IJICTR
International Journal of Information & Communication Technology Research

Volume 12- Number 1 – Winter 2020 (42 -55)

# Trend Detection and Prediction in Blogosphere based on Sentiment Analysis using PSO and Q-Learning

**Rezvan Mohamadrezaei**
Department of Computer Engineering,
Central Tehran Branch, Islamic Azad University
Tehran, Iran
rez.mohamadrezaeilarki.eng@iauctb.ac.ir

**Reza Ravanmehr***
Department of Computer Engineering,
Central Tehran Branch, Islamic Azad University
Tehran, Iran
r.ravanmehr@iauctb.ac.ir

Received: 7 August 2019 - Accepted: 2 December 2019

*Abstract*—**The blogosphere is an effective communication platform where users publish and exchange their opinions. By analyzing user behavior, current and future trends of a community can be discovered. The proposed model for processing the social data of users first extracts related sentiments of weblog comments. An improved PSO algorithm is then employed to detect the trend of users in the TRDT (TRend DeTection) phase. By the discovery of trends at a reasonable time and appropriate precision, this model predicts future trends of the blogosphere using the Q-learning algorithm in the TRPT (TRend PredicTion) phase. Given the ever-increasing processing requirements and a huge volume of data, our approach provides a distributed processing/storage platform for TRDT and TRPT phases. The precision and performance of the proposed model in the TRDT phase are measured by the Chi-squared standard test. Moreover, the evaluation of the TRPT phase shows the comparable precision of the proposed approach with real-world scenarios such as the Netflix predictive system.**

***Keywords-Social Networks; User Sentiment Analysis; Particle Swarm Optimization; Q-Learning; Trend Detection/Prediction; Blogosphere.***

## I. INTRODUCTION

Personal and social media on the web provide services in the form of social networks, weblogs, and microblogs to share interests and opinions. The rapid growth of social media enables web users to be faster informed of the opinions of others about various topics, follow collective behaviors of users, and detect the "most popular" content presented in these formats. Indeed, the content of blogs can effectively reflect the events in personal life experiences or public opinions. Microblogging services such as Twitter, Sina Weibo, and Twister facilitate easy sharing of status messages among different users. The real-time features of microblogging systems have turned them into a social hotspot for initializing various public events and related discussions. For example, 22 events of the total 138 events (about 16%) in 2010 were reported originally in microblogs. According to another paper, this percentage increased to 36% in 2011, i.e., 36% of the total hot real-time events were originally reported to microblogs [1]. The popularization of these microblog platforms has led to the development of the so-called blogosphere.

* Corresponding Author

The innovative aspect of this paper is responding to the two following fundamental questions: 1) is a distributed processing platform able to detect a general trend of content published in the blogosphere within an acceptable period using sentiment analysis and swarm intelligence procedures without violating or threatening scalability by the developed system? (The acceptable period is the interval between two changes in the most important trends of the blogosphere. If the trend is detected in a period longer than this interval, the users will lose some important information), 2) in the case of detecting the trend within an acceptable period of time, are future blogosphere trends can be detected by machine learning algorithms such as Q-Learning?

Insight extraction from a large volume of data and detection of similar patterns and complex relationships between these data seem impossible without data mining operations and data analysis on these large datasets. A wide range of databases and services from social networks such as Twitter and Facebook to search engines such as Google take advantage of data mining to exploit users' interests, refine the search results, and detect the general trend of users. In addition to enhancing the accuracy of searches and helping users to access their required data faster, these capabilities have extensively affected the social behavior of users. Detecting the general trend of users in the virtual world as a reflection of their real behavior is of great importance to be analyzed in business, advertising, and social sciences. Although data mining is not a new topic in the field of information technology, today's trends and needs in this area are facing huge data volumes and more complications. For example, Daily Active Users (DAU) Worldwide is 152 million for Twitter in the fourth-quarter of 2019 and should apply hundreds of millions of transactions per day while simultaneously extracting user trends [2]. Processing this enormous amount of data is associated with serious requirements in this area, such as more precise and intelligent processes in a reasonable time for the user. This requirement is a prelude to the introduction of artificial intelligence and optimization to achieve more accurate results in data mining, provide more intelligent analyses, as well as detect behavioral patterns on data streams published on the web.

In this paper, based on the swarm intelligence algorithms, different components of the proposed approach will search (in parallel) in the state space for detecting the general trends of users in different time intervals. In addition to the content of the weblog posts, which are used in the sentiment analysis component, publishing and activation times of different posts on a weblog are also available. The trend of published weblog posts can be extracted from these temporal tags. Considering the detection of a trend in an acceptable period, the weblog posts are evaluated, and the titles with the highest trend are specified. Then, the most trending titles will be predicted for this community for the next periods. To this end, the performance of the proposed approach in determining the group behavior of a community of users in a short time with acceptable accuracy is evaluated.

In addition to detecting user trends, the future direction of trends by the model is predicted using an appropriate machine learning algorithm (Q-learning). Extensive experiments are also performed on real-world social network data, the posts published on the Verge database, to demonstrate the accuracy and effectiveness of our proposed approach in comparison with other state-of-the-art methods. The Verge is an American technology news and media network that publishes news items, long-form feature stories, product reviews, etc.

In brief, we summarize the contributions of this research as follows:

- Developing a model for trend detection based on the improved PSO algorithm and social data of users within a reasonable time and appropriate precision.
- Prediction of future trends of blogosphere using Q-learning algorithm.
- Providing a distributed processing /storage platform for trend detection and prediction to speed up the process.

The paper is structured as follows. The literature is reviewed in the next section. Section III explains the foundation of the research. The proposed algorithm is discussed in Section IV. Experimental results are presented in Section V. Conclusions and future works are provided in Section VI.

## II. Literature Review

First, in recent years, research on the detection of trends and user interests in weblogs has been motivating for many scholars. The blogosphere is commonly known as the social network of weblogs and includes a community of users that interact with each other and form small and large communities. In this type of social network, a specific notable member begins an important conversation, and the reactions of others may appear as referring or collector nodes. In real-world situations, there may be a number of certain notable members beginning large-scale conversations and several other members who gather content from different discussions [3].

The valuable information of blogs is mostly obtained by monitoring and analyzing user-generated content. Different approaches have been addressed in the literature to analyze these extracted contents to detect or predict the trends. For this purpose, we consider four categories for reviewing the articles that are published in trend detection and prediction in weblogs: pattern mining, time series, sentiment analysis, and swarm intelligence. Since most of the published papers in recent years have been focused on sentiment analysis, this category will be further divided into three sub-categories to provide a more comprehensive overview in this domain. It is worth mentioning that our approach is a combination of sentiment analysis and swarm intelligence.

### *A. Trend Detection/Prediction Using Pattern Mining*

Originally, social network mining approaches are divided into different categories, such as graph mining, classification, clustering, and link mining. Indeed, social network mining can be applied in a static context, which ignores the temporal aspects of the network, or in a dynamic context, which takes temporal aspects into

consideration. In the static context, the analysis is directed at either: 1) finding patterns of the network, 2) clustering subsets of the networks, or 3) building classifiers to categorize nodes and links in a snapshot of the network. In the dynamic context, relationships between the nodes are identified in the network by evaluating the spatio-temporal co-occurrences of events [4]. A social network trend analysis approach has been proposed by Nohuddin et al. that is based on the frequent pattern mining and the Self Organizing Maps (SOM) clustering. They defined the trends of interest in terms of sequences of support values for specific patterns that appeared across a given social network [5].

Venant et al. revealed relationships between students' activities and their academic performance [6]. For this purpose, they gathered data from a remote laboratory learning environment. A sequential pattern mining approach has been developed, and it showed that a correlation exists between academic strategies and the learners' performance. Choi and Park proposed a method for detecting topics from Twitter streams using HUPM [7]. The proposed method includes a stage for calculating the utilities for words in each batch of tweets by the sliding window, a stage for determining the min-util on each batch, and a stage for extracting actual topic patterns from the candidate topic patterns by TP-Tree. An automatic, effective, and scalable method proposed by Kawabata et al. operates on semi-infinite collections of co-evolving streams and summarizes all the streams into a set of multiple discrete segments grouped by their similarities. it automatically and incrementally recognizes such patterns and generates models for each of them [8].

