Available online at www.sciencedirect.com

ScienceDirect

Procedia Computer Science 126 (2018) 451–461

Procedia Computer Science

www.elsevier.com/locate/procedia

International Conference on Knowledge Based and Intelligent Information and Engineering Systems, KES2018, 3-5 September 2018, Belgrade, Serbia

Identifying Fake News and Fake Users on Twitter

Costel-Sergiu Atodiresei*, Alexandru Tănăselea, Adrian Iftene

Faculty of Computer Science, Alexandru Ioan Cuza University, General Berthelot, 16, Iasi 700483, Romania

Abstract

In the last years big social networks like Facebook or Twitter admit that on their networks are fake and duplicate accounts, fake news and fake likes. With these accounts, their creators can distribute false information, support or attack an idea, a product, or an election candidate, influencing real network users in making a decision. In this paper, we present our system build with the aim of identifying fake users and fake news in the Twitter social network.

Keywords: Fake news; User credibility; Twitter

1. Introduction

Fake News focuses on classifying the credibility of a tweet post. It makes and presents some scores and their interpretation. Online news have the power to influence millions of people, but there are no ways to establish what it is true and what it is fake. We might guess what common sense says it's true, but sometime we might just need some impartial, trustable source of news.

Studies show that everybody has problems with identifying fake news, disregarding criteria's such as age or education. Some 82% of middle-schoolers could not distinguish between an ad labeled "sponsored content" and a real news story on a website, according to a Stanford University study of 7,804 students from middle school through college. Many students judged the credibility of newsy tweets based on how much detail they contained or whether a large photo was attached, rather than on the source [1].

Both Google and Facebook, along with Twitter, are under growing pressure to better manage their algorithms as more details emerge about how Russia used their platforms to interfere in the presidential election to sow discord [2].

The platforms have an immense influence on what gets seen and read. More than two-thirds of Americans report getting at least some of their news from social media, according to the Pew Research Center. A separate global study published by Edelman last year found that more people trusted search engines (63%) for news and information than traditional media such as newspapers and television (58%) [2].

Facebook rolled out a third-party fact-checking program with PolitiFact, FactCheck.org, Snopes.com, ABC News, and the Associated Press. Those partnerships, however, did not stop inaccurate reports from landing on Facebook's Crisis Response page.

* Corresponding author. *E-mail address:* atodiresei.costel.sergiu@info.uaic.ro

10.1016/j.procs.2018.07.279

Researchers have begun in recent years to address the issue of identifying false news and their credibility on television and YouTube [3], Twitter [4, 5], Facebook [6], displaying sites that train and help users who want to identify false news.

Methods of assessing veracity come from two major categories - linguistic approaches (with automatic learning) and social networking approaches. The analysis of Twitter data is increasingly used in analyzing feelings, identifying political opinions, rapidly identifying events occurring in the world, natural phenomena in progress, measuring people's satisfaction with services offered in health. The techniques used involve automatic learning, the use of dictionaries with terms specific to a domain we want to monitor, Naive Bayes classifiers, models based on Maximum Entropy and SVM. All these are used in combination with techniques and tools specific to natural language processing: POS, stop-word removal, identification of name-type entities, sentiment identification etc. [7-11].

Our application aims to receive the link to a tweet from the user and to compute its credibility, based on comparison to trustworthy news sources and based on the credibility (so far) of the user. It also computes other statistics, such as the overall emotion (sentiment) of the tweet, taking into consideration the emoji and hashtags used in the twitter (if any).

2. Proposed solutions

2.1 General Architecture of the System

The general architecture of the system is presented in Figure 1. There is a **Twitter crawler** component, which collects tweets and adds them to our **database**. When we will need tweets from trustworthy sources to compare with our current one, we can retrieve them directly from our database. The **Processing module**: when a user wants to know the credibility of a new tweet, he inputs the link of the tweet in our interface. Our algorithm then uses an **NER** (Named Entity Recognition) **component**, which split the text into its composing parts: it brings out the entities (generally, nouns and their relative importance in the context), the topics, the social tags, the overall tweet sentiment and the hashtag sentiment.

