

Customizable Twitter Bots for Improved QoS in Product Marketing

Balabhadrapathruni Ramya, Anush Baskaran, Taruni Anand, Senthil Kumar R, N.Ch.S.N Iyengar*

School of Computer Science Engineering

Vellore Institute of Technology,

Vellore, Tamil Nadu.

bramyasharma@gmail.com, {anush.baskaran2013,tarunianand.m2013,rsenthilkumar,nchsnriy}@vit.ac.in

Abstract—In recent times, users have started spending a significant amount of their time on social networking platforms (Twitter, Facebook, Orkut), sharing a plethora of personal information and opinions on different aspects of life. Internet users around the world make use of this powerful tool of communication a.k.a microblogging, using Twitter, a web application that provides dual benefits of microblogging and social networking. In this paper, we propose an application that uses the open structure of twitter to perform 2 main tasks: (1) analyze data for sentiment analysis and opinion mining of a particular product and classify this data into positive, neutral and negative reviews (2) use this data to create automated programs, known as bots, which are useful for generating benign tweets. We use this double-edged sword to provide companies with the opportunity to make use of these automated bots to generate the right tags to advertise products. This enhanced marketing opportunity provides Quality of Service in terms of cost, time and energy.

Keywords—Twitter; microblogging; sentiment analysis; opinion mining; automation; bots; QoS

I. INTRODUCTION

Microblogging is a combination of blogging and instant messaging that allows users to create short messages to be posted and shared with an audience online. Microblogging tends to offer a portable communication mode that feels organic and spontaneous to many and hence captures the public imagination. It attracts users due to its instant publication with few restrictions on content.

Social platforms like Twitter have become extremely popular means of this new type of blogging. As of March, 2016, Twitter has 320 million users worldwide. Twitter has become a favourite in data mining community due to its excellent API and support for academic research which makes scraping twitter data easy. Twitter has a limit of 140 characters, hence letting the users express their emotions in a concise manner through text, emoticons, hashtags, images etc; As twitter data has a variety of emotions expressed and is available for collection, it is the ideal dataset to perform sentiment analysis on.

The idea of sentiment analysis or opinion mining is to derive the opinion or attitude expressed by the writer. It is the process of computationally determining the emotion of the tweet. Using sentimental analysis, we have identified and categorized the nature of the target tweets. This has been

achieved through python. Python is an object-oriented, high level programming language with easily implementable syntax which focuses on readability. It is easier to create deep neural networks, that is, deep learning through python. Python also has primary built-in data analysis tools, packages and libraries which help keep the focus on the method of implementation rather than the implementation itself. In our paper we have automated the process of marketing by creating a bot which tweets a reply to the target tweet based on the sentimental analysis performed on it.

Traditionally bots have been used to reply to every tweet which causes spamming. These spam bots tend to like ancient posts, redirecting the user to spam websites which indirectly buries their PC in malware oblivion. An automated bot can be an extremely useful tool for marketing if used in the right way. We use keyword search to first find the target audience which are again refined based on their liking and positivity towards a particular product that we are marketing for. Our bot eliminates the idea of spamming as we customize and auto tweet only to a specific target audience based on their interest in that particular item.

This paper aims to create an enhanced system for marketing which improves the QoS(Quality Of Service) and is extremely client friendly as all the client needs to do is enter the keywords to find the target tweets and rest will be taken care of by the system.

II. RELATED WORK

Recently, there have been several methodologies to analyze sentiments of reviews and large textual matter, in order to aid in marketing and understand customer views and feedback. Using Twitter as a microblogging website, has also found its disadvantages in terms of large number of illegitimate botnets for marketing purposes. When sentiment analysis is used to find profiles of people who are currently interested in a specific product (based on search keys), the audience for marketing becomes targeted. To make this more time efficient, we can use bots to selectively tweet to interested users about the given product. This overcomes the disadvantage of bots spamming unnecessarily to irrelevant users.

