K&#x0131;l&#x0131;n&#x00E7; Deniz (Orcid ID: 0000-0002-2336-8831)

A Spark-based Big Data analysis framework for real-time sentiment prediction on streaming data

Deniz Kılınç
Manisa Celal Bayar University, Department of Software Engineering, Manisa, Turkey

Correspondence
Dr. Deniz Kılınç, Manisa Celal Bayar University, Faculty of Technology, Department of Software Engineering, 45400, Turgutlu/Manisa
Email: deniz.kilinc@cbu.edu.tr

SUMMARY

There are many data sources that produce large volumes of data. The Big Data nature requires new distributed processing approaches to extract the valuable information. Real-time sentiment analysis is one of the most demanding research areas that requires powerful Big Data analytics tools such as Spark. Prior literature survey work has shown that, though there are many conventional sentiment analysis researches, there are only few works realizing sentiment analysis in real-time. One major point that affects the quality of real-time sentiment analysis is the confidence of the generated data. In more clear terms, it is a valuable research question to determine whether the owner that generates sentiment is genuine or not. Since data generated by fake personalities may decrease accuracy of the outcome, a smart/intelligent service that can identify the source of data is one of the key points in the analysis. In this context, we include a fake account detection service to the proposed framework. Both sentiment analysis and fake account detection systems are trained and tested using Naïve Bayes model from Apache Spark's machine learning library. The developed system consists of four integrated software components: (i) Machine learning and streaming service for sentiment prediction (ii) A Twitter streaming service to retrieve tweets (iii) A Twitter fake account detection service to assess the owner of the retrieved tweet (iv) A real-time reporting and dashboard component to visualize the results of sentiment analysis. The sentiment classification performances of the system for offline and real-time modes are 86.77% and 80.93% respectively.

Keywords: Big data machine learning; real-time sentiment analysis, streaming data; Twitter streaming; fake account detection

1. INTRODUCTION

Big Data is mostly defined in terms of 3Vs, velocity, variability and volume, and describes the characteristics of new digital information. Data is continually being generated from many sources such as sensor data (medical devices, car-sensors, road cameras etc.), machine log data (clickstream, geo-location), digital media (images, video etc.), archives of any kinds of documents (e-mail, HTML, XML etc.) and social media (Twitter, Facebook etc.). The characteristics of Big Data requires automated data-analysis where machine learning (ML) algorithms can be used to realize this requirement. In particular, automated sentiment analysis (SA) systems are required to analyze social media data [1].

Sentiment is defined as the thought or opinion of people about any item. The textual opinions of people about products, services, brands, and their attributes are explored with the use of natural language processing techniques [2]. Social networking applications such as Twitter, Tumblr where people use short messages to express their opinions. The quantity of this information continuously increases to be named as Big Data. For instance, Twitter has 695m registered users and 9,100 tweets happen per second [3].

As the information growth rate increases, automation of data analysis tasks such as SA are performed with ML techniques. Many researches about automated SA are conducted on relatively small or static data sizes with conventional approaches. Nevertheless, the volume and velocity of generated data is so enormous that there is an increasing gap between analyzed data and data to be analyzed. An emergent solution to decrease this gap is to benefit from Big Data analytics solutions. There are mainly two big data processing strategies in SA: i) Static strictly-controlled structures, i.e. batch processing, which stores data on a distributed file system and then uses a distributed computational framework such as Hadoop MapReduce (MR). ii) Interactive or real time streaming data processing such as Apache Spark which collects data as "streams" and process the streaming data through an in-memory computation strategy such as resilient distributed datasets (RDD) [4, 5].

Considering the conventional SA researches and frameworks in the literature, there exists many SA studies related to different languages. However, infrequent researches (mostly in English) are conducted about real-time sentiment analysis in the literature [8-12]. There is an increasing need to analyze real-time stream data such as spot and stop fraudulent activity in the finance domain, inventory management, web analytics/content management (sales performance evaluation through searched keywords) and real-time customer behavior analysis to improve customer experience [6].

In addition to the need for real-time sentiment analysis, there exists an additional fundamental challenge about the reliability of owner of the generated data. As in any social environment, fake accounts in Twitter can be sources of manipulation about brands, people and services for unethical purposes. In this context, a 'fake account' can be defined as the account that does not belong to a real user and who may post misleading comments or news. The data generated by fake accounts consequently may decrease the quality of SA outcome and therefore there is a need to filter out suspicious user accounts with the support of a smart service.

The growing requirement for streaming data analysis is our first motivation point. Our second key motivation factor in this study is that, there are only a few big data oriented SA studies in Turkish. Our third motivation point and the most distinctive feature of the system is that a fake account detection that increases the overall precision is proposed as an integrated service for the first time in SA. To the best of our knowledge, this is the first real-time sentiment analysis framework including a fake account detection module conducted on SA.

In this paper, we demonstrate real-time sentiment analysis of Turkish tweets using a machine learning model from Spark MLlib package. The proposed system mainly consists of four interrelated software components: (i) Spark machine learning and streaming service to create and test prediction model for SA (ii) A Twitter streaming service to collect streaming tweets and real-time processing of the data that is fed to ML service (iii) A Twitter fake account detection service that controls the owner of the retrieved tweet (iv) A real-time reporting and dashboard software to monitor SA of collected tweets.

In SA, different types of ML algorithms such as Decision Tree (DT), Support Vector Machines (SVM) and Naïve Bayes (NB) are used. NB requires less computing power compared to the other methods and hence the NB learning model from MLlib is selected to create a Spark based real-time prediction service that controls the reliability of tweet's owner and analyzes sentiments of tweets collected depending on the query terms sent to the Twitter streaming service by the user. First the owner of a Tweet is classified as real or fake using Twitter fake account detection service in real-time. Sixteen attributes (*description*, *followers_count*, *friends_count*, *statuses_count* etc.) of a tweet are extracted and then tested utilizing a NB model that is trained with a dataset named TwFakeUsr [27]. If owner of the tweet is classified as fake, then the corresponding tweet message is discarded. On the other hand, if the identity is identified as genuine, then the related tweet is sent to next service for pre-processing and sentiment prediction.

The NB model is built following sequential data pre-processing steps explained in Section 3.6. The features of real-time tweets are obtained on the fly and they are fed directly to sentiment classification engine to extract their corresponding sentiments. The consequent statistics results through the prediction service, are displayed on the real-time reporting and a dashboard software developed to monitor the system. HUMIR [7] dataset which is a set of hotel and movie reviews collected from two well-known recommendation web sites is used in order to train and test NB model for SA.

