International Journal of *Geo-Information*

Article

Hessian Distributed Ant Optimized Perron–Frobenius Eigen Centrality for Social Networks

P.V. Kumaraguru [1], Vidyavathi Kamalakkannan [2], Gururaj H L [3], Francesco Flammini [4,*], Badria Sulaiman Alfurhood [5] and Rajesh Natarajan [6]

1 Department of MCA, Guru Nanak College (Autonomous), Velachery Main Road, Velachery, Chennai 600042, Tamilnadu, India; coe@gurunanakcollege.edu.in
2 Department of Electronics and Communication Engineering, Selvam College of Technology, Namkkal 637003, Tamilnadu, India; vidyavathiece.2010@gmail.com
3 Department of Information Technology, Manipal Institute of Technology Bengaluru, Manipal Academy of Higher Education, Manipal 570064, Karnataka, India; gururaj.hl@manipal.edu
4 IDSIA USI-SUPSI, University of Applied Sciences and Arts of Southern Switzerland, 6928 Manno, Switzerland
5 Department of Computer Sciences, College of Computer and Information Sciences, Princess Nourah bint Abdulrahman University, Riyadh 11671, Saudi Arabia; bsalfurhood@pnu.edu.sa
6 Information Technology Department, University of Technology and Applied Sciences-Shinas, Al-Aqr, Shinas 324, Oman; rajesh.natarajan@shct.edu.om
* Correspondence: francesco.flammini@supsi.ch

Abstract: Terabytes of data are now being handled by an increasing number of apps, and rapid user decision-making is hampered by data analysis. At the same time, there is a rise in interest in big data analysis for social networks at the moment. Thus, adopting distributed multi-agent-based technology in an optimum way is one of the solutions to effective big data analysis for social networks. Studying the development of a social network helps users gain an understanding of interactions and relationships and guides them in making decisions. In this study, a method called Hessian Distributed Ant Optimized and Perron–Frobenius Eigen Centrality (HDAO-PFEC) is developed to analyze large amounts of data (i.e., Big Data) in a computationally accurate and efficient manner. Designing an adaptable Multi-Agent System architecture for large data analysis is the primary goal of HDAO-PFEC. Initially, using a Hessian Mutual Distributed Ant Optimization MapReduce model, comparable user interest tweets are produced in a computationally efficient manner. Eigen Vector Centrality is a measure of a node's importance in a network (i.e., a social network), which allows association with other significant nodes (i.e., users), allowing for a greater effect on social networks. With this goal in mind, a MapReduce methodology in the Hadoop platform using Big Data, which enables quick and ordered calculations, is used in a distributed computing method to estimate the Eigen Vector Centrality value for each social network member. Lastly, extensive investigative experimental learning demonstrates the HDAO-PFEC method's use and accuracy as well as its time and overhead on the well-known sentiment 140 dataset.

Keywords: big data; Mutual Distributed; multi-AGENT; Hessian Optimization of Ant Frobenius; Perron; Eigen Vector Centrality

check for updates

Citation: Kumaraguru, P.V.; Kamalakkannan, V.; H L, G.; Flammini, F.; Sulaiman Alfurhood, B.; Natarajan, R. Hessian Distributed Ant Optimized Perron–Frobenius Eigen Centrality for Social Networks. *ISPRS Int. J. Geo-Inf.* **2023**, *12*, 316. https://doi.org/10.3390/ijgi12080316

Academic Editors: Wolfgang Kainz, Hangbin Wu and Tessio Novack

Received: 8 May 2023
Revised: 13 July 2023
Accepted: 26 July 2023
Published: 1 August 2023

1. Introduction

There has been interest in tackling optimization problems in a distributed model due to the popularity of multi-agent systems. By doing this, agents may only access information in a limited way and interact with neighbors, making it suitable for large data applications involving intensive computing and intricate network architecture, such as standard assessment and other similar tasks. In order to obtain encouraging results, different works have been explored in terms of error rate, optimality backdrops, overhead, and various optimization approaches. These investigations were motivated by optimization methods and contemporary works on distributed optimization.

Laplace three-level stochastic variational inference, often known as truth-finding based on views gathered from several agents constituting a social network, was introduced in [1]. In this study, the agents' dependabilities were connected with a group of people who had similar attitudes about the same incident. Also, it was claimed that the agents had access to several communities for a variety of activities, despite the fact that these communities were not known in advance.

Laplace variational inference techniques were used in the truth table to draw conclusions about the agents' social networks, which in turn assessed the reliabilities of the agents. For vast social networks, a stochastic variation inference model was created, adding accuracy with little error. The created approach only examines the confusion matrix and event states for the same community or switch communities. The Hessian Mutual Distributed Ant Optimization model is created in this work to solve this problem and helps achieve the goal with the use of Mutual Weights and the Hessian Matrix.

Multi-agent-based distributed architecture, a methodology to assess the best multi-agent architecture for collaborative processing for efficient machine processing, was introduced in [2]. Equipment breakdowns were effectively anticipated using multi-agent systems thanks to the possibility of collaborative learning.

Also, numerous multi-agent system designs were used to examine the cost and reliability variables, resulting in minimal error through the use of improved maintenance practices. Yet, the precision lacked emphasis. Perron–Frobenius Eigen Vector Centrality, which is created in this study to solve this problem, rapidly obtains the dimensionality-reduced tweets, processing enormous amounts of data (i.e., Big Data), utilizing a multi-agent based model, and improving accuracy rate.

The method in this paper uses a cutting-edge agent-based distributed system, such as Hessian Distributed Ant Optimized and Perron–Frobenius Eigen Centrality, for acquiring similar users' interests via tweets in social networks, as opposed to related works in [1,2], which use a standard truth table and collaborative learning method.

The main contributions of this paper are summarized as follows.

(1) To handle the stochastic variational inference problem, a Hessian Distributed Ant Optimized model is suggested, requiring just local Twitter assessments using combine vectors rather than explicit inference, making it ideal for applications requiring fast processing.
(2) The suggested technique employs a time-changing Hessian matrix, defining the algorithm suitable to use for digraphs, in contrast to most distributed optimization methods where the confusion matrix has been employed (e.g., [1,2]).
(3) Finally, by applying Perron–Frobenius Eigen Vector Centrality, dimensionality-reduced tweets are arrived at by means of the centrality factor, which uses less memory resources for data storage and processing in real-time application.

The remainder of the essay is structured as follows. The reviews that were performed are further described in Section 2. The suggested Hessian Distributed Ant Optimized and Perron–Frobenius Eigen Centrality technique is presented in Section 3. The effectiveness of the algorithms is illustrated in Section 4 through tables and graphics, and the findings are presented in Section 5.

2. Related Works

In order to solve a multi-agent optimization issue with specified constraints, a distributed optimization technique utilizing a randomized gradient-free model was given in [3]. In [4], a system based on the anytime property was presented to enhance distributed local search algorithms for distributed constraint optimization problems. The framework made use of a breadth-first search tree, which distributed the detection of an updated state as it occurred and added up the cost of the system state.

In decagons, where agents are required to achieve concurrence on some key variables through local information exchange, the consensus control problems of multi-agent systems have captured deep interest. Input, consensus state, and dual optimization, three optimization issues that helped the multi-agent system reach the best consensus, were given in [5]. From a multi-agent perspective, an overview of several diffusion methods used in social networks was put out in [6].