Agarwal [1] et al proposed that posts on microblogging sites like Twitter can be classified based on polarity using three models. The models were created for binary classification of

sentiment into positive and negative; as well as a three way classification of tweets into positive, negative and neutral polarity. This was performed using tree kernel based, feature based and unigram models, as well combinations of the same. It was concluded that the combination of tree kernel based model and feature based model outperformed the other models.

Pak [2] et al proposed to build a classifier for microblogging websites by collecting a corpus of positive, negative and objective sentiments, followed by linguistic analysis on the corpora. The analysis was based on key features, based on a “salience” value in addition to entropy. It was concluded that salience provided more accurate discrimination of n-grams than entropy, increasing its efficiency as an analyzer.

Wilson [3] et al proposed a method to determine contextual polarity of a statement based on prior polarity key words. This was done by first classifying the text as neutral or polar based on clues that determine a hint of polarity. This is followed by disambiguating the text as positive, negative, neutral or both based on an inter annotator study. It was concluded that this method could determine polarity of large volumes of data, along with significantly better results than baseline methods.

Go [4] et al proposed a classifier based on distant supervision for microblogging on Twitter, owing to their popularity. They used Twitter data as their training dataset, and trained a binary classifier, to determine either positive or negative polarity of a tweet. This was done by acquainting the classifier with emoticons, and limiting the trainer data to tweets that contained emoticons. It was concluded that machine learning algorithms provide high accuracy in classifying tweets based on polarity.

Chu [5] et al proposed a classification system that was based on an entropy based component, a machine learning based component, an account properties based component and a decision maker based component that uses a combination of features from a user to determine its possibility of being a human, bot or cyborg. The efficiency was determined by the test dataset used. It was concluded that the human to bot to cyborg ratio was nearly 5:4:1, making exploitation of twitter accounts relatively simple.

Kouloumpis [6] et al proposed a sentiment analyzer for Twitter specific to microblogging, by means of comparing it to an existing lexicon. The lexicon consisted of a Hashtag dataset, an Emoticon dataset, iSieve dataset, along with POS features. It was concluded that POS feature analysis was less effective in Twitter sentiment analysis, unlike those of reviews. Hashtags and emoticons analysis proved to be more beneficial in evaluating the sentiment of the given Tweet.

Haustein [7] et al examined the existence and effects of automated bots prevailing in the Twitter environment. It was stated that most of these bots follow tweeters with an intention of the users following the bot back. In addition, there was also rampant usage of cyborgs on Twitter, primarily for spam and marketing purposes. It was specifically noted to identify bots that disseminate links to scientific papers, and concluded that identification of social bots are far more difficult than arXiv.

Wald [8] et al proposed a method to identify bots and cyborgs on Twitter. This was done by engaging ten different feature algorithms to calculate a Klout score, and further using six classification learning algorithms. It was concluded that bots with higher number of followers and lengthier posts were responded to. Also, RF100, the ensemble learner outperformed other learning algorithms in accurately identifying bots.

Boshmaf [9] et al proposed a method to test the activity of bots on Online Social Networks (OSN) by setting up a web based botnet called Social Bot Network on Facebook. They had managed to infiltrate several user accounts, with an 80% success rate. They further found that this could breach a user’s privacy beyond the public domain. It was concluded that the in practice, several OSN security practices were ineffective in detecting or stopping social bots.

III. Motivation

While social media is taking up the front scene in communication, Twitter has emerged as one of the most popular websites for people to share messages in a 140-character length message. This data conveys interests and emotions, which can be used to predict several outcomes. Several companies, start-ups in particular, can use this data to determine the right tags to be included to use Twitter for easy advertising. There has been no solution for this purpose, and hence becomes an important need of the hour. The motivation of Twitter bots will remain unchanged, as they are already present, and easily executable. However, they lack in customizability. A customizable application for tweeting will enhance quality of tweets, improve product reach, and most importantly save lots of time. This will in turn allow cost cutting and easy profile analysis, which can in turn provide extensibility.

