Journal of Information & Knowledge Management
Vol. 18, No. 2 (2019) 1950013 (26 pages)

DOI: 10.1142/S0219649219500138

World Scientific
www.worldscientific.com

TwitSenti: A Real-Time Twitter Sentiment Analysis and Visualization Framework

Jamuna S. Murthy
PES University, India

Siddesh G. M.*
Ramaiah Institute of Technology, India
siddeshgm@gmail.com

K. G. Srinivasa
National Institute of Technical Teachers Training and Research
Chandigarh, India

Published

Abstract. Twitter is considered as one of the world's largest social networking sites which allow users to customize their public profile, connect with others and interact with connected users. The proposed work introduces a distributed real-time twitter sentiment analysis and visualization framework by implementing novel algorithms for twitter sentiment analysis called Emotion-Polarity-SentiWordNet. The framework is applied to build an interactive web application called "TwitSenti" which can benefit companies and other organizations in knowing the people's sentiment towards the aspects such as brands, current events, etc., which in turn helps in quick decision-making and planning marketing strategies. The algorithm is validated against three existing classifiers and hence proved that Emotion-Polarity-SentiWordNet provides highest accuracy value of 85%. Also, the framework showed best scalability results when evaluated through web app as four node clusters, proves to be fast and can scale well with massive data.

Keywords: Sentiment analysis; Twitter; Apache Storm; D3.js; Kafka; spout; bolt.

1. Introduction

Twitter is considered as one of the world's largest social networking sites which allow users to customize their public profile, connect with others and interact with connected users. A global survey from one of the well-known companies like "Statista" witnesses that as of the fourth quarter of 2017 there are average number of 330 million monthly active twitter users (Statista, 2017). Users post short messages of 140 characters called tweets based on a variety of topics ranging from simple ones such as "Hi #xyz☺am@college" to themes such as "#IPL-2017" using different ways such as blogging on Twitter website, using Twitter mobile application and also through other social network applications which allow virtual connections from one application to another such as Instagram. Being a most popular social networking

*Corresponding author.

site twitter uses a network model called "following". Any person can follow any other person who remains to be his/her friend. The person who follows is named as a follower and a follower on twitter can receive all the updates that he/she follows. Tweets are of different types which include normal tweet, reply and retweet. Normal tweets are the ones which people post based on their thoughts and opinions which is not a reply to any other tweets or a retweet for concerned tweet. Reply is a tweet which a twitter user posts with "@" symbol attached with name of replies' for example "@abc am not interested in you". Retweet is a message which commonly starts with "RT" and it is a post which is shared by the follower to his/her followers. Apart from this the other very important features of tweet are HashTags and URLs which make twitter stream more readable and provides an understanding of current topic or an event discussed on twitter (Lau *et al.*, 2012). Twitter is one of the best platforms for news information propagation and hence people post news information as tweets at real-time (Reis *et al.*, 2017). It may be any kind of news from Prime Ministers decision about any changes in autonomy or company's date of releasing their product, the following feature in twitter makes the news to spread within short interval of time. Therefore, twitter is considered as one of the best platforms for sentiment analysis (SA) or decision making. It is an ongoing field of research in text mining today and addresses a particular kind of classification problem where we classify the free form of natural language texts (containing words, emoticons and special symbols) as positive, negative and neutral feelings with aim of identifying the thoughts and opinions of people about concerned to an event or a topic or pertaining to any aspect of an entity, which can help companies and other organizations to take fast and effective decisions about their brand and marketing strategies. Tweets are usually informal in nature and are no more than 140 characters. Hence, people use irregular expressions, smilies, emoticons, abbreviations, etc to save their room of space for messages. This has increased the problems related to SA of tweets such as data sparsity and sarcasm. This indirectly leads to inaccurate classification (Yadollahi *et al.*, 2017).

1.1. *Motivation*

Over the years, many classifiers were trained by the researchers for sentiment classification of tweets but most of them used the traditional approaches such as Bag of words model, Unigrams-Bigrams model, POS Tagging, Machine Learning techniques, etc. which lead to less classification accuracy. Also twitter is a real-time user application in which data flows in the form of streams (i.e. people continuously post the tweets over real-time); this poses a new challenge to the companies and developers in terms of storage, processing and analysis (i.e. Sentiment Analysis), since the stream data are rigorous and have to be operated immediately. The existing frameworks for twitter sentiment analysis used Hadoop and MySQL for twitter feed classification. But we know the hadoop is used only for batch data processing and hence is least suitable for handling real-time twitter data. Also the traditional database systems such as MySQL support only conventional search and hence are least

suitable for handling unstructured data (i.e. tweets). Thus this research work aims at implementing a "Real-Time Twitter Sentiment Analysis and Visualization Framework" with a novel classifier algorithm called "EPS (Emotion-Polarity-SentiWordNet)".

1.2. *Contributions*

- The framework uses Apache Storm, a real-time stream data processing engine as a core part of the framework, and also Redis, a NoSQL database which handles the data inside the memory and quickens the tasks for analysis.
- The proposed system integrates different components to provide a best solution to the twitter sentiment analysis problems. Framework consists of major modules in terms of data collection, data parsing and data visualization.
- Data collection module consists of data collector and data filter. The second module is data parser and consists of a data pre-processor and a sentiment analyzer. Finally, the data visualization module focuses on real-time visualization of the classified tweets on real-time web application called "TwitSenti".
- The key aspects of the proposed system lie in building highly scalable cluster with highly availability of the framework over real-time.

The highlights of the proposed framework are: the data collection module is designed to poll tweets from twitter streaming API using Java library called twitter4j. The tweets are consumed by Kafka which is a distributed message queuing system integrated with the data parsing module Apache Storm which includes data pre-processing and sentiment classification for instantly processing tweets unlike Hadoop which stops after batch processing is done. Data pre-processor functions as natural language processing pipeline and does detection and analysis of slangs because the tweets contain number of abbreviations. Later, the process of Lemmatization and correction is carried out which involves the reduction of inflectional forms. The key aspect of this pipeline is it includes the removal of skip words which most of the existing systems for twitter data analytics lacked. The most important part of the framework is sentiment classifier and it provides a hybrid classification algorithm EPS (Emoticon-Polarity-SentiWordNet) for twitter feed classification. Finally, the data visualization module shows the classification results over real-time web app "TwitSenti" using JavaScript D3.js (Document Driven Documents) and python micro server Flask. The database integration of Apache Storm to view visualization results is Redis, which is a NoSQL database for semi-structured data analytics such as twitter data. And for best scalability results, the framework is built both locally using vagrant, virtual box and also on cloud platform as four-node clusters using Amazon EC2.

2. Literature Review

Sentiment analysis deals with analyzing the user generated data and hence many researchers have worked on the same and proposed different techniques which are

discussed here. The concept of name entity recognition (NER) was introduced by Allan *et al.* (2011), which is a process of extracting the tweets posted by users on twitter with aim of identifying their opinions and thoughts on any aspect of entity. The framework tried to rebuild Natural Language Processing (NLP) pipeline for twitter data analysis. The work included parts of speech tagging and named entity recognition which is to identify the names in tweets. Relative to the other NLP systems, this system increased performance by F1 score value of 23%. But the system lacked bag-of-words tagging and stop words removal, which is most important for increasing accuracy of data analysis. Ritter *et al.* (2012) introduced an open domain event extraction system which increased F1 score value by 14%. This framework categorizes events and presents them on a calendar. The processing included parts of speech tagging first and determining where the nouns represent events and categorizing them. This system lacked skip words removal which is most important for increasing accuracy. But the proposed system solves this issue by implementing a component called data pre-processor in Data Parsing module, which includes all the natural language processing tasks including skips words removal to enhance the accuracy of tweets.

Ritter *et al.* (2011) developed a system for analysing the twitter data. The system comprised of two modules lexicon builder and sentiment analyser. They considered MapReduce framework for running these two components and proved for scalability. But the Lexicon builder component lacked stop words removal which is most important factor for increasing accuracy and the sentiment analyzer component classified the sentiment value wrongly for positive and negative words which calculated wrong sentiment score. In contrast, the proposed work consists of two components in data parser module, data pre-processor and sentiment classifier. Data pre-processor solves the issue of lexicon builder component by removing stop words and Sentiment analyzer classified the tweets accurately because it included novel classification algorithms for the same. Here hadoop is used for processing the data and MySQL database for the storage. But most of the times hadoop requires larger clusters to process the data over real-time and MySQL database takes lots of time for processing. But the proposed work includes best distributed real-time data processing system, Apache Storm for twitter data analysis over multi-node. It also includes Redis database for visualizing the analytics results which is in memory key value store and therefore processing is superfast. Balamurali *et al.* (2011) proved that unigram and trigram model outperforms the trigram model when used with Naive Bayes classification for tweet sentiment analysis. In contrast, the reverse was true in case of SVM and MaxEnt, which was proved by Chang and Lin (2011). Although the above methods tried to increase the classification accuracy to some extent, the proposed EPS algorithm outperforms all the existing classifiers by increasing the classification accuracy to 85%.