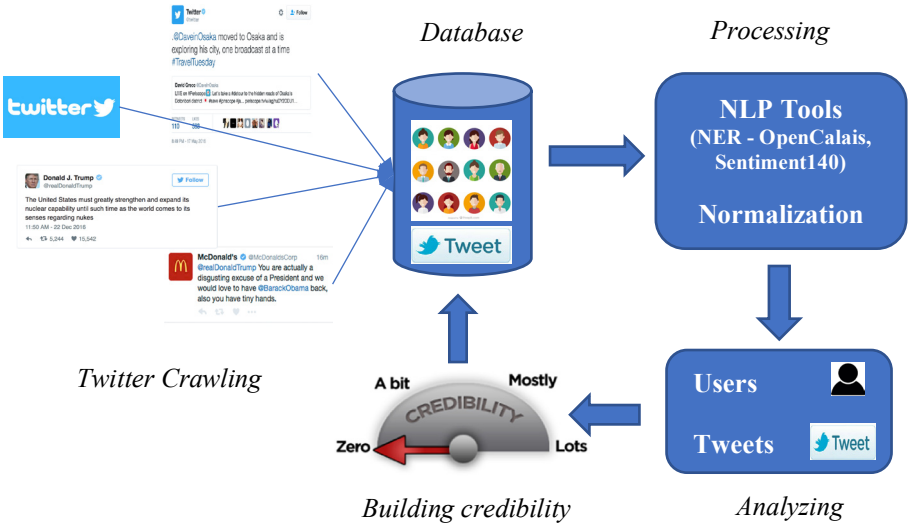

Fig. 1. System architecture

For the NER parsing, we used a public API called OpenCalais [12], while for the sentiment computation, we used Sentiment140 [13]. For the hashtag metric, we also used Sentiment140 on the regex formed by the said text.

Analyzing module: we combine this analysis of the current tweet with what we have stored locally in our database. Specifically, we look up similar tweets in trustworthy sources - this is why the application is better suited for news - similar pieces of information that can be also found online. **Building credibility module**: we retrieve this from the database, searching similar tweets, with similar NER decomposition - in this case, even if the order of the words is jumbled up, we can still identify similar tweets. If we find similar tweets, the score of our tweet increases - i.e. the chances that it is true are higher.

Meanwhile, we also take into consideration the score of the user. Originally, the user starts with the score of 0 - we assume that he is not saying the truth. The more tweets he posts that are similar and can be verified through our trustworthy sources, the more his own credibility score increases. The app outputs the User Score, the Tweet Score and a message describing the tweet as overall true, false, or unable to verify.

2.2 Implementation Details

This application is based on service-oriented architecture, where different modules are developed to interact and improve the efficiency of response. The main programming language used in most of the modules is Java and our component that handles the persistent storage is developed using Node.js. Our system is using a NoSQL database, the well-known MongoDb to keep unstructured information, and organize its content in two collections: tweets and users.

Node.js is an asynchronous, event-driven engine where the application makes a request and then continues working on other useful tasks rather than stalling while it waits for a response. On completion of the requested task, the application is informed of the results via a callback or a promise or Observable. This enables large numbers of operations to be performed in parallel – essential when scaling applications. MongoDB was also designed to be used asynchronously and so it works well with Node.js applications.

