Sentiment Analysis:
Android applications

by

Forum Kapadia

A Project Report Submitted
in
Partial Fulfillment of the
Requirements for the Degree of
Master of Science
in
Computer Science

Supervised by

Dr. Leon Reznik

Department of Computer Science

B. Thomas Golisano College of Computing and Information Sciences
Rochester Institute of Technology
Rochester, New York

May 2017
Abstract

Sentiment Analysis:
Android applications

Forum Kapadia

Supervising Professor: Dr. Leon Reznik

As there are millions of applications available today that serves various purposes, therefore its components like sentiment analysis of the user comments have become an important study of research. Sentiment Analysis is also known as opinion mining that helps in evaluating whether the content is negative, neutral or positive. It helps in understanding the attitude or how the user feels with respect to a topic. Most of the users provide comments and ratings to an application after getting acquainted with the application. We are mainly concerned with the opinions of other users while downloading an application. These feedback’s play an important role in shaping our views. Reviewing each and every comment becomes difficult and thus the natural language processing with respect to sentiment analysis comes into the picture. Users are not only concerned with the feedback but also various other factors such as security features, the number of downloading, ratings, compatibility with the phone etc. In this research paper, we discuss the importance of sentiment analysis, various libraries used, and compare the results.

The library used to perform the sentiment analysis is TextBlob python library and natural language understanding(NLU) API by Watson. They help us in extracting the metadata such as entities, contents, sentiments, emotions, part-of-speech tagging, classification,
NP(noun phrase) extraction and so on. These sentiment scores are fed as an input to the expert system which contains certain rules as a part of machine learning for evaluating the comments and provides the output as the short summary of the application. This summary helps the user to focus on important aspects quickly and make a smart decision. The experiment is performed on five social media application with more than 5000 feedback of each application which is cleaned and the results are evaluated. Our results suggest that text blob library outperforms the NLU API.
Contents

Abstract  ........................................................................................................................................... ii

1 Introduction  ................................................................................................................................... 1

2 Design and Implementation  ........................................................................................................... 2
  2.1 Related Work  ......................................................................................................................... 2
  2.2 Approaches:  .......................................................................................................................... 3
    2.2.1 TextBlob  ......................................................................................................................... 4
    2.2.2 Watson: Natural Language Understanding API  .......................................................... 7

3 Analysis  ......................................................................................................................................... 12

4 Conclusions  .................................................................................................................................. 15
  4.1 Future Work  .......................................................................................................................... 16
  4.2 Problems faced  ...................................................................................................................... 16

Bibliography  ..................................................................................................................................... 17
List of Figures

2.1 Working: TextBlob Library .................................................. 4
2.2 Sentiment score calculation ................................................. 6
2.3 Sentiment score results ...................................................... 6
2.4 Overall feedback ............................................................. 7
2.5 Working: NLU API ............................................................. 8
2.6 Calling the NLU API [4] ....................................................... 9
2.7 Sentiment score using NLU API .......................................... 9
2.8 Sentiment score using NLU API .......................................... 10
2.9 Entity score using NLU API ............................................... 10
2.10 Semantic score using NLU API .......................................... 11

3.1 Snapchat application: Summary TextBlob ............................ 12
3.2 Snapchat application: Summary NLU API ............................ 13
3.3 Snapchat application: Summary NLU API ............................ 13
3.4 Snapchat application: Summary NLU API ............................ 14
Chapter 1

Introduction

A lot of research is going on in the area of Sentiment analysis. It is a part of Natural language processing and uses Machine learning features like PartOfSpeech tagging, n-grams, phrase extraction, emotions, keywords etc. The main task of this project is to perform sentiment analysis on user feedback using TextBlob library [8] and Watson Natural Language Understanding(NLU) API[3]. The user feedback is collected for various applications which are in the excel format and further data cleaning is performed on the same. Data cleaning is accomplished by removing irrelevant attributes, and missing values. This cleaned data acts as an input to the Sentiment analysis model created using TextBlob library and we obtain the sentiment scores for a particular application in the form of subjectivity and polarity. These scores are fed as an input to the Java code that has machine learning algorithms which help in providing the overall feedback to the users.

Moreover, achieving sentiment analysis using NLU API had a huge learning curve. The preprocessed data is fed as an input to NLU and the output is in the form of overall sentiment, emotions, keywords, categories, semantic roles. This API is activated once the python code invokes it. Finally, once we obtain the sentiment results of both the approaches, we provide an overall rating for an application that helps the user to make smart and quick decisions. Moreover, it even helps the developer to understand the point of improvements if their application have been receiving negative feedback’s.
Chapter 2

Design and Implementation

2.1 Related Work

There is a significant amount of research done in sentiment analysis. A research paper [7] displays the fact that by making use of emoticons as a positive label leads to reduce machine learning algorithms dependencies. Alec Go [1] used a couple of classifier methods for training such as Naive Bayes, and SVM. This classifier acts as our base model and we can achieve good accuracy on a smaller training dataset.

