Opinion Mining over Twitterspace: Classifying Tweets Programmatically using the R Approach

Jinan Fiaidhi, Osama Mohammed*, Sabah Mohammed
Department of Computer Science
*Department of Software Engineering
Lakehead University, Thunder Bay,
Ontario P7B 5E1, Canada
{jfiaidhi, omohamme, mohammed}@lakeheadu.ca

Simon Fong#, Tai hoon Kim+
# Faculty of Science and Technology
University of Macau, Macau, China
ccfong@umac.mo
+ Department of Computer Engineering, Glocal Campus,
Konkuk University, Korea
taihoonn@kku.ac.kr

Abstract— Today the channels for expressing opinions seem to increase daily. When these opinions are relevant to a company, they are important sources of business insight, whether they represent critical intelligence about a customer’s defection risk, the impact of an influential reviewer on other people’s purchase decisions, or early feedback on product releases, company news or competitors. Capturing and analyzing these opinions is a necessity for proactive product planning, marketing and customer service and it is also critical in maintaining brand integrity. The importance of harnessing opinion is growing as consumers use technologies such as Twitter to express their views directly to other consumers. Tracking the disparate sources of opinion is hard – but even harder is quickly and accurately extracting the meaning so companies can analyze and act. Tweets’ Language is complicated and contextual, especially when people are expressing opinions and requires reliable sentiment analysis based on parsing many linguistic shades of gray. This article argues that using the R programming platform for analyzing tweets programmatically simplifies the task of sentiment analysis and opinion mining. An R programming technique has been used for testing different sentiment lexicons as well as different scoring schemes. Experiments on analyzing the tweets of users over six NHL hockey teams reveals the effectively of using the opinion lexicon and the Latent Dirichlet Allocation (LDA) scoring scheme.

Keywords-data mining; twitter; classification algorithms, sentiment analysis

1. INTRODUCTION

Twitter has become a very popular communication tool among microbloggers and Internet users to share opinions on different aspects of daily life. Much of that data is public and available for mining and it has fuelled the rapid growth of consumer-generated content such as consumer satisfaction, opinion extraction, ratings and sentiment analysis. Further, research suggests that the online purchase intent is significantly impacted by negative/positive sentiments found online [1]. Therefore, it makes a lot of sense that marketers and PR practitioners are spending a greater amount of time trying to measure the market opinion where the objective is to classify opinion according to a polar spectrum. The extremes on the spectrum usually correspond to positive or negative feelings about something, such as a product, brand, or person. However, an individual’s sentiment toward a brand or product may be influenced by one or more indirect causes and very likely may change over time according to a person’s mood, world events, and so forth. For this reason it is imperative to have a sufficiently sophisticated and rigorous enough methodology to look at data from the standpoint of time as well as from other analytical and contextual point of views. Certainly analyzing tweets is a daunting task due to the volumes of consumer generated data and the complexity of the ill format language used in writing the 140 character long tweets. Tweets are different from plain natural language sentences primarily because they are more casual (i.e. not as thoughtfully composed as movie reviews, for example). Moreover, tweets use variety of short hand symbols (e.g. Happy emoticons: “:-)”, “;-)”, “=)”, “:D” and Sad emoticons: “:-(”, “(“,”=-(“,”;-(“) and acronyms (e.g. FTW: (for the win)) which makes their analysis more difficult. However, some researchers argue that the ultimate goal for analyzing tweets is understand the online reputation of the object under analysis. In this direction there are many reputation management (ORM) toolkits (e.g. IRM1, RepAdvisor2, Klout3, TweetBeep4, Amazon Mechanical Trurk5) that use variety of techniques to neutralize negative publicity over the Internet including the Twitter. Actually, Marketer’s goals go beyond trying to fix business reputation to include the ability to reliably analyzing consumer satisfaction as well as analyzing their competiveness with similar vendors. However, many the marketers today rely on using primitive DIY (do-it-yourself) searching and tracking tools such as (e.g. Trackur6, Twitter Search Engine7, Hashtags8, Twitter Stream Graph9) to capture much of what’s being said about their brands, people, products, and competitive industries in real