The rest of the paper is organized as follows. We explain real-time sentiment prediction frameworks in Section 2. We describe our generic sentiment analysis framework in Section 3 that includes the method of Twitter real-time data access, used datasets, ML algorithm used in the proposed architecture and the implementation of the system as a whole. Section 4 presents the results of the experiments and the discussion drawn from results of the corresponding experiments. In Section 5 related works are presented. Finally, Section 6 concludes the paper and it presents proposed future work followed by an acknowledgement.

2. REAL-TIME SENTIMENT PREDICTION FRAMEWORKS

Microblogging platforms, particularly Twitter, are valuable sources to extract sentiments of their users about movies, products and events. In this manner, a real-time sentiment analysis may be used and integrated with recommendation systems or health status prediction system [4, 8].

From a data point of perspective, social networking platforms generate Big Data that is huge in volume and in production velocity. Real-time analysis of such data requires new tools and techniques such as distributed computing frameworks [14]. Hadoop depending on MR model and Spark relying on RDD theory are two widely-used distributed data processing approaches. The main difference between the two models is that while the former uses a disk-based processing approach, the latter makes use of an in-memory model resulting in much faster data processing.

2.1. Apache Hadoop

Hadoop MR following a functional programming model makes use of map() and reduce() functions while processing data stored on a distributed file system such as Hadoop Distributed File System (HDFS). In this model, a task is divided into jobs and processing of each job requires consecutive read-writes from/to a distributed file system. Since iterative data processing strategies, i.e machine learning algorithms, needs multiple rounds of computation on the same data, the performance of MR models becomes inefficient [15]. Though it is a batch-processing framework, Hadoop MR may also be used for real-time processing with the help of Hadoop Streaming utility.

2.1. Apache Spark

Spark in comparison to Hadoop MR uses an in-memory data structure (RDD) to process data without excessive reading and writing to disks. The main abstraction in Spark is that of an RDD, which represents a read-only collection of objects partitioned across a set of machines that can be rebuilt if a partition is lost [16]. RDDs can be created in three ways:

i. From a distributed file system such as HDFS,
ii. Dividing a Scala collection into number of slices and sending those pieces to multiple nodes, or
iii. Making transformations from existing RDDs into another RDD [16].

Spark is originally a batch data processing system [28] similar to Hadoop. However, support for a streaming library makes Spark a micro-batch processing system hence it becomes a near real-time framework that may process streaming data sources. With the micro-batch architecture, streaming data are divided into small batches, i.e. discretized streams (DStreams) [29] which are a sequence of RDDs in Spark memory, and processed by Spark. In other words, new batches are created from input DStreams depending on the batch interval length and those discrete streams are stored in memory as RDD sequences. RDDs are then executed by generating Spark jobs [17]. Figure 1 shows the architectural overview of the Spark Streaming scheme. In this flow, data can be consumed from several streaming data sources such as Kafka, Flume, ElasticSearch, MySQL, TCP sockets, and processed utilizing complex methods and algorithms.

Spark's machine learning library, MLlib, has a rich set of learning algorithms including classification, regression, clustering, collaborative filtering, and feature engineering methods. With the use of Spark streaming, machine learning algorithms can be used in two cases:

i. Machine learning models from MLlib are generated offline and then the trained model is used for analysis of streaming data.
ii. Training MLlib algorithms at run-time to evaluate streaming data.

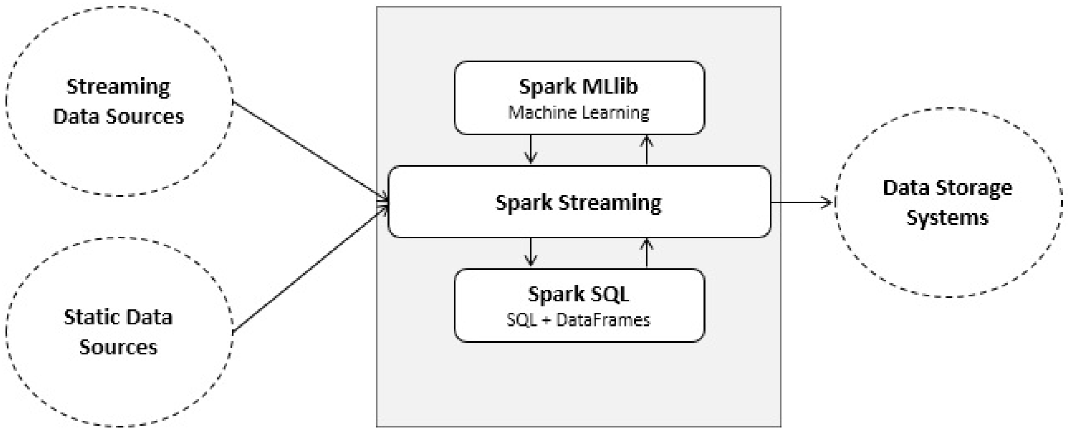


Fig. 1. Overview of the Spark Streaming.

