords
H.5.m. Information interfaces and presentation (e.g., HCI): Miscellaneous.

General Terms
Human Factors; Design.

INTRODUCTION

In 2012, more than 100 million users posted 340 million tweets a day and the service handled an average of 1.6 billion search queries per day [20]. In 2013, it was one of the ten most-visited websites and has been described as "the SMS of the Internet" [2]. As of February 2017, Twitter had more than 313 million monthly active users [19]. On the day of the 2016 U.S. presidential election, Twitter proved to be the largest source of breaking news, with 40 million tweets sent by 10 p.m. that day [13].

Every day, millions of people use Twitter to create, discover and share ideas with other. Now, people are turning to Twitter as an effective way to reach out to businesses, too. From local stores to big brands, and from brick-and-mortar to internet-based or service sector, people are finding great value in the connections they make with businesses on Twitter [1].

There are many great business uses for Twitter, like sending out news briefs or advertising the latest job opening. But believe it or not, there are even more personal uses for Twitter. With this round up, consider the seven that follow [9].

SIMILAR APPLICATIONS

IceRocket [4] is generally for blog searches, but it offers the possibility to search news on Twitter. Twitter search returns most recent tweets that relate to your query. If the query is also a user, it shows a fact box about the user, along with tweets by that user. This application lets you reply directly to the tweets. The IceRocket site is a free resource for people looking to monitor their brand, it is ad supported. IceRocket has an API that it licenses to social media monitoring firms as well as PR agencies [21].

Twitonomy [18] is an online platform and in order to use it you have to connect with your Twitter account. The user has the opportunity to monitor his account or any other Twitter user, along with lists and any keyword search he wants to watch. Twitonomy acts like a basic Twitter client. You can reply, retweet and favorite individual tweets. As can be observed in image 2 from below, this application has all kinds of statistics about your own activity. You can see things like: tweets per day, how many links you've shared, how often others mentioned you in their tweets, which of your tweets were retweeted the most [10].

APPLICATION DESCRIPTION

This application aims to offer users the possibility to follow news on Twitter, without having to read duplicate topics (this paper represent an extended version of results presented in [5, 11]). In order to remove similar news, it was used a similarity algorithm which calculates distance between two tweets. In the first phase, it was made an analysis, in terms of time and accuracy, of the next four similarity distance algorithms: *Levenshtein*, *Needleman-Wunsch*, *Jaro-Winkler* and *Smith-Waterman*. Upon review of the analysis result, Smith-Waterman turned out to be more competent to find similarities.

When the user performs a search in application, it is done a request to Twitter API in order to retrieve the latest and the most popular tweets. The search result was divided into subcategories. Each category was extracted from tweets and can be: *a location, a person, an organization* or *a date*. Also tweets was analyzed from the sentiment point of view. A tweet can be categorized as *positive, negative* or *neutral*.

Similar Algorithms

In computer science and statistics, the **Jaro–Winkler distance** is a measure of similarity between two strings [6, 22]. It is a variant of the **Jaro distance** metric a type of string edit distance, and was developed in the area of record linkage (duplicate detection). The lower the Jaro–Winkler

distance for two strings is, the more similar the strings are. The similarity score is normalized such that 0 equates to no similarity and 1 is an exact match. The Jaro score d_j of two given strings s_1 and s_2 is:

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

where m is the number of matching characters, t is half the number of transpositions (the number of matching characters (but different sequence order) divided by 2).

Jaro-Winkler distance uses a prefix scale p which gives more favorable ratings to strings that match from the beginning for a set prefix length l. Given two strings s_1 and s_2, their Jaro-Winkler distance d_w is:

$$d_w = d_j + (lp(1 - d_j))$$

where: d_j is the Jaro distance for string s_1 and s_2, l is the length of common prefix at the start of the string up to a maximum of 4 characters, p is a constant scaling factor for how much the score is adjusted upwards for having common prefixes (the standard value for this constant is Winkler's work is $p = 0.1$).

The **Levenshtein distance** is a string metric for measuring the difference between two sequences [8]. Informally, the Levenshtein distance between two words is the minimum number of single-character edits (i.e. insertions, deletions or substitutions) required to change one word into the other. Mathematically, the Levenshtein distance between two strings a, b (of length $|a|$ and $|b|$ respectively) is given by $lev_{a,b}(|a|, |b|)$ where:

$$lev_{a,b}(i,j) = \begin{cases} \max(i,j) & if\ \min(i,j) = 0 \\ \min \begin{cases} lev_{a,b}(i-1,j) + 1 \\ lev_{a,b}(i,j-1) + 1 \\ lev_{a,b}(i-1,j-1) + 1_{a_i \neq b_j} \end{cases} & otherwise \end{cases}$$

where $1_{a_i \neq b_j}$ is the indicator function equal to 0 when $a_i = b_j$ and equal to 1 otherwise, $lev_{a,b}(i, j)$ is the distance between the first i characters of a and the first j characters of b.