IV. Research Problem

For any product to be successful, the basic requirement is to have customers or buyers. To build customers, advertising is one of the easiest methods of communication. Recently, advertising has gone beyond the realms of banners and television. Microblogging serves as a mean for companies to explain their product while connecting to social media. This exploitation of social media has given rise to spamming, which in turn leads to negative results [10]. People tend to turn away from products that are constantly pushed to their notice. According to a survey conducted by the students of University of Illinois, at least a third of internet users do not respond positively to internet advertisements.

This brings us to a scenario where a huge network of active consumers is available, but there is no effective advertising strategy for the same. This brought us to think of reasons why a layman would not want to see advertisements. To do this, our sample data was available as a large stream of tweets. Most of these were opinions and comments that would provide valuable feedback and interest in a product. When these interests are segregated, into people who like a product and

those who don't, it paves way to an opportunity to advertise usefully.

We assume that users who show a liking and interest in a target product would benefit from advertisements related to it. So, classifying sentiments through a sentiment analysis algorithm must help in identifying interested customers.

While this is can be done manually, automating the advertising sector may be a lot more efficient in terms of cost and energy. Bots have been utilized for spamming, but there can be bots made to advertise sensibly. This can be done by allowing the bots to selectively post tweets according to the sentiment of a given user. A new product will then be introduced to only those who are genuinely interested, based on their online profiles. This can help the company to not just get a customer base, but also develop a sense loyalty in them. Technologically, this would make bots more useful and targeted in their approach.

V. Proposed Model

In this section, we present the structure of the application, starting from a high level perspective in which the system works, and then continue by explaining each level of the system architecture with the obtained results.

Since this application is designed to be used for marketing purposes, we assume that CompanyXYZ is promoting their product, and they decided to use twitter for their marketing. In order to obtain desirable results of a successful marketing campaign, they carry out their process in a systematic manner; the phases in which their campaign is carried out consists of using data mining techniques such as data collection and data processing, and finally ends with using an automation technique: creating an automated bot. The usefulness of carrying out their campaign in phases lies in the fact that their goal will be successful only if they reach out to the right set of customers. Once they find the potential set of customers, they can make use of automation to reach out to them, in order to save man power and other costs.

Each of these phases will be explained in further details in the coming sections. The high level view explaining the flow of this model is presented in Fig. 1.

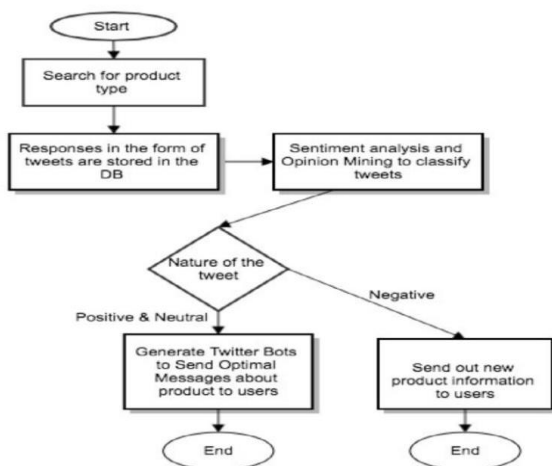


Fig. 1. Proposed Flow of the Model

A. System Design

The application works in layers with the layer below performing a function for the layer above. Some of these layers use the database for performing data mining tasks. In order to implement a working model of the application, we have used MongoDB to store the collected data and extract certain pieces of information for processing. Fig. 2 below gives the diagrammatic view of the systems architecture.