Convex value minimization over the fixed value points set is a unique optimization model that was developed in [7]. In addition, a discrete-time technique for finding the worldwide solution to the specified optimization problem was also described. In order to find a unique optimal solution, a distributed optimization technique utilizing first-order and second-order dynamic agents was suggested in [8]. A strategy for learning multi-agent strategies for competing against collective competitors was put forward in [9]. Moreover, it was claimed that deterministic policy gradients in recurrent neural networks allowed for agent collaboration during communication.

The idea and notation of distributed optimization have recently drawn a lot of interest from several walks of life. Owing to this, consensus-based flocking, distributed optimization, and other topics have become increasingly popular in the research sector. In [10], a distributed optimization model with a flocking component of a nonlinear continuous function and a differentiable convex objective function was examined. In order to achieve sophisticated cost optimization, Ref. [11] suggested a fusion of deep reinforcement learning with link-state protocol providing preliminary supervised learning.

Several disciplines of study have consistently used multi-agent systems. A decomposition technique for multi-project scheduling that is based on two stages and has a suitable CPU running time for large-size instances was devised in [12]. To identify deliberate activities on highways, a multi-agent system for smart roads was created in [13] utilizing Model Driven Engineering (MDE). For decentralized multi-agent learning addressing high dimensions, a traditional rewarding system in gaming notion utilizing a Deep Q-learning framework was presented in [14].

A collaborative learning-based industrial multi-agent system for large fleets was reported in [2]. The system's prediction and optimization mechanisms considerably reduced the cost of maintaining an asset over its full life. A multi-agent simulation framework for complicated urban transportation was proposed in [15] with the aim of lowering the overall journey duration and smart car adoption level. Another design for the distribution system in the electric power model was created in [16] utilizing an artificial neural network in order to obtain operational restrictions and decrease the incidence of faults.

Target search is one of the common research issues in the multi-agent field. Target discovery may need to be conducted in some real-time situations together with some sub-task analysis in order to meet a deadline. These temporal limits were highlighted in [17] as the agents recognizing targets within a predetermined window of time. Yet, the decision-making system was unable to handle this more complicated issue directly.

Distributed optimization is used in a distributed protocol for a network connection that was introduced in [18]. To reduce the expense associated with node-based designs, an adaptive method with linear dynamics based on multi-agent systems was developed in [19].

In this study, an agent-based distributed approach dubbed Hessian Distributed Ant Optimized and Perron–Frobenius Eigen Centrality (HDAO-PFEC) is developed for accurately and efficiently assessing sentiments in social networks with little operating time and data storage cost.

3. Perron–Frobenius Eigen Centrality and Hessian Distributed Ant Optimization (HDAO-PFEC)

This part uses the Sentiment140 dataset, which contains 1.6 million tweets, to analyze those tweets in order to determine user interest in a distributed and MapReduce-optimized way (i.e., Twitter agent). Hessian Distributed Ant Optimization and Perron–Frobenius Eigen Centrality are the names of the technique (HDAO-PFEC). The goal of investigating adaptive multi-agent systems with intelligence and context learning capabilities is what HDAO-PFEC is meant to do. Figure 1 below shows the HDAO-PFEC method's block diagram.

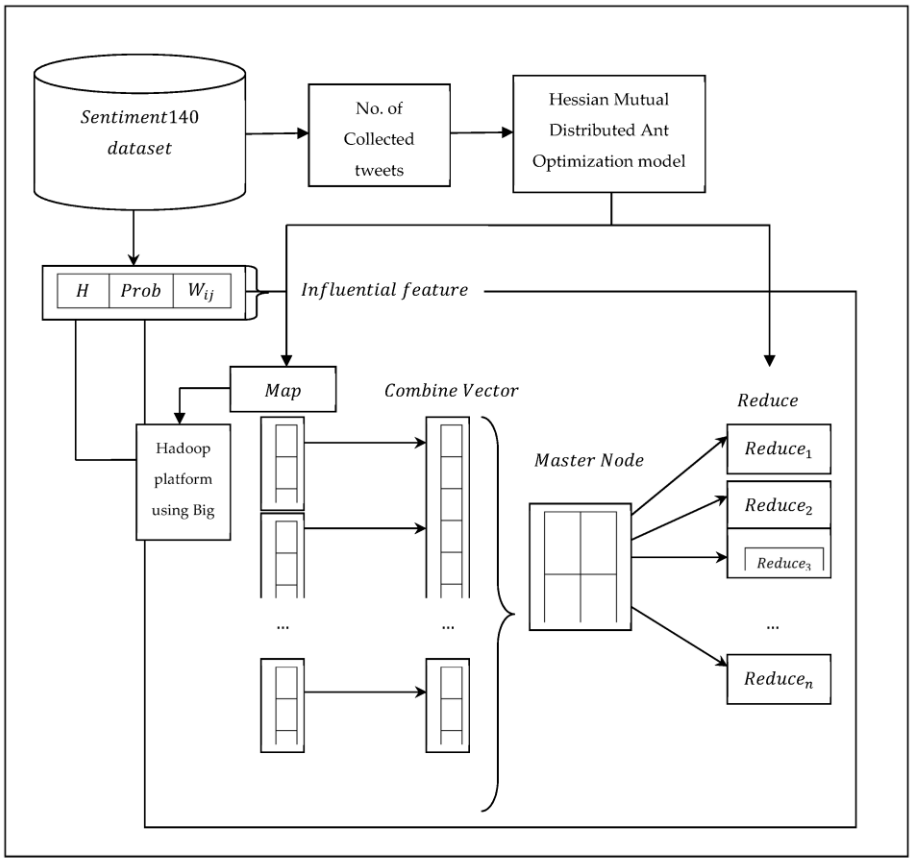


Figure 1. Block diagram of Hessian Distributed Ant Optimized and Perron–Frobenius Eigen Centrality (HDAO-PFEC).

The goal of this study is to develop a distributed, optimized multi-agent system for information retrieval that is computationally accurate and efficient, as seen in the following image using the Sentiment140 dataset (1.6 million tweets) as the input dataset. A MapReduce framework is created to achieve the goal by first analyzing three significant aspects, as shown in the above image.

Using the Hessian Mutual Distributed Ant Optimization model, comparable user interest tweets are acquired during the Map phase. Using these findings, the second phase generates the combined vector, and the last step, the reduction phase, uses the Perron–Frobenius Eigen Vector Centrality model to produce accurate and dimensionality-reduced tweets. Here is a detailed explanation of the suggested approach, along with information on graph theory and the identified issue.

3.1. Graph Theory

Let graph '$G = (V, E, AM)$' represents a digraph, where '$V = \{v_1, v_2, \ldots v_n\}$' represents the vertex set, '$E = \{(v_i, v_j) | v_i, v_j \in V\}$' represents the edge set, comprising of interaction links, 'AM' corresponding to the associated adjacency matrix, '$Prob$' is denotes the probability, 'H' is Hessian matrix and 'W_{ij}' is a corresponding weight. To be specific an edge 'v_i, v_j' denotes that agent 'j' can access the information of agent 'i' but not conversely. In addition, let 'AM_{ij}' be the matrix 'AM' for corresponding 'ith' row and 'jth' column. Then, the adjacency matrix 'AM_{ij}' is written as given below.

$$AM_{ij} = \begin{cases} 1, \ if \ edge \ (i,j) is \ in \ E \\ 0, \ otherwise \end{cases} \tag{1}$$

where i and j—refers the Agents, E—refers the Edge set.

