



T&TRS: robust collaborative filtering recommender systems against attacks

Fatemeh Rezaimehr¹ · Chitra Dadkhah¹

Received: 29 June 2022 / Revised: 16 July 2023 / Accepted: 23 August 2023 /
Published online: 18 September 2023

© The Author(s), under exclusive licence to Springer Science+Business Media, LLC, part of Springer Nature 2023

Abstract

In recent years, the Internet has had a main and important contribution to human life and the amount of data on the World Wide Web such as books, movies, videos and, etc. increase rapidly. Recommender systems allow users to quickly access items that are closer to their interests. One of the most popular and easiest models of recommender systems is the Collaborative Filtering (CF) model, which uses the items ratings given by users. The important challenge of CF is robust against the attacks which manipulated by fake users to reduce the efficiency of the system. Therefore, the impact of attacks on the item recommendations will increase and fake items will be easily recommended to users. The purpose of this paper is to design a robust CF recommender system, T&TRS, Time and Trust Recommender System, against user attacks. Our proposed system improves the performance of users clustering for detecting the fake users based on a novel community detection algorithm that is introduced in this paper. Our proposed system calculates the reliability value for all items ratings and tags them as suspicious or correct. T&TRS considered the rating time and implicit and explicit trust among users for constructing the weighted user-user network and detects communities as the nearest neighbors of the users to predict unknown items ratings. After detecting the suspect users and items using a novel community detection method, our proposed system removes them from rating matrix and predict the rating of unobserved items and generate the Top@k items according to the user interests. We inject the random and average attacks into the Epinion data set and evaluate our proposed systems based on Precision, Recall, F1, MAE, RMSE, and RC measures before and after attacks. The experimental results indicated that the precision of items recommendations increase after attack detection and show the effectiveness of T&TRS in comparison to the two base K-means methods such as KMCF-U, KMCF-I and graph-based methods such as TRACCF, and TOTAR.

Keyword Recommender systems · Collaborative filtering · Attacks · Time · Trust · Community detection · Graph

✉ Chitra Dadkhah
Dadkhah@kntu.ac.ir

Fatemeh Rezaimehr
F.rezaimehr@email.kntu.ac.ir

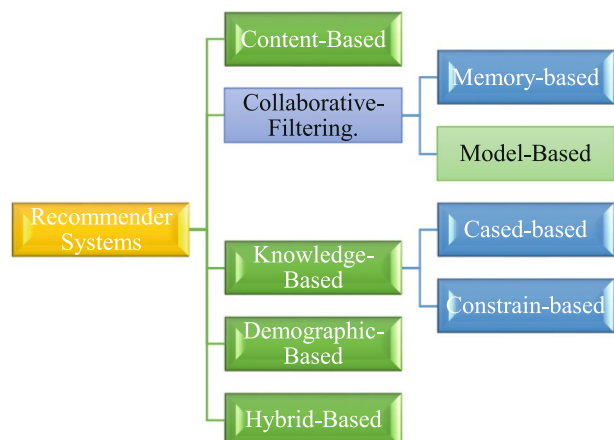
¹ Computer Engineering Faculty, K. N. Toosi University of Technology, Tehran, Iran

1 Introduction

Big data has become one of the significant topics in various research areas, including social networks and online shopping. A large volume of digital data is often produced by users when they purchase/ rate items. Such information, along with users/items contextual data can be used to obtain user-items interactions, model user's preferences, and make proper item recommendations for them. Recommendation Systems (RSs) have been developed to perform such tasks [1]. RSs have significant applications in many industrial systems in recent years, which work on the availability of large-scale social network data. RSs have been successfully used in many fields such as videos [2, 3], webpages [4], friends [5], mobile applications [6], OLAP sessions [7], travel packages [8–10], tourisms[11, 12], e-commerce [13], Business-to-Consumer (B2C) controls [14], business process models [15], jobs [16], routs [17], learning objects [18], and, etc. in industry and academia applications. Some of the papers have worked on optimizing the performance of their proposed recommender systems using evolutionary algorithms [19], and others used additional information such as time, location, etc.