1 http://www.internet-reputation-management.com/
2 https://twitter.com/#!/repadvisor
3 http://klout.com/home
4 http://tweetbeep.com/
5 https://www.mturk.com/mturk/
6 http://www.trackur.com/
7 https://twitter.com/#!/search-advanced
8 http://www.hashtags.org/
9 www.neoformix.com/Projects/TwitterStreamGraphs
time over the Twitter domain. Most of such DIY tools rely on using a simple keyword search and an RSS reader for content collection. The problem with using DIY techniques for tracking and analyzing consumer satisfaction is that we cannot go beyond simple steps to introduce measures for a specific tracking need for only specified period of time. Even if marketers try to use some more sophisticated tools and techniques the problem remains the same. For example some marketers may use the Yahoo Pipes to track group of related keywords or products [2] or the Venn Diagrams\(^{10}\) for tracking related tweets. Although these add-value DIY approaches provide estimates on some issues expressed in tweets, they are far from realistic as they rely on simple and traditional keyword classification techniques. More concrete approaches has been evolved based on analyzing the tweets based on natural language techniques to explore their contents and even to draw conclusions about their sentimental values (e.g. emotions such as "happy" "angry" and "sad").

Sentiment Analysis (SA) is an emerging field of analytics which focuses on identifying the polarity of a text and its extent on certain measurement dimensions. Besides measuring the polarity of the text, SA may also be used to understand the relevance and objectivity of the text [3]. SA has emerged due to the confluence of natural language processing systems and statistical analysis techniques. SA involves classifying opinions in text into categories like "positive" or "negative" often with an implicit category of "neutral". There are many other approaches and tools for detecting sentiment. Most of these approaches and tools\(^{11}\) use regression classifiers to arrive to some sort of sentimental analysis. In addition, there is no information provided by these tools on how to derive the sentimental summary. Most importantly, there is no way to scale or customize these tools for various types of sentiment analysis and comparison. This makes the ultimate solution for SA is to analyze tweets programatically. In this direction there are some general programming approaches that can be used for SA. Mostly they include text-analytics APIs – natural-language processing for information extraction and classification -- that can be applied for sentiment analysis (e.g. Python NLTK\(^{12}\), RapidMiner\(^{13}\), Gate\(^{14}\)). However, one can identify more sophisticated programming approaches involving machine learning techniques (e.g. WEKA data-mining workbench\(^{15}\) and the R platform\(^{16}\) ). Actually the R programming platform gains more developers attention than any other programming platform as R provides a wide variety of statistical and graphical techniques, including linear and nonlinear modeling, classical statistical tests, time-series analysis, classification, clustering, and many natural language processing packages. R is easily extensible through functions and extensions, and the R community is noted for its active contributions in terms of packages\(^{17}\). In this article we are presenting an R programming techniques for testing different sentiment lexicons as well as different scoring schemes. Our research goal is based on a bottom up approach that starts by identifying which sentiment lexicon is more suitable than others then proceeds to select more accurate sentiment scoring algorithm (e.g. TF-IDF, LDA (Latent Dirichlet Allocation)). Experimentation on analyzing Twitter sentiments for six NHL Hockey teams has been conducted to determine the suitability of the lexicons and scoring schemes classification. However, the experiments require the assistance of an expert trainer. For this reason the top level goal of our research is to identify suitable machine learning for training these classifiers. The authors are conducting experiments for achieving the top research goal concurrently with the work of this article [24].

II. THE R TWITTER CLIENT

Writing Twitter client in R is an easy programming task. All what is required is to import the TwitteR package

\[
\begin{align*}
\text{require(twitteR)}; \\
\text{require(plyr)}; \\
\text{require(stringr)}; \\
\text{tweets = searchTwitter(Subj, n=NumOfTweets, lang='en')} ; \\
\text{#transform tweets into pure text} \\
\text{tweets = laply(tweets, function(t) t$getText());} \\
\text{write(tweets, file=outFile)};
\end{align*}
\]

The searchTwitter function can be used to search for tweets that match a desired term(s) including hashtags along with the basic AND and OR logical connectors.