Wang *et al.* (2013) designed a real-time twitter sentiment analysis system called "SentiView", which was based on model-driven development technique with adjustment of time parameter. A new evolution model was put forth based on cellular

automata, which used time comparison feature for analysing the varying sentiments on twitter platform at real time. The system was effective in analysing the sentiment to some extent but it lacked accuracy and speed since the system was able to analyse only 300 sentimental tweets per minute but the proposed system analyses 1080 sentimental tweets per minute and also the visualization was very poor since Visual C++ 2008 Platform was used. But the proposed system uses D3 visualization which is very effective and also user friendly. Anjaria and Guddeti (2014) performed sentiment analysis using supervised learning using influential factor as key parameter for classification of tweets. A hybrid combination of Support Vector Machines (SVM), Naive Bayes, Maximum Entropy and Artificial Neural Networks-based supervised classifiers was designed and was applied on different datasets consisting of social scenarios with US Presidential Elections 2012 and Karnataka Assembly Elections 2013. This lead to an average accuracy of 73% for both datasets, but the proposed EPS Algorithm classifies tweets with accuracy of 85% on six different datasets leading highest accuracy.

Saif *et al.* (2016) designed a new concept called "SentiCircle" which was derived from lexion-based approach. SentiCircle considers the co-occurrence patterns of words in different contexts of tweets and absorbs their semantics, and later, the new strength and polarities of sentiment lexicons are reassigned to previously assigned strength and polarity. Three datasets were considered for experiment and this approach increased the accuracy to 4–5% compared with state-of-the-art technique with two datasets but lagged by 1% in F-measure in third dataset. But the proposed EPS technique shows accuracy of 85% with six different datasets and is highly stable. Zimbra *et al.* (2016) performed brand-related twitter sentiment analysis using feature engineering. Starbucks-brand related dataset was used for sentiment analysis with three and five classification levels. This approach provided accuracy of 80% compared with state-of-the-art technique, but the proposed technique showed 5% high accuracy compared with this technique and can be used for any kind of twitter datasets.

Azzouza *et al.* (2017) designed a real-time twitter sentiment analysis framework using unsupervised learning techniques. Apache Storm tool is used for real-time data processing of tweets and the whole architecture is divided into five different modules, which include Tweets acquisition module used for data collection, Tokenization module for obtaining lexicons, Tweets processing module for pre-processing of tweets with NLP Techniques, Opinion analysis module for sentiment classification of emoticons and words, Visualization module for real-time data graphs and histograms and finally Recommendation module for recommending the most used keywords related to their query. This system showed over all accuracy of 55% which seems to be very low. Also the huge module system with five different modules makes the system very complex compared with the proposed system which is simple and consists of only three modules. Also the Tweet Processing module in this system lacked removal of skip words which is very important to increase accuracy. In addition, the visualization showed basic details only which are present in almost all visualization frameworks today, whereas the proposed data visualization module

shows eight different charts with each describing the sentiment to very accurate and effective way for users over real time.

There are many websites which provide statistic results for Twitter Sentiment analysis such as tweetstats.com, WordCloudBot, TwitterCounter.com but all these applications are simple and provide not much information except tweet count, word cloud, etc. They are not available most of the times. But our TwitSenti web application provides all necessary information needed for sentiment analysis with eight different varieties of graphs. The graphs and charts used in our application are attractive and include location-based search with dynamic update of all the different graphs over real time.

3. Proposed Work

The proposed framework for twitter sentiment analysis consists of three major modules in terms of data collection, data parsing and data visualization, as shown in Fig. 1.

3.1. *Data collection*

With Twitter APIs, developers can only access 1% of public Twitter data. Twitter also formulates rules of rate limits to restrict the handling of APIs. Twitter APIs utilize 15 min windows to judge whether an application exceeds rate limits. Normally, Twitter APIs authorize 180 queries in 15 min time ranges, albeit for some expensive request, the rate limits are controlled within 15 queries per minute windows. Twitter status is the basic entity of Twitter message object, the system

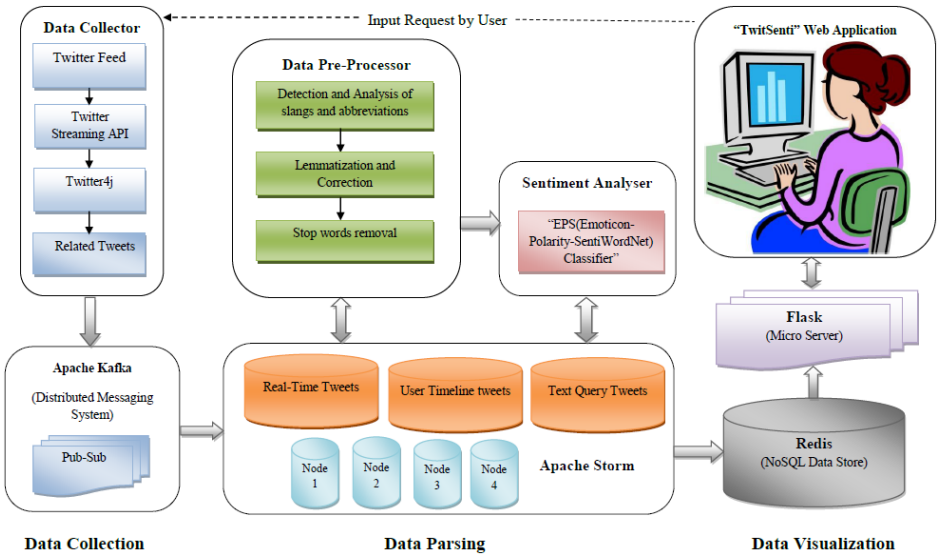

Fig. 1. Proposed framework for twitter sentiment analysis.

extracts five fields from it, namely, tweet content which may contain URL, HashTags, mentioned user, whether it is retweeted, text and emoji, screen name which is the account ID of user and is unique, created time which indicates when this message is made, geographic information including latitude and longitude, country code which is formatted in ISO alpha-2. The reason why we do not use country name is that sometimes country name may change by location, for example, the country name of Japan may become Japanese characters. The system involves Twitter API implementations of Java version which is called Twitter4j. In the Twitter API, there are some useful parameters such as, *count* which indicates the number of tweets, *lang* which filters tweets with given language, *geocode* which filters tweets within given area, *q* which indicates the contents of query string. To increase the varieties of data collection methods, the following are several implementations:

3.1.1. *Real-time data collection component*

Real-time data are most valuable in Twitter. Real-time data collection component collects real-time data by a Twitter stream listener and extracts desired information from Twitter statuses and then merges all information into one message.

3.1.2. *User timeline collection component*

According to the screen name input by users, user timeline collection component acquires all tweets in the timeline of this account and produces messages to Kafka clusters.

3.1.3. *Text query collection component*

By calling search APIs of Tweepy, text query collection component collects related tweets of input text and produces messages to Kafka clusters.

3.1.4. *Favourite list collection component*

Based on input account name, favourite list collection component collects all tweets in users' Favourite list and sends them to Kafka brokers.

3.2. *Data parsing*

This is very important module and includes two components: data pre-processor and sentiment classifier. Apache Storm is used to parse the data collected using Kafka. The main advantage of using storm is it consists of spouts and bolts where spout receives the input from Kafka and bolts are used to process the data and provide necessary output. Spouts and bolts are linked using some grouping mechanism to form a topology. The per-processor and sentiment classifier are treated as bolts and are run using storm in a parallel way. In contrast to hadoop which has only map and reduce phase, storm supports multiple stages with one spout and any number of bolts. The processed data are stored in Redis for analysing the results.

3.2.1. *Proposed pre-processing algorithm for feature extraction*

Data pre-processor processes each tweet individually and sends the refined tweets to sentiment classifier. For example, consider the tweet "Hey@xyz!!gunnyt:-)sleep tight".

Step 1: Check for each word from available dictionaries such as WordNet, JSpell and SpellCheck. The words which are not found in the dictionary are considered as either slangs or abbreviations. From the above tweet, gunnyt is slang and will not return any meaning. The slang from the above tweet gunnyt is replaced in the tweet as "Hey@xyz!!good night:-) sleep tight" using The Sms dictionary and Netlingo (Jansen and James, 2002).
Step 2: After detection and analysis of slang, lemmatization, i.e. stemming of each word, is carried out and later it is corrected using the JSpell or Jazzy Spell Checker or Snow Ball (Porter, 2001). If happier is stemmed to happi, replace it with happy using spell checkers.
Step 3: After lemmatization and correction, remove all the skip words from the tweets such as http, with, yours, etc. from our tweet remove you, hey using Stanford or wiki or Texifier (Zitouni, 2014).
Step 4: Remove all the special characters excluding emoticons and username starting with @user such as @xyz in our example is removed.
Step 5: There are other important features such as HashTags and URLs which are retained with emoticon and words since they are most important features for Sentiment analysis of different trending topics (Khuc *et al.*, 2012).