The digraph 'G' is referred to as firmly associated if for any pair 'i' and 'j' in the graph, there is a path by following which vertex 'j' can be reached by vertex 'i'. For any vertices set, '$i, j \in G$', the distance 'Dis_{ij}', is referred to as the range of the shortest path from 'j' to 'i'. To develop a multi-agent distributed system for social networks, the work proposed to provide each network centrality unit (i.e., eigen vector centrality) with a multi-agent (i.e., Twitter agent) responsible for the optimal operation of that unit. In digraph theory and analysis of social networks, network centrality refers to the most predominant vertices within a graph. In our work, this refers to the identification of the most influential tweets in a social network.

By utilizing vertex set 'V' and edge set 'E' to represent the agents and communication mediums, the above-bestowed graph symbols are utilized to detail the multi-agent distributed system for social networks. Since some mediums may not be utilized to send tweets at time 't', a different annotation '$G[T] = (V, E(T), AM(T))$' to express the communication between users during tweet at time 'T' taking into consideration only the utilized mediums. Edge '(i, j)' is in '$E(T)$', if agent 'j' sends tweet to 'i', at time 'T', otherwise, '$(i, j) \notin E(T)$'.

3.2. Problem Formulation

Let us consider a multi-agent system that comprises of 'n' agents tagged with index '$1, 2, \ldots, n$' and each agent advances based on the dynamics.

$$a_i(T) = t_i(T) \tag{2}$$

where $a_i(T)$—Agents state over time 't', $t_i(T)$—Each agent possessing certain amount of tweets evolved over time 't'.

The objective is to design '$t_i(T)$' by utilizing the Hessian Mutual Distributed Ant Optimization and eigen vector centrality score such that all twitter agents work in a cooperative manner to reach the optimal state with respect to time-invariant largest Eigen optimization problem as given below.

$$MIN \sum_{i=1}^{n} MAX[f_i(a_i, T)] \tag{3}$$

3.3. Hessian Mutual Distributed Ant Optimization Model

Identifying the equivalent users' tweets that connect two agents (i.e., Twitter agent) is the first step in the modeling of the proposed multi-agent-based distributed system.

This is performed in the proposed work in the Map Phase by applying Hessian Mutual Distributed Ant Optimization HM-DAO model. Figure 2 shows a sample HM-DAO configuration for a partitioned social network map.

As illustrated in the above HM-DAO configuration, social network map contains, '$u = 12$' users, '$K = 4$' computers, '$A = 6$' twitter agents, with users' tweets represented in the form of a circle and agents denoted in the form of squares. A distributed model for a multi-agent system is designed using ant colony optimization with a category of time-changing cost function by considering the Hessian Matrix. This is due to the reason that the users' tweet acquired as input by the multi-agent system changes or evolves over time and hence the ideal point of multi-agent would be changing over time.

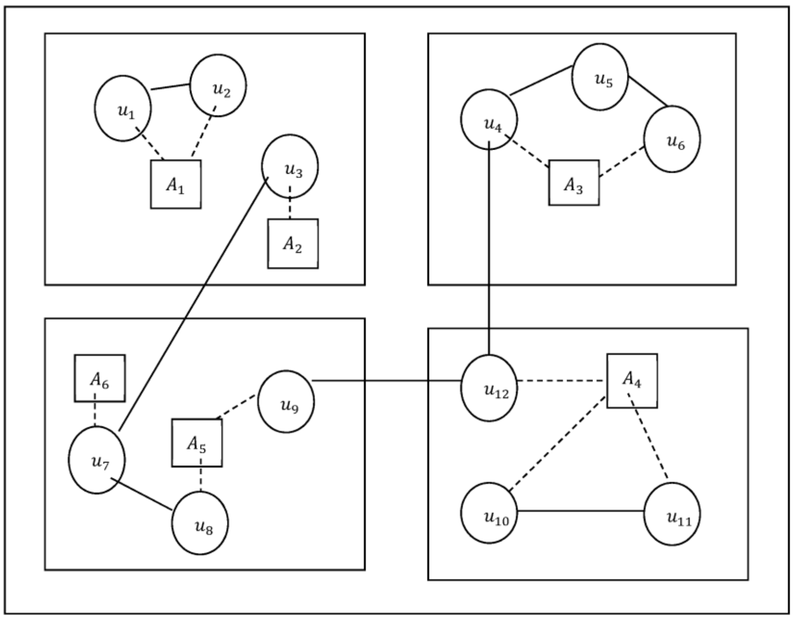


Figure 2. Sample HM-DAO configuration.

Let us consider a function '$f : R^n \to R$' considering as input users' tweets (i.e., simply tweets) procured by each agent as a vector '$a \in R^n$' and outputting a function '$f(a) \in R$'. With the assumption that all second partial derivatives of 'f' exist for a time-varying function (i.e., with users' tweets evolving over time), the hessian matrix of overall tweets procured for each agent is mathematically expressed as given below,

$$H = \begin{bmatrix} \frac{\partial^2 f}{\partial a_1^2} & \frac{\partial^2 f}{\partial a_1 \partial a_2} & \cdots & \frac{\partial^2 f}{\partial a_1 \partial a_n} \\ \frac{\partial^2 f}{\partial a_2 \partial a_1} & \frac{\partial^2 f}{\partial a_2^2} & \cdots & \frac{\partial^2 f}{\partial a_2 \partial a_n} \\ \cdots & \cdots & \cdots & \cdots \\ \frac{\partial^2 f}{\partial a_n \partial a_1} & \frac{\partial^2 f}{\partial a_n \partial a_2} & \cdots & \frac{\partial^2 f}{\partial a_n^2} \end{bmatrix} = H_{ij} = \frac{\partial^2 f}{\partial a_i \partial a_j} \tag{4}$$

where H_{ij}—Hessian matrix, $\partial^2 f$—Second order partial derivatives of function 'f', $\partial a_i \partial a_j$—Partial derivative of agent state of user i and j.

In this manner, from Equation (4), the time-changing users' tweets, are procured by the multi-agent system in the form of hessian matrix 'H_{ij}'. With the obtained hessian matrix of overall tweets procured for each agent, the distributed ants' environment (i.e., the nodes of the graph representing the users' tweets in the social network in our case) is designed as a set of interconnected Twitter agents.

In the proposed work, the Twitter agents (or agents) have their own action of control and determine in an independent manner to perform an action, interfaces with other agents' serial passing of messages between agents assigned using the agent identifier 'AID'. The agent-acquired users' tweets moving from user 'i' to user 'j' with probability '$Prob_{ij}$' for each hessian matrix 'H_{ij}' is measured as given below.

$$Prob_{ij}[H_{ij}] = \frac{(\tau_{ij})^{\alpha} (\eta_{ij})^{\beta}}{\sum_j (\tau_{ij})^{\alpha} (\eta_{ij})^{\beta}} \tag{5}$$

where H_{ij}—Hessian matrix, i & j—Input users, τ_{ij} —Amount of pheromone (agent acquired tweets) moving from user 'i' to user 'j', η_{ij}—Weight inverse, α &β—Criterion to manage the impact of 'τ_{ij}' and 'η_{ij}'.

However, to process a huge volume of data (i.e., Big Data), better solutions are required to be manifested with more pheromones (agent-acquired tweets). Hence, at any time agent 'a' with agent identifier 'AID' acquires a tweet 'T_a' of cost 'C_a' finer than the best-acquired tweet chunks, the agent will increase the pheromone strength on each edge of the tweet with a value '$\Delta\tau_{ij}^{A}$', that is equivalent to the characteristic of the solution. This is mathematically expressed as given below.

$$\Delta\tau_{ij}^{a} = \begin{cases} \frac{1}{C_a},\ if\ users\ tweet(i,j) \in\ T_a \\ 0,\ Otherwise \end{cases} \tag{6}$$

where $\Delta\tau_{ij}^{a}$—Amount of pheromone moving from user 'i' to user 'j' at any time agent 'a', $\frac{1}{C_a}$—Cost inverse at any time agent 'a', T_a—Tweet at any time agent 'a'.