The research paper [2] advances by designing a model with greater in-depth preprocessing stage and used a couple of features such as negative/positive POS tags and scores. Even if the features were not of any n-grams or traditional bag-of-words, still this model displays high accuracy’s.

Barbosa [5] has proposed a different method in sentiment analysis on twitter data. They took noisy labels to train a model by using a polarity prediction from three different sites and also use roughly thousand data to tuned it manually and another thousand data used for testing purpose.

Tsur, Davidov, and Rappoport 2010 have used an hash-tags feature for generating a training data but their limitations are that they have restricted their experiments to sentiment and non-sentiment classification. But the limitation of this approach is that it works well only for particular topic specific data.

A research paper [6] uses feature extraction method by data mining. This method basically removes the stop words and then performs feature extraction using NLTK library.
The tool used by the paper for analyzing the scores of words is termed as SentiStrength. The sentiment results are in the range of -1 to 5. Where positive number denote the positive sentiments and negative number denote negative sentiments.

### 2.2 Approaches:

The initial step of Sentiment Analysis is to select and extract the text features. Some of these features include identifying the presence of words and its frequency. It is also termed as word n-grams along with its frequency. This helps in understanding the important words by assigning binary weights. Part of speech tagging indicates extracting the adjectives as they play an important role in demonstrating opinions. Then the words and phrases used to describe opinions such as excellent and worst. Moreover, Negation words change the orientation of the opinions. Example, not nice signifies something that is bad.

The preprocessing techniques include data collection, data cleaning, and document conversion. The corpus was created by collecting data from disparate sources such as help-shift.com, appfollow.io, and searchman.com. The format of the data available from all these websites is different and collaboration into a single CSV file required a python code that combines the relevant features and eliminates the unnecessary attributes. For the scope of this project, I had conducted my experiments on five application i.e. Snapchat, Instagram, WhatsApp, IMO, Skype.

Once the file collaboration is successfully accomplished, I manually cleaned for any missing values, unexpected values, removal of irrelevant attributes etc. This phase was time-consuming and required lots of research. The corpus was created for each individual application which had approximately 5000 user feedback’s. Moreover, it also includes rating provided by each user, the number of downloads, and security features.

The problems that I encountered during this phase was majorly related to collecting the data. The websites mentioned above-provided feedback but they had certain restrictions on a number of feedback’s downloaded in a single request, so multiple requests had to be
made. The free version had restrictions where I could just download the data for two applications. Some of the data were in different format and collaboration of the data required every data to being in a single format.

2.2.1 TextBlob

The first step is to parse the input file i.e. user feedbacks file which was converted into CSV format and pass it as an input to the sentiment analysis model. To create this model, a python library named TextBlob is used. This library is an API used to perform the natural language activities such as extracting the noun phrase, translation, POS tagging, sentiment analysis, tokenization, n-grams etc. This library contains two implementations of the sentiment analysis i.e. NaiveBayes analyzer which comes under NLTK classifier and pattern analyzer which is dependent on pattern libraries.

A python code parses the CSV file and calls the sentiment model and once this model is executed, it provides the result in the form of polarity and subjectivity. The results are stored in a CSV file. This polarity and subjectivity are values that help in determining the sentiments of the comment. Polarity falls under the range of [-1.0 to 1.0] and subjectivity under the range of [0.0 to 1.1]. The polarity score denotes the amount of positive/negative
information which is present in the statement or entire comments. This scores helps us to understand the opinions of various users. If the score is declined towards the positive range then the comment is more likely to be positive, if the score is equivalent to 0 then the comment is neutral and if the score is in the negative range then the comment tends to be negative. The subjective score helps in understanding the objectivity of the sentence. i.e. very good, this sentence has a low subjective ranking as compared to, this application is good and has great video calling feature. If this score is declined towards 0.0 then we can say that the comment is objective in nature and if the score is near to 1.0 then the comment is subjective.

Once these polarity and subjectivity scores are saved in CSV file, they are fed as an input to the machine learning algorithms written in Java. These algorithms have a certain set of rules that help them to make a decision and determine the overall status of the application. The rules that are defined help in verifying the overall polarity and the subjectivity score for the application and give the application an overall ranking. The different set of rules defined are as follows: if the maximum of the comments has the polarity score >0.5 then the comments are excellent. If the polarity score falls under the range of 0.0 to 0.5 exclusive then, the overall comments are good. If the score falls under the range of -0.5 to 0.0 exclusive, then the comments are bad and finally, if the score falls under the range of -1.0 to -0.5 then the comments are extremely bad. The subjectivity threshold that I have selected is 0.4, which denotes if the maximum comments have a subjective score >= 0.4 then the subjective rating of the comments is good otherwise it is bad.