### *B. Trend Detection/Prediction Using Time Series*

Recent complex multivariate temporal datasets impose on developing algorithms capable of analyzing and learning from such data [9]. These datasets come from a variety of sources especially social networks. The data instances (which are usually time series of multiple variables) show traces of complex behaviors. Batal et al. provided several pattern mining techniques for finding recent temporal patterns in applications such as event detection. Therefore, they employed a time interval representation of a sequence of events [9].

Arthur et al. proposed a geometric model for trend detection in one-dimensional time series data [10]. Trends are detected using three natural parameters: granularity, support size, duration. These parameters can be configured for different environments. The model proposed by Anghinoni et al. tries to solve both the time-frame and noise problems inherent in stochastic processes. It sets out to propose a trend classification method based on community detection on a complex network derived from the stochastic time series [11]. Their proposed learning scheme is able to capture local and global patterns. Jadeja and Kotecha proposed an accurate and refined prediction algorithm that deals with the weight of the term and instrument Poisson distribution function to forecast the behavior of the term for the next cycles [12].

### *C. Trend Detection/Prediction Using Sentiment Analysis*

The sentimental analysis plays an essential role in trend detection and prediction in social networks. In fact, many techniques in recent years employ sentiment analysis for trend detection. Based on our literature review in this specific area, most of these techniques utilize lexicon or machine learning or hybrid approaches. For this purpose, we have conducted an extensive literature review, and the results are explained in the next subsections.

#### *1) Sentiment Lexicons Based Methods*

Sentiment analysis approaches based on the lexicons are preferred over learning-based methods when the training data is not adequate [13]. These approaches are simple and domain-independent. The Semantic Orientation CALculator, which is a method based on the dictionaries of the numerical values for the polarity of each word and phrase together with their semantic orientations has been developed by Taboada et al. [14]. Moreover, they attempted to classify sentiment based on the representation of negative or emphasis phrases. Xiaomei et al. proposed a model for sentiment extraction using hashtag analysis [15]. As a dataset, they employed the Sina Weibo corpus to build the co-occurrence graph, which consists of emoticons and words according to the words and emoticons similarity. This graph has been used to analyze the sentiment of microblogs for the classification of the detected bursts. Hung and Jeng Chen proposed a revised approach for sentiment lexicon [16]. For this purpose, they have adopted the SentiWordNet to a domain-oriented sentiment lexicon. They argue that most sentiment-based classification processes only extract the sentimental words from SentiWordNet without considering the word sense disambiguation (WSD).

To eliminate the interference of local daily events generated by noisy data, Yang et al. proposed a bursty event detection method for quantifying the influence of microblog text [17]. Through the analysis of high-impact microblogs, the burst words are mined, the potential bursty event data sets are constructed, and the k-means cluster method is applied to detect the event. Tattershall et al. have explored a stock market-inspired burst detection algorithm and used it to find bursty terms in over 30 years of computer science abstracts. These terms represent a snapshot of computer science researches. The proposed method requires little tuning and can be used on a large dataset [18].

#### *2) Machine Learning-Based Methods*

Due to the time-consuming task of the manual creation and evaluation of complex sentiment lexicon, many approaches have attempted to extract the sentiment-relevant features in the text through automated techniques [19]. Romanowski and Skuza illustrated the development of a sentiment analysis system for future predictions of the stock market. The authors utilized the classification of data extracting from the Twitter social network and the previous stock market records [20].

Qiu, et al. proposed a variant of the Conditional Random Field model, called SentiCRF, for generating

term pair and calculate their sentiment scores [21]. For this purpose, they employed a cumulative logit model based on the aspects and their sentiments in a review platform to accurately predict the non-rated reviews' ratings. For training their SentiCRF model, they utilized a heuristic re-sampling algorithm. Iyer et al. presented how predictive the sentiment or opinion of the users are in social media microblogs and how it compares to that of the experts. They find that the crowd wisdom in the microblogging domain matches that of the experts in most cases [22]. There are cases, however, where they don't match, but they observed that the crowd's sentiments are actually affected by the experts.

An intelligent event propagation model with the knowledge sets has been proposed in [23] that consists of three steps: first propagation, learning process, and consecutive propagation for event detection and prediction. Ali et al. have developed an event detection system using real-time data to simultaneously snapshot the whole dynamic dataset at different times [24]. For this purpose, they took spatial and temporal information of the tweets into consideration to construct location and time-based clusters of the events and find influential spreaders over time.

Gui et al. utilize a novel multiagent reinforcement learning method in which the user tweet selector and candidate-mentioned-user tweet selector cooperatively extract indicator content for mentioned prediction and recommendation task [25]. Another work proposed by Zhang et al. to aggregate all of the micro-blogs generated by the same individual user to calculate the user's influence power and to learn user's interest distribution over topics to discover new emerging events in microblogging and predict their future popularity [26].

*3) Hybrid Methods (Machine Learning And Lexicon Based)*

Many researchers have attempted to employ both machine learning methods and lexicon-based methods to improve sentiment analysis performance [21]. Hutto and Gilbert presented a technique called VADER to analyze the trends in social media utilizing a rule-based model. They developed a gold-standard list of lexical features for sentiment analysis, which can be applied in different contexts, such as microblogging social networks [19].

The approach known as SentiTrend is presented in [27] for trend detection in Twitter using the Maximum Entropy Classifier and TF-IDF model (Term Frequency–Inverse Document Frequency). In this approach, the posts are collected and processed in order to discover the trending topics in the Spanish language. The proposed system in [28] attempted to detect events in real-time based on deep learning on sentiment trajectory. For this purpose, they employed trajectory analysis and sentiment about the special keywords. They utilized the combination of CNN and LSTM for the prediction model. HassanKhan et al. provided a general-purpose sentiment analysis approach based on the combination of semi-supervised machine learning and sentiment lexicon [29]. To define word semantics, they utilized the expected likelihood estimation smoothed odds ratio.

Dey et al. [13] presented a methodology to create a lexicon called Senti-N-Gram. The Senti-N-Gram is an n-gram sentiment dictionary developed along with the related scores. To extract the sentiment score, they employed the rule-based machine learning approach. Dridi and Recupero investigated the role of semantics in sentiment analysis of social media [30]. To this end, frame semantics and lexical resources such as BabelNet are employed to extract semantic features from social media that lead to more accurate sentiment analysis models. Liu et al. proposed a novel model called MLEM for multi-lingual event detection and evolution graph generation. They merged the same entities and similar phrases and presented multiple similarity measures by incremental word2vec model [31].

*D. Trend Detection/Prediction Using Swarm Intelligence*

In recent years, incorporating swarm intelligence in analyzing social media to detect/predict trends is a novel idea. Different techniques of swarm intelligence are suitable approaches for complex pattern classification and clustering.

Banerjee and Agarwal presented a nature-inspired theory to model collective behavior in social media using the blog's data to predict future trends. For this purpose, ACO (Ant Colony Optimization) is trained with the behavioral trend from the blog data [32]. Lv et al. proposed a new solution called Twitter Optimization for analyzing the behavior model on Twitter [33]. They utilize one of the swarm intelligence bio-inspired algorithm techniques, which is Particle Swarm Optimization (PSO). In [34], a hybrid quantum swarm intelligence method has been proposed for link prediction. This approach can be used in event detection in social networks utilizing the fluctuation discovery and optimal weight techniques for link prediction.

Orkphol and Yang proposed a novel combination of an artificial bee colony (ABC) to improve K-means for sentiment analysis of microblogging. To overcome the problem of short and noisy, TF–IDF is used to transform a message to a vector form, and the SVD technique is also used to reduce dimensions by selecting the most important relevant terms which are desirable by K-means [35]. Albert et al. presented an automatic method to find trends. The model bases his success on the weights of the structures that are used to predict a given trend. Weights are adjusted by a Neural Network with particle swarm (HydroPSO) to optimize the fitness function, the final slope is calculated based on the number of the linear structures and their adjusted weights [36]. Kumar and Jaiswal demonstrated the use of two meta-heuristic swarm-based optimizers, binary grey wolf and binary moth fame, for selecting a subset of relevant features used to train and test the learning model, which predicts the sentiment in a microblog [37].