When an agent completes a tour (or acquires all the users' tweets collected at a particular time interval), it will uncover backwards its moves plotting the users on the way with pheromone (agent-acquired tweets). The update also considers pheromone evaporation. In our experiments, we used the mutual weights of each tweet. This is mathematically evaluated as given below.

$$W_{ij} = \frac{t_{ij}/N}{\sum f_{ij}/N} \tag{7}$$

From the above Equation (7), the weights of each tweet '$\frac{t_{ij}}{N}$' are obtained based on the number of times that tweet is found to the number of twitter API in which the tweets under consideration are found '$\frac{f_{ij}}{N}$', respectively. With the aid of '$\Delta\tau_{ij}^{a}$' and its corresponding weight 'W_{ij}', similar user interests tweet 'ST' are arrived at. The Algorithm 1 representation of the Hessian Mutual Distributed Optimization is given below.

Algorithm 1. Hessian Mutual Distributed Optimization.

```
Input: Tweets '$Tweet = t_1,\ t_2,\ \ldots t_n$', Users '$U = u_1,\ u_2,\ \ldots u_n$', agent identifier '$AID$', Agent '$A = a_1,\ a_2,\ \ldots a_n$'
Output: Computationally efficient similar user interests tweet '$ST$'
Step 1: Begin
Step 2: For each Users '$U$' with Tweets '$T$'
Step 3: Obtain Hessian time-changing tweet function using (4)
Step 4: Evaluate probability factor for each obtained hessian matrix using (5)
Step 5: Obtain better solutions using (6)
Step 6: Evaluate mutual weight of each tweets using (7)
Step 7: Return (similar user interests tweet '$ST$')
Step 8: End for
Step 9: End
```

As given in the above Hessian Mutual Distributed Optimization algorithm, for each user's tweets obtained by the multi-agent (i.e., the Twitter agent), the objective here remains in obtaining the similar user interests tweets in a computationally efficient manner. This is performed in our work by applying the Hessian time-changing tweet function as the tweets evolved changes over time. Next, better solutions for each Twitter agent are arrived at by means of hessian distributed optimization model weight mutual weight. In this manner, similar user interest via a Twitter agent is obtained in a computationally efficient manner.

3.4. Perron–Frobenius Eigen Vector Centrality Model

With similar user interest tweets obtained via Twitter agent, dimensionality-reduced tweets are generated. These are generated by means of the Perron–Frobenius Eigen Vector Centrality (PF-EVC) model. The Perron–Frobenius Eigen Vector Centrality is an assessment of the impact of a user's tweet on a social network. It allocates correlative scores to all users' tweets in the social network on the basis of the hypothesis that relationships to high-scoring users' tweets bestow more to the score of the users' tweets than equivalent relationships to low-scoring users' tweets. To obtain relevant tweets and at the same time to reduce the dimensionality of data (i.e., users' tweets), in this work, the Perron–Frobenius Eigen Vector Centrality model is used. The flow diagram of the PF-EVC model is given below Figure 3.

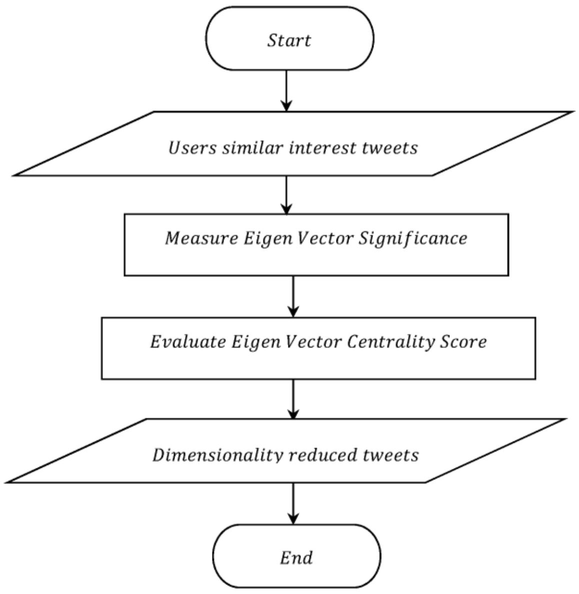

Start
Users similar interest tweets
Measure Eigen Vector Significance
Evaluate Eigen Vector Centrality Score
Dimensionality reduced tweets
End

Figure 3. Flow diagram of PF-EVC model.

The elaborate description of the PF-EVC is given below. To measure the Forbenius Eigen Vector Centrality of a users' tweet (or simply tweet), the significance of all other tweets that tweet '*i*' is associated has to be evaluated. On the basis of this correlative significance, the Frobenius Eigen Vector Centrality of a tweet is calculated as given below. Let us assume the vector index '$V = [v_1,\ v_2,\ \ldots, v_n]$' and weight index '$W = [w_1,\ w_2,\ \ldots, w_n]$', the eigen vector significance is measured as given below.

$$EVS(t_i) = w\left(st_j\right) + v\left(st_j\right) \tag{8}$$

where EVS—Eigen Vector Significance, t_i—User tweet, w—Weight, v—Vertices, st_j—Neighbor users similar interest tweet.

From the above Equation (8), the 'EVS' of each user's similar interest tweet 'st_i' is arrived at based on each of the neighbor users' similar interest tweet (i.e., 'j' to 'i') with corresponding to the weight and 'vertices. Then, the eigen vector centrality score (i.e., dimensionality-reduced tweets) is measured as given below.

$$x_i = \frac{1}{\lambda}\sum x_j = \frac{1}{\lambda}\sum am_{ij}x_j \tag{9}$$

where x_i—Eigen vector centrality score, λ—Constant value called the Perron–Frobenius value, am—Adjacency matrix.

From the above Equation (9), 'λ' with vertex 'i' is linked to vertex 'j' for the corresponding adjacency matrix 'am', respectively. The Algorithm 2 representation of Perron–Frobenius Eigen Vector Centrality is given below.

