A Topic based Approach for Sentiment Analysis on Twitter Data

Pierre FICAMOS*, Yan LIU[Ψ]
School of Software Engineering
TONGJI University
Shanghai, China

***Abstract*—Twitter has grown in popularity during the past decades. It is now used by millions of users who share information about their daily life and their feelings. In order to automatically process and analyze these data, applications can rely on analysis methods such as sentiment analysis and topic modeling. This paper contributes to the sentiment analysis research field. First, the preprocessing steps required to extract features from Twitter data are described. Then, a topic based method is proposed so as to estimate the sentiment of a tweet. This method requires to extract topics from the training dataset, and train models for each of these topics. The method allows to increase the accuracy of the sentiment estimation compared to using a single model for every topic.**

Keywords—sentiment analysis; opinion mining; natural language processing; feature extraction; topic modeling

I. INTRODUCTION

Twitter is a social network which allows its users to post and share short messages (up to 140 characters) called tweets[1]. Over the past decades, Twitter has spread worldwide and has become one of the major social networks. Overall, social media are growing more and more popular, they are now one of the main way of communication for both people and companies. Twitter's slogan is: "Twitter it's what's happening". Indeed, many users are sharing about events of their daily life. Hence, following Twitter's flow of data may allow to monitor events which are occurring and understand people's feelings.

In order to automatically process Twitter data, several data analysis methods, such as sentiment analysis and topic modeling, can be applied. The outcomes of these analyses may be used by several applications, such as event monitoring, and opinion mining about products or brands. Indeed, companies always need fast and accurate information in order to be able to react the market trends.

This paper proposes a sentiment analysis method: First, topics are extracted from the training dataset. Then an algorithm is trained for each topic. Finally, the method estimates the sentiment of a sentence according the best topic related algorithm results.

This paper's contributions contain: 1) Detail the possible steps for data preprocessing in order to extract bags of words from Twitter data. 2) Propose a sentiment analysis method which relies on tweets topics for improving the estimation accuracy.

This paper is organized as follows. Section II describes related works on the sentiment analysis research field. Then, the section III introduces the preprocessing steps which are applied in order to extract bags of words from the samples. The method applied so as to estimate the sentiment of a tweet is explained in section IV, and section V presents the evaluation of this method. Finally, the section VI concludes and introduces future works.

II. RELATED WORK

A. General Research Field

Sentiment analysis research field is anterior to the social network era. Most of the first studies led on sentiment analysis were focusing on review, such as movie review, since reviews are often associated to a score. Hence, it was simple to get the sentiment score of a review. The researchers did not have to manually label their datasets.

The first approaches for extracting sentiments from texts were relying on human generated baseline. These methods were not able to handle the complexity of the language, and were providing low accuracy results. Indeed, a random approach for classifying a text between positive and negative would already be 50% accurate. The human generated baselines seems to have difficulty to provide better accuracy than 70% for the sentiment prediction [1], [2].

One of the most popular approach for sentiment analysis is to rely on supervised machine learning techniques. The three main machine learning algorithms which are applied for sentiment analysis are: Naive Bayes [3], Maximum Entropy [4], and Support Vector Machine (SVM) [5]. The accuracy of these three algorithm depends on the feature extraction method which is applied and the analyzed datasets. For example, SVM shows better performance when it only use unigrams, and adding bigrams features will reduce its accuracy [1], [2].

According to anterior research, the feature extraction methods have to take the following specificity into account:

- **Presence is better than Frequency**: Two possible methods for extracting features from a text are either to generate a bag of words which contains each words present in the text, or to count the frequency at which each word appears in the text. Previous researches have shown that analyses are more accurate when focusing on the word presence [1].

[1]https://twitter.com/

- **Negation Handling:** Negation allows to alter the meaning of a word to its opposite meaning. Therefore, during the feature extraction, it is important to indicate process whether or not a word is negated [3]. If the negation is not handling the algorithm will understand the opposite meaning of the sentence, and will have less accurate predictions.
- **Bigrams:** Several feature extraction methods will use bigrams in order to extract a more accurate representation of the sample [2], [3]. Indeed, n-grams allow to capture the context of a word, thus it allows the algorithms to be more accurate.
- **Part of Speech (POS) tags:** POS (Part Of Speech) tags are representation of the lexical category of a word [6]. Part of speech may allow to disambiguate words meaning [1], and it may also be used so as to generate pattern for extracting features from the samples [7].
- **Lemmatizing / steaming word:** Both lemmatization and steaming allow to ignore the possible variation of a word. These methods are often applied for extracting features from a text. They allow to reduce the amount of features generated, and regroup similar features [8].