3.2.2. *Proposed EPS algorithm*

The sentiment analyzer implements three classifiers in the form of Improved Emoticon Classifier (IEC), Enhanced Polarity Classifier (EPC) and SentiWordNet Classifier (SNC) in one single algorithm, and the algorithm description is given below.

The set of tweets T is defined as shown in Eq. (1):

$$T = \{t_1, t_2, \ldots, t_n\}. \tag{1}$$

If each tweet t contains w words, then set of words W is shown in Eq. (2):

$$W = \{w_1, w_2, \ldots, w_m\}. \tag{2}$$

Then the sentiment score S, calculated for each word, is given as shown in Eq. (3):

$$\text{Score} = \sum_{i=1}^{n} \sum_{j=1}^{m} S_{t_i w_j}. \tag{3}$$

Step 1: In this step, emoticons are classified using IEC. The emoticons present in the tweets are first detected by pattern matching with regular expression. Next the manually tagged list of emoticons with positives and negatives are initialized for extraction of emoticons. Positive and negative emoticon scores are obtained by matching the positive and negative emoticons present in tweet against manually

tagged list. Later. aggregate score is obtained by adding two scores. Finally, if the aggregate score is greater than zero the tweet is given a value 1, it is given the value -1 if it is less than zero and if the sum is zero then the value assigned to tweets is also 0. Next the tweets are classified further using EPC and SNC in step 2 and step 4. Let $P_E = \{\text{positive emoticons list}\}$ and $N_E = \{\text{negative emoticons list}\}$ be two lists tagged as input to IPC, then Step 1 is represented as shown in Eq. (4):

$$\text{Score}(e) = \begin{cases} 1, & (w_x \in W) \wedge (t \in T) \wedge (w_x \in P_E), \\ -1, & (w_y \in W) \wedge (t \in T) \wedge (w_y \in N_E), \\ 0, & (w_z \in W) \wedge (t \in T) \wedge (w_x \notin P_E) \wedge (w_z \in N_E), \end{cases} \tag{4}$$

where $\text{Score}(e)$ is emoticon score and w_x, w_y, w_z are the words from W and t is a tweet from T.

Step 2: EPC follows "bag of words tagging" approach in which a rich set of positive and negative words is given as input. The words list used is collected from Bing Liu List of Words (2017) and both the lists are combined to obtain roughly around 9500 words. Generally, words are domain-independent. EPC is an enhancement from Bill McDonald List of Words (2017) and hence the name. It works similar to IEC except that their emoticons are used and here words are used. EPC works only with words with correct spelling and if there is a combination of positive or negative in tweet it is classified as neutral and addressed using SNC. Let $P_W = \{\text{positive words list}\}$ and $N_W = \{\text{negative words list}\}$, then Step 2 is represented as shown in Eq. (5):

$$\text{Score}(w) = \begin{cases} 1, & (w_x \in W) \wedge (t \in T) \wedge (w_x \in P_W), \\ -1, & (w_y \in W) \wedge (t \in T) \wedge (w_y \in N_W), \\ 0, & (w_z \in W) \wedge (t \in T) \wedge (w_z \notin P_W) \wedge (w_z \in N_W), \end{cases} \tag{5}$$

where $\text{Score}(w)$ is the word score of IPC, w_x, w_y, w_z are the words from W and t is a tweet from T.

Step 3: SNC is based on the SentiWordNet dictionary. The tweets are usually classified using POS (Parts of Speech) Tagging here. Different weights are assigned to different words present in tweets and are classified as positive and negative. SNC is similar to EPC, whereas here the words are separated using delimiter and the sentiment values are assigned by referring to SentiWordNet library. The aggregate sentiment score is calculated by adding the weight of each word which was assigned using SentiWordNet. Finally, the tweet is given the value 1 if the aggregate score is greater than zero, it is assigned -1 if less than zero and the score 0 indicated that the calculated sum is zero. Step 3 is represented as shown in Eq. (6):

$$\text{Score}(s) = \begin{cases} 1, & (w_x \in W) \wedge (t \in T) \wedge (\text{weight}(w_x) > 0), \\ -1, & (w_y \in W) \wedge (t \in T) \wedge (\text{weight}(w_y) < 0), \\ 0, & (w_z \in W) \wedge (t \in T) \wedge (\text{weight}(w_z) < 0), \end{cases} \tag{6}$$

where Score(s) is sentiment score of SNC and weight (w_x), weight (w_y), weight (w_z) are the words from W and t is a tweet from T.

Step 4: To the pre-processed tweets, first IEC is applied in step 1, after all the emoticons are classified and scores are recorded in step 2; the neutral tweets from step 1 are classified using EPC and positive and negative scores are recorded; finally, SNC is applied in step 3 to calculate sentiment value. The tweets which are not classified using any classifier are considered to be neutral. The final classification step can be expressed as shown in Eq. (7):

$$\text{Class} = \begin{cases} \text{Positive } (S_e > 0) \vee (S_e = 0 \wedge S_w > 0) \vee (S_e = 0 \wedge S_w = 0 \wedge S_s > 0), \\ \text{Negative } (S_e < 0) \vee (S_e = 0 \wedge S_w < 0) \vee (S_e = 0 \wedge S_w = 0 \wedge S_s < 0), \\ \text{Neutral } (S_e = 0) \wedge (S_w = 0) \wedge (S_s = 0). \end{cases} \tag{7}$$

The results of classification are stored in in-memory NoSQL database Redis, which is visualized using the Data Visualization Module. The proposed algorithms are implemented using Apache Storm programming module, which is given as follows.

3.2.3. *Sentiment topology*

The sentiment topology can process data from Twitter directly and generate personal and country sentiment. The personal sentiment is generated by classifying tweets contents, whereas country sentiment is calculated by counting average values of personal sentiments in the same country. Tweets contain text, emoticons, emoji, URLs, HashTags and punctuations. Among them, only text, emoticons, emojis and exclamation points contribute to sentiment classification. The topology handles extractor of Twitter library to remove irrelevant content and keep related ones. This topology applies Stanford NLP library to classify the content of tweets into different sentiments. The proposed system implements the pre-processing algorithm and the novel EPS algorithm as two different bolts of the topology. The following are details of its bolts, input and output:

Input: The inputs of this topology are basic Twitter contents. The country sentiment function only works for tweets which have country code.

Output: The outputs are messages which end with personal sentiment and country sentiment.

Pre-processing bolt: This bolt adopts the proposed pre-processing algorithm steps to pre-process the twitter messages.

EPS bolt: This bolt implements the proposed EPS algorithm for twitter feed sentiment classification.

Count bolt: The count bolt handles a distributed word count method to calculate the average sentiment of countries.

3.3. *Data visualization*

Data visualization components are implemented by D3.js and python micro server ask Flask which is a web framework and facilitates to implement a web server for proposed framework. D3.js allows users to build visualization by themselves. The data of the web framework are sent from the server continuously. To address real-time data, there are four options: polling, long polling, WebSocket and Server-Sent Event. Polling and long polling are techniques that the client side sends requests to the server side periodically.

They are the easiest to implement, even though they are costly. WebSocket and Server-Sent Event (SSE) are more popular, in which WebSocket allows both clients and servers to send messages to each other, while SSE, as its name shows, is only responsible for sending messages from the server side to the client side. For only the server side to send data to the front end periodically, the project exploits SSE to update stream data, which only need to set Content-Type to text/event-stream in HTTP header. In order to update data on graphs, the website checks whether data updated every second with window.setInterval function. The user interface for "TwitSenti" web application is shown in Fig. 2. It consists of eight different charts which are initiated by the user visiting the website. The web application provides both searches by query and also the time-based analysis results which are displayed as updates on dashboard for the visiting users. The three different sentiments classified using the EPS algorithm are represented as green for positive, red for negative and grey for neutral sentiment. The details of different dashboards used in web application are described subsequently.