Main modules

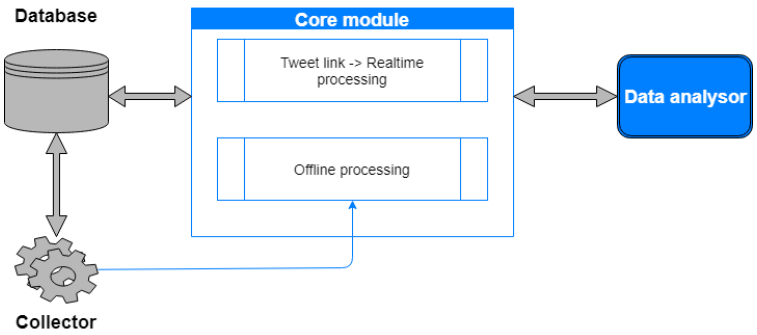

Fig. 2. Interactions between main modules

Core module: this module aggregates all components and orchestrates their behavior (see Figure 2). There is an endpoint where all the magic begins and behind the scenes online and offline processing is starting during different requests from users or from another module. The online processing means a decision over a tweet; giving an URL of a tweet, the application returns, in real time, a response to its trustworthy. On the other way, there is an offline processing mechanism for progress evaluation of a user.

Collector module: to collect information means life for this application. An outdated system is useless and can mislead to respond with a wrong decision about trustworthy of a user or tweet. This module collects users and tweets and fires the action to core module to start offline processing whenever is needed. At the beginning is initialized with some trusted sources that increase the validity of content.

A **Database module** that handles the requests to the database for add, remove, update and retrieve some data to reduce the complexity of architecture of each module, by implementing the layer of persisting data plays its role like a charm. Moreover, using a programming language that exploits the capabilities of the chosen database, give us a generous bonus of performance.

Analyzing module is one of the most important modules and it is responsible for the processing and analyzing the tweets. There are many components that helps us to decide the trustworthy of a tweet. The most important components are *hashtag sentiment*, *emoji sentiment*, *text sentiment* and *named entity recognition*. Based on above tools, the business logic behind this module can process the tweet and returns very accurately the needed response.

Obtaining the credibility score

The algorithm analyzes the link received and returns a string message that includes the *Decision*, the *User score,* and the *Tweet Score*. The twitter link is composed of twitter address, user screen name and status id (tweet id). Twitter has a user-friendly manner of taken this tweet link.

Next, the algorithm works as it follows:

- It gets the user details from our database;
- If we do not have the user in the database, get its values from Twitter and initialize its score to zero;
- It gets the tweet's details from Twitter;
- Creates a tweet score code based on the received data from different APIs (OpenCalais, Sentiment140) used to process (similar) tweets (Text Processing Service);
- Creates a user score code as an arithmetic mean of its prior score and actual tweet score.

Tweet score: this is, in brief, how the algorithm (used by Text Processing Service) considers the different measurements for the "fakeness" of a tweet:

1. It's start with a score of 0 and we extract the entities (companies, people, places, products, etc.) with OpenCalais (NER decomposition).
2. If the search text does not have entities, the response is: *We cannot tell with precision whether the tweet is fake or not*. The score will be -500.
3. The program queries for similar tweets posted by verified users (text similarity over 0.25).
4. If 0 tweets are found, the response is: *We cannot tell exactly if a tweet is fake or not*. The score will be -500.
5. For each similar tweet found, we the extract entities with OpenCalais and 10 points are added to the score. If the similarity score between entities is 1, then 40 points are added to the score. If the similarity score is between 0.5 and < 1, then 10 points are added to the score. If the similarity score is between 0.1 and 0.5, then 5 points are added to the score. Otherwise, we do not change the score.
6. If during parsing of the tweets, the score exceeds 150 points, break and the score becomes 100. Also if the score is higher than 100 at the end.
7. Finally, if the score is greater or equal to 50, the response will be: *Tweet is not fake. Confidence: score*. Else, if the score is less to 50, then the response will be: *Tweet is fake. Confidence: 100 - score*.
8. If only one tweet is found, the score will be –score (negative) and the response will specify: *Only one source found*.
9. If something went wrong, the score is -1000.

In conclusion, the *tweet score* can be 1000, -500 or in interval [-50, 100].

The **user score** is based on his tweets, being highly influenced by the latest tweets inserted into the application. Given a user with a popular tweet, after a while, he gets a credibility as high as its tweet credibility.