B. Sentiment Analysis on Twitter

Since social network has become more and more popular, several researchers have been focusing on applying sentiment analysis on short text message. The majority of these researches are dealing with Twitter samples. Indeed, it is convenient to retrieve data from Twitter with its available APIs. Then, in order to train the algorithms, sentiment analysis requires to have annotated samples. Contrary to review, twitter data does not have associated scores, the data have to be manually annotated. An alternative method for annotating these data relies on the emoticons which are contained in the samples. Emoticons are used to convey the sentiment of the writer, thus it is possible to label every text which contains positive emoticons to positive and negative ones to negative [2], [9]. This method is convenient since it allows to automatically generate important set of data, but the dataset generation will be less accurate than a manually annotated dataset.

Performing feature extraction on Twitter messages rises new challenges:

- **Short messages:** Sentiment analysis is usually performed on longer text. Because of the text limitations, Twitter messages are short, and the algorithm has less features available for analysis.
- **Internet language:** Twitter users adopt the "internet language" when writing their messages. This language differs from the traditional English: new words, repeated letters [2], emoticons.
- **Twitter characteristics:** Twitter allows users to add three specific entities to their messages: hashtags, user references, and URLs. These entities require to be processed differently than common words.

III. DATA PREPROCESSING

The preprocessing steps aim to begin the feature extraction process and start extracting bags of words from the samples. One of the main focus is to reduce the final amount of features extracted. Indeed, features reduction is important in order to improve the accuracy of the prediction for both topic modeling and sentiment analysis. Features are used to represent the samples, and the more the algorithm will be trained for a specific feature the more accurate the results will be. Hence, if two features are similar it is convenient to combine them as one unique feature. Moreover, if a feature is not relevant for the analysis, it can be removed from the bag of words.

- **Lower uppercase letters:** The first step in the preprocessing is to go through all the data and change every uppercase letter to their corresponding lowercase letter. When processing a word, the analysis will be case sensitive and the program will consider "data" and "Data" as two totally different words. It is important that, these two words are considered as the same features. Otherwise, the algorithms will affect sentiments which may differ to these two words. For example, on these three sentences: "data are good", "Awesome data", and "Bad Data". The first and second sentences both contain "data" and are positive, the third sentence contains "Data" and is negative. The algorithm will guess that sentences containing "data" are more likely to be positive and those containing "Data" negative. If the uppercases had been removed the algorithm would have been able to guess that the fact that the sentence contains "data" is not very relevant to detect whether or the sentence is positive. This preprocessing step is even more important since the data are retrieved from Twitter. Social media users are often writing in uppercase even if it is not required, thus this preprocessing step will have a better impact on social media data than other "classical" data.
- **Remove URLs and user references:** Twitter allows user to include hashtags, user references and URLs in their messages. In most cases, user references and URLs are not relevant for analyzing the content of a text. Therefore, this preprocessing step relies on regular expression to find and replace every URLs by "URL" and user reference by "AT_USER", this allows to reduce the total amount of features extracted from the corpus [2]. The hashtags are not removed since they often contain a word which is relevant for the analysis, and the "#" characters will be removed during the tokenization process.
- **Remove digits:** Digits are not relevant for analyzing the data, so they can be removed from the sentences. Furthermore, in some cases digits will be mixed with words, removing them may allow to associate two features which may have been considered different by the algorithm otherwise. For example, some data may contain "iphone", when other will contain "iphone7". The tokenization process, which will be introduced later,

will not separate "iphone" from "7", so these two features will be considered as different. Removing the "7" will allow the features representing these two words to be the same.

- **Remove stop words:** In natural language processing, stop words are often removed from the sample. These stop words are words which are commonly used in a language, and are not relevant for several natural language processing methods such as topic modeling and sentiment analysis [10]. Removing these words allows to reduce the amount of features extracted from the samples.
- **Remove repeated letters:** This preprocessing step refers to the fact that Twitter users will often repeat some letters several times when they want to highlight a word [2]. For example, the following tweets contain the word "love" with several repeated letters: "I looooovvveee that!". Repeated letter will be reduced to their first two occurrences. Hence, for the previous example "looooovvveee" will become "loovvee".
- **Tokenize:** Tokenization is almost implicit since the English language is already segmented. Each word is separated by space, thus the token can be created by splitting the sentence on each space. The tokenization applied for this project also include other functionalities such as separated punctuation tokens from the word. The experiments will use the NLTK word_tokenize method for tokenizing its samples [11].
- **Detect POS tags:** Part of speech may have two uses for data analysis. First, it may be used so as to disambiguate the meaning of a word. Even if it makes sense for a reader that in the sentences: "I like that" and "I am not like you", "like" have two different meaning, when computing the bag of words, the algorithm will consider them as the same [1]. The second use for POS tags is to allow to categorize words and process them differently according to which type they correspond to [7]. The detection of the samples POS tags will rely on the NLTK method pos_tag [11].
- **Lemmatize:** When processing samples, "word" and "words" would be considered as two different features. Hence, in order to improve the features reduction process, the unigrams can be lemmatized. This preprocessing step mainly allows to remove plurals and conjugations. The lemmatization will be based on the WordNet implementation which is including in the NLTK distribution [11].