3.3.1. *Heat Map*

Figure 3 shows the Heat Map which describes the average positivity and negativity of the tweets by users on Google Charts. The snapshot of the dashboard is taken in context by querying the results for keyword "Narendra Modi" and location "India". The map highlights the location of keyword queried and also the sentiment analysis

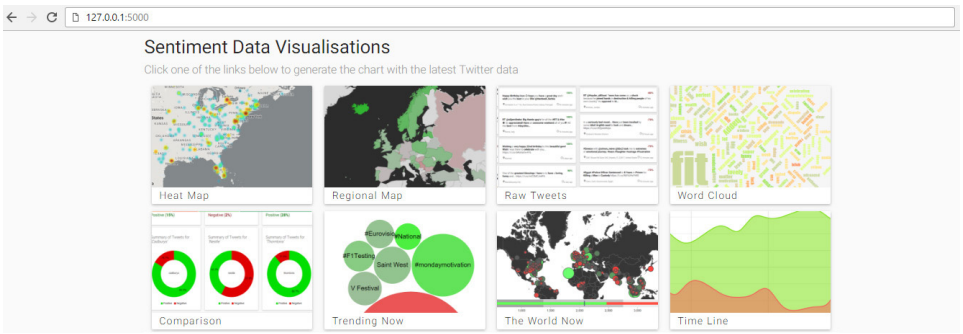

Fig. 2. User interface of "TwitSenti" web application.

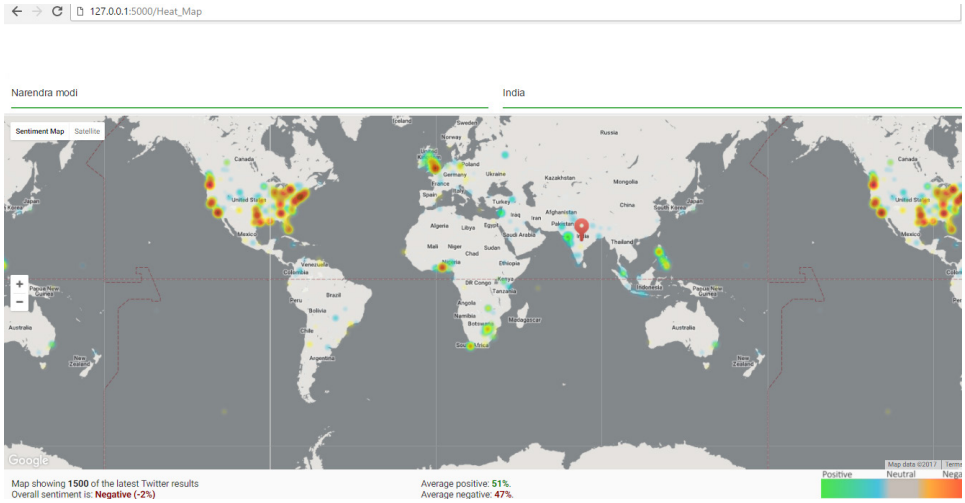

Fig. 3. Heat Map of average positive and negative tweets.

results on Google map. This is useful in a context to know the influence of keyword in twitter network.

3.3.2. *Regional map*

Figure 4 shows the regional map. This is a dashboard which updates the mood (sentiment) of the people based on each country or region of the map. The speciality of this dashboard is it displays the sentiment of countries which talk about that

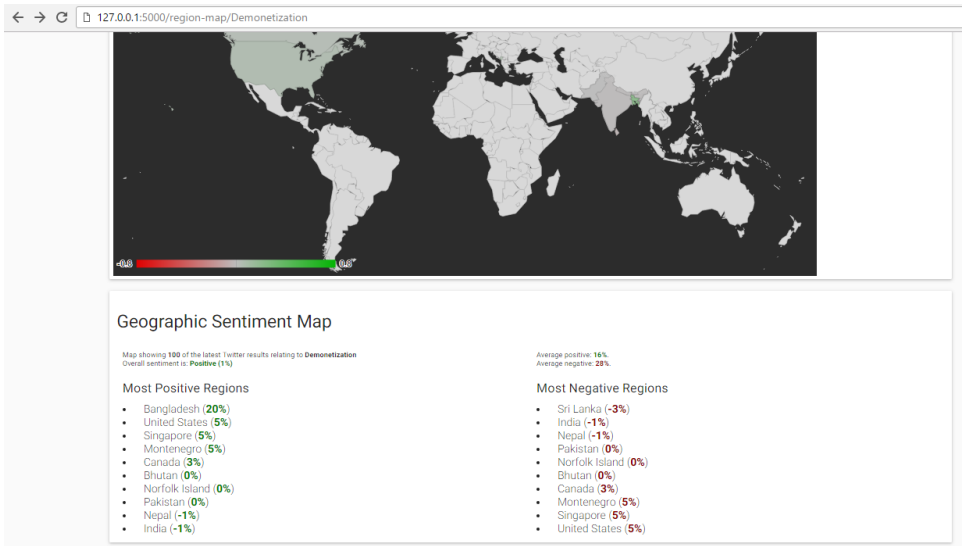

Fig. 4. Regional map updating sentiments of people.

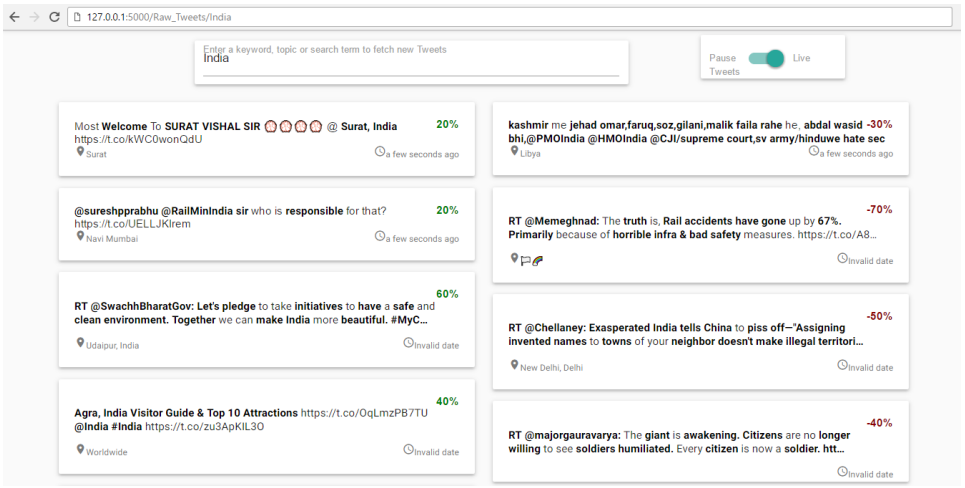

Fig. 5. Raw tweets for keyword "India".

particular keyword. The snapshot for this dashboard is taken by querying the keyword "Demonetization".

3.3.3. *Raw tweets*

Figure 5 shows the dashboard of raw tweets for keyword "India". The raw tweets are the most recent tweets which are displayed on the dashboard with time and location based on keyword search. This is useful in a way to know the most recent information which is flowing in the twitter.

3.3.4. *Word Cloud*

Figure 6 shows the Word Cloud application, a dashboard of informative spells used by the twitter users in tweets. The snapshot of the dashboard is taken for a particular search keyword "India" and hence the relative spells used in the tweets related to love get updates on dashboard over real time. From the dashboard we observe that "Blocked" is one such word which is heavily used by the twitter users when queried for "india" keyword.

3.3.5. *Comparer*

Figure 7 shows Comparer dashboard. This is a very useful dashboard which provides the comparison of four different keyword inputs by the user. The results of the dashboard shown in Fig. 8 are taken by comparing four keywords "Redmi", "Samsung", "Apple" and "Lenovo". From the dashboard results, we observe that Lenovo is heavily used by the users and hence the opposite brand companies can query the tweets related to that brand and find out what they like in Lenovo brand and in turn can find the pitfalls of their brand.

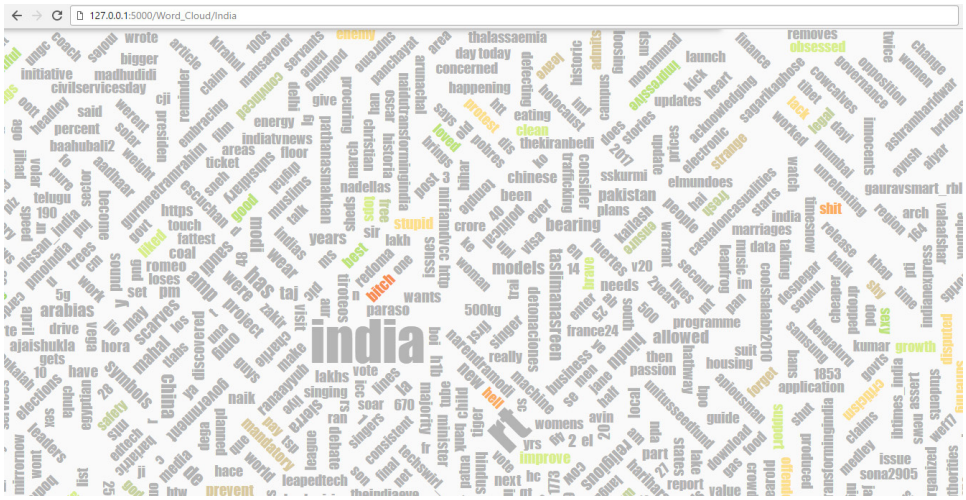

Fig. 6. Word Cloud application.