Pandey et al. proposed a method for event detection and sentiment analysis in Twitter that clusters the input tweets in three phases; (i) preprocessing of the tweets, (ii) feature extraction, and (iii) hybrid clustering using K-means and cuckoo search [38]. The review and assessment of the different swarm intelligence methods have been proposed by Silambarasi et al. [39]. For this purpose, they analyzed the performance of cuckoo

search algorithm, genetic algorithm, and particle swarm optimization in trend detection/prediction using Twitter dataset. Sayed et al. proposed a metaheuristic method based on PSO and k-means on an Apache Spark platform to collect and analyze the stream of tweets for event detection purposes [40]. Similar work has been performed by Salam et al. using another swarm intelligence technique, the Whale Optimization algorithm [41].

### *E. Discussions*

Considering the methods discussed in the previous subsections, it is obvious that the approaches based on sentiment analysis and machine learning have been very effective in trend detection. On the other hand, in recent years, swarm intelligence techniques (such as PSO) have undeniably played an important role in improving trend detection approaches in text-based social media, such as the blogosphere, Twitter, and Sina Weibo. In fact, every post in a text-based social media contains features such as title, author, text, and some comments, which makes it suitable to be employed as the input of a PSO algorithm, as explained in chapter IV, subsection "Detection of Users' Trend (TRDT)". This is mostly due to the nature of the social networks that are formed based on the collaboration between different users and sharing of posts for various subjects. This is an excellent motivation for employing swarm intelligence techniques to analyze the users' sentiments.

As briefly explained in the introduction and will be discussed later in the next sections, our proposed trend detection method utilizes the regular expressions for frequent pattern matching and searching of users' posts to extract the intended sentiments. Then, a swarm intelligence technique (improved PSO algorithm) is employed to detect the trend of users in the TRDT phase. For trend prediction, a popular machine learning algorithm called Q-learning, which is model-free reinforcement learning, is used to predict the future trends of the blogosphere. It should be mentioned that reinforcement learning techniques (such as Q-learning) are very effective in prediction models in different areas of social network analysis and have been employed in different fields recently. TRPT is developed based on a modified version of Q-learning that employs the combination of PAC (Probably Approximately Correct) learning and Markov decision processes, as explained in chapter IV, subsection "Prediction of Users' Trend (TRPT)".

## III. Fundamental Concepts

A brief overview of the principles used for the proposed approach of this paper has been explained in this section.

### *A. PSO Algorithm*

PSO (Particle Swarm Intelligence) is a meta-heuristic algorithm that consists of a specified number of particles that randomly take an initial value. For each particle, the two values of position and velocity are modeled by a position and velocity vector, respectively. These particles iteratively move in the n-dimensional space of the problem to search for new probable options by computing the optimum value as a criterion for assessment [42]. The problem space dimension is equal to the number of parameters in the function to be optimized. A memory saves the last best position of each particle in the past, and another memory saves the best position among all the particles. The particles decide about their movement on the next turn with the experience gained from these memories. In each iteration, all the particles move in the n-dimensional space of the problem to find the global optimal point in the end. The particles update their velocity and positions in terms of the best absolute and local responses. Equations (1) and (2) show how the particles update themselves, and Fig. 1 illustrates this concept in a diagram.

$$\boldsymbol{\vartheta_{m,n}^{new}} = \vartheta_{m,n}^{old} + \Gamma_1 \times r_1 \times \left(p_{m,n}^{localbest} - p_{m,n}^{old}\right) + \Gamma_2 \times r_2 \times \left(p_{m,n}^{globalbest} - p_{m,n}^{old}\right) \quad (1)$$

$$\boldsymbol{p_{m,n}^{new}} = p_{m,n}^{old} + \vartheta_{m,n}^{new} \quad (2)$$

The variables of the above equations are as follows:

- ✓ Velocity of particle : $v_{m,n}$
- ✓ Variables of particle : $p_{m,n}$
- ✓ Independent random numbers with uniform distribution $r_2$ , $r_1$
- ✓ Learning factors :$\Gamma_1$ , $\Gamma_2$
- ✓ Optimum local answer : $p_{m,n}^{localbest}$
- ✓ Optimum absolute answer :$p_{m,n}^{globalbest}$

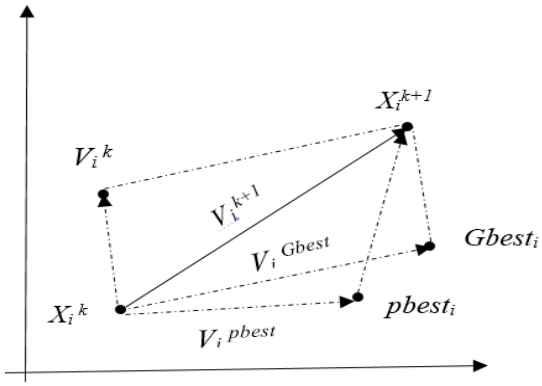


Figure 1. Velocity and position vectors' update in PSO (Clerc et al., 2006).

### *B. Q-learning Algorithm*

Q-learning is a reinforcement learning technique that follows a specific guideline to carry out different actions in a different state by learning an action/value function. A strong point of this method is the ability to learn the mentioned function without having a certain model of the space. A recent modification has been made to Q-learning, known as delayed Q-learning, which caused a significant improvement. In this new model, PAC (Probably Approximately Correct) learning has been combined with Markov decision processes [43]. To each pair (state, action), a Q(s, a) value is assigned. This value includes the sum of rewards when we begin from s state, perform the action a, and follow the available guideline. For learning the Q function, we can use a table with <s, a> pair as entry together with the approximation of agent from the real value of Q. The values in this table are filled with a random initial value (usually zero). The agent

alternatively detects the current s state and performs an action like a. Then, the reward of r(s, a) and the new resulting state of performing the action [s'= d(s, a)] are observed. The values in the table of Q function change based on equation (3).

$$\hat{Q}(s,a) \leftarrow r(s,a) + \gamma \max_{a'} \hat{Q}\left(s',a'\right) \quad (3)$$

## IV. PROPOSED MODEL

Fig. 2 shows a high-level overview of the proposed approach. In the following subsections, each part of the model is explained in detail.

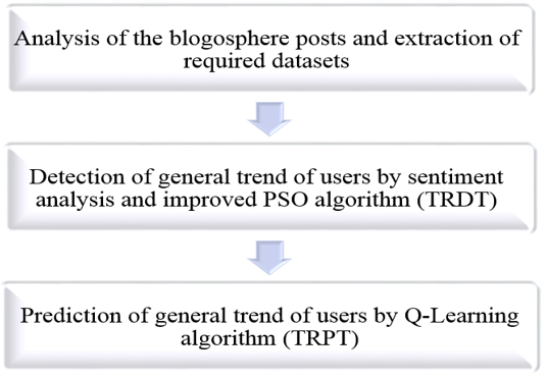


Figure 2. The overview of the proposed model.

### A. *Analysis Of Blogosphere Posts And Extraction Of Required Datasets*

In the proposed model, we collect different types of data to provide an effective trend detection and prediction system. For this purpose, the posts published on The Verge[2] database and social data of users are employed. Each post has a title, an author, a text, and some comments. Table I investigates the effect of each feature in the dataset on the proposed model. The Verge dataset will be discussed in detail in subsection C in section V.

### B. *Detection of Users' Trend (TRDT)*

Following PSO algorithm discussions in subsection A of section III, the particles should align with each other in terms of data values to be able to move towards the optimal point together with the swarm, which in fact occurs during the initialization step. It was already mentioned that the PSO algorithm has a lower performance than the ant colony algorithm, although its accuracy is three times higher [44].

One of the significant factors that reduce the PSO performance is the initialization step [42]. The optimal values produced by these algorithms are specific to a particular dataset, and if the dataset is changed, the optimal value should be re-calculated. In such circumstances, since the algorithm is invoked in large numbers and possibly for a large volume of data, the steps of the algorithm should be designed in such a way that repetitive tasks could be eliminated and the complex steps demanding a considerable computing power could be simplified as much as possible.