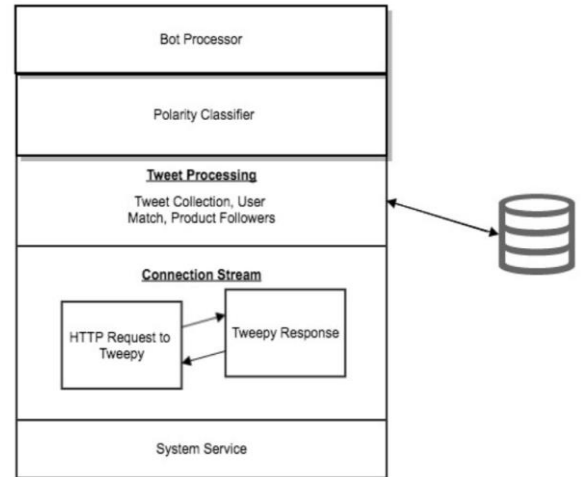



Fig. 2. System Architecture

- **System Service** - The system service component invokes all other components and provides a command line console for twitter to process its search for tweets, based on the search command.
- **Connection Stream** - Once the HTTP server pings the Tweepy API with the search option, Tweepy induces the generation of the Tweet processing stream. This HTTP connection is open till a single request is served and is open every time a request is made.
- **Tweet Processing** - The Tweet Processing component is generate during the middle of the Connection Stream process. The collection of new published tweets containing the search term is gathered. It also gathers tweet status id, tweet author id, and tweet author name. Information collected is stored in the database. This component also gathers the product tweeted about's followers.
- **Polarity Classifier** - The polarity classifier is used to analyze tweets to determine whether a tag is positive, negative or neutral. The results are then written to the database to be used in the next stage. While neutral and negative tweets are ignored, the positive tweets are taken to be used for marketing purposes.
- **Bot Processor** - The bot processor is responsible for analysis positive reviews to gather the user_id of the

person who tweeted, and tag him in an automated tweet that a company wishes to publish. Also, in order to make sure that the messages do not spam user accounts by continuously tagging them, a counter is set to tag them in optimal number of posts, in order to serve a positive purpose.

B. Data Collection

For a company trying to promote its product via twitter, it can optimize its marketing campaigning only if it tries to reach TABLE 1. Using synonymous terms to optimize collection of data

Word	Image
Shoe	 A screenshot of a tweet from a user with a profile picture of a shoe. The tweet text says: "So I got these shoes from @... and they're great, with awesome customer service 🙌🙌🙌... Contact her for more 🙌". It is 7 hours old.
Footwear	 A screenshot of a tweet from a user with a profile picture of a shoe. The tweet text says: "Fresh hot from oven. #Seunclan footwear with elegant footwears that suits all dress sense. @...". It is 18 hours old.

While twitter provides several API's for its developers to provide access to its twitter servers, we used the tweepy API written in Python, due to the support and the features it provides. We also preferred to use the streaming API as compared to the REST API to access live stream data by establishing a continuous HTTP connection. Twitter's API ensures the security of user accounts by providing an OAuth authentication model which supplies a unique set of tokens and secrets to access a user's account through an application. This ensures that only the user that possesses these tokens can post or access the twitter account.

For this project, we ran the streaming API and collected around 2000 tweets from users. In order to store these tweets, we used MongoDB, which in turn neatly stores this data in the form of a JSON object. This allows for easy parsing of the components of twitter data such as text, hashtags, user_id, screen_name etc. The pseudo code for collection of data is shown below in Fig 3.

Algorithm 1 Data Collection
<pre> procedure COLLECTDATA MONGO_HOST ← path to which database you want to collect data search_words ← keywords for tweet search open streaming connection to twitter. try: consumer_key ← consumer key to twitter consumer_secret ← consumer secret to twitter access_token ← access token to twitter access_secret ← access secret to twitter auth ← tweepy.OAuthHandler(consumer_key, consumer_secret) auth.set_access_token(access_token, access_secret) client ← create cursor to mongoDB db ← client.database listener ← set up the listener for streaming and set rate limit streamer ← tweepy.stream(auth, listener) streamer.filter(search_words) json_data ← collect data as a JSON object and store the same except: if on_error(status_code) then print error connecting to the database </pre>

Fig. 3. Psuedo Code for Collection of Twitter Data

out to twitter users who may be potential buyers. For this purpose, we try to gather twitter data whose search terms are similar to the product being sold. For instance, if a company is selling shoes, we try to capture tweets about users review about a particular shoe. However, when a person is tweeting about a shoe, he may not always use the word shoe. Hence, we tried to use words that are synonymous to shoes like [boots, footwear] etc as shown in Table 1.