3. USE CASE: GENERIC SENTIMENT ANALYSIS FRAMEWORK

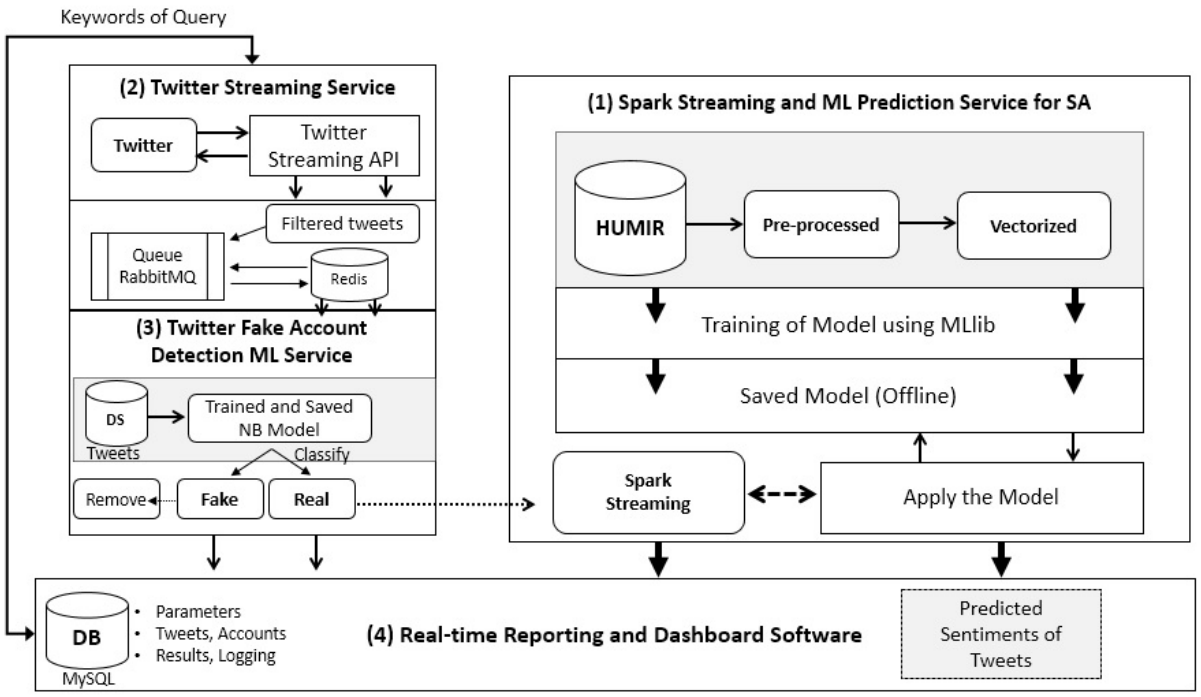
Our proposed sentiment analysis framework has four major software components as shown in Fig 2. The description of each component is as follows.

(1) “Spark Streaming and ML Prediction Service” has two main objectives. The first one is to construct offline NB based learning model using the HUMIR dataset. The second objective is pre-processing collected tweets on the fly and realizing corresponding sentiment prediction of each one in real-time.

(2) “Twitter Streaming Service” uses keyword queries to filter and retrieve tweets and their respective account information. The service then acts as a producer and posts the generated results to the RabbitMQ messaging system. A consumer reads the messages from the queue and stores them in an in-memory database named Redis. Finally a service reads each data from Redis and sends it to a fake account detection service.

(3) “Twitter Fake Account Detection Service” controls the owner (account) of the retrieved tweet and predicts it as real or fake using MLlib NB model trained on the TwFakeUsr dataset. All the users and their corresponding tweets are sent to Reporting software for logging and visualization. The tweets of genuine users are sent to Spark Streaming framework.

(4) “Real-time Reporting and Dashboard Software” is used to query, report and visualize the sentiment classification results of the prediction service. It has also a temporary database to store some metadata information such as IP addresses of the services and the keywords of the users.



Keywords of Query
(2) Twitter Streaming Service
Twitter
Twitter Streaming API
Filtered tweets
Queue RabbitMQ
Redis
(3) Twitter Fake Account Detection ML Service
DS
Tweets
Trained and Saved NB Model
Classify
Remove
Fake
Real
(1) Spark Streaming and ML Prediction Service for SA
HUMIR
Pre-processed
Vectorized
Training of Model using MLlib
Saved Model (Offline)
Spark Streaming
Apply the Model
DB
MySQL
Parameters
Tweets, Accounts
Results, Logging
(4) Real-time Reporting and Dashboard Software
Predicted Sentiments of Tweets

Fig. 2. Outline of the proposed system.

3.1. Twitter real-time data access and filtering

As shown in Table 1, the Twitter Application Programming Interface (API) supports many functions that can be performed. Real time twitter messages are filtered and retrieved using the Twitter Streaming API which requires both a persistent HTTP connection and a user authorization supported by OAuth protocol. The protocol helps users to access the Twitter API without sharing their credentials. Filtering of tweets can be utilized according to the different categories such as terms, hashtags or locations.

Table 1. Samples of Twitter API functions.

API	Supported Functions
Accounts and users	– Mute, block and report users – Follow, search, and get users – Create and manage lists – User Profile Images and Banners – The number of tweets user liked, the number of followers the user has, the number of users the account is following
Tweets	– Post, retrieve and engage with tweets – Get tweet timelines – Search tweets – Filter real-time tweets
Entities	– Metadata and additional contextual information about content – Hashtags, media, urls in the tweet
Places	– Named locations with corresponding geo coordinates

There are various clients that support the Streaming API, tweepy [18] which supports Python and handles errors properly is used in the study. A sample stream listener code segment to filter tweets that mentions “Ayla film” (“Ayla movie”) shown in Figure 3.

```
auth = tweepy.OAuthHandler(TWITTER_APP_KEY, TWITTER_APP_SECRET)
auth.set_access_token(TWITTER_KEY, TWITTER_SECRET)
streamListener = StreamListener()
stream = tweepy.Stream(auth = api.auth, listener = streamListener)
stream.filter(track=["ayla film", "ayla_film"])
```

Fig. 3. Tweepy sample code.

3.2 Training and test dataset for sentiment analysis

The number of datasets created for SA is very limited in Turkish. In this paper we have used HUMIR [6]. This is one of the most important datasets having sentiment features in Turkish. HUMIR is used to train and test the NB machine learning algorithm for SA and contains 65,000 instances which were extracted from two well-known Turkish movie and hotel recommendation web sites, “beyazperde.com” and “otelpuan.com”. The dataset consists of 53,400 movie reviews and 11,600 hotel reviews with an average of 74 and 33 words per review respectively. The number of reviews having negative and positive polarity are balanced in the dataset.

3.3 Training and test dataset for fake account detection

In our proposed service, we make use of dataset named TwFakeUsr to classify the users as real or fake. The dataset contains 1,000 samples which belong to 501 fake and 499 real user accounts. The features of the dataset is presented in Table 2. The first 13 features are obtained from User Object of Twitter API and the remaining 3 features are created manually using Twitter API. For instance, ‘*mentions_average’* feature is calculated with the use of the last 20 tweets and their corresponding mention counts.

Table 2. Features of TwFakeUsr dataset.