TABLE I. EMPLOYING THE BLOG FEATURES IN THE MODEL

| **Blog feature in the dataset** | **Application in TRDT model** |
|---|---|
| content and title of post | A context to extract the histogram of keywords |
| Histogram of keywords | Candidate trends extracted from posts. Each candidate trend has a rank that is a positive value |
| Metadata of posts | The starting point for the development of keywords' histogram |
| Post's comments | Numerical coefficient in the calculation of user sentiments. This coefficient is computed by analysis of text comments |
| Post's sentiment | Dimensions of particle movement: computation of sentiment for each particle results in fitness value, which should be maximized. |
| Post's author | A coefficient in velocity of particles. If a notable popular author writes a post, the normal velocity of the particle is increased. |
| date/time of post publication | The factor causing the particle to enter the CAQ |
| Related content | A coefficient in particle direction vector that coordinates the particle movement direction with its related contents. |
| The Blog posting rate | Increased the generation and initialization of new particles, and the development of agent-sets |

In the PSO algorithm, the particle initialization step is executed each time for all data, no matter the remaining steps of the algorithm. Thus, the previous iterations of the initialization phase can be used as long as the initialized particles are saved somewhere in the memory. The proposed pseudo-code for the TRDT phase shows this process in Algorithm1. Storing all particle data is so expensive in blogosphere space where over tens of millions of daily blog posts are published, which requires that the user trends are generated in real-time. If the information on each blog post is only 1MB, a computer with a 10TB or more RAM is required (only for storing the data for one day), while considering the detection of trends is not related to the previous days.

In these conditions, the need to run the algorithms on more computers with a higher distributed processing/storage platform is revealed. Indeed, more than one workstation or server is required to process/store this volume of data, each of which is known as a processing node. Therefore, we should first store the data of initialized particles to avoid their recalculation in later iterations of the algorithm. Second, this should be done by a number of computers to resolve the physical limitations that we currently face. In other words, the algorithm must be implemented in a distributed environment. In the TRDT

[2] www.theverge.com

phase, two specific modules, called CAQ and Aggregator, have been utilized to meet the above requirements.

**Algorithm1.** The proposed pseudo-code for TRDT phase

```
Procedure (process context A)
While  (true) {
        Check for new element (blog post / comment)
        Parse new element and get keywords
        Wrap element and set related objects
        Put wrapped element in CAQ
}
-------------------------------------------
Procedure (process context B)
If there is new element in CAQ
   For each new element
      Initialize particle
   END
DO
   For new element
      Extract publisher (blogger / commenter) name
      If publisher name already ranked
         Get the rank
      Else
         Rank the publisher and save the rank
//sentiment is the fitness value here and defines that popular
words from
//histogram got what feedback
    Calculate sentiment for this element (by regular expression)
    If the sentiment value is better than the best sentiment value
(pBest) in history
      Set current value as the new pBest //set the trend for this
agent set
    END
Send the particle with best fitness value to Aggregator  // Send
the trend to Aggregator
   Choose the particle with the best fitness value of all the
particle as the gBest
   For each particle
      Calculate particle velocity
      v = v + c1 × rand()× (pbest-present) + c2 × rand()× (gbest-
present)
      Update particle position present = present + v
   END
END
```

1. *CAQ*

CAQ (Change Aware Queue) is a new data structure responsible for distinguishing the new data from duplicate ones, and the principles of inserting and deleting of CAQ are based on the queue data structure. In fact, CAQ is a data queue capable of recognizing new data. During the initialization phase, the PSO algorithm receives the newly added particles through this data structure and only initializes them to prevent several initializations of data in frequent iterations of the algorithm. In each iteration of the algorithm, the whole CAQ is fed into the algorithm as a transaction, and the initialization step occurs only on new data. As shown in Fig. 3, each set of algorithms has one (or more) CAQs.

2. *Aggregator*

The aggregator is a coordinator of the results produced by the algorithm. In the pseudo-code in Algorithm1, the code related to (process context B) is independently run on different nodes. Each node sends its results to the Aggregator, which means that each node calculates the generated trends based on its local data and sends it to the Aggregator. The Aggregator applies the same algorithm on input data to identify the most popular trends (Fig. 3).

As mentioned earlier, the whole CAQ is fed into the algorithm as a transaction upon each iteration of the algorithm, and the initialization stage occurs only on new data. Now, to reduce the duplicate initializations that directly affect the performance of the algorithm, we proceed as follows: If we assume that:

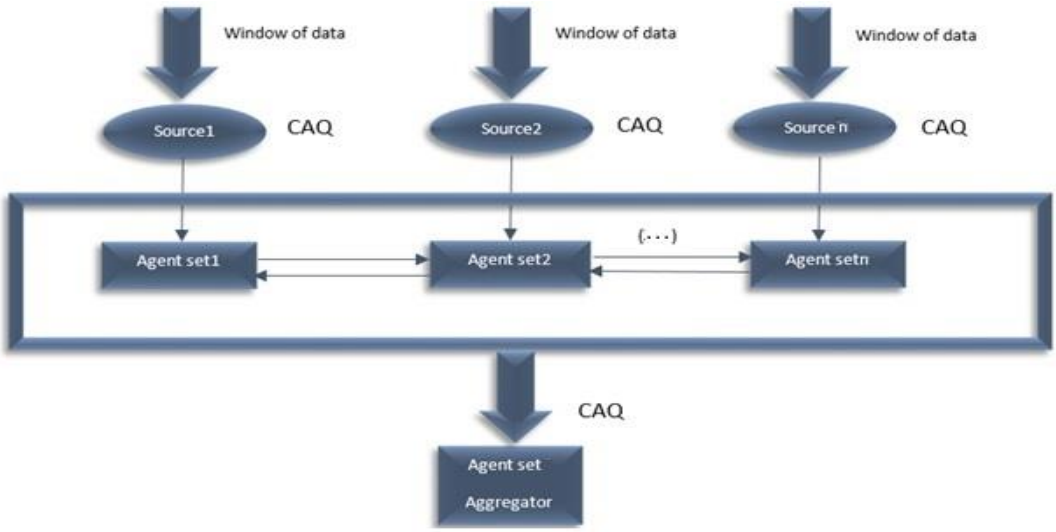


Figure 3. CAQ and Aggregator in TRDT.

- ✓ T: is the time window for which the general trend should be extracted in seconds. For example, if we say that the general trend of the blogosphere is "Computer Network" in the last three days, then T= 3 *24*60*60.
- ✓ B: is the number of weblog posts published during the T period
- ✓ I: is the number of iterations of the algorithm in the T period. The value of this parameter is mostly dependent on the user's requirements.
- ✓ R: is the number of particles with duplicate initializations.

We assume that the distribution of published posts per unit time is a normal distribution. Therefore, if the standard form of the PSO algorithm is used, the R value will be equal to:

$$R = \left(0 \times \frac{B \times T}{I}\right) + \left(1 \times \frac{B \times T}{I}\right) + \cdots + \left(I \times \frac{B \times T}{I}\right) \quad (4)$$

$$R = \frac{B}{2} \times T \times (I + 1) \quad (5)$$

As mentioned before, the CAQ is used to store the weblog's posts. Suppose then:

- ✓ W: is the desired time window to be stored by CAQ (per second)
- ✓ n: is the number of CAQs in the system

Thus, R, in this case, will be equal to:

$$R = \left(0 \times \frac{B \times T}{n \times W}\right) + \left(1 \times \frac{B \times T}{n \times W}\right) + \cdots + \left(\left(\frac{T}{n \times W}\right) \times \frac{B \times T}{n \times W}\right) \quad (6)$$

$$R = \frac{B}{2} \times \frac{T^2 \times (n \times W + T)}{nW^3} \quad (7)$$

With regard to equations 5 and 7, it is clear that:

a. $\frac{T^2 \times (n \times W + T)}{nW^3} > T \times (I + 1)$:

The number of initializations is increased for duplicate particles.

b. $\frac{T^2 \times (n \times W + T)}{nW^3} = T \times (I + 1)$:

The number of initializations is equal for duplicate particles.

c. $\frac{T^2 \times (n \times W + T)}{nW^3} < T \times (I + 1)$:

The number of initializations is decreased for duplicate particles.

Therefore, option (c) should be selected to reduce the number of duplicate initializations. For this purpose, if n and W are chosen so that:

$$\frac{T \times (n \times W + T)}{nW^3} < I + 1 \quad (8)$$

the number of initializations is decreased. In equation (8), the smaller the $\frac{T \times (n \times W + T)}{nW^3}$ value from I+1, the lower the number of initializations. Therefore, the number of initializations can be minimized by this method. As can be seen from equation (7), the number of initializations will never be zero because the PSO algorithm needs to initialize the new particles. However, in the calculation of the minimum value for R, it is evident that the minimum point of function is equal to B, which is the same as the number of posts in the weblog. Therefore, no matter how optimized our algorithm, one-time initialization of the whole particles is still required.