C. Data Processing

Upon obtaining a stream of tweets in the form of a JSON file, we follow a few steps to process the text. This involves extraction of the text from the file, preprocessing, tokenization and finally analysis.

Extraction - Text from the JSON object is extracted into Python by using simple list methods, like list.append and list.insert. This ensures that only the textual matter from the JSON object is removed and available for processing. This is also done by reading the JSON file into a Python directory, followed by pretty printing to make the text readable.

Preprocessing and Tokenization - The language used in Twitter does not always conform to the rules and regulations of the English language. In order to pre process the collected tweets, we use regular expressions to separate the hashtags, emoticons, tags, hyperlinks and normal text. The text is then tokenised- separated into individual words, after removing stop words like prepositions and articles. This is called Natural Language Processing or NLP. This narrows down the data that needs to be analyzed and makes analysis more efficient.

Analysis - The tokenised data is now ready to be analysed. Analysis of text has been done using the textblob API. Textblob has a very user friendly and simple mechanism for analysing sentiments, using the bag of words method. The bag of words method consists of three steps- tokenisation, lexicon comparing and polarity calculation. As our data has been tokenised, the stop words and punctuation have been removed for easy and efficient analysis. Lexicon analysis, the next step, basically utilizes a pre existing lexicon (or bag) of words to compare every word in a given tweet. Assuming the tweet contains the word "awesome", the lexicon helps to classify this as a positive word, and vice versa for a negative term. The step that follows this is most important. This involves checking every word in the tweet, and calculating its positivity

or negativity by means of scoring. Assume that a given tweet consists of the words “awesome”, “amazing” and “boring”. The two positive words (awesome and amazing) are given a positive score, and the word boring is given a negative score. The scores of the entire tweet is then calculated by averaging the individual scores. This is repeated for the entire repository of tweets.

Sample Tweet: Playing in the beach is awesome and amazing but can get boring #weird :P
 Tweet after preprocessing: Playing in the beach is awesome and amazing but can be boring
 Hashtags: weird
 Emoticon: :P
 Tokenised tweet: Beach, amazing, awesome, boring
 Scoring the tweet: Beach: 0, amazing: 1, awesome: 1, boring: -1
 Tweet score for given tweet= 0.25

The bag of words method otherwise follows the Naïve Bayes Classification algorithm. This means that it uses the common probabilistic method to classify data as positive and negative. The simple formula that was mentioned above can be deduced directly from the one below, that uses Laplacian smoothing. P is the polarity, $P(C)$ is the probability of d occurring in class C , and $\text{count}(d, C)$ is the number of times d has occurred in class C . The decision tree of how this has been carried out has been given in Fig 4.

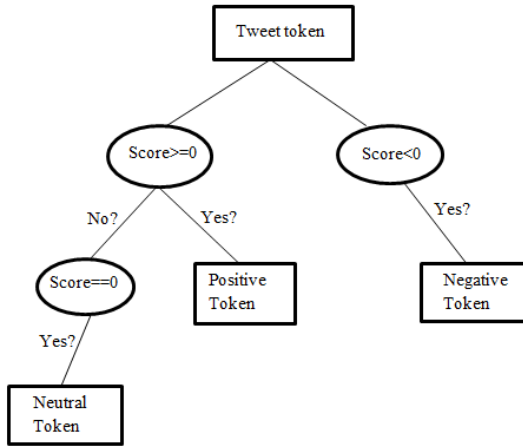


Fig. 4. Decision tree for categorization of tweets

Application of this formula to the entire dataset can provide with a polarity for every tweet. This ensures that our data is successfully classified. The pseudo code for how we've carried out data processing is given in Fig 5.