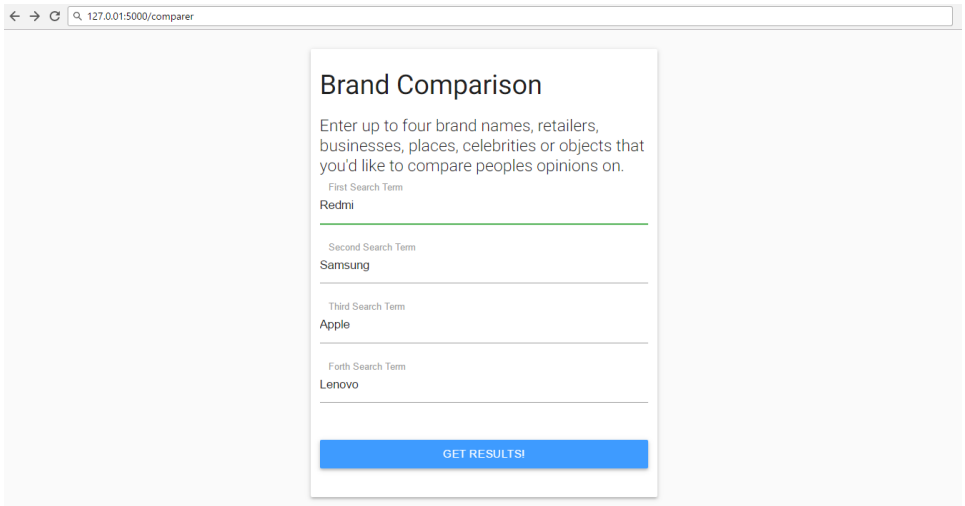

Fig. 7. Comparison of four brands.

3.3.6. *Trending Now*

Figure 9 shows the dashboard of "Trending Now" which display the top 10 trends based on the location. The snapshot results of the dashboard are taken for location India. The dashboard shows the top 10 trends with sentiment attached to them. Compared with other applications which show top trends, our application highlights the weightage of top trends and by clicking on trends the tweets related to them are shown so that people can look at the reviews related to trends.

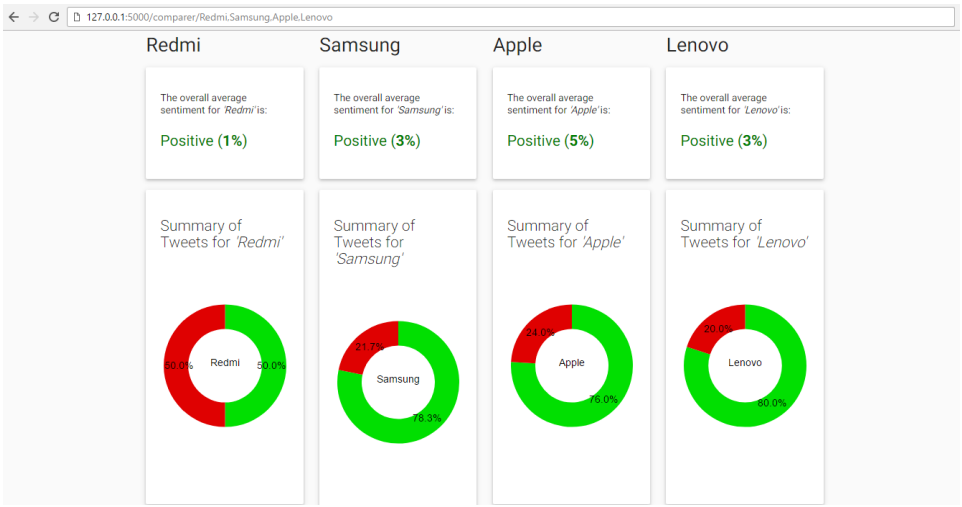

Fig. 8. Comparison results for Redmi, Samsung, Apple and Lenovo.

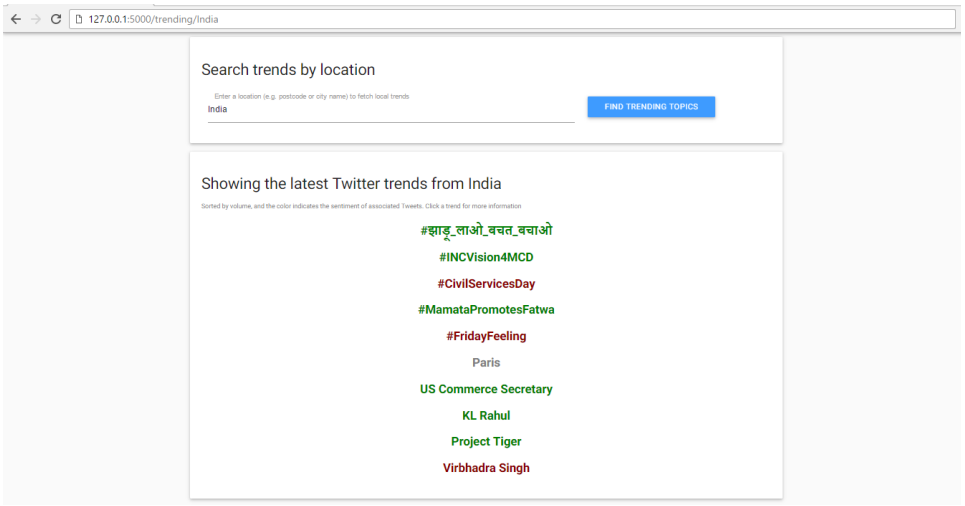

Fig. 9. Latest trends on Twitter from India.

3.3.7. *The World Now*

Figure 10 shows the dashboard for "World Now", a real-time dashboard which keeps updating the stats for positive and negative tweets on landing of the webpage. It also shows the live sentimental tweets by each location. Zooming into the particular region, the tweets of that particular region are highlighted and the users can read the tweets.

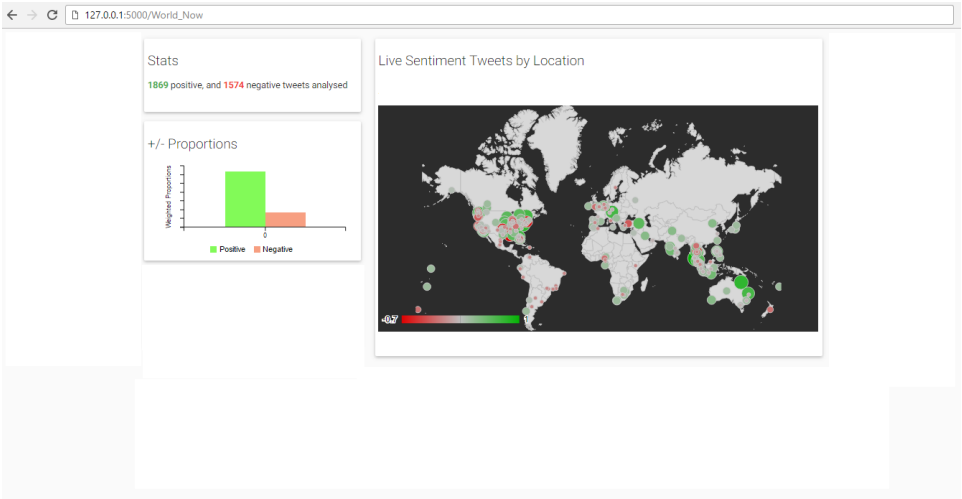

Fig. 10. Live sentiment tweet by location.

3.3.8. *Timeline*

Figure 11 shows the timeline dashboard, a real-time graph which shows the variation of sentiment of the query word input by the user. This is very crucial at times of knowing the real-time events such as voting, disasters and news, etc. The graph shows the hourly variations with history of 24 h. "Demonetization" keyword is used to take the snapshot for this dashboard. Here we observe that most of the people are supporting demonetization; on the other hand, there are also rationally equal negative opinions on demonetization.