### C. *Prediction of Users' Trend (TRPT)*

After detection of the blogosphere trends based on the enhanced PSO algorithm in the first phase, these data are fed into a Q-Learning algorithm to perform the blogosphere trend prediction. The Q-Learning algorithm is a common machine learning algorithm introduced in subsection B of section III. Since the velocity vector error is somewhat associated with trend detection error, the reward values and Q-Learning algorithm's errors are determined based on the PSO velocity vector.

At each step of reward and error calculations (equation 3), the table of PSO velocity vectors is referred, and the desired action is performed on the Q-Learning algorithm according to the same vector. Fig. 4 shows the TRPT phases. The proposed algorithm has another advantage in the detection of the most influential authors in determining the blogosphere trend. Given the fact that the trends detected in the blogosphere have been determined based on a series of posts that have been published by the authors, those who have published these posts can be ranked. These results will be discussed in detail in section V.

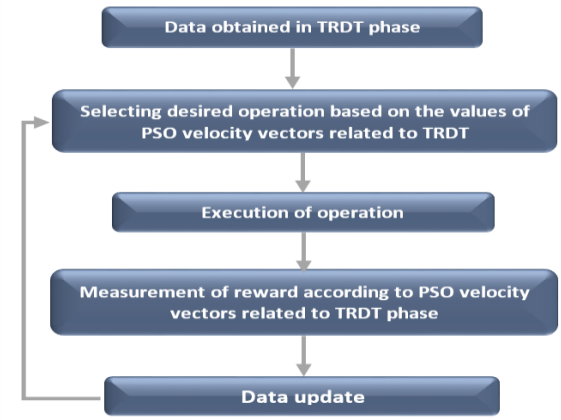


Figure 4. The future trend of users predicted by the Q-Learning algorithm (TRPT).

## V. Experimental Results and Evaluations

In this section, simulation results and evaluations are presented for TRDT, and TRPT approaches. At first, a summary of the hardware configuration of the system, and the related software setup and parameters are presented. Then, the dataset, together with the related fields which have been used in the model, are explained. In the next subsections, both TRDT and TRPT approaches are evaluated, and the simulation results are presented in different scenarios.

### A. *Hardware Configuration*

The required hardware should be provided in such a way that it can perform the Chi-squared test method in addition to executing both TRDT and TRPT. The Chi-squared test is employed for validation of the TRDT's results in this paper. However, considering the Chi-squared test is not in the distributed form in the standard mode, the deployment features of these two

methods are different in practice. Therefore, the processing and storage resources available for Chi-squared test and TRDT have been similarly deployed and configured for validation of the comparison. The minimum and maximum hardware specifications supporting the Chi-squared test and TRDT are shown in Table II.

TABLE II. PROFILE OF REQUIRED HARDWARE TO RUN THE CHI-SQUARED TEST AND TRDT/TRPT

| Hardware | Chi-Squared Test | TRDT |
|---|---|---|
| **CPU** | 1 node, 16 cores | 1node – 1 core / 4 nodes – 4 cores |
| **RAM** | 1 – 16 GB | 1 – 16 GB |
| **H.D.D** | 10 – 400 GB | 10 – 400 GB |

### B. Software Configuration

Before running the experiments, the required parameters of the TRDT algorithm should be determined. It should be noted that to run the TRDT algorithm, JSwarm software library was used, which requires the $r_1$ and $r_2$ (random parameters, as explained in equation1) to set up a group of particles to search in the problem state space. Therefore, to align the traditional PSO algorithm as the basis of the approach used in TRDT, the values of these parameters should be calculated once. For this purpose, the results expected from a simple example were fed into the PSO algorithm, and parameters were calculated for different values (Table III).

TABLE III. DIFFERENT VALUES FOR PSO PARAMETERS CALCULATION

| Particle dimension | Iteration number for each run | Particle population | Random parameters | |
|---|---|---|---|---|
| | | | $r_1$ | $r_2$ |
| **2** | 400 | 25 | 2.5586 | 1.3358 |
| **2** | 400 | 30 | -0.6504 | 2.2073 |
| **2** | 4000 | 150 | 2.1304 | 1.0575 |
| **2** | 4000 | 250 | 0.4862 | 2.5067 |
| **5** | 1000 | 60 | -0.7238 | 2.0289 |
| **5** | 1000 | 50 | 0.5287 | 3.1913 |
| **10** | 2000 | 60 | 1.6319 | 0.6239 |
| **10** | 2000 | 200 | -0.3344 | 2.3259 |
| **10** | 20000 | 50 | -0.2746 | 4.8976 |
| **20** | 40000 | 70 | -0.2699 | 3.395 |

The calculated values of PSO parameters lie in the specific ranges. Fig. 5 shows the cumulative points of these values. These cumulative points indicate that the best results of PSO are achieved when $r_1$ and $r_2$ values are in the specific ranges. So, we have used these values in experiments.

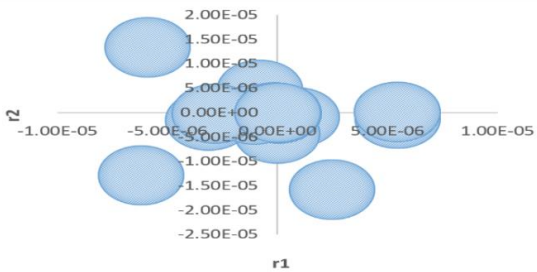


Figure 5. Cumulative distribution of $r_1$ and $r_2$ parameters.

### C. Dataset

The dataset employed in this paper includes the posts published on The Verge database (www.theverge.com) from October 8, 2013, to November 8, 2013, which is specifically dedicated to review and critique the mobile phones. The Verge is an American technology news and media network operated by Vox Media. Like a social network, this database involves active discussions among users, which provides a suitable forum for analysis of social behavior and sentiment analysis that will be effective in discovering the general trend. Each blog post has a title, an author, a text, and a few comments. On each blog post, the post's author can add comments as well as those who are the members of the Verge network. As shown in Fig. 6, the relationship of the author with blog posts is one-to-many, and the visitor to a blog post is many-to-many. After publication, the blog posts enter into the trend detection system, and the data related to the author are extracted from the post. Each processing node saves a mapping of authors' name to their ranking in RAM, which is updated in each data entry. The rank of the authors is important since more popular authors have a greater impact on the general trend.

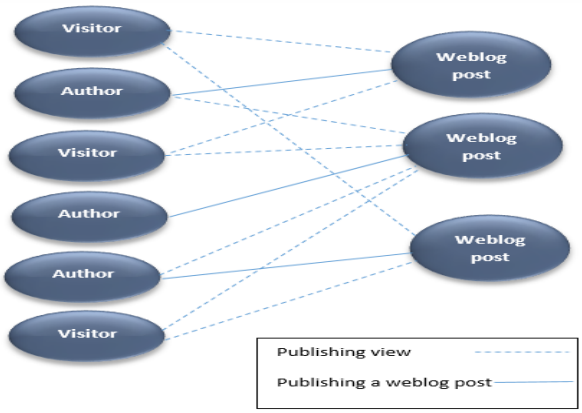


Figure 6. Relationship between users and blog posts.

Table I investigates the effect of each feature in the dataset on the proposed model in subsection A of section IV. The author of each blog post is extracted and ranked by a simple procedure using equation (9). The rank value is a decimal number between 1 (initial value) and 2, which will be higher if it gets closer to number 2. In equation (9), the repeat parameter represents the number of posts the author has published on the blog.

$$Rank = \text{Rank} + \frac{repeat}{(repeat+1)} \quad (9)$$

This ranking affects the PSO algorithm in TRDT, and thus the posts published by influential authors have a greater impact on determining the blogosphere trend.

### D. TRDT Evaluations

In Fig. 7, the total velocity curve of all particles is calculated, which shows that the position of particles is not changing during the whole period of the blogosphere trend calculation. This means that some posts had no effect on trend change in the blogosphere. However, the interesting thing about this diagram is that in 1500 to 2000 iteration, a sudden sharp change has occurred in the blogosphere trend, after which the

position change has become zero and turned into particle stability mode.