The **Needleman-Wunsch algorithm** is an algorithm used in bioinformatics to align protein or nucleotide sequences [7, 12]. It was one of the first applications of dynamic programming to compare sequences. The Needleman-Wunsch algorithm, which is based on dynamic programming guarantees finding the optimal alignment of pairs of sequences. The algorithm essentially divides a large problem (e.g. the full sequence) into a series of smaller problems and uses the solutions to the smaller problems to reconstruct a solution to the larger problem.

In order to perform a Needleman-Wunsch alignment, a matrix is created which allows us to compare the two sequences. The first line and the first column from the matrix was initialized with line index, respectively column index. The score of the rest cells was calculated as follows:

$$M[i][j] = min \begin{cases} M[i-1][j] + gap \\ M[i][j-1] + gap \\ M[i-1][j-1] + cost \end{cases}$$

where: *gap* is the gap penalty (in this implementation the gap has value 2), *cost* was 0 if the characters matches and 1 otherwise. The Needleman-Wunsch distance has a value between 0 (identical strings) and 1 (different strings), and is calculated based on the next formula:

$$D = \frac{M[|s_1|][|s_2|]}{\max(|s_1|, |s_2|) * gap}$$

where $|s_1|$, $|s_2|$ is the length of s_1, respectively s_2 string.

The **Smith-Waterman algorithm** is a dynamic programming method for determining similarity between nucleotide or protein sequences [16]. The algorithm was first proposed in 1981 by Smith and Waterman and is identifying homologous regions between sequences by searching for optimal local alignments. The Smith-Waterman algorithm is built on the idea of comparing segments of all possible lengths between two sequences to identify the best local alignment. It is based on calculation of local alignments instead of global alignments of the sequences and allowing a consideration of deletions and insertions of arbitrary length [16]. The Smith-Waterman algorithm is the most accurate algorithm when it comes to search databases for sequence homology, but it is also the most time consuming [16].

Named Entity Recognizer Module

This module deals with extracting information, localize and classify named entities in tweets into pre-defined categories such as the names of persons, organizations, locations, expressions of times, etc. For all these operations it was used "Stanford Named Entity Recognizer (NER)" library with 7 class model.

Named Entity Recognition (NER) labels sequences of words in a text which are the names of things, such as person and company names, or gene and protein names. It comes with well-engineered feature extractors for Named

Entity Recognition, and many options for defining feature extractors. Stanford NER is also known as CRFClassifier. The software provides a general implementation of (arbitrary order) linear chain Conditional Random Field (CRF) sequence models [17].

Tweets Classifier Module

Formerly known as Twitter Sentiment, Sentiment140 [15] is a service that lets users discover the current sentiment around a brand, product or topic on Twitter. Sentiment140 uses classifiers based on machine learning algorithms and allow users to see the classification of individual tweets. The API lets users classify tweets and integrate sentiment analysis classifier functionality into their own websites or applications. The API uses RESTful calls and responses are formatted in JSON. As can be seen in the table 3, the response is the same as the request, except a new field "polarity" added to each object. The polarity values are: 0: *negative*; 2: *neutral*; 4: *positive* [14].

The approach is to use different machine learning classifiers and feature extractors. The machine learning classifiers are Naive Bayes, Maximum Entropy (MaxEnt), and Support Vector Machines (SVM). The feature extractors are unigrams, bigrams, unigrams and bigrams, and unigrams with part of speech tags. We build a framework that treats classifiers and feature extractors as two distinct components [3].

The emoticons are stripped out from training data. If the emoticons are left in, there is a negative impact on the accuracies of the MaxEnt and SVM classifiers, but little effect on Naive Bayes. The difference lies in the mathematical models and feature weight selection of MaxEnt and SVM. Stripping out the emoticons causes the classifier to learn from the other features (e.g. unigrams and bigrams) present in the tweet. The classifier uses these non-emoticon features to determine the sentiment [3].

Best Similarity Algorithm

In the first phase of application development was aimed to select the best similarity algorithm. The four algorithms introduced above were applied on a set of 2,000 tweets, tweets which were saved in XML files. For detecting which tweets from the entire source data are similar, each tweet was compared with all that follow it.

In the next table can be observer how long did the execution of each algorithm. In order to improve the execution time, it was used a caching mechanism from Microsoft. In cache was kept the comparison result for every algorithm and the key format is: "*AlgorithmName_*TextTweetOne_*TextTweetTwo*".

Another way to improve the comparisons accuracy and time was to remove the stop words (most common words like "the", "and", "an" etc.) from tweets. After implementing this step and with the help of caching mechanism, the time was improved (see Table 1).

	Without cache	**With cache**	**With cache after removing stop words**
Jaro-Winkler	06:05.35	03:50.11	01:46.06
Levenshtein	22:35.37	12:10.08	11:26:82
Needleman-Wunsch	51:11.47	34:21.63	21:22.49
Smith-Waterman	39:46.32	35:33.20	23:51.46

Table 1. Algorithms execution time.