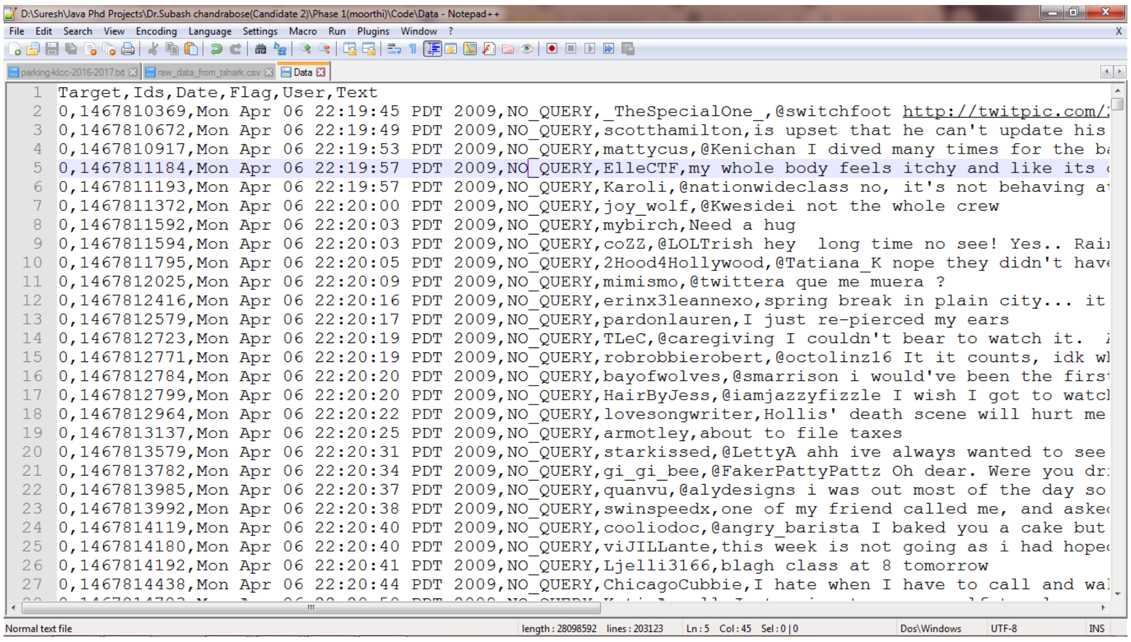
Algorithm 2. Perron–Frobenius Eigen Vector Centrality.

```
Input: Tweets '$T = t_1, t_2, \ldots t_n$', Users '$U = u_1, u_2, \ldots u_n$', Vertex set '$V = v_1, v_2, \ldots, v_n$', Edge set '$E = e_1, e_2, \ldots, e_n$', users similar interest tweets '$ST = st_1, st_2, \ldots, st_n'$
Output: Dimensionality reduced tweets
Step 1: Begin
Step 2: For each Users '$U$' with similar interest tweets '$ST$'
Step 3: Measure correlative significance using (8)
Step 4: Measure eigen vector centrality score using (9)
Step 5: Return dimensionality reduced tweets ($x_i$)
Step 6: End for
Step 7: End
```

As given in the above Perron–Frobenius Eigen Vector Centrality algorithm, for each user with similar interests tweets provided as input, the objective here remains in obtaining the dimensionality reduced tweets for selecting their choices of interests for social networks have been designed. With this objective, two factors are concerned. They are measuring the correlative significance based on the Eigen Vector Centrality and eigen vector centrality score using Perron–Frobenius value. With these two factors, only the greatest eigen value results are acquired, therefore contributing to dimensionality reduced tweets and eliminating the lesser influence eigen values.

3.5. Experimental Setup

In our experiments, both the Hessian Mutual Distributed Optimization algorithm and the Perron–Frobenius Eigen Vector Centrality algorithm are implemented in JAVA program language using the Cloud infrastructure and MapReduce parallel programming model. The version of 2.7.3 is adopted for Hadoop cluster. In the clusters, one node acts as the master and the others act as slaves. Both the single machine and the nodes in the clusters have the same hardware configuration, namely Core2 Duo CPU @ 2.20 GHz, 2 CPUs and 2 GB of RAM. Operating system and SUN JAVA JDK1.8.0_131 trained on Sentiment140 dataset obtained from https://www.kaggle.com/kazanova/sentiment140 (accessed on 1 February 2023). The dataset comprises of the following features as given in Figure 4 and Table 1.


D:\Suresh\Java Phd Projects\Dr.Subash chandrabose(Candidate 2)\Phase 1(moorthi)\Code\Data - Notepad++
File Edit Search View Encoding Language Settings Macro Run Plugins Window ?
parking-klcc-2016-2017.txt
raw_data_from_tshark.csv
Data

```
Target,Ids,Date,Flag,User,Text
0,1467810369,Mon Apr 06 22:19:45 PDT 2009,NO_QUERY,_TheSpecialOne_,@switchfoot http://twitpic.com/
0,1467810672,Mon Apr 06 22:19:49 PDT 2009,NO_QUERY,scotthamilton,is upset that he can't update his
0,1467810917,Mon Apr 06 22:19:53 PDT 2009,NO_QUERY,mattycus,@Kenichan I dived many times for the b
0,1467811184,Mon Apr 06 22:19:57 PDT 2009,NO_QUERY,ElleCTF,my whole body feels itchy and like its
0,1467811193,Mon Apr 06 22:19:57 PDT 2009,NO_QUERY,Karoli,@nationwideclass no, it's not behaving a
0,1467811372,Mon Apr 06 22:20:00 PDT 2009,NO_QUERY,joy_wolf,@Kwesidei not the whole crew
0,1467811592,Mon Apr 06 22:20:03 PDT 2009,NO_QUERY,mybirch,Need a hug
0,1467811594,Mon Apr 06 22:20:03 PDT 2009,NO_QUERY,coZZ,@LOLTrish hey  long time no see! Yes.. Rai
0,1467811795,Mon Apr 06 22:20:05 PDT 2009,NO_QUERY,2Hood4Hollywood,@Tatiana_K nope they didn't hav
0,1467812025,Mon Apr 06 22:20:09 PDT 2009,NO_QUERY,mimismo,@twittera que me muera ?
0,1467812416,Mon Apr 06 22:20:16 PDT 2009,NO_QUERY,erinx3leannexo,spring break in plain city... it
0,1467812579,Mon Apr 06 22:20:17 PDT 2009,NO_QUERY,pardonlauren,I just re-pierced my ears
0,1467812723,Mon Apr 06 22:20:19 PDT 2009,NO_QUERY,TLeC,@caregiving I couldn't bear to watch it.
0,1467812771,Mon Apr 06 22:20:19 PDT 2009,NO_QUERY,robrobbierobert,@octolinz16 It it counts, idk w
0,1467812784,Mon Apr 06 22:20:20 PDT 2009,NO_QUERY,bayofwolves,@smarrison i would've been the firs
0,1467812799,Mon Apr 06 22:20:20 PDT 2009,NO_QUERY,HairByJess,@iamjazzyfizzle I wish I got to watc
0,1467812964,Mon Apr 06 22:20:22 PDT 2009,NO_QUERY,lovesongwriter,Hollis' death scene will hurt me
0,1467813137,Mon Apr 06 22:20:25 PDT 2009,NO_QUERY,armotley,about to file taxes
0,1467813579,Mon Apr 06 22:20:31 PDT 2009,NO_QUERY,starkissed,@LettyA ahh ive always wanted to see
0,1467813782,Mon Apr 06 22:20:34 PDT 2009,NO_QUERY,gi_gi_bee,@FakerPattyPattz Oh dear. Were you dr
0,1467813985,Mon Apr 06 22:20:37 PDT 2009,NO_QUERY,quanvu,@alydesigns i was out most of the day so
0,1467813992,Mon Apr 06 22:20:38 PDT 2009,NO_QUERY,swinspeedx,one of my friend called me, and aske
0,1467814119,Mon Apr 06 22:20:40 PDT 2009,NO_QUERY,cooliodoc,@angry_barista I baked you a cake but
0,1467814180,Mon Apr 06 22:20:40 PDT 2009,NO_QUERY,viJILLante,this week is not going as i had hope
0,1467814192,Mon Apr 06 22:20:41 PDT 2009,NO_QUERY,Ljelli3166,blagh class at 8 tomorrow
0,1467814438,Mon Apr 06 22:20:44 PDT 2009,NO_QUERY,ChicagoCubbie,I hate when I have to call and wa
```

```
Normal text file    length : 28098592   lines : 203123    Ln : 5   Col : 45   Sel : 0 | 0    Dos\Windows    UTF-8    INS
```

Figure 4. Sentiment140 dataset features.

Table 1. Sentiment140 dataset features and description.