After the *tweet score* has been calculated, it is used to calculate processed tweed score t (which is used then in calculating of the *user score u*):

- $t = 0$ (initially);
- if *tweet score* is -1000 or -500, then t remains 0;
- if *tweet score* is between -50 and 50, then $t = 1$;
- if *tweet score* is between 51 and 70, then $t = 9$;
- if *tweet score* is between 71 and 90, then $t = 10$;
- if *tweet score* is between 90 and 99, then $t = 11$;
- if *tweet score* is 100, then $t = 12$.

Now we can calculate the *user score* based on variable *t*:

1) $u = 0$ (initially – 0 analyzed tweets)
2) $u = \frac{u+t}{2} \rightarrow u = \frac{t}{2}$ (after one analyzed tweet)
3) $u = \frac{u+t}{2} \rightarrow u = \frac{3t}{4}$ (after two analyzed tweets)

... ...

where *u* = user score, *t* = processed tweet score and numbers 1), 2), 3), ... are iterations.

In order to have a high score, the user (the person that tweets) should have high scores at all his tweets. Anyway, if a visitor to our app, press the button *Check* for the same tweet for a couple of times, the user score is stabilized at the inserted tweet score. By introducing the possibility of restricting the user at influencing the score just once, we can justify that the *user score* is influenced by popular tweets. Also, it is necessary for the future to change the mean to take less into consideration just the last tweet and be a mean of all the tweets inserted.

The overall range of the user score is from 0 to 12, higher the value, higher the trust. A score of 6 means we don't say it is true, but we don't say either that it is false.

3. Use Cases

In the next examples are snapshots of received raw data from the different APIs (Open Calais, sentiment140) used to process the tweets (Online processing), where *stats_code* represents *tweet score*.

Use Case 1 – The First Real Tweet

Tweet text: "*Live pictures from United States show early stages of #SuperBlueBloodMoon, which hasn't been seen for more than 150 years ◯.*"

We have obtained the following result and statistics (see Table 1):

Table 1. The result returned by the NLP services for the first tweet.

Twitter text	Named entities	Topics - confidence	Sentiment score
Live pictures from United States show early stages of #SuperBlueBloodMoon, which hasn't been seen for more than 150 years :)	*United States*	*Entertainment_Culture - 1* *Environment - 0.537* *Human Interest - 0.576*	*Text - 2* *Emoji - 4* *Hashtag - 2*

```
stats: "Tweet is not fake. Confidence: 100.0",
stats_code: 100,
hashtag_sentiment: 2,
similar_tweets: {
    nr_analyzed_tweets: 1369,
  - tweets: {
      - Live pictures from United States show early stages of #SuperBlueBloodMoon, which hasn't been seen for more than 150…: {
            similiraty_entities: 1,
            user_name: "BBC News (World)",
            user_descripton: "News, features and analysis from the World's newsroom. Breaking news, follow @BBCBreaking. UK news,
            similiraty_text: 0.8357142857142857
        },
      - RT @BBCWorld: Live pictures from United States show early stages of #SuperBlueBloodMoon, which hasn't been seen for more
            similiraty_entities: 1,
            user_name: "Tom Ackerman",
            user_descripton: "LEX 18 Morning/Noon Prognosticator (CBM 260), Dad, Navy Vet, runner, hiker, gardener & minder of my
            similiraty_text: 0.8920863309352518
        }
    },
    nr_verified_users: 3
},
```