Algorithm 1 Data Processing

```

1: procedure PROCESSDATA
2:   screen.names ← list to capture positive tweeters
3:   consumer_key ← consumer key to twitter
4:   consumer_secret ← consumer secret to twitter
5:   access_token ← access token to twitter
6:   access_secret ← access secret to twitter
7:   auth ← tweepy.OAuthHandler(consumer_key, consumer_secret)
8:   auth.set_access_token(access_token, access_secret)
9:   client ← create cursor to mongoDB
10:  db ← client.database
11:  tweets_iterator ← object to iterate through the records in db collection
12:  loop:
13:    if tweet_iterator(i) != null then
14:      text ← tweet_iterator[i]['text']
15:      text_data ← TextBlob(text)
16:      if text_data.sentiment.polarity is greater than 0 then
17:        add the twitter user name to screen.names
18:      i ← i + 1.
19:    goto loop.
20:  close;

```

Fig. 5. Pseudo Code for Processing of Twitter Data

D. Bot Processing

Once the screen names for the positive tweeters have been obtained, we use twitter bots (which were traditionally used for spamming) to our advantage. These twitter bots tag screen names to promote a product of the company by sending each user a tweet with an image about the product. In order to keep to the tweet limit, these bots send tweets after a random time interval to each of the twitter users from the screen name dataset as shown in Table 2 in the bot processing output phase. In doing this, we make sure that the users remember the product name and eventually as their sent posts, they might tend to buy the product. In doing so, we make sure that the users are not continuously spammed with messages about the product. This concept of marketing is called as *positioning*. In such a scenario, the user tends to form a perception about a product. The pseudo code for the creation of a bot is shown in Fig 6.

Algorithm 1 Bot Processing

```

procedure CREATEBOT
  consumer_key ← consumer key to twitter
  consumer_secret ← consumer secret to twitter
  access_token ← access token to twitter
  access_secret ← access secret to twitter
  auth ← tweepy.OAuthHandler(consumer_key, consumer_secret)
  auth.set_access_token(access_token, access_secret)
  screen.names ← list of positive tweeters
  loop:
    if screen.names[i] != null then
      status ← tag screen.names[i] in status
      fn ← path in local host to image
      api.update_with_media(fn, status)
      sleep(random amount of time)
    i ← i + 1.
  goto loop.
  close;

```

Fig. 6. Pseudo Code for Bot Processing

VI. Results

In this section, we present the outputs of each phases talked above in Table 2 presented below. As discussed, the first phase involves an application created for the company where they enter relevant product terms. The output for the data collection phase is stored in the form of a JSON object which

Initial Application



Data Collection

```
{
  "id": "14391672",
  "text": "Good 2001 #Nike Air Jordan #Retro XI Concord 11 White Black Size 9.5 #Shoes #Share",
  "source": "https://t.co/6RQytqCbcz",
  "in_reply_to_status_id": null,
  "in_reply_to_user_id": null,
  "retweet_count": 0,
  "favorite_count": 0,
  "created_at": "2012-01-24 10:15:07",
  "lang": "en",
  "contributors_enabled": false,
  "is_quote_status": false,
  "extended_entities": {
    "media": [
      {
        "id": "14391672",
        "media_url_https": "https://pbs.twimg.com/media/Cq17WvMA550t.jpg",
        "type": "photo",
        "sizes": {
          "thumb": {
            "w": 150,
            "h": 150,
            "crop": "crop"
          },
          "small": {
            "w": 453,
            "h": 453,
            "crop": "crop"
          },
          "medium": {
            "w": 1066,
            "h": 1066,
            "crop": "crop"
          },
          "large": {
            "w": 1066,
            "h": 1066,
            "crop": "crop"
          }
        },
        "url": "https://t.co/6RQytqCbcz"
      }
    ]
  },
  "user": {
    "id": "14391672",
    "name": "Anush Baskaran",
    "screen_name": "anushbaskaran17",
    "location": "San Francisco",
    "description": "I love anything that sparkles. #Kaleidoscope #Kaleidoscope #Kaleidoscope #Kaleidoscope #Kaleidoscope #Kaleidoscope #Kaleidoscope #Kaleidoscope #Kaleidoscope #Kaleidoscope",
    "url": "https://t.co/6RQytqCbcz",
    "followers_count": 1449,
    "friends_count": 1449,
    "retweet_count": 0,
    "favorite_count": 0,
    "created_at": "2012-01-24 10:15:07",
    "lang": "en",
    "contributors_enabled": false,
    "is_quote_status": false,
    "extended_entities": {
    }
  }
}
```