S. No	Feature Name	Description
1	Target	The polarity of the tweet
2	Ids	The id of the tweet
3	Date	The date of the tweet
4	Flag	Flag value
5	User	User that tweeted
6	Text	The text of the tweet

Simulations are conducted with three parameters, computational time, computational overhead and accuracy. Fair comparison is made with the proposed Hessian Distributed Ant Optimized and Perron–Frobenius Eigen Centrality (HDAO-PFEC) and existing two methods, laplace three-level stochastic variational inference [1] and multi-agent based distributed architecture [2] for agent-based distributed system in social networks.

4. Discussion

To measure the performance of the proposed method comparative analysis with laplace three-level stochastic variational inference [1] and multi-agent based distributed architecture [2] are performed based on the metrics, running time, data storage overhead and accuracy score with respect to different number of tweets.

4.1. Performance Analysis of Running Time

The first parameter of importance for any multi-agent distributed system is the running time. Running time refers to the time consumed in performing certain task. In our work, running time refers to the time consumed in obtaining the computationally efficient similar user interests tweet. This is mathematically formulated as given below.

$$RT = \sum_{i=1}^{n} Tweet_i * Time\left[\Delta\tau_{ij}^{a}(W_{ij})\right] \tag{10}$$

where, RT—Running Time, $\Delta\tau_{ij}^{a}$—Agent acquired tweets moving from user 'i' to user 'j', W_{ij}—Corresponding weight.

From the above Equation (10), 'RT' is evaluated based on the number of tweets considered for conducting simulations and the time incurred in obtaining similar user interests tweets '$Time\left[\Delta\tau_{ij}^{a}(W_{ij})\right]$'. It is measured in terms of milliseconds (ms). The running time performance of three different methods, HDAO-PFEC, laplace three-level stochastic variational inference [1] and multi-agent based distributed architecture [2] are shown in Table 2.

Table 2. Running time using proposed HDAO-PFEC, and existing laplace three-level stochastic variational inference and multi-agent based distributed architecture.

Number of Tweets	Running Time (ms)		
	HDAO-PFEC	Laplace Three-Level Stochastic Variational Inference	Multi-Agent BASED Distributed Architecture
10,000	2500	2630	2655
15,000	2635	2685	2735
20,000	2655	2755	2815
25,000	2690	2815	2925
30,000	2735	2835	3015
35,000	2755	2925	3235
40,000	2780	3055	3340
45,000	2855	3155	3455
50,000	2950	3215	3615
55,000	3035	3355	3730

Figure 5 given above shows the running time performance of three different methods used for analyzing the agent based distributed system method in social network. While obtaining tweets of similar user interests, a significant amount of time is being consumed. As represented in above figure, the running time of proposed HDAO-PFEC method is reduced as compared to existing methods. With 10,000 number of tweets considered for experimentation the running time is measured for 10 different simulation runs. With the increase in number of tweets made for sentiment analysis, the running time also gradually increases. However, with simulations conducted for 1000 tweets, the overall running time for modeling multi-agent based distributed system using HDAO-PFEC was observed to be 1055 ms, 1130 ms with the aid of laplace three-level stochastic variational inference [1] and 1155 ms using and multi-agent based distributed architecture [2]. From the simulation results it is inferred that the running time was comparatively lesser using HDAO-PFEC when compared to laplace three-level stochastic variational inference [1] and multi-agent-based distributed architecture [2]. This is because of the application of the Hessian Mutual Distributed Ant Optimization model in the Map phase. By applying this model, second partial derivatives for a time-varying function are generated using the Hessian matrix. With this, the running time of the HDAO-PFEC method is found to be minimized by 6% compared to laplace three-level stochastic variational inference [1] and 11% compared to multi-agent-based distributed architecture [2], respectively.

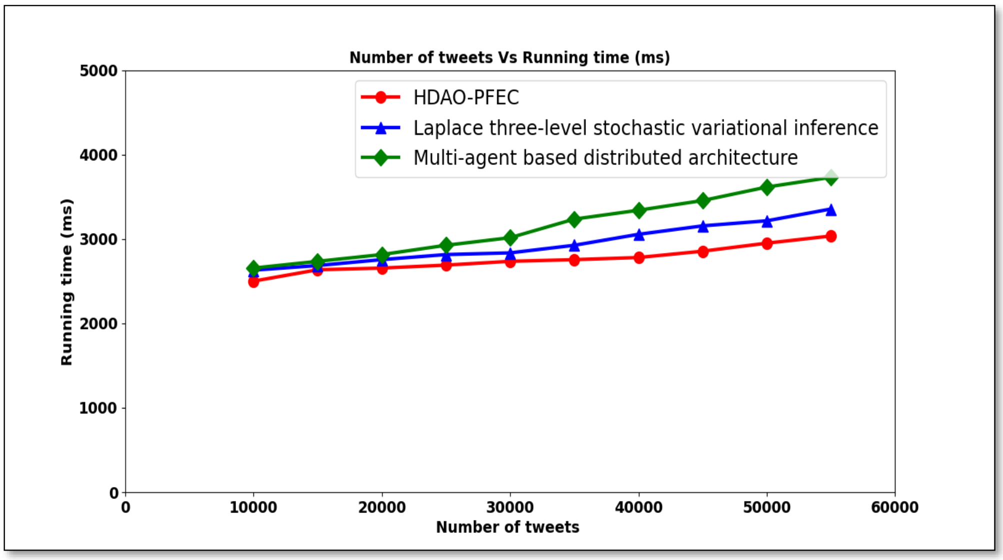


Figure 5. Graphical representation of running time.

4.2. Performance Analysis of Data Storage Overhead

The next parameter used in the analysis of multi-agent distributed system involving social networks is the data storage overhead. Data storage overhead is the memory consumed in performing a task or in other words, a certain amount of storage overhead is said to occur while obtaining similar user interests tweet. This is mathematically formulated as given below.

$$DSO = \sum_{i=1}^{n} Tweet_i * MEM\left[\Delta\tau_{ij}^{a}(W_{ij})\right] \tag{11}$$

where DSO—Data Storage Overhead, $\Delta\tau_{ij}^{a}$—Agent acquired tweets moving from user 'i' to user 'j', W_{ij}—Corresponding weight.

From the above Equation (11), 'DSO' is evaluated on the basis of the tweets involved and the storage overhead generated while acquiring similar user interests tweets '$MEM\left[\Delta\tau_{ij}^{a}(W_{ij})\right]$'. It is measured in terms of kilobytes (KB). The data storage overhead performance of three different methods, HDAO-PFEC and existing methods i.e., laplace three-level stochastic variational inference and multi-agent based distributed architecture are shown in Table 3.

Table 3. Data storage overhead using proposed HDAO-PFEC, and existing laplace three-level stochastic variational inference and multi-agent based distributed architecture.