Fig. 3. The content of the JSON returned by the main service for the first tweet

The tweet is not fake because there were 2 tweets posted by verified users (BBC News and Tom Ackerman) with a text similarity over 25% (see Figure 3). The searched tweet has only one entity found, and because the similar tweets contain this entity it turned out that the trust score *(stats_code*) is 100 points. So, for each similar tweet found, 10 points are added to the score and because the similarity score between entities is 1, then 40 points are added to the score (2×10 + 2×40 = 100).


```
stats: "We can not say precisely if the tweet is false or not",
stats_code: -500,
hashtag_sentiment: 2,
similar_tweets: { },
text_sentiment: 2,
ner: {
  - entities: [
      - {
            name: "United States",
            type: "Country",
            relevance: 0.2
        }
    ],
  - topics: [
      - {
            confidence: 0.537,
            name: "Environment"
        },
      - {
            confidence: 1,
            name: "Entertainment_Culture"
        },
      - {
            confidence: 0.576,
            name: "Human Interest"
        }
    ]
},
text: "Live pictures from United States show early stages of #SuperBlueBloodMoon, which hasn't b
emoji_sentiment: 4
```


Fig. 4. The content of the JSON returned by the main service for the first tweet changed

If we change the text from the tweet with this: "*Live pictures from United States show early stages of #SuperBlueBloodMoon, which hasn't been seen for more than **500** years ◯*".

It can be seen that no similar tweets have been found and in this case, the stats_code is -500 and the result is: "*We cannot say precisely if the tweet is false or not*" (See Figure 4). The method query from Ttwitter4j API is quite sensitive to changes, even if they are minor.

Use Case 2 – The Second Real Tweet

Tweet text: "*Donald Trump uses State of the Union address to say: "There has never been a better time to start living the American dream"*".

We have obtained the following result and statistics (see Table 2):

Table 2. The result returned by the NLP services for the second tweet.

Twitter text	Named entities	Topics - confidence	Sentiment score
Donald Trump uses State of the Union address to say: "There has never been a better time to start living the American dream"	*Donald Trump*	*Labor - 0.865*	*Text - 0* *Emoji - -1* *Hashtag - -1*


```
stats: "Tweet is not fake. Confidence: 50.0. Only one source found.",
stats_code: -50,
hashtag_sentiment: "-1",
similar_tweets: {
    nr_analyzed_tweets: 84,
  - tweets: {
      - Donald Trump uses State of the Union address to say: "There has never been a better time to start living the Americ…: {
            similiraty_entities: 1,
            user_name: "BBC News (World)",
            user_descripton: "News, features and analysis from the World's newsroom. Breaking news, follow @BBCBreaking. UK news,
            similiraty_text: 0.8285714285714286
        }
    },
    nr_verified_users: 1
},
```


Fig. 5. The content of the JSON returned by the main service for the second tweet

The tweet is not fake, with a 50% confidence, because a single tweet (posted by a verified user - BBC News) was found with a text similarity over 25% (see Figure 5). The searched tweet has a single entity and the similar tweet contains this entity, it turned out that the trust score is 50 points (10 points for text similarity over 25% and 40 points for identical entities). A single source of trust has been found and for that, the *stats_code* is -50.

If we change the text from the tweet with this: "*Donald Trump uses State of the Union address to say: "There has never been a better time to start living the American **nightmare**"*", it can be seen again that no similar tweet was found, as in the case of the first example (see Figure 6).


```
stats: "We can not say precisely if the tweet is false or not",
stats_code: -500,
hashtag_sentiment: "-1",
similar_tweets: { },
text_sentiment: 0,
ner: {
  + social_tags: […],
  - entities: [
      - {
            confidence: "0.998",
            name: "Donald Trump",
            type: "Person",
            relevance: 0.8
        }
    ],
  - topics: [
      - {
            confidence: 0.909,
            name: "Labor"
        }
    ]
},
text: "Donald Trump uses State of the Union address to say: "There has never been a better time to start living the American nightmare",
emoji_sentiment: -1
```


Fig. 6. The content of the JSON returned by the main service for the second tweet changed

Use Case 3 – The First Real User

Initially, users have a credibility score set at 0, both online processing and offline processing updates the score, and trustworthy of a user, expecting trusted sources, may vary.

For the following example, we will use an account not checked in the application before. The text is taken from Wikipedia and it is true [14] (see Figure 7).