II. EXPERIMENT ON THE ROLE OF LEXICON

Sentiment analysis has been handled as a Natural Language Processing (NLP) task at many levels of granularity. Starting from being a document level classification task [4, 5], it has been handled at the sentence level [6, 7] and higher levels (e.g. phrase level [8]). However, the Twitter messages do not follow a strict NLP model due to the inherent limitations of the Twitter messaging system. In addition, Twitter posts tend to be more informal in language, resulting in a completely different vocabulary that along with the challenges in classifying very short messages, makes the task of Twitter sentiment classification a difficult one. The simplest solution to the sentiment analysis problem seems to come from using the “bag of words” model\(^{18}\) rather than words, sentences and paragraphs arranged in a specific order. It is a simplifying assumption used in information retrieval and text mining [9] where single terms are used as features for representing the documents and they are treated independently. Some researchers recently put their focus on trying to find suitable lexicon for classifying tweets sentiments by annotating tweets for negative or positive polarity through using identifying words as positive and negative sentiment wise. Actually most of such researchers use their experience on mining larger text

---

\(^{10}\) http://www.neoformix.com/2008/TwitterVenn.html

\(^{11}\) http://www.sentiment140.com/

\(^{12}\) http://www.nltk.org/

\(^{13}\) http://rapid-i.com/content/view

\(^{14}\) http://gate.ac.uk/sentiment

\(^{15}\) http://www.cs.waikato.ac.nz/ml/weka/

\(^{16}\) http://cran.r-project.org/web/packages/tm/index.html

\(^{17}\) http://en.wikipedia.org/wiki/R_(programming_language)

like the movie review comments [5] in developing these lexicons. Actually lexicons become a big part of the research in opinion mining for many of the online messages (SMS, emails, tweets) [10]. A good attempt in this direction has been made by Hu and Liu where they developed a general purpose opinion lexicon [13] that classifies English words into positives (1967 positive words) or negatives (4783 negative words) sentiment wise. Their complete list of words can be downloaded online19. We can use the opinion lexicon (referred as Type 1) to classify the positivity or negativity of tweets using a simple scoring scheme like the one used in [15]. This scheme counts number of positive and negative words in a given tweet and the final tweet sentiment score is to subtract positives from negatives. We started to experiment with this lexicon on a topic that is rich with sentiment like supporting NHL Hockey teams. We collected a sample of 1500 tweets for six NHL teams in April 17 2012 during the main NHL games competition season. Figure 2 illustrates the sentiment scoring for these NHL hockey teams using the first 100 tweets using the opinion lexicon. Figure 3 illustrates the same experiment described in Figure 3 but with a different lexicon (the Dadvar et al lexicon [18]) which contains 136 positive20 and 109 negative21 words. The Dadvar lexicon is referred to us as Type 2. Based on figures 1 and 2 one can easily finds the effect of using lexicons on sentiment analysis.

![Fig. 1: Sentiment Scoring Using Opinion Lexicon.](image)

19 http://www.cs.uic.edu/~liub/FBS/sentiment-analysis.html
20 http://wwwhome.ewi.utwente.nl/~dadvarm/dir2011/positive.txt
21 http://wwwhome.ewi.utwente.nl/~dadvarm/dir2011/negative.txt
Table 1 describes a comparison between the average sentiment scoring for each NHL Hockey team using the two lexicons (Type 1 and Type 2). It clearly shows that the choice of a lexicon affects the outcome of the sentiment analysis. With a larger lexicon and more relevant words to opinion mining, we are getting a spectrum of positivity and negativity, while using a smaller lexicon with less relevant words, we are getting more positive spectrum. Certainly, the choice of lexicons affects the average sentiment ranking of the teams.

<table>
<thead>
<tr>
<th>Team Name</th>
<th>Averages (Type 1)</th>
<th>Averages (Type 2)</th>
<th>Ranking (Type 1)</th>
<th>Ranking (Type 2)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Montreal Canadiens</td>
<td>0.01</td>
<td>0.01114206</td>
<td>3</td>
<td>6</td>
</tr>
<tr>
<td>Vancouver Canucks</td>
<td>-0.24</td>
<td>0.03220738</td>
<td>4</td>
<td>2</td>
</tr>
<tr>
<td>New York Rangers</td>
<td>-0.34</td>
<td>0.02957151</td>
<td>6</td>
<td>3</td>
</tr>
<tr>
<td>Ottawa Senators</td>
<td>0.16</td>
<td>0.02932551</td>
<td>2</td>
<td>4</td>
</tr>
<tr>
<td>St. Louis Blues</td>
<td>0.23</td>
<td>0.03678679</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>Toronto Maple Leafs</td>
<td>-0.31</td>
<td>0.01994852</td>
<td>5</td>
<td>5</td>
</tr>
</tbody>
</table>