Because recommendations are usually personalized in RS(s), different users or user groups receive a variety of recommendations. For personalized recommendations, the system provides a list of ranked items. In performing this ranking, the system attempts to predict the most appropriate and the best products and services for users based on preferences and limitations, and in order to complete such operations, it collects the users' past transactions [20]. RS(s) use different data sources such as rating matrix and user and item features to extract correlations between users and items. Based on the recommendation technique and method, recommender systems can be divided [1] into Content-Based (CB), Collaborative Filtering (CF), Knowledge-Based (KB), Demographic-Based (DB), and Hybrid System (HR) as shown in Fig. 1. The system uses only the item rating given by active user and items features are called CBRS. In contrast, the system recommends the items to active users based on users-items interaction data called CFRS. There are also HRSs that use both data types. In KBRS users define explicitly her/his requirements and according to them system recommends the items. The CFRS is divided into model-based and memory-based. Model-based CF uses machine learning algorithms such as matrix factorization and Singular Value Decomposition (SVD), to train the data and construct the model. Memory-based

Fig. 1 The models of recommender systems



methods, such as user/item-based approach, identifies the nearest neighbors of active user or items which observed by an active user based on similarity measures. This approach works on the assumption that similar users have similar tastes and preferences, and a target user gives positive ratings to the items that have already been positively rated by his/her neighbors. Item-based approach assumes that the target user gives the same rating to similar items. Although CFRS has been successfully applied to many applications and are indeed the most widely used RSs in the industry, they suffer from several issues, including rating sparsity, cold-start, scalability, and attack. When users rate a small number of items, the sparsity issue occurred. The cold-start problem happens when a few item ratings exist in the rating matrix for new users or new items, and similar users or items cannot be found easily. Several techniques have been used in the literature to handle these issues [21]. CFRS are easy to understand; that is why the fake users attacked these systems. Recently, methods have been introduced to detect and remove these attacks. Contextual data, friendship, membership, and social trust relationships are approaches to solving the cold start problem in social networks and community structure. Purchase/rating time is another important factor that, if available, can be used to improve the efficiency of recommendation lists. Traditional RSs ignore this factor and consider all the ratings as the same value for prediction, regardless of their transaction time. However, users might change their preferences over time, and also the popularity of items change over time.

Shilling attacks are one of the new challenges of recommender systems where the user attempts to increase /decrease the popularity of a particular item by using one or more fake profiles and add/remove item to/from the Top@K items recommendations. Shilling attacks are divided into push or nuke categories in terms of the attack target item. In push attacks, the attacker tries to add a specific item to the recommendation list with the fake items ratings. Nuke attacks try to remove a specific item from the recommendations list of recommender systems. In recent studies, researchers are trying to reduce the destructive effects of shilling attacks on recommender systems. Each attack profile has different dimensions such as: *attack model*, *attack content*, *profile size*, and *attack size* [22]. The *attack content* describes the attacker's target, which includes both push and nuke target item. *Profile size* is the number of ratings that an attacker has assigned to an attack profile. If the attack profile contains a large number of ratings, it will be detected sooner. For this reason, the attacker does not increase the profile size. The number of profiles that an attacker adds to the system is the *size of the attack*. The *attack model* describes the properties of the items used in the attack [23].

Lam and Reidl in [24] classify attacks into two main categories of *random* and *average*, based on the attackers' behavior patterns and ratings. Attack profiles contain; selected items (I_s), filler items (I_F), unrated items (I_0) and target item (I_T). In the random and average attack, the value of I_F is normal distribution around the system mean items rating and normal distribution around the mean rating value for the specific item; $i \in I_F$, respectively. In both attacks, $I_T = r_{min}/r_{max}$, $I_s = \varnothing$ and the value of I_0 determines by filler size.