Id	Name of the feature	Description of the feature
1	Description	Indicates the length of the description of account
2	Protected	Indicates the user's account protection choice
3	followers_count	The number of followers account has
4	friends_count	The number of users account is following
5	statuses_count	The number of tweets (including retweets) posted
6	favourites_count	The number of tweets user liked
7	listed_count	The number of public lists that user is a member of
8	Verified	Indicates that the user has a verified account or not
9	profile_use_bck_image	True indicates the user wants their uploaded background image to be used
10	contributors_enabled	Indicates that the user has an account with "contributor mode" enabled
11	default_profile	Indicates that the user has not altered the theme, background of user profile
12	default_profile_image	Indicates whether the user has uploaded own profile image or not
13	is_translator	Indicates that the user is a participant in Twitter's translator community
14	hashtags_average	Number of hashtags that user has used in last 20 tweets
15	mentions_average	Number of mentions that user has used in last 20 tweets
16	urls_average	Number of URL links that user has used in last 20 tweets

3.4 Pre-processing of filtered tweets

The first step of SA is named pre-processing and utilized before the feature representation of the dataset. Pre-processing involves a series of methods as shown in Fig. 4.

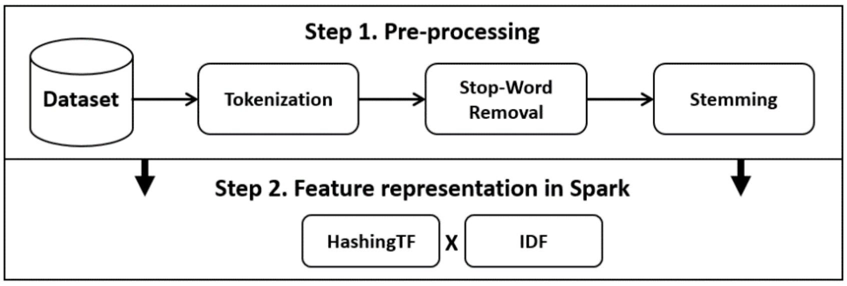


Fig. 4. Pre-processing and feature representation.

3.4.1 Tokenization

Tokenization is the first pre-processing phase that splits the text of documents into minimal meaningful units called tokens. Tokenization also performs some sort of removal task while splitting the tokens, because punctuations and symbols are used for selecting token boundaries in addition to white spaces. Further, minimum and maximum lengths of tokens are set during tokenization.

3.4.2 Stop-Word Removal

Stop word removal is one of the most commonly used phase of pre-processing in order to reduce the dimensionality of the dataset. Stop words are highly common words such as "şey (thing)", "sen, siz (you)", "tarafından (by)" that are not valuable to select documents matching a need and must be removed from documents.

3.4.3 Stemming

In SA and machine learning, stemming is the procedure which reduces derived words to their base forms using a language dependent stemming algorithm [19]. Both derivational affixes and inflectional suffixes of terms are stripped and terms are converted into root form. For example, a stemming algorithm reduces the words "sevdim (loved)", "seviyorum (loving)", and "severek (loving)" to the "sev (love)". In this study, Zemberek [20], a well-known NLP toolkit is utilized to stem tokens in Turkish.

3.5 Feature representation of text in Apache Spark

After pre-processing step, the text of a tweet is required to be converted to an appropriate form of feature representation so that the machine learning algorithm can perform classification. Bag of words (BoW) is the most popular feature representation method for text classification. Each text document is represented as a vector, and each column corresponds to a feature [21]. The most critical step of BoW is feature weighting which is calculated considering three parts; i) Term frequency factor (TF), ii) the inverse document frequency factor (IDF) and iii) document length normalization. The weighting of term *k* in document *i* is calculated as shown in Eq. 1.

$$w_{ki} = \frac{tf_{ik}\log\left(\frac{N}{n_k}\right)}{\sqrt{\sum_{\mathrm{k=1}}^{\mathrm{n}}(tf_{ik})^2\left[\log\left(\frac{N}{n_k}\right)\right]^2}} \qquad (1)$$

Where t_k is the k^{th} term in document d_i. tf_{ik} is the frequency of word t_k in document d_i. $log(N/n_k)$ is inverse document frequency of word t_k in dataset. n_k is the number of documents containing the word t_k. N is the total number of document in dataset.

The most important problem of text classification is its high dimensionality which is also called "curse of dimensionality" in the literature [22]. Apache Spark uses a hash function to map feature values to indices in the feature vector called "Hashing Trick". The method is very fast, preserves sparsity and is well suited for online learning scenarios [23]. Accordingly, in the study, Spark's feature hashing technique named HashingTF is utilized to vectorize features of documents.

3.6 Sentiment prediction and fake account detection and with Naïve Bayes learning model

Naïve Bayes is a well-known algorithm which is used in sentiment analysis effectively. The supervised-learning model is obviously based on Bayes theorem [24]. However, it assumes naively that there is no dependence between every pair of features. In Bayes theorem, the relationship between a class variable **y** and the dependent feature vectors x_n is defined as in Eq. 2.

$$P(y \mid x_1,...,x_n) = \frac{P(y)P(x_1,...,x_n \mid y)}{P(x_1,...,x_n)} \qquad (2)$$

Spark MLlib NB may be trained within a single pass through dataset and therefore it does not need to cache training data. Spark MLlib NB model implements a multinomial Bayesian approach which is particularly efficient for text analysis tasks [25]. NB takes RDD of class labels and related features and produces a model for the prediction/evaluation purposes.

Since the proposed framework has two datasets for two particular tasks (fake account detection and sentiment classification), two different NB models were trained and tested for each task. In order to train/test the NB algorithm, we split datasets into train and test with the ratios of 80:20 respectively. We evaluated the performance of the models with the use of classification accuracy (ACC). In a supervised classification, a ML algorithm is first trained with a one portion of dataset and a predictive model is generated. The trained model is then tested with remaining samples to evaluate its prediction ACC. Prediction ACC of a model is then calculated with the following relation:

$$ACC = \frac{TP + TN}{TP + FP + TN + FN} \qquad (3)$$

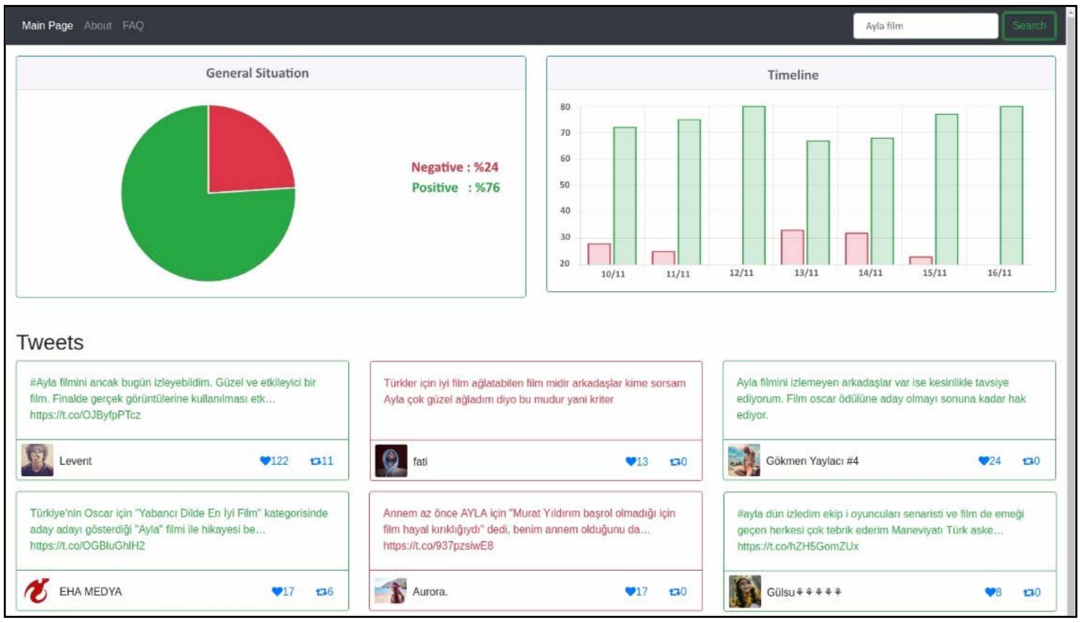
In the above relation TP, TN, FP, FN are True Positive, True Negative, False Positive and False Negative respectively [26]. Though the two datasets used to train the NB algorithm are balanced, we additionally calculate the F-measure score which is a sensitive performance measure that takes both precision and recall parameters into account. The corresponding relation is given in Equation 4.

$$F - measure = \frac{2 \times precision \times recall}{precison + recall} \qquad (4)$$

3.7 Implementation details

The application of the proposed system was developed based on Apache Spark 2.2.1. The Spark Python API (PySpark version 2.1.2) which supports the Spark programming model for Python was utilized. Django web framework was used for rapid development of secure and maintainable web applications. The whole system was hosted on an Ubuntu running Apache web server using Web Server Gateway Interface (WSGI). Four software components of the system were run on AWS (Amazon Web Services) m4.large servers (nodes) having two Intel Xeon E5-2686V4 2.3GHz processors (VCPUs) with 4 cores and 8 GB of RAM on different Spark cluster configurations. Depending on the need for scalability, all software components can also be configured to run as services on the separate servers.

The configuration parameters of NB was set as default. Tweepy v3.1.0 Twitter Streaming API was used for real-time data access and for filtering as explained in Section 3.1. Metadata information, parameters, keywords, account information and sentiments of retrieved tweets were stored in MySQL database. RESTful Node API functions were implemented to connect Twitter Streaming Service. Most frequently accessed Twitter real-time data was stored in an in-memory to the database based on Redis 3.2 with the Python helper library redis-py. A messaging system was constructed using RabbitMQ 3.2.4 and used with the Pika Python client. The Angular java script library was utilized for the frontend of the system to access the API. Figure 5 shows a sample screen of the report and dashboard component.



#Ayla filmini ancak bugün izleyebildim. Güzel ve etkileyici bir film. Finalde gerçek görüntülerine kullanılması etk... https://t.co/OJByfpPTcz
Levent ♥122 11

Türkler için iyi film ağlatabilen film midir arkadaşlar kime sorsam Ayla çok güzel ağladım diyo bu mudur yani kriter
fati ♥13 0

Ayla filmini izlemeyen arkadaşlar var ise kesinlikle tavsiye ediyorum. Film oscar ödülüne aday olmayı sonuna kadar hak ediyor.
Gökmen Yaylacı #4 ♥24 0

Türkiye'nin Oscar için "Yabancı Dilde En İyi Film" kategorisinde aday adayı gösterdiği "Ayla" filmi ile hikayesi be... https://t.co/OGBluGhlH2
EHA MEDYA ♥17 6

Annem az önce AYLA için "Murat Yıldırım başrol olmadığı için film hayal kırıklığıydı" dedi, benim annem olduğunu da... https://t.co/937pzsiwE8
Aurora. ♥17 0

#ayla dün izledim ekip i oyuncuları senaristi ve film de emeği geçen herkesi çok tebrik ederim Maneviyatı Türk aske... https://t.co/hZH5GomZUx
Gülsu ♥8 0

Fig. 5. Sample dashboard screen of the reporting software component.

4. EXPERIMENTAL RESULTS AND DISCUSSION

4.1 Pre-processing

Pre-processing is an important task for sentiment prediction systems. Stop-word removal and stemming are the critical pre-processing steps to reduce the size of corpus before the learning model is trained and tested. Table 3 shows the results of the pre-processing steps in terms of number of instances and number of features. It should also be noted that the minimum length of the words are set to three (3) which means features having less than three letters are removed from the corpus to reduce noisy words that decreases prediction performance of the system.

In Turkish, minimum feature length that is meaningful from sentiment analysis of point of view is 3. One of the aims in pre-processing is to decrease feature size with removing noisy features to increase prediction ability of the system and it is clearly seen from Table 3 that the feature dimension is nearly decreased to 35% of the whole feature set.

Table 3. Number of instances and features before and after pre-processing.

	Original		After pre-processing	
Dataset	Number of instances	Number of features	Number of instances	Number of features
movies	53,400	28,312	52,689	18,312
hotels	11,600	7,936	10,533	5,372
Sum	65,000	36,248	63,222	23,684

The accuracy of the proposed SA prediction system is evaluated in two modes: i) Offline sentiment classification (extraction) performance of trained NB model, and ii) Real-time performance of the model by evaluating the sentiments of tweets with a group of experts.

4.2 Offline sentiment classification

Two different versions of HUMIR dataset were created to evaluate the proposed sentiment prediction framework. The first one is the original dataset (HUMIR-Org) with no pre-processing and the second one is the pre-processed dataset (HUMIR-Proc) including punctuation cleaning, tokenization, stop-word removing and stemming steps as described in previous sections. The NB learning model, was first trained with 80 % of data and then tested with the remaining 20 % of data. In the first run the accuracy and F-measure values were obtained 85.21 % and 85.12 % for the original dataset respectively. In the second run on HUMIR-Proc, the accuracy and F-measure values were improved about 2 % as shown in Table 4. Since the dataset is balanced, having equal number of negative-positive sentiments, the corresponding ACC and FM values are obtained close to each other.