There are other hybrid rules which the machine learns and makes the decision like if the polarity score is good and even the subjectivity score is bad then various other parameters are considered and the decision is made. Moreover, at this stage, the stars provided by the user also help in influencing the overall decision of the application. Once all these factors are evaluated and security features are extracted an overall short summary is provided for an application that helps the user in making decision.

Sample snap for parsing the data and collecting subjectivity and polarity scores:
Figure 2.2: Sentiment score calculation

```python
import csv
import xlrd
from xlrd import open_workbook
from textblob import TextBlob
wb = xlrd.open_workbook('snapchat.xlsx')
sheet = wb.sheet_by_index(0)
f = open('out.txt', 'w')
for i in range(sheet.nrows):
    blob = TextBlob(sheet.cell_value(i, 0))
    print >> f, blob.sentiment
f.close()
```

Figure 2.3: Sentiment score results

These scores are fed into the java code which enables the machine learning algorithms to evaluate the overall description of the application. The results are as follows:

**Tools and Technologies:**

The tools and technologies used for implementing this model: Python 3.5, PyCharm, Java 7 or above, Java Eclipse or other IDE, and Python library TextBlob.
2.2.2 Watson: Natural Language Understanding API

Natural language Understanding (NLU) API analyzes the unstructured text and extracts the metadata like entities, keywords, sentiments, relations, categories, and emotions. This API is called using a python code as seen in figure 2.6:

The python code consists of importing the watson_cloud environment and connecting to the IBM Watson credentials. The cleaned data is passed as an input to the NLU API. The output that we receive consists of categories, emotion, entities, keywords, metadata, relations, semantic roles etc. The sentiment output(fig. 2.7), emotion score(fig. 2.8), entity score(fig. 2.9), and semantic score(fig. 2.9) in JSON format is as follows:

These scores help in understanding the overall sentiments, emotions, frequently used
keywords, the entities (e.g. application name, phone model) etc. This overall summary helps the user to understand the sentiments of other users feedback’s and can make an opinion based on this scores.

**Tools and Technologies:**

The tools and technologies used for implementing this model: Python 3.5, PyCharm, Java 7 or above, Java Eclipse or other IDE, IBM Bluemix, IBM Watson natural language understanding API.
Figure 2.6: Calling the NLU API [4]

```python
# the input file to be used for sentiment analysis
inputFile = 'inputfile.csv'
with open(inputFile, 'r') as f:
    reader = csv.reader(f)
    cnt = 0
    for row in reader:
        # skipping the first row which is the heading
        if cnt == 0:
            cnt += 1
        else:
            if len(row) == 0:
                pass
            # if empty review
            else:
                inputText = row[0]
                cnt += 1
                totalCount += 1
                # sentiment analysis of every review
                response = natural_language_understanding.analyze(
                    text=inputText,
                    features=[features.Sentiment(), features.Emotion()],
                    language='en')
                print(totalCount)

                # counting the number of positive and negative review with the score
                if response['sentiment']['document']['label'] == 'positive':
                    positiveCount += 1
                    score += response['sentiment']['document']['score']
                elif response['sentiment']['document']['label'] == 'negative':
                    negativeCount += 1
                    score += response['sentiment']['document']['score']

        return positiveCount, negativeCount, score, totalCount

if __name__ == '__main__':
    posCount, negCount, score, totCount = sentimentMining()
    print('Positive Count', posCount)
    print('Negative Count', negCount)
    print('Final Score', score)
```

Figure 2.7: Sentiment score using NLU API
Figure 2.8: Sentiment score using NLU API

```json
{
   "emotion": {
      "document": {
         "emotion": {
            "sadness": 0.522566,
            "joy": 0.670037,
            "fear": 0.07263,
            "disgust": 0.050079,
            "anger": 0.10602
         }
      }
   }
}
```

Figure 2.9: Entity score using NLU API

<table>
<thead>
<tr>
<th>Entity</th>
<th>Type</th>
<th>Score</th>
</tr>
</thead>
<tbody>
<tr>
<td>instagram</td>
<td>Company</td>
<td>0.75</td>
</tr>
<tr>
<td>facebook</td>
<td>Company</td>
<td>0.51</td>
</tr>
<tr>
<td>samsung</td>
<td>Company</td>
<td>0.42</td>
</tr>
<tr>
<td>Mehzabin quresh</td>
<td>Person</td>
<td>0.33</td>
</tr>
<tr>
<td>MySpace</td>
<td>Company</td>
<td>0.31</td>
</tr>
<tr>
<td>Insta</td>
<td>Company</td>
<td>0.31</td>
</tr>
</tbody>
</table>
Figure 2.10: Semantic score using NLU API

I think the given length of the video that we can upload should more than one minute.