Number of Tweets	Data Storage Overhead (KB)		
	HDAO-PFEC	Laplace Three-Level Stochastic Variational Inference	Multi-Agent Based Distributed Architecture
10,000	30,000	35,000	40,000
15,000	32,000	36,000	42,000
20,000	33,000	38,000	45,000
25,000	35,000	40,000	46,500
30,000	38,000	42,000	47,000
35,000	42,000	44,000	48,000
40,000	45,000	46,000	50,000
45,000	46,000	48,000	50,500
50,000	48,000	50,000	52,000
55,000	50,000	52,000	54,000

Figure 6 above illustrates the data storage overhead generated with respect to 10,000 tweets collected at different time intervals. While generation of similar user interests tweets, the intermediate tweets generated are stored in the combined vector. From the acquired tweets in the combined vector, an overhead is said to take place. Hence, increasing the number of tweets results in an increase in the data storage overhead. Therefore, data storage overhead is directly proportional to the number of tweets posted on social networks. However, with simulations conducted with 1000 tweets, the data storage overhead using HDAO-PFEC was observed to be 2000 KB, 3000 KB when applied with laplace three-level stochastic variational inference [1] and 4000 KB using multi-agent based distributed architecture [2]. Comparative better performance of data storage overhead observed in HDAO-PFEC is observed due to the application of the Hessian Mutual Distributed Optimization algorithm. By applying this algorithm, the mutual weight of each tweet is taken into consideration similar user interests tweet are arrived at due to the storage of intermediate results in combined vector and final results in the reduce phase. This in turn reduces the data storage overhead using HDAO-PFEC by 8% compared to laplace three-level stochastic variational inference [1] and 16% compared to multi-agent-based distributed architecture [2], respectively.

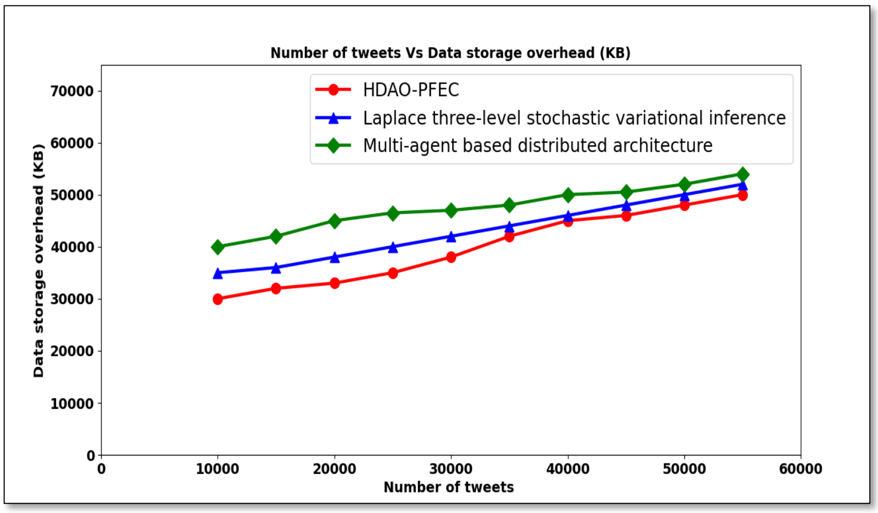


Figure 6. Graphical representation of data storage overhead.

4.3. Performance Analysis of Accuracy Score

Finally, the accuracy score of estimating agents' analysis of users' choices of interest is evaluated. This is mathematically expressed as given below.

$$A = \sum_{i=1}^{n} \frac{Tweets_{C\_A}}{T_i} * 100 \tag{12}$$

From the above Equation (12), the accuracy score 'A' is evaluated based on the tweets correctly grouped by Twitter agent '$Tweets_{C\_A}$' to the total tweets 'T_i' sampled. It is measured in terms of percentage (%). The accuracy score performance of three different methods HDAO-PFEC, laplace three-level stochastic variational inference [1], and multi-agent-based distributed architecture [2] are shown in Table 4.

Table 4. Accuracy score using proposed HDAO-PFEC and existing laplace three-level stochastic variational inference and multi-agent based distributed architecture.

Number of Tweets	Accuracy Score (%s)		
	HDAO-PFEC	**Laplace Three-Level Stochastic Variational Inference**	**Multi-Agent Based Distributed Architecture**
10,000	80.00	78.50	75.00
15,000	78.85	76.25	74.15
20,000	77.00	75.65	73.55
25,000	76.25	74.55	72.15
30,000	75.00	74.15	70.15
35,000	74.35	73.00	69.80
40,000	73.86	71.35	69.25
450,00	73.25	70.45	68.60
50,000	72.80	69.75	68.10
55,000	70.00	68.00	66.75

Figure 7 shown above illustrates the graphical pattern of accuracy score. Here, the accuracy score refers to the estimation of Twitter agents' analysis of users' choices of tweets to detect sentiment. As shown in the figure, to start with 6000 tweets, the accuracy scores decline gradually and observed a small hype at 7000 tweets and again decreased. The increase and decrease in accuracy score are said to occur due to different reasons like, positive emoticons, negative emoticons variation are said to observe. Consider 1000 tweets involving both positive, negative and neutral annotations in simulation conduction, 98% of accuracy score was found by applying HDAO-PFEC, 96.5% and 95% using laplace three-level stochastic variational inference [1] and multi-agent based distributed architecture [2]. From the results it is inferred that the accuracy score is comparatively better when applied with HDAO-PFEC than laplace three-level stochastic variational inference [1] and multi-agent based distributed architecture [2]. This is due to the application of Perron–Frobenius Eigen Vector Centrality algorithm. By applying this algorithm, dimensionality reduced tweets are obtained that in turn are utilized for determining correlative significance based on the eigen vector centrality. This eigen vector centrality most influential tweets in the social network are identified that in turn aids in improving the accuracy score of HDAO-PFEC by 3% compared to laplace three-level stochastic variational inference [1] and 6% compared to multi-agent based distributed architecture [2], respectively.

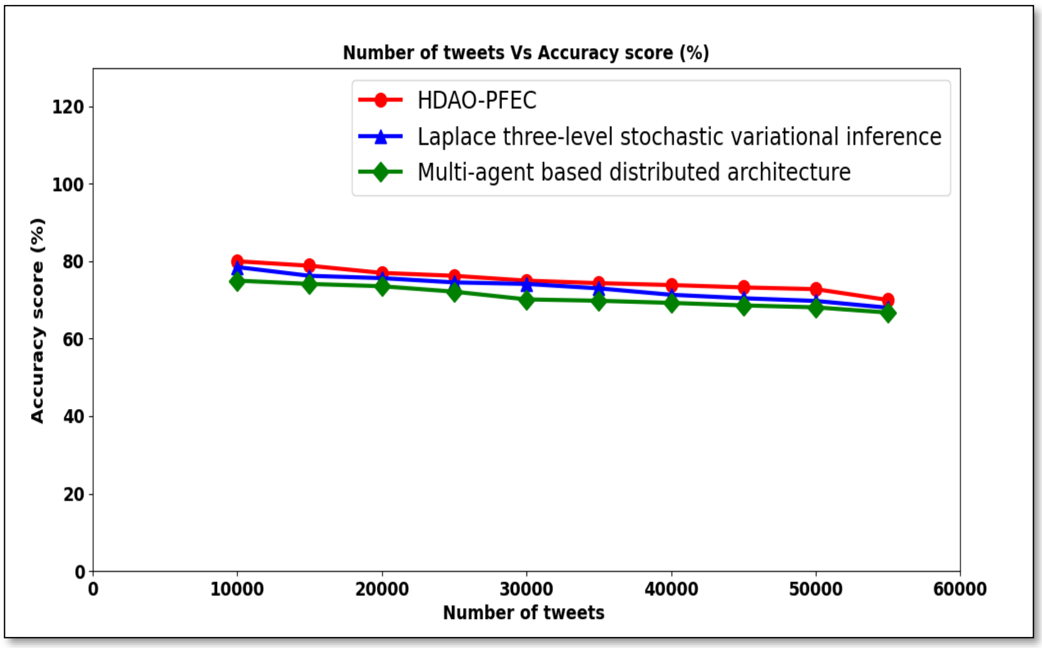


Figure 7. Graphical representation of accuracy score.