However, there are many other attempts to develop more comprehensive lexicons for opinion mining which we can experiment with. Among such notable attempts are:
- Subjectivity Lexicon [11]
- Holistic Lexicon [12]
- SentiWordNet\(^{22}\) Lexicon

We can use the following R sentimentAnalyzer program (see figure 3) to compute the scores given the paths to any lexicon positive and negative word files.

\(^{22}\) http://sentiwordnet.isti.cnr.it/
III. EXPERIMENTS ON SENTIMENT SCORING

Opinion detection is a very complicated task involving leveraging opinion evidences (e.g., suck, cool), opinion collocation (e.g., I believe, to me), and opinion morphology (e.g., sooooo) based on a relevant opinion lexicons [14]. However, there is no general-purpose sentiment lexicon that is optimal for all domains, because it is well known that sentiments of words are sensitive to the topic domain [19]. For example, “unpredictable” is negative in the electronics domain while being positive in the movie domain. Indeed, sentiment lexicons adapted to the particular domain or topic have been shown to improve task performance in a number of applications, including opinion retrieval [20, 21], and expression level sentiment classification [22]. For this purpose, using an opinion lexicon with a simple scoring scheme may not be effective for all domains. There are many attempts to enhance the classification performance of an opinion lexicon by removing the stop words as well as by treating the effect of negation. Stop words represent common English words that have no effects on the sentiment weighting. The negation effect is performed by identifying parts of text affected by a negating statement (ex: “not good” as opposed to “good”). Then, the document is scored based on terms found and whether it is negated. There are attempts to use part of speech (POS) taggers to mark the grammatical components for the given text (e.g. Stanford Log-linear POS Tagger25), however, the tweets do not follow a strict natural language structure and hence the use of such taggers will not be of great help. Since we are following the Bag-of-Words26 model in dealing with the sentiment analysis of tweets, we can solve the effect of negation by using an empirical method that counts the sentiment score for the negation operator (e.g. no, not, rather, hardly) and the word that follows the negation operator as one component for the scoring purpose. Using this empirical consideration, if we have a negation operator followed by a positive word then the score is -1 and in case the negation operator is followed by a negative word then the score is +1. There are many other improvements to make the sentiment analysis more context oriented or to increase the scope of the sentiment classifications from binary (positive, negative) to a multi-sentiment values (e.g. anger, disgust, fear, joy, sadness, surprise27).

However, if one is satisfied with one lexicon for ranking sentiments for a given domain, then the next task is to improve the scoring scheme. The simple scoring scheme used earlier treats all the terms as equal. From the basic research in information retrieval we know that not all the terms that are in a text hold the same importance. There are some terms are more important than others related to the overall opinion outcome of any given sentence or text. From this type of research, the most recommended measure for identifying important terms is the TF-IDF ranking technique [16, 17]. TF-IDF stands for Term Frequency-Inverse Document Frequency. This method evaluates words in each sentence to minimize the influence of those that are very common across the document and do not carry much meaning. The importance of a word is high if it is frequent in a particular sentence, but less frequent in other. TF-IDF is equal to TF*IDF. Although there are many different weighting formulas, TF and IDF are commonly computed as follows:

\[
\text{TF} (i,j) = \text{Term Frequency is the number of the occurrences of the word } i \text{ in sentence } j \text{ divided by the sum of the occurrences of all the words in sentence } j.
\]

\[
\text{IDF} (i) = \text{Document Frequency} = \log \left( \frac{|S|}{S_s} \right), \text{where } |S| \text{ is the total number of sentences in the input text, and } S_s \text{ is the number of sentences in which the word } i \text{ appears.}
\]

Figure 4 illustrates the use of the TF-IDF weighted scheme for scoring the sentiment of the six NHL Hockey teams for the same sample of tweets used at our previous examples.