IV. METHODOLOGY

This paper proposes an approach which can be extended to several sentiment analysis problems. The concept of this approach is that sentiment analysis algorithms can perform better when the data, which are processed, deal with a less wide category of topic. Hence, topic modeling techniques may help to divide data into several datasets. The vocabulary diversity of these dataset will be inferior to the original data, thus training the sentiment analysis is more simple.

After the data preprocessing, the tweets are represented by bags of words. These bags of words contain words which has been lemmatized and their associated POS tags. The proposed method uses this bag of words in order to extract topics from the text and train its algorithms.

A. Topics extraction

The first requirement of this method is to extract topics from the samples. Topic extraction can either be supervised or unsupervised. Supervised topic extraction require to have manually analyzed the training dataset, and to associate a topic to each tweet. Therefore, unsupervised topic modeling technique is more simple to implement and will be applied for this paper.

First, features have to be extracted from the training data set. The testing dataset is ignored for the topics extraction process since, in real application cases, these testing data would not be available. In order to represent the samples: nouns, verbs, adjectives, adverbs and interjections are extracted from the bags of words according to their POS tags. By empirical study, this feature extraction method has provided better topic distribution for applying sentiment analysis. Then, these features are used to train the model and extract topics from the samples.

The topic modeling model which is applied is a Latent Dirichlet Allocation (LDA), the core estimation is based on the algorithm of Hoffman *et al.* [12]. This model has been chosen since it allows inference of topic distribution. Other equivalent model could also be applied.

B. Train the Algorithms

After having extracted the topics from the samples, the training dataset can be split in several subsets. The topics extraction process provides a probability distribution of the topics for each sample. Hence, a sample may be associated to several topics. The training process takes into account this probability distribution: each topic subset contains all the samples which probability distribution is superior to a threshold. Thus, some samples may be contained in several subsets, when others may be contained by only one.

```
FOR each topic related to the tweet
  FOR each sample
    estimate the topic probability distribution
    IF probability > threshold THEN
      ADD sample to the training subset
    END IF
  END FOR
  extract features from the subset
  train algorithm
END FOR
```

Fig. 1. Algorithms training process (pseudocode)

Once all the subsets have been generated, sentiment analysis algorithms can be trained for each of these subsets. The training of these algorithms requires to extract features from the subsets. After the preprocessing, the samples are already represented by bags of words, thus features can be directly extracted from these bags of words. The feature extraction method which have been applied allows to extract unigrams from the bag of words, and handle negation words.

Other alternative feature extraction could also be applied. The pseudocode of the training process is detailed in Fig 1.

C. Sentiment estimation

Since a tweet may be associated to several topics and a sentiment analysis algorithm has been trained for each of these topics, a method for exploiting the results of all of these algorithms needs to be defined.

In order to estimate the sentiment of a tweet, the sentence is first preprocessed, and then features are extracted from the bag of words. Finally, the topic probability distribution of the tweet is estimated. For each topic whose probability is superior to a threshold, the algorithm trained for this specific topic is applied in order to estimate the sentiment of the tweet. The estimation which has the highest probability is kept as the final estimation. This estimation process is described as a pseudocode in Fig 2.

```
FOR each topic related to the tweet
  IF topic probability distribution > threshold THEN
    estimate the sentiment of the tweet
  END IF
END FOR
sentiment = maximum probability of the estimation
```

Fig. 2. Sentiment estimation process (pseudocode)

V. Evaluation

The experiments of this paper mainly rely on the NLTK libraries [11]. As it has been introduced in the third section, several preprocessing steps rely on these libraries. Furthermore, the algorithm which will be applied for this experiments, Naive Bayes (NB), is also implemented by NLTK. Other sentiment analysis algorithm such as maximum entropy or SVM could also have been selected.

So as to train the algorithm and estimate its accuracy, the experiments will process the same dataset which was introduced by Go *et al.* [2]. The data have been retrieved from Twitter: the training set of 1.6 million tweets, was annotated according their emoticons, and the testing set composed of 177 negative tweets and 182 positive ones were manually annotated. Because of the processing time required for training the algorithms, the training dataset has been reduced to 5 000 positive tweets, and 5 000 negative ones. Three dataset have been generated by randomly extracting samples from the complete dataset. Then the experiments have been conducted on these three dataset. The results obtained are the average results of these three experiments.

The experiment focuses on two parameters: N the total amount which will be extracted from the samples, and ξ the threshold for the topic probability distribution. The aim is to demonstrate that these two parameters are linked, and the correct combination of these parameters allows to increase the global estimation accuracy.