Three of the algorithms find 505 tweets with distance 0 (tweets are similar), while Smith-Waterman find 569 tweets with distance 0 (it sees a tweet and a retweet being the same). In the end, we decided to use Jaro-Winkler algorithm to group similar tweets due to its fast running time.

In the application interface can be observed how the groups are display and the tweets that were classified as positive have a green background and the negative ones have a red background (see Figure 1).

Figure 1. Viewing clusters on Google Maps.

CONCLUSION

Although there are some applications which extract tweets and perform a sentiment analysis ("Social Mention", "Twitter Sentiment Visualization"), "News on Twitter" application keeps those benefits and provides other functionality: tweets which contain your searched data, but also grouped by relevant information like: *date, location, person* and *organization*. Having these clusters, user can decide to read only some topics and not all the tweets returned by API.

When a tweet contains a link, the url is altered to 23 characters even if the link itself is less than 23 characters long. So it is possible to have two tweets that refer to the same thing, have the same destination url, but the links are displayed differently. Because of this thing, Smith-Waterman algorithm can find these two tweets as different.

In order to improve the accuracy of the algorithm, before a comparison, the links can be removed from tweets.

Due to the modular structure, the application functionality can be very easily extended and provide functionality like: which tweets are most retweeted, monitor tweets from users etc.

ACKNOWLEDGMENT

This work is partially supported by POC-A1-A1.2.3-G-2015 program, as part of the PrivateSky project (P_40_371/13/01.09.2016).

REFERENCES

1. Ask a Ronlee: http://askaaronlee.com/10-reasons-why-your-business-should-use-twitter/ (Last accessed on 29 May 2017)
2. Business Standard: http://www.business-standard.com/article/technology/swine-flu-s-tweet-tweet-causes-online-flutter-109042900097_1.html (Last accessed on 29 May 2017)
3. Go, R., Bhayani, L. and Huang, L. Twitter Sentiment Classification Using Distant Supervision. *Stanford University*, Technical Paper (2009).
4. IceRocket: http://www.icerocket.com/ (Last accessed on 29 May 2017)
5. Iftene, A., Dudu, M. Ș. and Miron, A. R. Scalable system for opinion mining on Twitter data. Dynamic visualization for data related to refugees' crisis and to terrorist attacks. In *Proceedings of 26th International Conference on Information System Development*, ISD 2017, September 6-8, Larnaca, Cyprus (2017).
6. Jaro. M. A. Advances in record linkage methodology as applied to the 1985 census of Tampa Florida. *Journal of the American Statistical Association*, 84 (406), (1989), 414-420.
7. Lesk, A. *Introduction to Bioinformatics*. Oxford University Press (2002).
8. Levenshtein, V. I. Binary codes capable of correcting deletions, insertions, and reversals. *Soviet Physics Doklady*. 10 (8), (1966), 707-710.
9. Lifewire: https://www.lifewire.com/ways-for-beginners-to-use-twitter-3486595 (Last accessed on 29 May 2017)
10. Marketingland Twitonomy: http://marketingland.com/twitonomy-the-best-twitter-analytics-tool-youve-never-heard-of-55389 (Last accessed on 29 May 2017)
11. Minea, I. and Iftene, A. News on Twitter. In *Proceedings CMSM4 (The Fourth Conference of Mathematical Society of the Republic of Moldova)*, June 25 - July 2, Chisinau, Republic of Moldova (2017), 535-538.
12. Needleman, S. B. and Wunsch, C. D. A general method applicable to the search for similarities in the amino acid sequence of two proteins. *Journal of Molecular Biology*, 48 (3), (1970), 443-453.
13. New York Times: https://www.nytimes.com/2016/11/09/technology/for-election-day-chatter-twitter-ruled-social-media.html (Last accessed on 29 May 2017)
14. Programmable Web: https://www.programmableweb.com/api/sentiment140 (Last accessed on 29 May 2017)
15. Sentiment 140: http://help.sentiment140.com/api (Last accessed on 29 May 2017)
16. Smith, T. F. and Waterman, M. S. Identification of Common Molecular Subsequences. *Journal of Molecular Biology*, 147 (1981), 195-197.
17. Standford CRF: http://nlp.stanford.edu/software/CRF-NER.shtml (Last accessed on 29 May 2017)
18. Twitonomy: http://www.twitonomy.com/?gclid=CO-gi_fZ8dECFQuMGQodbgYCkQ (Last accessed on 29 May 2017)
19. Twitter Company: https://about.twitter.com/company (Last accessed on 29 May 2017)
20. Twitter: https://blog.twitter.com/2012/twitter-turns-six (Last accessed on 29 May 2017)
21. Wikipedia IceRocket: https://en.wikipedia.org/wiki/IceRocket (Last accessed on 29 May 2017)
22. Winkler, W. E. String Comparator Metrics and Enhanced Decision Rules in the Fellegi-Sunter Model of Record Linkage. *Proceedings of the Section on Survey Research Methods*. American Statistical Association (1990), 354-359.