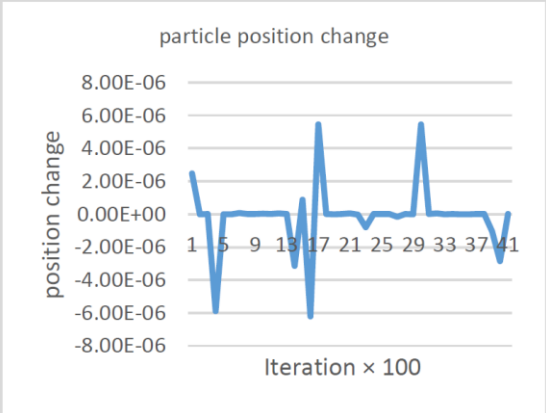


Figure 7. Particle position changes.

Fig. 8 shows the trend detection error in TRDT based on the particles' movement. The results indicate that particle movement is first scattered and far from the trend of the blogosphere but is gradually converged to the desired result.

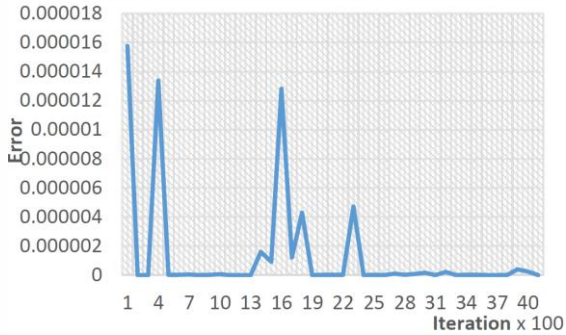


Figure 8. Trend detection error in TRDT.

Fig. 9 compares the trend detection error with the velocity vector error. In any change in the position of particles in TRDT, the velocity vector shows the direction of particles' movement towards a new trend. As shown in Fig. 9, the velocity vector error is related to the trend detection error. Therefore, obtaining velocity vector error at long time intervals can converge the blogosphere errors with a higher rate. Whenever two trends are repeated simultaneously in more posts, their distance decreases.


Figure 9. Comparison of trend detection error with velocity vector error in TRDT.

Moreover, we consider a rank for each trend, which is the fitness value in the TRDT algorithm. This parameter has been illustrated by size and color in Fig. 10 (a-f), i.e., the higher the trend rank, the larger its size and color. As shown in these figures, first, the number of detected trends is low (Fig. 10-a), and in Fig. 10-b, the number of discovered trends is increased, but their ranking relative to each other is reduced. However, the trends with higher rankings are gradually identified, and it is interesting to note that the distance between the top trends is reduced over time, which indicates that the blogosphere trends in each period are placed together in an ontology. Therefore, if we look at this from another perspective when a trend becomes popular in the blogosphere, the popularity of the related trends (which are on the same ontology) are likely to increase.

### *E. Comparison of TRDT with Chi-squared Test*

The comparison results show that the proposed approach has a lower accuracy than the Chi-squared test in terms of the posts without comments.

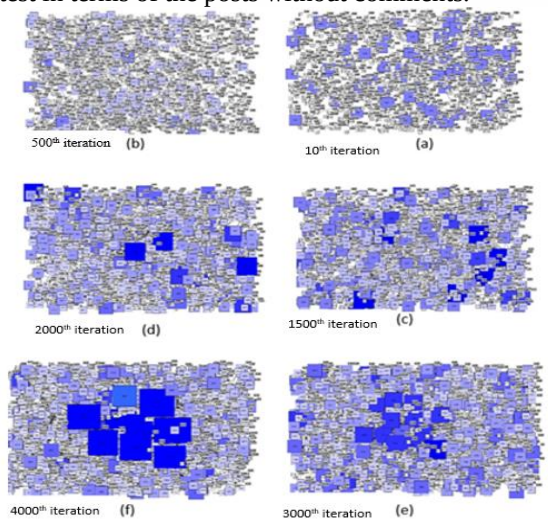


Figure 10. Illustration of TRDT behavior.

However, TRDT provides better accuracy for the posts that include users' comments (Table IV). In fact, the posts that have attracted users' feedback have also increased the accuracy of TRDT. The chi-squared test is primarily used to determine whether a sample of data has been derived from a sample space with a specific distribution. The chi-squared test can be applied to categorized data (for example, data that have been placed in different classes) that it is very similar to the system here to categorize the posts. Moreover, this approach can be employed on discrete distributions such as Binary and Poisson distributions. A chi-squared test was chosen due to its low dependency on the relationship between information content and the features of a blog's post. That is why it has worked well on the posts without comments.

TABLE IV. PRECISION AND RECALL OF TRDT AND THE CHI-SQUARED

| Weblog features | Chi-squared test | | TRDT | |
|---|---|---|---|---|
| | Precision | Recall | Precision | Recall |
| Posts without comments | 85.45 | 87.43 | 84. 52 | 86.47 |

| | | | | |
|---|---|---|---|---|
| **Posts with neg. or pos. comments** | 82.08 | 80.16 | 86.79 | 88.82 |

In addition, the efficiency (operations per second) of the chi-squared test approach is much lower than TRDT, as shown in Table V. It is interesting to note that increasing the number of CPUs has increased the difference between the efficiency of the proposed approach and the Chi-squared test. In this scenario, the Chi-squared test is executing on a single node (from 1 core to 16 cores) while TRDT is running in a distributed platform (from 1 node/1 core to 4 nodes/4 cores). As shown in below Table, even the interconnection latency between different nodes in the distributed execution of TRDT has been increased, the total execution time is still better than the chi-squared test. In fact, Chi-squared test is executed in a centralized server while TRDT is deployed on a cluster of 4 computers with some network latencies (the number of cores is the same in each configuration). Considering all these conditions, which is in favor of the Chi-squared test method, it can be seen from Table V, last column, that the results of the TRDT method are better.

TABLE V. EXECUTION TIME IN TRDT AND THE CHI-SQUARED

| CPU Cores | Chi-squared test Exe. Time (sec) | TRDT Exe. Time (sec) | Improvement factor |
|---|---|---|---|
| **1** | 1256 | 546 | 2.3 |
| **2** | 844 | 359 | 2.35 |
| **4** | 511 | 208 | 2.46 |
| **6** | 358 | 136 | 2.64 |
| **8** | 226 | 81 | 2.78 |
| **10** | 147 | 52 | 2.85 |
| **12** | 116 | 40 | 2.9 |
| **16** | 83 | 29 | 2.9 |

Fig. 11 shows the chi-squared test and TRDT execution time. Also, the improvement of execution time in TRDT related to the chi-squared test has been depicted.

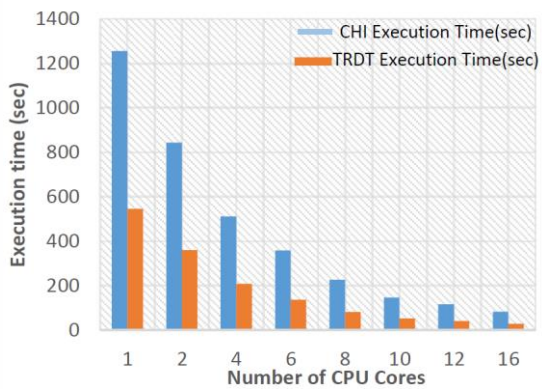


Figure 11. The improvement of execution time in TRDT related to the chi-squared test.

## F. TRPT Evaluations

As previously explained, the reward and error values of the Q-Learning algorithm are based on the velocity vector in PSO. In fact, the PSO velocity vector table is employed to calculate the Q-learning's reward and error values in each iteration of TRDT (Equation 3). This table is the same as the action/value table in the Q-Learning algorithm, as discussed in subsection B of section III. The desired action in the Q-learning algorithm is selected and performed based on this velocity vector. Results of one thousand iterations of TRPT are given in Fig. 12.

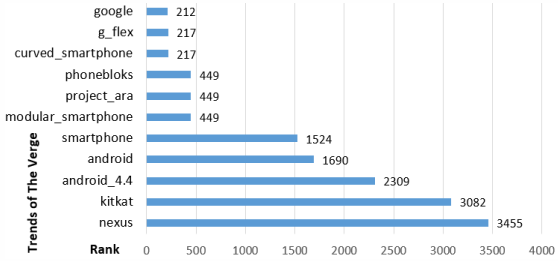


Figure 12. The TRPT predicted trends for 1000 iterations.