Numerous methods have been presented for attack detection to improve the robustness of collaborative filtering recommender systems. The purpose of these methods is to reduce the impact of shilling attacks by identifying and eliminating attack profiles. Fake accounts and fake followers are also dangerous on social platforms because they may change concepts such as popularity and influence, which may have a great impact on the economy, politics, and society. Caruccio et al. in [25] presented a new technique for detecting fake accounts. Their technique extracted knowledge automatically from big data to identify the patterns of fake accounts. This article discusses topics related to fake account discrimination. Today, one of the user's worries is privacy in social media. Cerruto et al. in [26]

used an image-recognition technique to recognize users from their pictures, aiming to collect their personal data accessible through social networks. They analyzed the amount of extrapolated data for evaluating privacy threats when users share information on different social networks. Also, Cirillo et al. in [27] proposed an intuitive framework that improves users' awareness of their data privacy on social network platforms. The article discusses topics related to user privacy awareness in the social network domain.

In this paper, we proposed a collaborative filtering recommender System, called T&TRS which is robust against the Random and Average attacks for nuke and push strategy attack. T&TRS system considered time of item ratings and user trust in its recommendation process.

The contributions of our proposed system are as follows:

- New method of community detection has been introduced, that has increased the precision of clustering for finding fake users. The most important problem to identify communities is the initial number of clusters and also the centers of the clusters which are selected randomly. In T&TRS, the nodes with maximum links in the generated user-user network are selected as the center of the clusters. In overlapping community detection of the proposed method, a user may belong to more than one community. So, systems can consider more users in the prediction process.
- The proposed system removed the fake users and items from the rating matrix and predicted the rating of unobserved items based on the new rating matrix to improve the accuracy of collaborative filtering recommender systems and being resistant to the attacks.
- The proposed system employed an imputation method integrating with a reliability measure for estimating the missing ratings. This approach is used to solve the sparsity problem in recommender systems.
- We extracted the implicit trust between users and present a new formula to calculate the similarity between users based on the implicit and explicit trust for constructing the weighted user-user networks.
- The new fitness function, which is used to assign a node to the best possible detected communities has been applied in our proposed system.

The structure of the paper is as follows: Section 2 reviewed the related works. In Section 3, T&TRS is explained in detail and also the novel community detection algorithm is described. Section 4 presents the evaluation of our proposed system, and finally, Section 5 describes the conclusions and future works.

2 Related work

Recently, some methods have been introduced to detect and remove the attacks in recommender systems by researches. Rezaeimehr and Dadkhah in [28] studied the methods to identify the attacks and classified them into four categories; 1) clustering, 2) classification, 3) probabilistic methods and 4) features extraction.

Recently, Narayanan et al in [29] extracted dynamic features by Convolutional Neural Networks (CNN) and Recurrent Neural Network (RNN) techniques for detecting shilling attacks. Ran et al in [30] proposed a schema to address the difficulty of analyzing sensitivity by exploiting the polynomial representation of the objective function in a large-scale

recommender system. Ovaisi et al in [31] presented a software library for comprehensive robustness as Robustness Gym for Recommender System (RGRecSys) that evaluates the robustness of different recommender systems.

According to the topic of our proposed system, we focused our studied papers on the area of time and trust recommender system, the community detection algorithms, prediction, and the approaches for detecting the fake user or review as follows:

2.1 Trust based recommender systems

Trust among users can be explicitly collected from users or can be implicitly inferred from users' behaviors such as items rating information. Ahmadian et al in [32] used a probabilistic model to determine how many ratings are required for each user to produce an accurate prediction. Moreover, they used the implicit ratings based on a reliability measure, for enhancing the rating profiles of users with insufficient ratings. In implicit methods, a trust network is constructed for each user based on her/his ratings in the system. Implicit relationships in the recommendation process can be effective to improve the performance of social recommender systems, especially for the users whose explicit relationships are insufficient to make accurate recommendations. Ahmadian et al. in [33] proposed a social recommender system based on reliable implicit relationships. Dempster–Shafer theory is used as a powerful mathematical tool to calculate the implicit relationships. Moreover, a new measure is introduced to evaluate the reliability of predictions, where unreliable predictions are recalculated using a neighborhood improvement mechanism. This mechanism used a confidence measure between the users to identify ineffective users in the neighborhood of the target user. Finally, new reliable ratings are calculated by removing the identified ineffective neighbors. The standard content-based or collaborative-filtering recommendation approaches may address this problem by asking users to share their data. Wahab et al in [34], solved the cold-start problem for new items using a federated learning-based approach. The originality of their solution compared to existing federated learning-based solutions comes from applying federated learning specifically to the cold-start problem and proposing a mechanism to calculate trust rating and then a double deep Q learning scheduling approach.

2.2 Time based recommender systems

The main motivation for a time-aware recommender system is that in realistic scenarios users' tastes might change over time (user drift). Most traditional recommender algorithms exploit the data as static and do not consider this issue into account. Both memory-based and model-based CF approaches have been recently adopted to take into account users' drifts in their recommendation processes. In time window algorithms only a number of the most recent ratings are considered in the recommendation process. In time decay-based methods, the temporal effect of the data is handled by boosting recent ratings and penalizing older ratings. Rezaimehr et al. in [35] introduced a novel time-aware recommendation algorithm that is based on identifying overlapping community structures among users. As mentioned earlier, user interests may change over time. Accurately dynamic user preferences modeling is a challenge in designing efficient personalized recommendation systems. System popularity is constantly changing as new selection emerges. Thus, modeling temporal dynamics should be important. Classical time-window or instance-decay approaches have a lot of challenges, as they lose too many data instances. Rendle et al. in

[36] proposed a matrix factorization method that includes temporal effects in RSs. Their behavior model change with the time throughout the life span of the data. They exploit the relevant components of all data instances, while discarding only what is modeled as being irrelevant. Daneshmand et al. in [37] proposed a time-aware recommender system that models dependencies between items using statistical diffusion. They indicated that future ratings of a user can be detected from his/her rating history. There is a hidden network structure among the items and each user tracks a sequence of items in this network. Moradi et al. in [38] introduced a novel trust-aware recommendation algorithm based on overlapping community structures between users. Users' interests might change over time, and thus dynamic modeling of users' tastes accurately is a challenging issue to design the efficient personalized recommendation systems.

2.3 Community detection

Community detection algorithms are used to cluster users and find neighbors of active users and it is very crucial in social network analysis. Many methods used topological features and node attributes, including community size, modularity, clustering, dynamic, and overlapped clusters [39]. These algorithms concentrated on the undirected/directed graph that used real and artificial datasets. Most existing community detection algorithms, ignore the structural information of the posts and the semantic information of users' interests and only use network topology. Jiang et al. in [40] Used an Interest Community Detection model to address these challenges. Their model is based on modified Hypertext. Recently, Al-Gorabi et al in [41] proposed a Node Influence k-Nearest Neighbor (NI-KNN) algorithm for detecting the community using an undirected graph. In addition, there are various applications that benefit community detection methods, including healthcare, economics, social media, e-commerce, communication networks, etc. In Table 1, we compared the mentioned algorithms from the following dimensions.

2.4 Sentiment analysis

Shubham and Krishan Kumar used the concept of lexicon-based approach to give star rating to the reviews. Instead of classifying reviews as positive, negative or neutral, they have categorized them into five subcategories, i.e., Excellent, Good, Neutral, Bad and Worst using the star-scale method. They got the results in a reasonable time by using the concept of Virtual Machines [42].