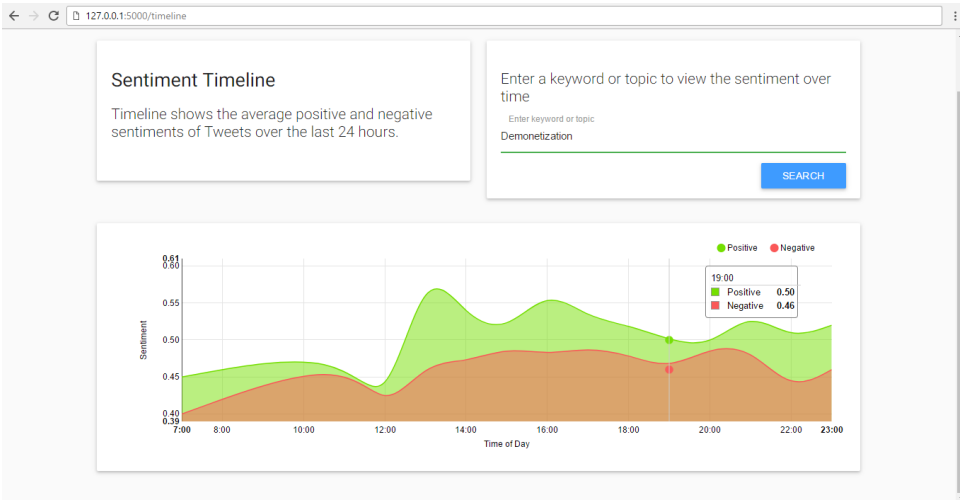

Fig. 11. Timeline for "Demonetization".

Table 1. Type 1 message string format.

Tweet content	Screen name	Created time	Geographical Info (long, lati)	Country code (ISO-alpha)	Sentiment value

Table 2. Type 2 message string format.

Keyword	Value

3.4. *Data format*

In order to allow data parsing components to be connected freely, this project defines a general data format. In the system, data can only be transformed with single string in Apache Kafka, every message is a string. The system defines two types of messages, the first type is shown in Table 1, it contains at least six fields, and DELIMITER is used as separator of each data field. The first five fields are fixed: Tweet Contents, Screen Names, Created Time, Latitude and Longitude and Country Codes. Later, the Sentiment Value is added after the calculation of sentiment score. If corresponding data are not available, it will be set as n/a. Other fields will be added after in sequence. Furthermore, the second type consists of only two fields and fields are split with delimiter as shown in Table 2.

4. Evaluation Results

4.1. *Evaluation of EPS algorithm*

The proposed EPS algorithm is validated by using six different datasets generated using the Data Collection module of our framework. The query strings for preparing the datasets are Paytm, H1B Visa, Redmi, Myntra, Deepika Padukone and Demonetization. The details of the Dataset are shown in Table 3. At first, crowd sourcing method was used to obtain the human judgement for datasets collected (Machedon *et al.*, 2013). The classified tweets had 50% of positive tweets, 41% of negative tweets and 9% of neutral tweets. Later, data parsing module of our framework which implements EPS algorithm is run on the different datasets to obtain the sentiment classified tweets. The Confusion Matrix, Accuracy, Recall, Precision, F-measure metrics are used for evaluating classified results and are

Table 3. Confusion matrix.

		Predicted classes		
		Positive (X)	Negative (Y)	Neutral (Z)
Known class	Positive (X)	$\mathbf{tp}\boldsymbol{X}$	eXY	eXZ
	Negative (Y)	eYX	$\mathbf{tp}\boldsymbol{Y}$	eYZ
	Neutral (Z)	eZX	eZY	$\mathbf{tp}\boldsymbol{Z}$

Table 4. Dataset details.

	Query string	No. of tweets analysed
Dataset 1	Demonetization	1000
Dataset 2	PayTm	99
Dataset 3	h1b1visa	105
Dataset 4	Redmi	100
Dataset 5	Myntra	300
Dataset 6	Deepika Padukone	512

compared with existing algorithms for sentiment analysis such as Emoticon Classifier, Polarity Classifier and Naïve Bayes classifier for which the same datasets were run. Confusion Matrix for the proposed algorithm can be defined as in Table 4. Here X = Positive, Y = Negative, Z = Neutral and the diagonal elements tp$X$, tp$Y$, tp$Z$ are true positives which are correctly classified. The other elements are called false positives which are incorrectly classified. Precision (P) is defined as ratio of true positives to sum of true positives and false positives. It is represented in the equation form as:

$$P(X) = \frac{\text{tp}X}{\text{tp}X + \text{tp}Y + \text{tp}Z}. \tag{8}$$

Recall is defined as ratio of true positives to true positives and false negatives which are manually classified. It is represented in the equation form as:

$$R(X) = \frac{\text{tp}X}{\text{tp}X + \text{e}XY + \text{e}XZ}. \tag{9}$$

F-measure is defined as mean of both precision and recall. It is represented in the equation form as:

$$\text{FM} = 2 * \frac{P * R}{P + R}. \tag{10}$$

Accuracy is defined as fraction of correctly classified tweets (tp$X$, tp$Y$, tp$Z$) to all possibilities (true positives, true negatives, false positives, false negative, true neutrals and false neutrals). It is represented in the equation form as:

$$\text{Accuracy} = \frac{\text{tp}X + \text{tp}Y + \text{tp}Z}{\text{tp}X + \text{e}XY + \text{e}XZ + \text{tp}Y + \text{e}YX + \text{e}YZ + \text{tp}Z + \text{e}ZX + \text{e}ZY}. \tag{11}$$

The evaluation results of six different datasets are listed from Tables 5 to 10. The graph for comparison of accuracy with the existing state-of-the-art techniques such as Emoticon Classifier (EC), Polarity Classifier (PC) and Naive Bayes Classifier (NBC) is shown in Fig. 12. From the figure, we can clearly observe that the proposed EPS Classifier outperforms all the three existing algorithms with highest accuracy of 85% over six different datasets.

Table 5. Dataset 1 results.

Dataset 1		Confusion matrices			Results			
		X	*Y*	*Z*	*P* (%)	*R* (%)	FM (%)	Accuracy (%)
EPS	*X*	39	5	2	95.12	84.78	89.66	88.89
	Y	2	38	0	84.44	95.00	89.41	
	Z	0	2	11	84.62	84.62	84.62	
EC	*X*	8	0	38	100.00	17.39	29.63	21.21
	Y	0	0	40	00.00	00.00	00.00	
	Z	0	0	13	14.29	100.00	25.00	
PC	*X*	19	9	18	100.00	41.30	58.46	65.66
	Y	0	33	7	78.57	82.50	80.49	
	Z	0	0	13	34.21	100.00	50.98	
NBC	*X*	27	17	2	62.79	58.70	60.67	61.62
	Y	16	23	1	54.76	57.50	56.10	
	Z	0	2	11	78.57	84.62	81.48	

Table 6. Dataset 2 results.

Dataset 2		Confusion matrices			Results			
		X	*Y*	*Z*	*P* (%)	*R* (%)	FM (%)	Accuracy (%)
EPS	*X*	38	3	0	82.61	92.68	87.36	82.86
	Y	8	32	2	80.00	76.19	78.05	
	Z	0	5	17	89.47	77.27	82.93	
EC	*X*	4	0	37	100.00	09.76	17.78	52.54
	Y	0	0	42	00.00	00.00	00.00	
	Z	0	1	21	21.00	95.45	34.43	
PC	*X*	16	3	22	76.19	39.02	51.61	68.57
	Y	5	22	15	81.48	52.38	63.77	
	Z	0	2	20	35.09	90.91	50.63	
NBC	*X*	30	10	1	65.22	73.17	68.97	61.62
	Y	14	25	3	65.79	59.52	62.50	
	Z	2	3	17	80.95	77.27	79.07	

Table 7. Dataset 3 details.

Dataset 3		Confusion matrices			Results			
		X	*Y*	*Z*	*P* (%)	*R* (%)	FM (%)	Accuracy (%)
EPS	*X*	35	3	0	79.55	92.11	85.37	86.00
	Y	7	46	1	92.00	85.19	88.46	
	Z	2	1	5	83.33	62.50	71.43	
EC	*X*	13	0	25	100.00	34.21	50.98	23.00
	Y	0	2	52	100.00	03.70	07.14	
	Z	0	0	8	09.41	100.00	17.20	

Table 7. (*Continued*)

Dataset 3		Confusion matrices			Results			
		X	*Y*	*Z*	*P* (%)	*R* (%)	FM (%)	Accuracy (%)
PC	*X*	4	3	21	87.50	36.84	51.85	48.00
	Y	2	27	25	87.50	50.00	63.53	
	Z	0	1	7	13.21	87.50	22.95	
NBC	*X*	24	10	4	58.54	63.16	60.76	67.00
	Y	15	38	1	77.55	70.37	73.79	
	Z	2	1	5	50.00	62.50	55.56	

Table 8. Dataset 4 details.

Dataset 4		Confusion matrices			Results			
		X	*Y*	*Z*	*P* (%)	*R* (%)	FM (%)	Accuracy (%)
EPS	*X*	188	10	0	87.44	94.95	91.04	85.00
	Y	27	67	0	78.82	71.28	74.86	
	Z	0	8	0	0.00	0.00	0.00	
EC	*X*	16	0	182	100.00	8.08	14.95	8.00
	Y	0	1	93	50.00	1.06	2.08	
	Z	0	1	7	2.48	87.50	4.83	
PC	*X*	78	7	113	96.30	39.39	55.91	43.33
	Y	0	1	43	85.71	51.06	64.00	
	Z	0	1	7	4.29	87.50	8.19	
NBC	*X*	180	18	0	77.92	90.91	83.92	75.00
	Y	49	45	0	65.22	47.87	55.21	
	Z	2	6	0	0.00	0.00	0.00	

Table 9. Dataset 5 details.