For more assessment of the TRPT predicted trends, we have implemented the approach presented by Kumar and Jaiswal in [37] using our dataset extracted from Verge. The predicted trends for 1000 iterations have been depicted in Fig. 13.

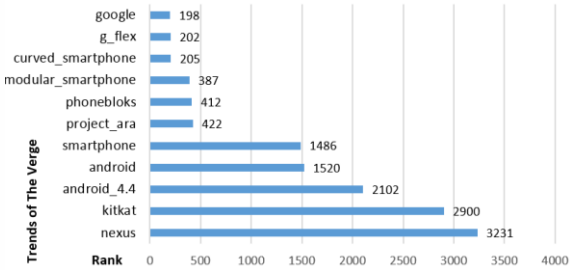


Figure 13. Kumar and Jaiswal [37] predicted trends for 1000 iterations.

It can be concluded from Fig. 12 and Fig. 13 that TRPT has better prediction results of different trends in the Verge dataset compared with [37].

Moreover, the content of The Verge website was employed to evaluate the overall prediction accuracy of TRPT. This evaluation has been done by a small program for extracting and storing the trends announced by The Verge on the first page in an interval overlapping with our dataset interval. Therefore, it can be distinguished that how much the results of TRPT conform (meet) the real-world trends. Considering TRPT feeds the weblog posts into the algorithm in order of occurrence, if the TRPT predicts a trend, the same trend likely should be announced by The Verge website in a future period. Because Q-learning should search all the states of the problem to be able to achieve the best results, we can see in Fig. 14 that the completion of the learning process reduces the error.

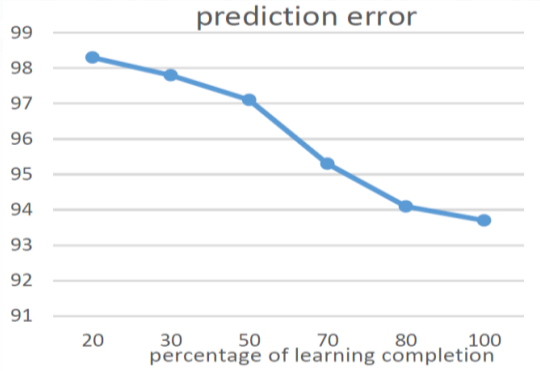


Figure 14. Prediction error in TRPT.

However, there is still a high error rate in the prediction of TRPT, even in the best case (89.95%) due to the high level of noise in data. Generally, the prediction of natural events with a Poisson-like distribution with machine learning algorithms causes a high error rate [45]. If the high error rate is because of noise in the data, we can achieve a higher level of accuracy by increasing the training period, normalizing the data samples, and increasing the number of samples. Now, the important point is whether more precision is always required? Real-world examples give different answers to this question. The Netflix website, an accredited provider of video streaming, has been organizing a competition every year to develop an algorithm to accurately predict future user trends. As shown in Table VI, the best result in these competitions has been 88.70% prediction error by BellKor [46, 47], which is 89.95% in this paper for the TRPT approach. Therefore, the performance of TRPT is comparable to the Netflix prize-winning algorithm.

TABLE VI. PREDICSTION ERROR RATE OF TRPT AND THE BEST METHOD IN NETFLIX COMPETITION

| Approach | Prediction error rate |
|---|---|
| The best method in Netflix competition | 88.70% |
| TRPT | 89.95% |

## VI. CONCLUSION AND FUTURE WORKS

In this paper, a model has been proposed to detect the general blogosphere trend in an acceptable period of time with reasonable accuracy based on the exchanged messages, which also provides the platform for the prediction of future blogosphere trends. In the TRDT phase of this approach, after analysis, the blogosphere posts and extraction of the required dataset, the trend of users is detected based on the sentiment analysis of their published posts and employing the improved PSO algorithm. Then, in the TRPT phase, the trend of users in the blogosphere is predicted, and the most influential authors in determining the blogosphere trend are specified using the Q-Learning algorithm based on the data obtained from the TRDT phase. The results obtained from the proposed model mention that in addition to the detection/prediction of user behavior with the required precision, the scalability feature is achieved. Moreover, employing a non-deterministic solution such as PSO rather than deterministic algorithms like Chi-squared test (that assess the whole space of the state) improves the response time of the trend detection model. For optimal use of all system resources, the processing load of the algorithm has been divided on different resources in the existing distributed system.

For future works, we would like to broadcast the overall processing results to processing nodes in the aggregator. Therefore, the convergence of computing nodes will happen more quickly. The CAQs can also be designed in such a way to serve multiple processes for better utilization of system resources and increasing the overall performance. Additionally, we are also interested in using a directed graph describing the trends and their relationships. Due to the fact that some directed graph traversal is faster than normal graphs, a number of TRPT operations will be performed faster.

## REFERENCES

[1] A. Goswami and A. Kumar,"A survey of event detection techniques in online social networks", Social Network Analysis and Mining, vol. 6, no.1, pp.1-41, 2016.

[2] Twitterinc.[online]https://investor.twitterinc.com/home/default.aspx (Accessed Nov 2020)

[3] H. Ullah Khan, A. Daud, U. Ishfaq, T. Amjad, N. Aljohani, R. A. Abbasi and J. S. Alowibdi, "Modelling to identify influential bloggers in the blogosphere: A survey", *Computers in Human Behavior*, vol. 68, pp.64-82, 2017.

[4] P. N. E. Nohuddin, R. Christley, F. Coenen, Y. Patel, C. Setzkorn and S. Williams, "Social Network Trend Analysis Using Frequent Pattern Mining and Self Organizing Maps". In: Bramer M. and Petridis M.,Hopgood A. (eds) Research and Development in Intelligent Systems XXVII. pp.311-324, 2010, Springer, London.

[5] P. Nohuddin, F. Coenen and R. Christley, "The application of social network mining to cattle movement analysis: introducing the predictive trend mining framework", Social Network Analysis and Mining, vol. 6, no. 1, pp.1-17, 2016.

[6] R. Venant, K. Sharma, P. Vidal, P. Dillenbourg and J. Broisin, "Using Sequential Pattern Mining to Explore Learners" Behaviors and Evaluate Their Correlation with Performance in Inquiry-Based Learning', Proceeding of 12th European Conference on Technology Enhanced Learning, pp.286-299, 2017, Tallinn, Estonia.

[7] H. J. Choi and C. H. Park, "Emerging topic detection in twitter stream based on high utility pattern mining", Expert Systems with Applications, 2019, vol. 115, pp.27-36, 2019.

[8] K. Kawabata, Y. Matsubara and Y. Sakurai, "Automatic Sequential Pattern Mining in Data Streams. In Proceedings of the 28th ACM International Conference on Information and Knowledge Management (CIKM '19). Association for Computing Machinery, New York, NY, USA, pp. 1733–1742, 2019.

[9] I. Batal, G.F. Cooper, D. Fradkin, J. Harrison, F. Moerchen and M. Hauskrecht, "An efficient pattern mining approach for event detection in multivariate temporal data", Knowledge and Information Systems, vol. 46, no.1, pp.115–150, 2016.

[10] G. Arthur Van, F. Staals, M. Löffler, J. Dykes and B. Speckmann, "Multi-Granular Trend Detection for Time-Series Analysis", IEEE Transactions on Visualization and Computer Graphics, Vol. 23 No. 1, pp. 661-670, 2017.

[11] L. Anghinoni, L. Zhao, Q. Zheng and J. Zhang, "Time series trend detection and forecasting using complex network topology analysis". Neural Networks, vol.117, pp.295-306, 2019.

[12] P. Jadeja and K. Kotecha, "Forecasting the behaviour of Trending Terms in Microblogs", 2018 IEEE Punecon, Pune, India, pp. 1-6, 2018.

[13] A. Dey, M. Jenamani and J. J. Thakkar, "Senti-N-Gram: An n-gram lexicon for sentiment analysis", Expert Systems with Applications, vol. 103, pp.92-105, 2018.

[14] M. Taboada, J. Brooke, M. Tofiloski, K. Voll and M. Stede, "Lexicon-Based Methods for Sentiment Analysis", Association for Computational Linguistics, vol. 37, no. 2, pp.267–307, 2011.

[15] Z. Xiaomei, Y. Jing and Z. Jianpei, "Sentiment-based and hashtag-based Chinese online bursty event detection", Multimedia Tools and Applications, vol. 77, no. 16, pp.21725-21750, 2018.