Negi et al. in 2021 used the residual learning-based framework, "ResNet-50", with transfer learning to propose an intelligent human action recognition system that can automatically recognize daily human activities. They present extensive empirical evidence demonstrating that residual networks are simpler to optimize and can gain accuracy from significantly higher depths [43].

Kumar et al. in 2017 employed a lightweight Docker container over the cloud as a utility for sentiment analysis using the four classification approaches. They analyzed the reviewer's comment on a product across multiple websites. The analyzed information can be used as a product recommendation to a customer [44].

Negi et al. in 2021 proposed an advanced CNN technique to care for farming by classifying a recognizing citrus disease that helps grow healthy plants. Their proposed model uses different training epochs, batch sizes, and dropouts. The dataset includes images of unhealthy and healthy citrus leaves and fruits used in Deep learning technique [45].

Table 1 The characteristics of different algorithms

	Paper	Purpose	Approach	Data set
Trust based	Ahmadian et al. [32]	Solve the sparsity	Probabilistic	Epinions Flixster
	Ahmadian et al. [33]	Calculate the implicit relationships between user	Dempster–Shafer theory	FilmTrust Flixster Epinions
Time based	Wahab et al [34]	Solve cold start problem Calculate trust rating	Federated learning-based double deep Q learning	MovieLens 1 M Epinions
	Rezaimehr et al. [35]	Identifying overlapping community structures among users	Overlapping community detection	MovieLens-100K MovieLens-1M
	Rendle et al. [36]	Exploit the relevant components of all data instances	Matrix factorization method	Tmall CIKM
	Daneshmand et al. [37]	Model the dependencies between items using statistical diffusion	Hidden network structure among the items	Netflix MovieLens
Community detection	Moradi et al. [38]	Modeling of dynamic users' tastes	Overlapping community detection Dynamic modeling of users' tastes	MovieLense-1G
	Jiang et al. [40]	User Interest Community Detection	Community detection based on modified Hypertext	
	Al-Gorabi et al. [41]	Detect the community using undirected graph.	Node Influence k-Nearest Neighbor (NI-KNN)	The data obtained from Google locations

Table 1 (continued)

Paper	Purpose	Approach	Data set
Shubham and Krishan [42]	Classify the reviews into five sub-categories, i.e., Excellent, Good, Neutral, Bad and Worst	Lexicon-based approach	Different amazon product
Negi et al. [43]	Propose an intelligent human action recognition system that can automatically recognize daily human activities	Residual learning	UTKinect Action-3D public
Kumar et al. [44]	Design an utility for sentiment analysis using the four classification approaches	Lightweight Docker container over cloud	NLTK benchmark movie review
Negi et al. [45]	Classify citrus disease that helps grow healthy plants	Advanced CNN	Canker, Black spot, Greening, Scab and Melanose
Negi et al. [46]	Face mask detection in real-time video streams.	Advanced Deep Learning	ImageNet
Sharma et al. [47]	Dictionary-based sentiment analysis	Advanced model for a dictionary-based sentiment analysis algorithm	Twitter dataset (self-labeled) movie reviews
Sharma et al. [48]	Recognize emotions based on lip structure over time	Deep facial expression recognition system	Pictures
Kumar et al. [49]	Summarize the events in multi view videos	local-alignment-based FASTA approach	Office Lobby BL-7F
Vijayvergia et al. [50]	Solve the problem of review ratings	STAR model	Amazon Electronics Amazon Apps for android Amazon Digital music products Amazon Book Kaggle Movie reviews Kaggle Combined reviews

Table 1 (continued)

Prediction approach	Paper	Purpose	Approach	Data set
	Kumar et al. [51]	Predict the price in the stock market.	recurrent neural network	The price of Google
	Negi et al. [52]	Identify the leaf disease	Deep convolutional neural network	Large agricultural plant
	Alok et al. [53]	Classify the malaria infected	Deep convolutional neural networks	Pathologist-cured visual
	Kumar et al. [54]	Classify of Indian liver patients	classification	Therapeutic
	Negi et al. [55]	Recognize whether or not a person uses a mask	deep learning	RMFD LFW SMFD
	Kumar et al. [56]	Event detection and summarization in multi-view surveillance video	boosting	Office Lobby BL-7F

Negi et al. in 2021 proposed an advanced Deep Learning model for face mask detection in real-time video streams. They analyzed a series of video streams/images to recognize individuals who comply with the regulation of wearing medical masks by the government. This could allow the government to take sufficient action against non-compliant persons [46].