Table 4. The evaluation results with and without pre-processing.

	Datasets	
	HUMIR-Org	HUMIR-Proc
Accuracy	84.83 %	86.77 %
F-measure	84.69 %	86.74 %

A Spark cluster generally is composed of a cluster manager (master) and worker nodes. On the other side NB model can be easily parallelized as it requires only one pass over dataset [27]. In order to evaluate the cluster effect of Spark, experiments were performed using four different clustering configurations; i) one master node, ii) one master node + one worker node servers, iii) one master node + two worker node servers, and iv) one master node + four worker node servers. Master and worker nodes are AWS m4.large servers as mentioned in section 3.7. Figure 6 shows the training time of each configuration in seconds when different number of nodes are used. 156, 132, 96, and 87 seconds as training time were obtained using the first, second, third, and fourth configurations respectively.

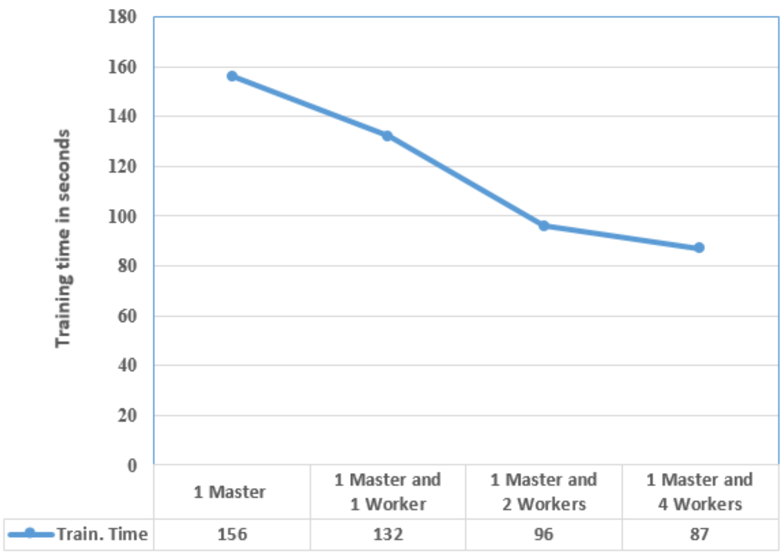


Fig. 6. Training time with respect to the number of nodes.

To compare the training time and accuracy of NB with commonly utilized algorithms in text categorization domain, we have also tested Logistic Regression (LR) and Support Vector Machine (SVM) using the same Spark cluster configurations mentioned. SVM has been trained in 354 seconds having an accuracy value of 87.02% using one master and 4 worker Spark nodes. On the other side LR algorithm has been trained in 120 seconds and its value of accuracy has been obtained as 80% using the same configuration. It can be seen from the experiments that SVM has a slight performance increase of ~ 0.4% in terms of accuracy in comparison to NB. Though SVM has a slightly better accuracy, it has a remarkable slowness in training time (267 seconds) compared to NB training time (87 seconds). Therefore NB is selected as evaluator algorithm for both sentiment analysis and fake account detection systems.

4.3 Real-time performance of the sentiment prediction model

Having the offline model trained and tested, it was converted to a service for the prediction of streaming tweets that were query results of the keyword(s) sent via the developed software component. The system made predictions of streaming tweets on the fly and the prediction performance of the system was about 80.93% out of 2,000 tweets. The overall prediction accuracy of the real-time system is then calculated as follows:

- First, the query keyword is sent to Twitter Streaming Service using the interface of the software.
- Then respondent tweets and their owner accounts are sent to RabbitMQ messaging system. A service consumes these messages from queue and stores them to Redis database. As a last step, another service includes account related data from Redis and sends it to fake account detection service.
- Fake account detection service controls the related account information of the tweet and makes a prediction as real or fake. If the users are real, the corresponding data is accepted to be valid and sent to Spark streaming and sentiment prediction service.
- Received tweets are pre-processed to extract the features and pipelined to ML Prediction Service.
- Having extracted features of each tweet, the system predicts corresponding sentiment analysis of each generated tweet based on the query keyword.
- While the predictions of the system are reported on the dashboard as the tweets are analyzed, the results are also stored in a database with corresponding predictions.

To validate the performance of the system on streaming data, the predictions of the system is reassessed by a proficient research group that consists of 5 academic staff. A software tool is developed and randomly selected tweets are shown to each staff. If the sentiment of a tweet is agreed by at least 3 people, it is marked as a true positive sentiment and is not shown to the staff again. The manual evaluation of the predictions have shown that the system is acceptably able to identify the class of sentiments with the accuracy of % 80.93.

4.4 Impact of real-time fake account detection system

To develop a fake account detection system, a new offline NB model from Sparks MLlib was trained and tested using TwFakeUsr dataset. The accuracy and F-measure values were obtained 89.16% and 88.65 % respectively that were similar to results of [27]. The real-time performance of the system was also tested during the selection of 2,000 tweets. System classified 38 accounts as fake and filtered out the corresponding bogus tweets. Fake accounts were manually controlled to validate the accuracy of the system, and 6 of them were found to be from real users. The true prediction rate of the system was evaluated to be as 84% (32/38) which may be acceptable. The common features and corresponding values of the fake accounts are provided in Table 5.

Table 5. The common features of the fake accounts.

Feature	The value of the feature
description	The length of the description is generally zero or low for fake the accounts.
followers_count	The follower count of a fake account is generally low.
friends_count	The friend count of a fake account is generally high.
default_profile	The default profile is generally true for fake accounts.
statuses_count	The number of tweets posted including the retweets is generally high for the fake accounts.

4.5 Discussion

Sentiment analysis of tweets has two difficulties including topic coverage and dealing with irrelevant or noisy tweets. The tweets to be analyzed are collected as a result of a special keyword search. If the corresponding search is too specific, the number of the related tweets have limited coverage of the searched domain. If the number of the keywords is increased, the system ingests noisy or irrelevant tweets. These two factors are major difficulties of tweet analysis. Another important challenge is to classify the owners of the retrieved tweets as fake or real. For this reason, real-time sentiment analysis systems should have a fake account detection capabilities. Pre-processing is also an important and difficult task for sentiment prediction systems. Considering these challenges, it can be concluded that the overall performance of the proposed real-time sentiment analysis framework having an accuracy of 80.93% is promising.