[16] C. Hung and S. Jeng Chen, "Word sense disambiguation based sentiment lexicons for sentiment classification", Knowledge-Based Systems, vol. 110, pp. 224-232, 2016.

[17] W. Yang, D. Li and F. Liang, "Sina Weibo Bursty Event Detection Method," in IEEE Access, vol. 7, pp. 163160-163171, 2019.

[18] E. Tattershall, G. Nenadic and R. D. Stevens, "Detecting bursty terms in computer science research". Scientometrics 122, pp. 681–699, 2020.

[19] C.J. Hutto and E. Gilbert, "VADER: A parsimonious rule-based model for sentiment analysis of social media text", Proceeding of the 8th International Conference on Weblogs and Social Media, pp. 216-225, 2014, Michigan, USA.

[20] A. Romanowski and M. Skuza, "Towards Predicting Stock Price Moves with Aid of Sentiment Analysis of Twitter Social Network Data and Big Data Processing Environment", In Pełech-Pilichowski, T., Mach-Król, M., Olszak, C. (Eds.): 'Advances in Business ICT: New Ideas from Ongoing Research', pp.105-123, 2017, Springer, Cham.

[21] J. Qiu, C. Liu, Y. Li and L. Zhangxi, "Leveraging sentiment analysis at the aspects level to predict ratings of reviews", Information Sciences, vol. 451–452, pp.295-309, 2018.

[22] R. Iyer, R. Zheng, Y. Li and K. P. Sycara, "Event Outcome Prediction using Sentiment Analysis and Crowd Wisdom in Microblog Feeds" CoRR, abs/1912.05066 (2019)

[23] L. L. Shi, L. Liu, Y. Wu, L. Jiang, and A. Ayorinde, "Event Detection and Multi-source Propagation for Online Social Network Management", J Netw Syst Manage 28, 1–20, 2020.

[24] M. Ali, L. Lu and M. Farid, "Detecting present events to predict future: detection and evolution of events on Twitter", In: 2018 IEEE Symposium on Service-Oriented System Engineering (SOSE). IEEE . 2018.

[25] T. Gui, P. Liu, Q. Zhang, L. Zhu, M. Peng, Y. Zhou and X. Huang. "Mention Recommendation in Twitter with Cooperative Multi-Agent Reinforcement Learning", In SIGIR. ACM, 535–544, 2019.

[26] X. Zhang, X. Chen, Y. Chen, S. Wang, Z. Li, and J. X. Xia, "Event detection and popularity prediction in microblogging", Neurocomputing, vol. 149, 1469–1480, 2019.

[27] M. P. Salas-Zárate, J. Medina-Moreira, P. J. Álvarez-Sagubay K. Lagos-Ortiz, M. A. Paredes-Valverde, R. Valencia-García "Sentiment Analysis and Trend Detection in Twitter". In: Valencia-García R., Lagos-Ortiz K., Alcaraz-Mármol G., del Cioppo J., Vera-Lucio N. (eds) Technologies and Innovation. CITI 2016. Communications in Computer and Information Science, vol. 658. pp.63-76, 2016, Springer, Cham.

[28] S.Y. Yoo, J. Song and O.R. Jeong, "Social media contents based sentiment analysis and prediction system", Expert Systems with Applications, 2018, vol. 105, pp.102–111, 2018.

[29] F. HassanKhan, U. Qamar and S. Bashir, "Lexicon based semantic detection of sentiments using expected likelihood estimate smoothed odds ratio", Artificial Intelligence Review, vol. 48, no. 1, pp.113-138, 2017.

[30] A. Dridi and D. Reforgiato Recupero, "Leveraging semantics for sentiment polarity detection in social media. International journal of machine learning and cybernetics". vol. 10, pp. 2045–2055, 2019.

[31] Y. Liu, H. Peng, J. Li, Y. Song and X. Li, "Event detection and evolution in multi-lingual social streams". Frontiers of Computer Science. vol. 14, 145612, 2020.

[32] S. Banerjee and N. Agarwal, "Analyzing collective behavior from blogs using swarm", Knowledge and Information Systems, vol. 33, no. 3, pp.523–547, 2012.

[33] Z. Lv, F. Shen, J. Zhao and T. Zhu, "A Swarm Intelligence Algorithm Inspired by Twitte". In: Hirose A., Ozawa S., Doya K., Ikeda K., Lee M., Liu D. (eds) Neural Information Processing. ICONIP 2016. Lecture Notes in Computer Science, vol. 9949, pp.344-351, 2016, Springer, Cham.

[34] W. Hu, H. Wang, Z. Qiu and T. Zhu, "An event detection method for social networks based on hybrid link prediction and quantum swarm intelligent", World Wide Web Journal, vol. 20, no. 4, pp.775–795, 2017.

[35] K. Orkphol and W. Yang, "Sentiment Analysis on Microblogging with K-Means Clustering and Artificial Bee Colony", International Journal of Computational Intelligence and Applications, vol. 18, no. 03, 1950017, 2019.

[36] A. Albert, L. F. López and N. GómezBlas, "Multilinear Weighted Regression (MWE) with Neural Networks for trend prediction", Applied Soft Computing, vol. 82, 105555, 2019.

[37] A. Kumar and A. Jaiswal, "Swarm intelligence based optimal feature selection for enhanced predictive sentiment accuracy on twitter". Multimedia Tools and Applications, vol. 78, pp. 29529–29553, 2019.

[38] A. Pandey, R. Dharmveer Singh and M. Saraswat, "Twitter sentiment analysis using hybrid Cuckoo search method", Information Processing & Management, vol. 53, no. 4, pp. 764–779, 2017.

[39] P. Silambarasi and K. L. N. Eranki, "Performance Evaluation of Meta-Heuristic Algorithms in Social Media Using Twitter", In: Khanna A., Gupta D., Bhattacharyya S., Snasel V., Platos J., Hassanien A. (eds) International Conference on Innovative Computing and Communications. Advances in Intelligent Systems and Computing, vol 1087. Springer, Singapore, 559-567, 2020.

[40] A. A. Sayed, M. M. Abdallah, A. M. Zaki, and A. A. Ahmed, "Big Data analysis using a metaheuristic algorithm: Twitter as Case Study," in IEEE International Conference on Innovative Trends in Communication and Computer Engineering (ITCE), pp. 20-26, 2020.

[41] M. A. Salam, V. Shukla and A. Pandey, " Enhancing Sentiment Analysis using Enhanced Whale Optimization Algorithm", International Journal of Intelligent Information and Database Systems, vol 3, Issues 2-4, 208-230, 2020.

[42] M. Clerc, "Particle Swarm Optimization", Wiley, ISTE (International Scientific and Technical Encyclopedia), 2006.

[43] A. L. Strehl, L. Li, E. Wiewiora, J. Langford and L. Michael, "Pac model-free reinforcement learning". In Proceedings of the 23rd international conference on Machine learning (ICML '06). ACM, New York, USA, pp.881-888, 2006.

[44] S. Das and A. Abraham, "Pattern clustering using a swarm intelligence approach", In Maimon, O., Rokach, L. (Eds.): 'Data Mining and Knowledge Discovery Handbook', Springer, Boston, MA, pp.469-504, 2009.

[45] S. Ghahramani, "Fundamentals of Probability: with Stochastic Processes", 3th edition, Chapman and Hall/CRC, 2015.

[46] Y. Koren, "The BellKor Solution to the Netflix Grand Prize", Netflix prize documentation 81, pp.1-10, 2009.

[47] A. Toscher and M. Jahrer, "The BigChaos Solution to the Netflix Grand Prize", Netflix prize documentation, pp.1-52, 2009.

**Rezvan Mohamadrezaei** received the B.Sc. and M.Sc. degrees in Computer Engineering. She is currently pursuing the Ph.D. degree at Department of Computer Engineering, Central Tehran Branch, Islamic Azad University. Her current research interests include social data mining, recommender system and deep learning.

**Reza Ravanmehr** graduated in Computer Engineering from Shahid Beheshti University, Tehran in 1996. After that, he gained M.Sc. and Ph.D., both in Computer Engineering from Islamic Azad University, Science and Research Branch, Tehran in 1999 and 2004, respectively. His main research interests are distributed/parallel systems, large scale data management systems, and social network analysis. He is the faculty member of Computer Engineering Department at Central Tehran Branch, Islamic Azad University, from 2001.