First, the method has been applied with one unique topic (N = 1). The results of this first experiment will be used as a reference for the other experiments. Indeed, with only one topic, this method corresponds to the basic sentiment analysis method. This method has provided 74.09% accurate results. When the size of training dataset is increased to its maximum, this method can provide 81.34% accuracy which is consistent with the results obtained by Go et al. [2].

TABLE I. Average Accuracy of the Estimation for N and ξ Given

ξ / N	2%	4%	6%	8%	10%
1	74.09	74.09	74.09	74.09	74.09
2	74.09	74.06	74.22	74.16	**74.25**
3	74.05	74.08	74.12	74.22	**74.59**
4	74.50	74.24	**74.61**	74.35	73.64
5	74.22	74.31	**74,95**	73.48	73.51
6	73.59	**74,20**	73.28	73.20	73.44
7	74.58	**74.86**	73.53	73.52	73.57
8	74.39	**75.17**	73.45	73.40	73.24
9	73.44	**74.07**	73.31	73.23	73.29
10	**73.54**	73.17	73.14	73.16	73.13

Then different combinations have been tried for N and ξ, the results are shown in Table I. Since LDA model uses randomness in the topic extraction process, the estimations have been repeated ten times, and the final results correspond to the average of these results.

According to these results, it can be estimated that as N increases, the threshold ξ needs to be reduced. This observation can be explained: the more the amount of topics increase, the more spread are the distributions of topics, thus the threshold needs to be decreased.

VI. Conclusion and Future Work

This paper's first contribution to the sentiment analysis research field is to detail the preprocessing steps which have to be applied to extract bags of words from Twitter data. The second contribution is to propose a topic based sentiment analysis approach. This approach relies on the fact that the complexity of an analysis can be reduced when the algorithm focus on a smaller range of topic. The algorithms have to deal with less vocabulary and its estimation can be more accurate. Hence, the experiments have shown that, applying this method performs better than the classical sentiment analysis model. The proposed approach can be extended to other sentiment analysis problems.

The method can be improved. The paper has focused on exploiting the results of the default parameter for the topic modeling method. More research on the topic extraction may allow to have more distinct topics, hence the estimation should be more accurate.

Moreover, the method proposed here cannot handle neutral data. In order to apply sentiment analysis on real application cases, neutral data need to be handled. Therefore, future works should focus on these neutral data. Two methods may allow to detect neutrality:

- Use supervised topic modeling techniques to differentiate opinionated data from the neutral data.
- Adapt the sentiment analysis method for detecting either opinionated data or neutral data.

REFERENCES

[1] B. Pang, L. Lee, and S. Vaithyanathan, "Thumbs up?: sentiment classification using machine learning techniques," in Proceedings of the ACL-02 conference on Empirical methods in natural language processing, vol. 10, pp.79-86, Association for Computational Linguistics, July 2002.

[2] A. Go, R. Bhayani, and L. Huang, "Twitter sentiment classification using distant supervision," CS224N Project Report, Stanford, 1, 12, 2009

[3] V. Narayanan, I. Arora, and A. Bhatia, "Fast and accurate sentiment classification using an enhanced Naive Bayes model," in International Conference on Intelligent Data Engineering and Automated Learning, pp. 194-201. Springer Berlin Heidelberg, October 2013.

[4] A. L. Berger, V. J. D. Pietra, and S. A. D. Pietra, "A maximum entropy approach to natural language processing" Computational linguistics, 22(1), 39-71, 1996.

[5] C. J. Burges, "A tutorial on support vector machines for pattern recognition," Data mining and knowledge discovery, 2(2), 121-167, 1998.

[6] E. Brill, "A simple rule-based part of speech tagger," in Proceedings of the workshop on Speech and Natural Language, pp. 112-116, Association for Computational Linguistics, February 1992.

[7] P. D. Turney, "Thumbs up or thumbs down?: semantic orientation applied to unsupervised classification of reviews," in Proceedings of the 40th annual meeting on association for computational linguistics, pp. 417-424, Association for Computational Linguistics, July 2002.

[8] M. Hu and B. Liu, "Mining Opinion Features in Customer Reviews," in AAAI, vol 4, no. 4, pp. 755–760, July 2004.

[9] A. Pak, and P. Paroubek, "Twitter as a Corpus for Sentiment Analysis and Opinion Mining," in LREc, vol. 10, pp.1320-1326, May 2010.

[10] C. Fox, "A stop list for general text," *ACM SIGIR Forum*, vol. 24, no 1-2, pp. 19–21, September 1989.

[11] S. Bird, E. Loper and E. Klein "Natural language processing with python," O'Reilly Media Inc. 2009.

[12] M. Hoffman, D.M. Blei, and F.R. Bach, "Online learning for latent dirichlet allocation," in advances in neural information processing systems, pp. 856-864, 2010.