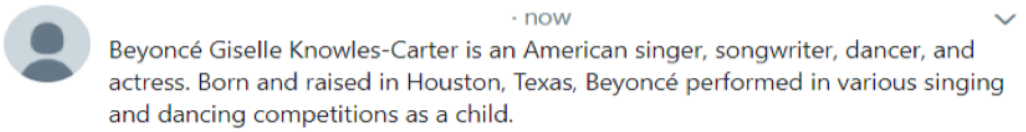

Fig. 7. The text from Wikipedia is inserted in our application

Running the application a few times, we receive the results indicated below (see Table 3). We can see how the user score becomes stable, closed to the processed tweet score.

Table 3. The result returned by the NLP services for the second tweet.

Run	Message
1	We cannot say precisely if the tweet is false or not. User score: 2.5, Tweet score: 5.0.
2	We cannot say precisely if the tweet is false or not. User score: 7.5, Tweet score: 6.0.
3	We cannot say precisely if the tweet is false or not. User score: 6.75, Tweet score: 6.0.

4	We cannot say precisely if the tweet is false or not. User score: 5.875, Tweet score: 5.0.
5	We cannot say precisely if the tweet is false or not. User score: 5.9375, Tweet score: 6.0.

After a couple of days, after checking the same link, we can see that the user score has started approximately with the same score it had before "*We cannot say precisely if the tweet is false or not. User score: 5.5625, Tweet score: 5.0.*".

Use Case 4 – The Second Real User

For the second example, we were curious what says the application of Centric, an IT company (see Figure 8).

Fig. 8. The text for Centric Company

It was the first time searching for a tweet for Centric IT Company, so the user score started again from 2.5, then stabilizing between 5 and 6: "*We cannot say precisely if the tweet is false or not. User score: 5.2675, Tweet score: 5.0.*"

4. Error Analysis

Similar Tweets

We want to test how does our software [15] behave when we input the same tweet, but from different users. We start with this initial tweet, from CNN [16]:

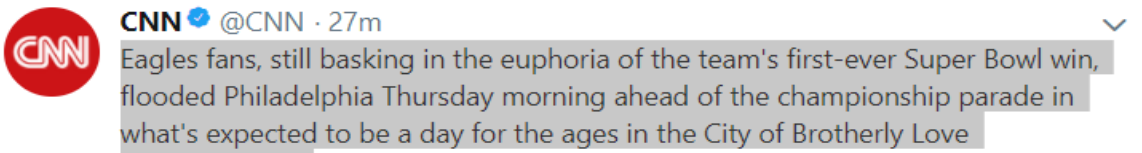

Fig. 9. The text for Centric Company

We obtain the following result (only the data about the similarity to other tweets): initial it found 88 similar tweets and the confidence is 60.0. When we take the same tweet, this time retweeted by another user, we get 99 similar tweets and the confidence score is still 60.

We can try our application with several other retweets, hence the number of similar tweets will increase (to include the ones in our database, too), but the confidence score stays the same. In the future, a possible improvement is to take into consideration the number of retweets in our algorithm and to embed it the formula for the confidence score.

Recognizing Opinion Tweets

We want to test out the following tweet, which is a review of a book:

Fig. 10. Review for a Book

Although this is a bit beyond the scope of our project, we want to see whether the algorithm can recognize a tweet that states an opinion versus one that is a piece of news and claims to be authentic. The text sentiment returned, computed with Sentiment140, is 2-Neutral. Since this is a 5-star review (out of 5), it goes to show that the algorithm could be improved to figure out on its own if a tweet is actually a review or an opinion and should not be taken into consideration when fact-checking news.

We observed that the results of our application are reliable, but it has a few drawbacks:

- It only works with tweets in English, that contain primarily text;
- It can be fooled when the news is from one of the trustworthy sources, but it's new (so it won't find similar tweets).