5. RELATED WORK

There are a lot of works on sentiment data analysis that make use of numerous methods from data mining or machine learning fields. Since the scope of this study is real-time sentiment analysis including a fake account detection module, our literature survey covers both streaming opinion mining approaches particularly in Turkish and Twitter fake account detection methods.

Recently, streaming Big Data analysis topic attracts attention of researchers and the number of the studies in this field increases. In [6] the authors compare real-time data processing systems and related machine learning algorithm libraries. In another study [8], the researchers train a NB algorithm on top of Spark framework to classify real-time tweets as positive, negative or neutral. Cheng et. al in their work [9] develop a framework to analyze consumer opinions about products using Apache Hadoop and the Hadoop stream processor, Storm. In [10], tweets are analyzed to discover their sentiment with the use of the Hadoop-Storm pair. The study compares tweet sentiment classification performance with the use of scalable classifiers and uniprocessor algorithms. A recent work [11], uses the Spark framework with PySpark, a Python API to Spark, to observe the performance of distributed NB and SVM classifiers to detect tweet sentiment in real-time.

One of the motivations of this study is the lack of streaming sentiment analysis on Turkish tweets. There are only a few opinion mining studies that make use of Big Data analytics solutions in Turkish. In [12] the authors develop a sentiment analysis dataset for Turkish with the use of similarity algorithms on top of Hadoop. They use the Map Reduce distributed programming framework to process huge amounts of microblog data to obtain a Turkish sentiment corpora. A scalable but non-streaming Turkish text analysis is conducted in [13] making use of Hadoop and Spark. As far as we know from the literature search, this is one of the first studies to analyze streaming Turkish data. Hence, this literature survey shows that real-time sentiment studies, particularly in Turkish, is an emergent topic.

Since none of the mentioned studies take into account the trustworthiness of the tweet's owner, tweet messages of fake accounts were included in the SA process that probably led to decrease the accuracy of the systems. Different researches have been proposed in literature to detect fake accounts. In general, fake account detection methods are categorized as user-based, content-based and hybrid methods. User-based methods consider the demographic and statistical features such as number of followers, number of friends, age of account, frequency of tweet [30, 31, 32]. In this category, there are also many studies trying to solve specific problems like "Twitter Spam Drift" [37, 38, 39] for data-driven cybersecurity prediction tasks [40]. On the other hand content-based approaches make use of the properties of a posted tweet that includes the count of mentions, count of hashtags, links, trending topics, and duplicate tweets [33, 34, 35]. Hybrid methods uses the features of both user-based and content-based methods [27, 36].

6. CONCLUSION AND FUTURE WORK

Sentiment analysis is mainly a classification problem that can be solved with the use of machine learning algorithms. There are numerous methods that are used to extract emotions from microblog texts. In this work we have presented a real-time Spark-based sentiment analysis framework having four software components: (i) Spark machine learning and streaming service (ii) A Twitter streaming service (iii) A Twitter fake account detection service (iv) A real-time reporting and dashboard software.

Since the posts of fake accounts may manipulate the analyzed domain, a fake account detection service that can detect consistency of the tweet owner is one of the critical points in SA. Therefore, we have designed a fake account detection service to the proposed framework and to the best of our knowledge, this is the first real-time sentiment analysis framework includes a fake account detection service based on Apache Spark.

For machine learning tasks (sentiment analysis and fake account detection) we have trained two Naïve Bayes classifiers from Spark ML library and transformed these models to machine learning services to predict the sentiment of tweets and reliability of the tweet owner. We have evaluated the performance of the designed ML services and obtained promising results in terms of accuracy. The machine learning algorithms in Spark are scalable methods and therefore they can handle enormous datasets. The real-time nature of the proposed system enables execution of queries for a special movie or hotel and obtain the sentiments of streaming tweets concurrently. The analyzed tweets and results of the predictions are also reported on a dashboard that makes real-time visualization possible.

This work may be adapted to any field with slight modifications. With more clear terms, the sentiment prediction service may be trained for a new application domain such as recommendation/brand tracking and it may be used to extract sentiment for those fields in real-time. There are some other widely used microblogging platforms such as Instagram and Tumblr. As a future work, the framework may be developed to make streaming predictions through a unified API that searches Twitter, Tumblr and Instagram concurrently to obtain synthesis of sentiments with an information fusion approach. Since the statistical properties of users and tweets vary over time, the performance of machine learning algorithms can also decrease. To cope with this problem, specific methods in [37, 38] can be adapted to learn the flows of the proposed system as a future work. Furthermore, the accuracy of the sentiment analysis may possibly be increased with the use of a training dataset that is collected and labelled from Twitter.

ACKNOWLEDGEMENTS

Funding for this work was partially supported by Research and Development Center of Commencis Technology accredited on Turkey - Ministry of Science. The modules and services of the work were parts of a Commencis project named AppConnect which was funded by governments including the Scientific and Technological Research Council of Turkey (TUBITAK) having Award No. 3170030.

REFERENCES

1. Aldoğan D, Yaslan Y. A comparison study on active learning integrated ensemble approaches in sentiment analysis. *Computers & Electrical Engineering*. 2017;57:311-323.
2. Liu B. *Sentiment Analysis and Opinion Mining*. Morgan & Claypool Publishers; 2012.
3. STATISTIC BRAIN RESEARCH INSTITUTE. Statistic Brain. https://www.statisticbrain.com/. [Accessed September 4, 2018].
4. Nair LR, Shetty SD, Shetty SD. Applying spark based machine learning model on streaming big data for health status prediction. *Computers & Electrical Engineering*. 2018;65:393-399.
5. Karanasou M, Ampla A, Doulkeridis C, Halkidi M. Scalable and real-time sentiment analysis of twitter data. In: *2016 IEEE 16th International Conference on Data Mining Workshops (ICDMW)*, 2016:944-951.
6. Singh D, Reddy CK. A survey on platforms for big data analytics. *Journal of Big Data*. 2014;2(1):8-32.
7. Ucan A, Naderalvojoud B, Sezer EA, Sever H. SentiWordNet for new language: automatic

translation approach. In: *2016 12th International Conference on Signal-Image Technology Internet-Based Systems (SITIS)*. 2016:308-315.