If that is done it would be more convinient.

This as an amazing app! I have been using this app for years and have over 3k followers!
Chapter 3

Analysis

The sentiment analysis test was conducted on five different applications: Snapchat, Instagram, WhatsApp, IMO, Skype.

**Application: Snapchat**

Snapchat has the database of more than 30000 user feedbacks. I selected this application as it had recently gained negative popularity because of the CEO’s comment and its rating fell down drastically.

When the python code is executed, it parses the database and calls the textBlob library. The textBlob library sends the sentiments score as a CSV file to the machine learning algorithms written in Java. This Expert system has certain set of rules that help in determining the overall efficiency of the application. The summary for this application is as follows:

```
The overall sentiments of the user feedback is as follows: maximum comments fall under the category of good rating, and the subjectivity of maximum comments is good. This indicates that the application is worth giving a try.

Excellent comments: 1412
Good comments: 2892
Bad comments: 151
Worst comments: 49

Number of downloads: 500 million+
Average ratings: 3.5*
```

Figure 3.1: Snapchat application: Summary TextBlob

The above summary helps in evaluating the feedbacks and as we can see a maximum
number of feedbacks fall under the positive category so we can conclude that the application is good.

The same database is fed to the NLU API as well and the detailed results obtained from this API is as follows:

Figure 3.2: Snapchat application: Summary NLU API

The emotions help us understand how angry, happy or sad the users are:

Figure 3.3: Snapchat application: Summary NLU API

The keyword explain the most frequently used words by all the users who have given their comments on the application. This tab is helpful for the developers.
<table>
<thead>
<tr>
<th>Term</th>
<th>Score</th>
</tr>
</thead>
<tbody>
<tr>
<td>good app</td>
<td>0.72</td>
</tr>
<tr>
<td>instant app</td>
<td>0.65</td>
</tr>
<tr>
<td>funniest app</td>
<td>0.62</td>
</tr>
<tr>
<td>great app</td>
<td>0.61</td>
</tr>
<tr>
<td>Best app</td>
<td>0.61</td>
</tr>
<tr>
<td>Awesome app</td>
<td>0.60</td>
</tr>
<tr>
<td>snap chat</td>
<td>0.60</td>
</tr>
<tr>
<td>Tablet app</td>
<td>0.59</td>
</tr>
<tr>
<td>Tp app</td>
<td>0.59</td>
</tr>
<tr>
<td>pointless app</td>
<td>0.57</td>
</tr>
</tbody>
</table>

Figure 3.4: Snapchat application: Summary NLU API
Chapter 4

Conclusions

Sentiment analysis/opinion mining is an active area of research that studies the sentiments, emotions, or attitude of people towards a particular entity. This analysis helps the users and developers from a different perspective. The users are benefitted as they get a short summary of the application and there is no need to review all the feedbacks in details to make a decision. Whereas the developers are benefitted as they understand various aspects if their application isn’t doing well and they could improve on that particular area. This paper helps in understanding the importance of sentiment analysis of the user feedbacks and highlights two API's to implement the model. The textBlob library and Watson NLU API does a great job of providing the sentiment results and the implemented model helps in agitating the results.

The input data consists of data collaborated from three websites for five different applications. This acts as an input to the sentiments models created. The sentiment analysis results generated from python textBlob library provides a short summary that helps the user to shape their opinion about an application. Moreover, the results obtain from NLU API provides other details such as emotions, entities, categories, keywords, sentiments, semantic roles etc. These results not only help the users but it also helps the developers in various aspects. For example, if their application is constantly receiving negative feedbacks, the developer could understand the reasons where people are facing problems while using their application by reviewing the summary.
4.1 Future Work

1. Extracting the user feedback in real time. This is a big challenge as it requires the data cleaning to be automated and if multiple sources are used to extract the data then it creates more problems due to the different data storage techniques.
2. Creating the web application where the user has to just input the application name, and they obtain the summary of the application. Moreover, it could be a full-fledged application which not only performs sentiment analysis but also provides all the details or answer any questions related to the application.
3. Implementing the sentiment model using many other libraries available such as Tokenizer, CoreNLP, SentiStrength, WordSegmentor and compare the efficiency of each method.

4.2 Problems faced

1. Data collection from various websites was a big problem, as every website had different format of data storage. So collecting important attributes from each website and merging them into a single database file took great efforts. Moreover, the NLU API had limited number of API calls, so it cannot take the entire database for a particular application.
2. Learning about the API’s and calling them programmatically, creating the sentiment model and training the model was time-consuming. Moreover, the model had difficulties in identifying the idioms/sarcasm etc.
Bibliography


[2] Vasudeva Varma Ayushi Dalmia, Manish Gupta. Iiit-h at semeval 2015: Twitter sentiment analysis the good, the bad and the neutral!