Dataset 5		Confusion matrices			Results			
		X	*Y*	*Z*	*P* (%)	*R* (%)	FM (%)	Accuracy (%)
EPS	*X*	177	16	2	80.45	90.77	85.30	85.55
	Y	37	220	4	89.80	84.29	86.96	
	Z	6	9	41	87.23	73.21	79.61	
EC	*X*	21	1	173	72.41	10.77	18.75	16.02
	Y	8	6	247	75.00	2.30	4.46	
	Z	0	1	55	11.58	98.21	20.72	
PC	*X*	80	5	110	90.91	41.03	56.54	59.77
	Y	8	171	82	96.61	65.52	78.08	
	Z	0	1	55	22.27	98.21	36.30	
NBC	*X*	149	38	8	64.22	76.41	69.79	71.68
	Y	76	176	8	79.64	67.43	73.03	
	Z	7	7	42	71.19	75.00	73.04	

Table 10. Dataset 6 details.

Dataset 6		Confusion matrices			Results			
		X	*Y*	*Z*	*P* (%)	*R* (%)	FM (%)	Accuracy (%)
EPS	*X*	443	45	4	88.60	90.04	89.31	85.90
	Y	45	366	6	83.18	87.77	85.41	
	Z	12	29	50	83.33	54.95	66.23	
EC	*X*	38	4	450	63.33	7.72	13.77	13.40
	Y	20	117	386	57.89	2.64	5.05	
	Z	2	4	85	9.23	93.41	16.80	
PC	*X*	209	54	292	74.64	42.48	54.15	52.90
	Y	62	243	112	80.46	58.27	67.59	
	Z	9	5	77	18.42	84.62	30.26	
NBC	*X*	315	172	5	84.45	64.02	72.83	74.20
	Y	35	376	6	66.55	90.17	76.58	
	Z	23	17	51	82.26	56.04	66.67	

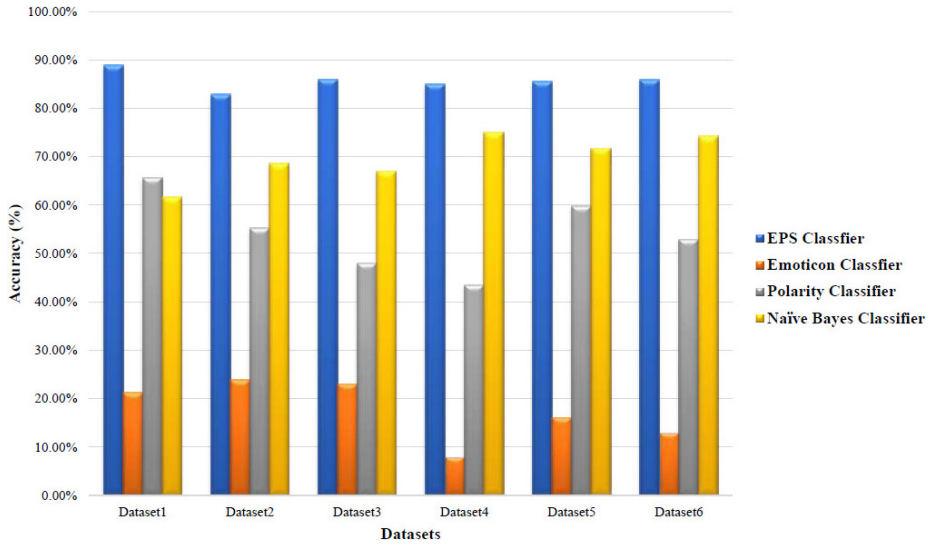

Fig. 12. Comparison of classification accuracy.

4.2. *Evaluation of proposed framework*

The proposed framework is evaluated using three metrics: throughput, runtime and scalability.

4.2.1. *Evaluation of throughput*

The throughput for the application is evaluated as the rate of twitter stream, i.e. the number of messages received from twitter with geographical or location information

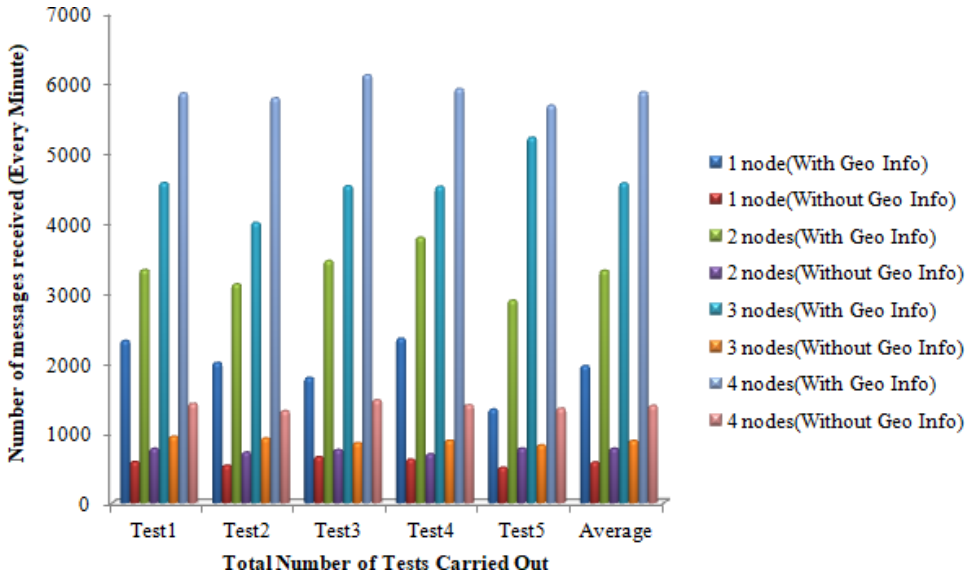

Fig. 13. Evaluation of throughput.

and without any filter, i.e. without any location constraint over multi-node Storm cluster with six vCPUs.

As mentioned before, all data collection components integrate java library twitter4j with Kafka, thus the proposed system handles a simple Kafka consumer to count the number of messages using the Kafka Manager tool which displays the message count on the UI. Figure 13 shows the results drawn for different worker nodes with two cases. One is messages received with geogaphical information as the location of tweets plays a vital role in our visualization. Next is the tweets received without any restriction of location. It is observed that maximum number of tweets are received when there is no restriction of location information since to receive the messages with geographical information the tweets have to pass through the location filter of twitter4j, i.e. "geocode". A single node cluster receives nearly 1995 messages per minute without restriction and 584 messages with geographical information. Only 27% of tweets are received when it comes to location informative tweets. As we see in Fig. 13, the number of messages increases by increasing the nodes of a cluster. A four-node cluster receives a maximum of 5847 tweets without restriction and 1414 geo informative tweets every minute. This shows that the proposed framework can receive maximum number of tweets and scale well for real-time data analytics and visualization.

4.2.2. *Evaluation of runtime of framework*

Figure 14 shows the runtime calculation of different datasets over real-time and tabulated in terms of seconds. All the four classifier algorithms were run on four-node

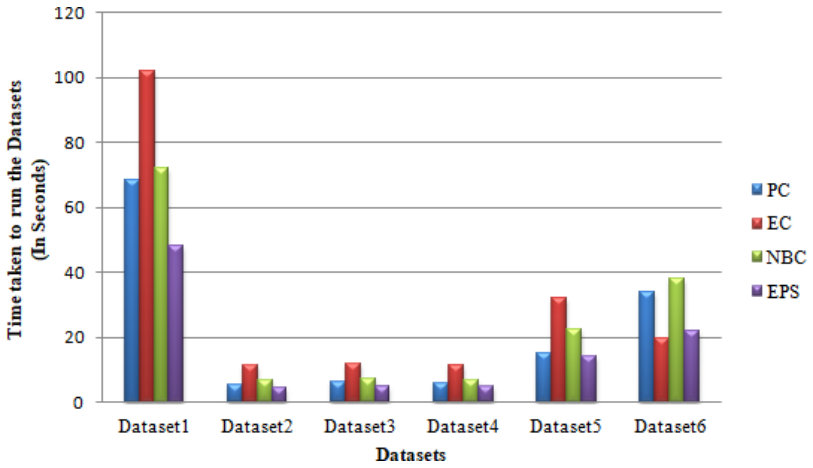

Fig. 14. Evaluation of runtime.