5. Conclusions

A novel multi-agent system for social network tweets involving Big Data is proposed in this work. An efficient method called Hessian Distributed Ant Optimized and Perron–Frobenius Eigen Centrality (HDAO-PFEC) user group based on their tweets for social networks is developed with less running time and overhead in a more accurate pattern. Initially, the numbers of users and tweets made by the users in the social network for sentiment analysis are taken as input. The Mutual Distributed Ant Optimization model is used as the graphical model in the Map phase in a distributed pattern to present links in social networks. With this, similar user interest tweets are obtained and stored in the combined vector. From there, these tweets are then passed on to the Reducer phase using Perron–Frobenius Eigen Vector Centrality to obtain dimensionality reduced tweets. With the application of these two phases in a distributed manner, accurate scores are arrived at by means of eigen vector centrality score. Extensive experiments are also conducted on Sentiment140 dataset, and the experimental results show that the proposed method achieves higher accuracy score by 13% and 20% as compared existing laplace three-level stochastic variational inference [1] and multi-agent based distributed architecture [2], respectively. Also, running time of proposed method is minimized by 21% and 29% as compared existing laplace three-level stochastic variational inference [1] and multi-agent based distributed architecture [2], respectively. Similarly, data storage overhead is reduced by 27% and 42% when compared existing methods. In limitation of this proposed work, failed to analyze the error rate and improve the precision.

Author Contributions: Formal analysis, P.V. Kumaraguru, Vidyavathi Kamalakkannan, Gururaj H L and Badria Sulaiman Alfurhood; Funding acquisition, Francesco Flammini; Investigation, Francesco Flammini; Methodology, P.V. Kumaraguru, Vidyavathi Kamalakkannan, Francesco Flammini and Rajesh Natarajan; Validation, Gururaj H L and Badria Sulaiman Alfurhood; Visualization, Rajesh Natarajan; Writing—original draft, Vidyavathi Kamalakkannan and Rajesh Natarajan; Writing—review & editing, P.V. Kumaraguru, Vidyavathi Kamalakkannan, Gururaj H L, Francesco Flammini, Badria Sulaiman Alfurhood and Rajesh Natarajan. All authors have read and agreed to the published version of the manuscript.

Funding: This research is supported by Princess Nourah bint Abdulrahman University Researchers Supporting Project number (PNURSP2023R359), Princess Nourah bint Abdulrahman University, Riyadh, Saudi Arabia.

Data Availability Statement: Not applicable.

Acknowledgments: The authors thank Princess Nourah bint Abdulrahman University Researchers Supporting Project number (PNURSP2023R359), Princess Nourah bint Abdulrahman University, Riyadh, Saudi Arabia.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Yang, J.; Wang, J.; Tay, W.P. Using Social Network Information in Community-Based Bayesian Truth Dis-covery. *IEEE Trans. Signal Inf. Process. Over Netw.* **2019**, *5*, 525–537. [CrossRef]
2. Palau, A.S.; Dhada, M.H.; Parlikad, A.K. Multi-agent system architectures for collaborative prognostics. *J. Intell. Manuf.* **2019**, *30*, 2999–3013. [CrossRef]
3. Pang, Y.; Hu, G. Randomized Gradient-Free Distributed Optimization Methods for a Multi-Agent System with Unknown Cost Function. *IEEE Trans. Autom. Control* **2020**, *65*, 333–340. [CrossRef]
4. Zivan, R.; Okamoto, S.; Peled, H. Explorative anytime local search for distributed constraint optimization. *Artif. Intell.* **2014**, *212*, 1–26. [CrossRef]
5. Wang, Q.; Duan, Z.; Wang, J. Distributed Optimal Consensus Control Algorithm for Continuous-Time Multi-Agent Systems. *IEEE Trans. Circuits Syst. II Express Briefs* **2020**, *67*, 102–106. [CrossRef]
6. Jiang, Y.; Jiang, J.C. Diffusion in Social Networks: A Multiagent Perspective. *IEEE Trans. Syst. Man Cybern. Syst.* **2015**, *45*, 198–213. [CrossRef]
7. Alaviani, S.S.; Elia, N. Distributed Multi-Agent Convex Optimization Over Random Digraphs. *IEEE Trans. Autom. Control* **2020**, *65*, 86–998. [CrossRef]
8. Sun, C.; Ye, M.; Hu, G. Distributed Optimization for Two Types of Heterogeneous Multiagent Systems. *IEEE Trans. Neural Netw. Learn. Syst.* **2021**, *32*, 1314–1324. [CrossRef] [PubMed]
9. Park, Y.J.; Cho, Y.S.; Kim, S.B. Multi-agent reinforcement learning with approximate model learning for competitive games. *PLoS ONE* **2019**, *14*, e0222215. [CrossRef] [PubMed]
10. Zhang, Q.; Gong, Z.; Yang, Z.; Chen, Z. Distributed Convex Optimization for Flocking of Nonlinear Multi-agent Systems. *Int. J. Control Autom. Syst.* **2019**, *17*, 1177–1183. [CrossRef]
11. Mukhutdinov, D.; Filchenkov, A.; Shalyto, A.; Vyatkin, V. Multi-agent deep learning for simultaneous optimization for time and energy in distributed routing system. *Futur. Gener. Comput. Syst.* **2019**, *94*, 587–600. [CrossRef]
12. Li, F.; Xu, Z. A multi-agent system for distributed multi-project scheduling with two-stage decomposition. *PLoS ONE* **2018**, *13*, e0205445. [CrossRef] [PubMed]
13. Bose, S.; Chandra, S.; Alfurhood, B.S.; Gururaj, H.L.; Flammini, F.; Natarajan, R.; Jaya, S.-K. Decision Fault Tree Learning and Differential Lyapunov Optimal Control for Path Tracking. *Entropy* **2023**, *25*, 443. [CrossRef] [PubMed]
14. Tampuu, A.; Matiisen, T.; Kodelja, D.; Kuzovkin, I.; Korjus, K.; Aru, J.; Aru, J.; Vicente, R. Multiagent cooperation and competition with deep reinforcement learning. *PLoS ONE* **2017**, *12*, e0172395. [CrossRef] [PubMed]
15. Kamiński, B.; Kraiński, Ł.; Mashatan, A.; Prałat, P.; Szufel, P. Multiagent Routing Simulation with Partial Smart Vehicles Penetration. *J. Adv. Transp.* **2020**, *2020*, 3152020. [CrossRef]
16. Shirazi, E.; Jadid, S. A multiagent design for self-healing in electric power distribution systems. *Electr. Power Syst. Res.* **2019**, *171*, 230–239. [CrossRef]
17. Yan, F.; Di, K.; Jiang, J.; Jiang, Y.; Fan, H. Efficient decision-making for multiagent target searching and occupancy in an unknown environment. *Robot. Auton. Syst.* **2019**, *114*, 41–56. [CrossRef]
18. Chandan, R.R.; Balobaid, A.; Cherukupalli, N.L.S.; Gururaj, H.L.; Flammini, F.; Natarajan, R. Secure Modern Wireless Communication Network Based on Blockchain Technology. *Electronics* **2023**, *12*, 1095. [CrossRef]
19. Liu, S.; Jiang, H.; Zhang, L.; Mei, X. Distributed Adaptive Optimization for Generalized Linear Multiagent Systems. *Discret. Dyn. Nat. Soc.* **2019**, *2019*, 9181093. [CrossRef]