---

24 http://www.home.ewi.utwente.nl/~dadvarm/dir2011/negation.txt
27 http://www.cse.unt.edu/~rada/affectivetext/
Table 2 provides the average scores for using the TF-IDF weights for the same six Hockey teams tweet files. Obviously TF-IDF has some effects of the overall sentiment ranking of the six Hockey teams.

<table>
<thead>
<tr>
<th>Team Name</th>
<th>Av. Senti. Scores</th>
<th>Ranking</th>
</tr>
</thead>
<tbody>
<tr>
<td>Montreal Canadiens</td>
<td>0.000144309</td>
<td>5</td>
</tr>
<tr>
<td>Vancouver Canucks</td>
<td>0.0004810987</td>
<td>3</td>
</tr>
<tr>
<td>New York Rangers</td>
<td>-0.0000258460</td>
<td>6</td>
</tr>
<tr>
<td>Ottawa Senators</td>
<td>0.0001075418</td>
<td>2</td>
</tr>
<tr>
<td>St. Louis Blues</td>
<td>0.001831214</td>
<td>1</td>
</tr>
<tr>
<td>Toronto Maple Leafs</td>
<td>0.0001488757</td>
<td>4</td>
</tr>
</tbody>
</table>

However, using the standard tf-idf by viewing a sample of tweets as a document, we would perhaps be able to discover important terms that appear in very few tweets, but we might not be able to discover central topic terms that are common to several tweets (since the idf tends to punish terms that appear in more than one tweet). Tweets that make use of such topic terms are expected to be more helpful than tweets that do not for the purpose of sentiment analysis, because in a sense the existence of such terms guarantees that the tweet also covers the main aspects of the subject topic at hand. For this purpose we modified the original sentiment scoring program (Fig. 4) to assign higher scores to topic terms. For this purpose, we used the Latent Dirichlet Allocation (LDA) method to identify topic terms. The LDA algorithm is already included in the TopicModel package\(^{28}\). LDA is a powerful learning algorithm for automatically and jointly clustering words into "topics" and documents into mixtures of topics [23]. Table 3 illustrates the average sentiment scoring under LDA for the same samples of the six Hockey teams.

<table>
<thead>
<tr>
<th>Team Name</th>
<th>Av. Senti. Scores</th>
<th>Ranking</th>
</tr>
</thead>
<tbody>
<tr>
<td>Montreal Canadiens</td>
<td>0.008982036</td>
<td>3</td>
</tr>
<tr>
<td>Vancouver Canucks</td>
<td>0.00785546</td>
<td>4</td>
</tr>
<tr>
<td>New York Rangers</td>
<td>-0.007242004</td>
<td>6</td>
</tr>
<tr>
<td>Ottawa Senators</td>
<td>0.01857283</td>
<td>2</td>
</tr>
<tr>
<td>St. Louis Blues</td>
<td>0.04129129</td>
<td>1</td>
</tr>
<tr>
<td>Toronto Maple Leafs</td>
<td>0.001930502</td>
<td>5</td>
</tr>
</tbody>
</table>

V. Conclusions

In this article we presented a systematic way to program and analyze Twitter sentiment using the R programming paradigm. Figure 5 illustrates our overall framework.

\(^{28}\) http://cran.r-project.org/web/packages/topicmodels/
The presented approach classifies tweets in two stages. The first stage uses a simple scoring method that relies on the number of positive and negative keywords available at the tweet text according to a provided lexicon. In this stage all the keywords are of equal weights. The second stage enhances the simple scoring technique by assigning some keywords higher weights. In this framework the programmer is the trainer. In first stage different opinion classification lexicons have been tried and in the second stage different algorithms for assigning weights to important keywords has been tried (e.g. TF-IDF and LDA). According to the experimentation conducted by the trainer, the opinion lexicon along with the LDA algorithm is found to be more effective for sentiment analysis than the Dadvar lexicon and the TF-IDF algorithm. The experiments conducted for the purpose of analyzing the Twitter user sentiments of six NHL Hockey teams. We intend to expand our research work to introduce scoring schemes that utilize training models instead of the human trainer using more useful R packages like the RTweetTools. Our research in [24] extends this current work by adding machine learning algorithms. We also intend to compare our approach with some recent works [25, 26, 27].

REFERENCES