5. Conclusions

Until now, the presented project has reached its goal, meaning that based on tweet's text and tweet's user, it returns a set of statistics about the tweet's veracity. The project is still under developing and we still work on presented modules in order to improve the quality of everyone and the entire overall quality of the system.

Detecting indeed whether a news is fake or not, based solely on its popularity in the same social network might not be the best idea. In present, Facebook is using human resources that investigate the popular posts in order to decide if they are fake or not.

In the next period, we will try to add other information from other sources (like well-known newspapers or the results from Google searches, in order to deal with first seen news, which hasn't similarities in the Twitter network).

Acknowledgements

We would like to thank the master students involved in developing the first version of the application presented in this paper. This work is partially supported by POC-A1-A1.2.3-G-2015 program, as part of the PrivateSky project (P_40_371/13/01.09.2016).

References

1. Shellenbarger S. Most Students Don't Know When News Is Fake, Stanford Study Finds. *The Wall Street Journal*, 21 November 2016: https://www.wsj.com/articles/most-students-dont-know-when-news-is-fake-stanford-study-finds-1479752576 accessed last time on March, 2018.
2. Pierson D. Facebook and Google pledged to stop fake news. So why did they promote Las Vegas-shooting hoaxes? Los Angeles Time, 2 October 2017: http://www.latimes.com/business/la-fi-tn-vegas-fake-news-20171002-story.html accessed last time on March, 2018.
3. Clark CS. *Fake news? A survey on video news releases and their implications on journalistic ethics, integrity, independence, professionalism, credibility, and commercialization of broadcast news*. PhD Thesis, College of Communication and Information Sciences in the Graduate School of the University of Alabama, 2009.

4. Castillo C, Mendoza M, Poblete B. Information Credibility on Twitter. WWW 2011 – Session: Information credibility, ACM, 1, Hyderabad, India, 2011, p. 675-684.
5. Iozzio C. Reuters built a bot that can identify real news on Twitter. Who says AI can't spot fake news? *Popular Science, Technology*. 2016: https://www.popsci.com/artificial-intelligence-identify-real-news-on-twitter-facebook accessed last time on March, 2018.
6. Allcott H, Gentzkow M. Social Media and Fake News in the 2016 Election. *Journal of Economic Perspectives*, 2017: **31** (2), p. 211–236.
7. Agarwal B, Xie I, Vovsha O, Rambow R, Passonneau R. Sentiment Analysis of Twitter Data. In *Proceedings of the ACL 2011*, Workshop on Languages in Social Media, 2011, p. 30-38.
8. Barbosa L, Feng J. Robust Sentiment Detection on Twitter from Biased and Noisy Data. *COLING* 2010, p. 36-44.
9. Gamallo G, Garcia M. Citius: A Naive-Bayes Strategy for Sentiment Analysis on English Tweets. *8th International Workshop on Semantic Evaluation* (SemEval 2014), 2014, p. 171-175.
10. Kharde VA, Sonawane SS. Sentiment Analysis of Twitter Data: A Survey of Techniques. International Journal of Computer Applications (0975 – 8887), 2016: **139** (11), p. 5-15.
11. Gînscă AL, Popescu A, Lupu M, Iftene A, Kanellos I. Evaluating User Image Tagging Credibility. In *Experimental IR meets Multilinguality, Multimodality, and Interaction*. Lecture Notes in Computer Science, 2015: **9283**, p. 41-52.
12. Open Calais: http://www.opencalais.com/ accessed last time on March, 2018.
13. Sentiment140: http://www.sentiment140.com/ accessed last time on March, 2018.
14. Wikipedia Beyoncé: https://en.wikipedia.org/wiki/Beyonc%C3%A9 accessed last time on March, 2018.
15. Tweet Analyzer: http://tweetanalyzer-env.us-east-2.elasticbeanstalk.com/ accessed last time on March, 2018.
16. CNN: https://twitter.com/CNN accessed last time on March, 2018.