8. Baltas A, Kanavos A, K. Tsakalidis A. An apache spark ımplementation for sentiment analysis on Twitter data. In: *International Workshop of Algorithmic Aspects of Cloud Computing*, 2017:15-25.
9. Cheng OK, Lau R. Big data stream analytics for near real-time sentiment analysis. *Journal Computer and Communnication*, (2015); 3(05): 189.
10. Rahnama AHA. Distributed real-time sentiment analysis for big data social streams. In: *2014 International Conference on Control, Decision and Information Technologies (CoDIT)*. ; 2014:789-794.
11. Nirmal VJ, Amalarethinam DIG. Real-Time Sentiment prediction on streaming social network data using ın-memory processing. In: *2017 World Congress on Computing and Communication Technologies (WCCCT)*, 2017:69-72.
12. Makinist S, Hallaç İR, Karakuş BA, Aydın G. Preparation of improved Turkish dataset for sentiment analysis in social media. *ITM Web Conf.*, 2017; 13:01030.
13. Çakir MU, Güldamlasioğlu S. Text mining analysis in Turkish language using big data tools. In: *2016 IEEE 40th Annual Computer Software and Applications Conference (COMPSAC)*, 2016:614-618.
14. Ramsingh J, Bhuvaneswari V. Data analytic on diabetic awareness with Hadoop streaming using map reduce in python. In: *2016 IEEE International Conference on Advances in Computer Applications (ICACA)*, 2016:346-350.
15. Shi J, Qiu Y, Minhas UF, et al. Clash of the Titans: MapReduce vs. Spark for large scale data analytics. *Proc VLDB Endow*. 2015; 8(13):2110–2121.
16. Zaharia M, Das T, Li H, Shenker S, Stoica I. Discretized streams: an efficient and fault-tolerant model for stream processing on large clusters. In: *Proceedings of the 4th USENIX Conference on Hot Topics in Cloud Computing*, 2018.
17. Salloum S, Dautov R, Chen X, Peng PX, Huang JZ. Big data analytics on Apache Spark. *International Journal Data Science and Analytics*, 2016; 13: 1-20.
18. Tweepy. http://www.tweepy.org/. [Accessed September 7, 2018].
19. Kılınç D, Özçift A, Bozyigit F, Yıldırım P, Yücalar F, Borandag E. TTC-3600: A new benchmark dataset for Turkish text categorization. *Journal of Information Science*. 2017;43(2):174-185.
20. Akin DM. Zemberek, an open source NLP framework for Turkic Languages. 2018;10.
21. Salton G, Buckley C. Term-weighting approaches in automatic text retrieval. *Information Processing & Management*. 1988;24(5):513-523.
22. Bach F. Breaking the curse of dimensionality with convex neural networks. *Journal of Machine Learning Research*, 2017; 18(19):1-53.
23. Weinberger K, Dasgupta A, Langford J, Smola A, Attenberg J. Feature hashing for large scale multitask learning. In: *Proceedings of the 26th Annual International Conference on Machine Learning*, 2009:1113–1120.
24. Pedregosa F, Varoquaux G, Gramfort A, et al. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 2011;12:2825–2830.

25. Yildirim P, Birant D. Naive Bayes classifier for continuous variables using novel method (NBC4D) and distributions. In: *2014 IEEE International Symposium on Innovations in Intelligent Systems and Applications (INISTA) Proceedings*, 2014:110-115.

26. Kohavi R, Provost F. On Applied Research in machine learning. *Machine Learning*, 1998; 30(3): 127-132.

27. Erşahin B, Aktaş Ö, Kılınç D, Akyol C. Twitter fake account detection. In: *2017 International Conference on Computer Science and Engineering (UBMK)*, 2017:388-392.

28. Zaharia M, Chowdhury M, Das T, Dave A, Ma J, Mccauley M, Franklin M, Shenker S, Stoica I. Fast and interactive analytics over hadoop data with spark. In: *Proceedings of the 4th USENIX Conference on Hot Topics in Cloud Computing*, 2012;45–51.

29. Zaharia M, Chowdhury M, Franklin MJ, Shenker S, Stoica I. Spark: Cluster computing with working sets. In: *Proceedings of the 2nd USENIX Conference on Hot Topics in Cloud Computing*, 2010:10–10.

30. Flores M, Kuzmanovic A. Searching for Spam: Detecting fraudulent accounts via web search. *Passive and Active Measurement*, 2013:208-217.

31. Yang Z, Wilson C, Wang X, Gao T, Zhao BY, Dai Y. Uncovering Social Network Sybils in the Wild. *ACM Trans Knowl Discov Data*, 2014;8(1):2:1–2:29.

32. Malhotra A, Totti L, Meira Jr. W, Kumaraguru P, Almeida V. Studying User Footprints in Different Online Social Networks. In: *Proceedings of the 2012 International Conference on Advances in Social Networks Analysis and Mining*, 2012:1065–1070.

33. Kontsevoi V, Lujan N, Orozco A. DETECTING SUBVERSION ON TWITTER. :15.

34. Stringhini G, Kruegel C, Vigna G. Detecting spammers on social networks. In: *26th Annual Computer Security Applications Conference*, 2010;1-9.

35. Egele M, Stringhini G, Kruegel C, Vigna G. Detecting Compromised Accounts on Social Networks. In: *NDDS*, 2013.

36. Conti M, Poovendran R, Secchiero M. FakeBook: Detecting Fake Profiles in On-Line Social Networks. In: *2012 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining*, 2012:1071-1078.

37. Chen C, Zhang J, Xiang Y, Zhou W. Asymmetric self-learning for tackling twitter spam drift. In: *2015 IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS)*, 2015: 208-213.

38. Chen C, Wang Y, Zhang J, Xiang Y, Zhou W, Min G. Statistical features-based real-time detection of drifted Twitter spam. *IEEE Transactions on Information Forensics and Security*. 2017;12(4):914-925.

39. Chen C, Zhang J, Xiang Y, Zhou W, Oliver J. Spammers are becoming" Smarter" on Twitter. *IT professional*. 2016;18(2):66-70.

40. Sun N, Zhang J, Rimba P, Gao S, Xiang Y, Zhang LY. Data-driven cybersecurity incident prediction: A survey. IEEE Communications Surveys & Tutorials, DOI: 10.1109/ COMST.2018.2885561.