Apache Storm cluster using different datasets. The figure clearly describes that Emoticon Classifier, i.e. EC, and Naïve Bayes Classifier, i.e. NBC, take more time than Polarity Classifier (PC) and Proposed EPS Classifier. Apache Storm considers each classifier algorithm as separate bolts and process bolts using the datasets provided as input CSV files. The runtime of each bolt is recorded here. Among all the classifier algorithms proposed, the EPS algorithm outperforms to take less runtime and proves that it is an efficient classifier algorithm.

4.2.3. *Evaluation of scalability*

The proposed framework is evaluated by considering scalability factor as how well the "TwitSenti" web application can scale with massive tweets. By altering the number of nodes and vCPUs, series of five tests were conducted as Test1, Test2, Test3, Test4 and Test5 every minute to obtain the average which is discussed here. From Fig. 15, for the first two results, it is observed that when the number of nodes increases from 1 to 4 with 1 vCPU then the output rate decreases massively. The reason behind this is single CPU cannot handle multiple thread since each worker thread uses 120% of the CPU for processing. Thus to obtain more accurate results vCPU should be monitored 1.2 times beyond the number of nodes. The results in Fig. 15 show that raising in number of nodes by keeping the computing components such as vCPU constant the performance (output rate) of the system cannot be enhanced. For instance, if we observe the first and third results by keeping the number of nodes constant and increasing the vCPUs from 1 to 4 increases the output rate from 562 to 789. Thus we can conclude that one vCPU cannot cover computing resources of "TwitSent" Web application. From third and fourth results of graph, by increasing the number of nodes to 2, the average data fetch rate increases from 752 to 1201 per minute. In the same way, in fifth and sixth rows by rising the number of vCPUs and node increases, the output rate increases from 1453 to 1656. In rare cases

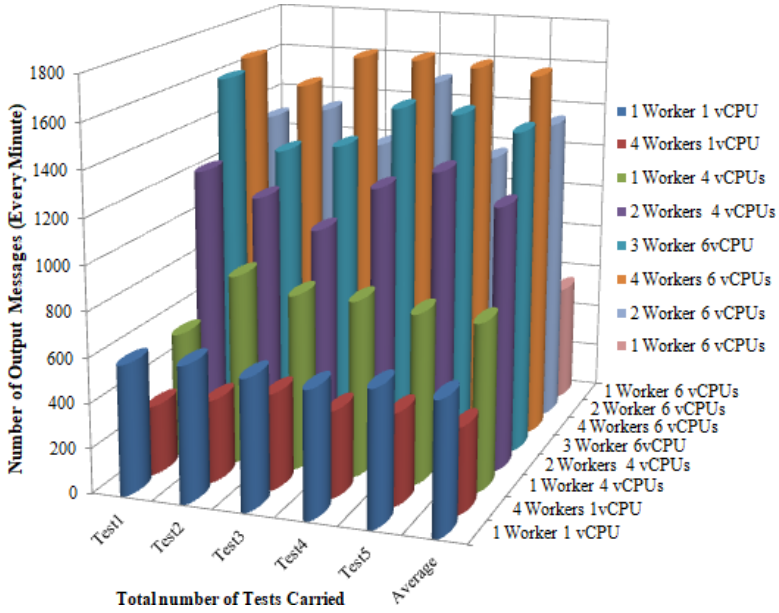

Fig. 15. Evaluation of scalability.

if we provide six VCPs for one-node cluster and two-node cluster as shown in results 7 and 8, the messages output gradually increases because of more number of VCPs. Thus if the computing resources are adequate then increasing the number of nodes leads to high output rate. Since the proposed framework system uses Twitter4j library to fetch the tweets, whose average fetch rate is 1071 messages per minute, the proposed system can completely process all tweets collected, with four virtual CPUs and two nodes.

5. Conclusion

This research work has proposed a distributed real-time twitter sentiment analysis and visualization framework by implementing a novel algorithm for twitter sentiment analysis called EPS. The whole framework is implemented in the form of three major modules called data collection, data parsing and data visualization which are flexible and reusable. Implementation of Apache Storm as a core part of the framework makes it distributed and provides real-time stream processing capabilities which lacked in existing systems. The framework was applied to build an interactive web application called "TwitSenti", which can benefit companies and other organizations in knowing the people's sentiment towards the aspects such as brands, current events, etc. which in turn helps in quick decision-making and planning marketing strategies. The algorithm was validated against three existing classifiers and hence proved that EPS provides highest accuracy value of 85%. Also, the framework showed best scalability results when evaluated web app as four-node

cluster and is proved to be fast and can scale well with massive data. This proves that our framework is best suitable for real-time bigdata analytics and visualization.

6. Future Work

The proposed system focuses on text mining only; besides, text files, images and videos contain beneficial information in twitter network as well. Hence, in future, the framework can be modified to support videos and audios and image files as well. The data visualization component is implemented using Google Charts and D3.js to present data on "TwitSenti" web app. In future, data visualization can be improved by combining other libraries, such as Leaflet.js and Mapbox. Furthermore, the appearance of the website is another potential improvement of the system. Finally, in order to implement dynamic topologies, Storm topologies are used as basic processing units which increase the flexibility of the system and decrease the time of developing new processing functions, however, the size of every topology becomes much bigger. In the future, the proposed framework intends to change the source code of Storm to allow topology to be changed dynamically.

References

Anjaria, M and RMR Guddeti (2014). Influence factor based opinion mining of Twitter data using supervised learning. In *Communication Systems and Networks (COMSNETS), 2014 Sixth International Conference*, pp. 1–8. IEEE: New York.

Azzouza, N, K Akli-Astouati, A Oussalah and SA Bachir (2017). A real-time Twitter sentiment analysis using an unsupervised method. In *Proceedings of the 7th International Conference on Web Intelligence, Mining and Semantics*, p. 15. ACM: New York.

Balamurali, AR, A Joshi and P Bhattacharyya (2011). Harnessing wordnet senses for supervised sentiment classification. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pp. 1081–1091. Association for Computational Linguistics.

Bill McDonald List of Words (2017). Available at http://www3.nd.edu/mcdonald/Word_Lists.html. Accessed on 10 February 2017.

Bing Liu List of Words (2017). Available at http://www.cs.uic.edu/liub/FBS/opinion-lexicon-English.rar. Accessed on 10 February 2017.

Chang, CC and CJ Lin (2011). LIBSVM: A library for support vector machines. *ACM Transactions on Intelligent Systems and Technology*, 2(3), 27.

Jansen, E and V James (2002). NetLingo: The Internet dictionary. Netlingo Inc.

Khuc, VN, C Shivade, R Ramnath and J Ramanathan (2012). Towards building large-scale distributed systems for twitter sentiment analysis. In *Proceedings of the 27th Annual ACM Symp. Applied Computing*, pp. 459–464. ACM: New York.

Lau, JH, N Collier and T Baldwin (2012). On-line trend analysis with topic models: # twitter trends detection topic model online. In *Proceedings of COLING 2012*, pp. 1519–1534.

Machedon, R, WM Rand and YV Joshi (2013). Automatic classification of social media messaging using multi-dimensional sentiment analysis and crowdsourcing. *SSRN Electronic Journal*. doi:10.2139/ssrn.2244353.

Porter, MF (2001). Snowball: A language for stemming algorithms.

Reis, J, H Kwak, J An, J Messias and F Benevenuto (2017). Demographics of News Sharing in the US Twittersphere. arXiv:1705.03972.

Ritter, A, O Etzioni and S Clark (2012). Open domain event extraction from twitter. In *Proc. 18th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 1104–1112. New York: ACM.

Ritter, A, S Clark and O Etzioni (2011). Named entity recognition in tweets: An experimental study. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pp. 1524–1534. Association for Computational Linguistics.

Saif, H, Y He, M Fernandez and H Alani (2016). Contextual semantics for sentiment analysis of Twitter. *Information Processing and Management*, 52(1), 5–19.

Statista (2017). Available at https://www.statista.com/statistics/282087/number-of-monthly-active-twitter-users/. Accessed on 14 April 2017.

Wang, C, Z Xiao, Y Liu, Y Xu, A Zhou and K Zhang (2013). SentiView: Sentiment analysis and visualization for internet popular topics. *IEEE Transactions on Human-Machine Systems*, 43(6), 620–630.

Yadollahi, A, AG Shahraki and OR Zaiane (2017). Current state of text sentiment analysis from opinion to emotion mining. *ACM Computing Surveys (CSUR)*, 50(2), 25.

Zimbra, D, M Ghiassi and S Lee (2016). Brand-related Twitter sentiment analysis using feature engineering and the dynamic architecture for artificial neural networks. In *System Sciences (HICSS), 2016 49th Hawaii International Conference*, pp. 1930–1938. IEEE: New York.

Zitouni, I (Ed.) (2014). *Natural Language Processing of Semitic Languages*, pp. 299–334. Berlin: Springer.