VII. Conclusion

It can be concluded that bots can be utilised for a positive outlook in terms of marketing. The bot can be tailored to work towards attracting a customer base who is genuinely interested in utilising the product after analysing their sentiment towards it. This has been found to reduce spam, owing to the selective nature of delivering its advertisements. Also, having sentiments of the users known, the user has ample scope to review, revise and upgrade their products. The efficiency in terms of time and energy has also been optimized by

makes extracting certain parts of the tweet easier in later stages. In the data processing phase, we classify the tweets as positive, negative, and neutral and store the screen names of those people whose tweets seem positive. In the final phase, a tweet is sent out by our bot giving details about the company's products.

Data Processing

#Good 2001 #Nike Air Jordan #Retro XI Concord 11 White Black Size 9.5 #Shoes #Share
are <https://t.co/6RQytqCbcz> <https://t.co/Eto0u0c48t>

Sentiment(polarity=0.17777777777777778, subjectivity=0.34444444444444445)

Bot Processing



automating the process using bots. This in turn helps in reducing delays, and cuts monetary costs too.

References

- [1] Agarwal, Apoorv, et al. "Sentiment analysis of twitter data." Proceedings of the workshop on languages in social media. Association for Computational Linguistics, 2011.

- [2] Pak, Alexander, and Patrick Paroubek. "Twitter as a Corpus for Sentiment Analysis and Opinion Mining." *LREc*. Vol. 10. No. 2010. 2010.
- [3] Wilson, Theresa, Janyce Wiebe, and Paul Hoffmann. "Recognizing contextual polarity in phrase-level sentiment analysis." *Proceedings of the conference on human language technology and empirical methods in natural language processing*. Association for Computational Linguistics, 2005.
- [4] Go, Alec, Richa Bhayani, and Lei Huang. "Twitter sentiment classification using distant supervision." *CS224N Project Report, Stanford 1.12* (2009).
- [5] Chu, Zi, et al. "Who is tweeting on Twitter: human, bot, or cyborg?." *Proceedings of the 26th annual computer security applications conference*. ACM, 2010.
- [6] Kouloumpis, Efthymios, Theresa Wilson, and Johanna D. Moore. "Twitter sentiment analysis: The good the bad and the omg!." *Icwsn* 11.538-541 (2011): 164.
- [7] Haustein, Stefanie, et al. "Tweets as impact indicators: Examining the implications of automated "bot" accounts on Twitter." *Journal of the Association for Information Science and Technology* 67.1 (2016): 232-238.
- [8] Wald, Randall, et al. "Predicting susceptibility to social bots on twitter." *Information Reuse and Integration (IRI), 2013 IEEE 14th International Conference on*. IEEE, 2013.
- [9] Boshmaf, Yazan, et al. "The socialbot network: when bots socialize for fame and money." *Proceedings of the 27th annual computer security applications conference*. ACM, 2011.
- [10] Azeem, Mohd Abdul. "Consumers' Attitudes toward Commercial E-mail Spam and Web pop-ups: Interference, Perceived Loss of Control, and Irritation." (2012).
- [11] Bonzanini, Marco. *Mastering Social Media Mining with Python*. Packt Publishing Ltd, 2016.