Sharma et al. in 2017 presented an advanced model for a dictionary-based sentiment analysis algorithm for human sentiment analysis. They believed that public sentiment in future and past events, such as public gatherings, and government policies, reflects public beliefs and can be used to analyze measures of support, disorder, or disruption in such situations [47].

Sharma et al. in 2017 used a new approach, D-FES, to recognize emotions based on lip structure over time. Using a recurrent neural network to analyze the pattern over time provided a classification of emotions into 6 different classes. They believed that their proposed system can work in a real-time environment to accurately track and classify emotions [48].

Kumar et al. in 2017 proposed a local-alignment-based FASTA approach for summarizing the events in multi view videos as a solution to the aforementioned problems. A deep learning framework is used to extract features to solve the problem of changes in illumination and to remove fine texture details and identify objects in a frame. Interview dependencies are then captured across multiple views of the video through local alignment via the FASTA algorithm. Finally, object tracking is applied to extract frames with low activity [49].

Vijayvergia et al. in 2018 proposed the STAR model and highlighted two Phases of the model based on novel approach using a transfer learning framework to analyze the reviews by exploiting the variation in human emotions [50]. Their experimental results have shown that the star model outperforms the state-of-the-art models.

2.5 Prediction approach

Kumar et al. in 2021 proposed a price model predicted in the stock market. Their model predicted the price of Google Stock using two deep learning models with the least possible error. They applied the Recurrent Neural Network for predicting the price on a short-term basis [51].

Negi et al. in 2021 described a deep convolutional neural network model to care for farming by identifying leaf disease. They believed that leaf disease detection helps in growing up the healthy plants. They used CNN techniques on large datasets of agricultural plants to accurately diagnose plant leaf diseases [52].

Alok et al. in 2021 investigated whether artificial intelligence can use low-cost, reliable and accurate deep learning solutions to predict lethal malaria. They reported some findings on the accurate classification of malaria infected cells using deep convolutional neural networks. They also discussed strategies for assembling a pathologist-treated visual dataset for training deep neural networks, as well as some data augmentation approaches used to significantly increase the dimensionality of the data, given the over fitting problem, especially in deep CNN training [53].

Kumar et al. in 2021 proposed a classification algorithm in the therapeutic dataset. Their approach used the dataset of Indian Liver Patient and their experimental result clearly showed that grouping classification algorithms efficiently improve the rate of illness prediction [54].

Negi et al. in 2020 trained a customized deep learning model to recognize whether or not a person uses a mask. They studied the concept of model pruning with Keras-Surgeon. Model pruning can be efficient in reducing model size so that it can be easily implemented and inferred in embedded systems [55].

Kumar et al. in 2018 proposed an accurate and efficient technique for event detection and summarization in multi-view surveillance videos using boosting as machine learning technique. Interview dependencies are captured across multiple views of the video are captured through weakly learning classifiers in boosting algorithm. The light changes and still frames are controlled by the deep learning framework with moving an object in the frame. It helps to make the correct decision for active frame and inactive frame without any prior information about the number of issues in a video [56].

Table 1 shows the characteristics of different methods which have discussed in this section.

3 Proposed method

The proposed T&TRS system consists of seven steps as shown in Fig. 2. The step 3 contains the novel similarity measure for constructing the user-user-network based on implicit and explicit trust between users. The step 4 contains the new community detection algorithm for clustering the users based on the time of rated items and trust between users. According to the detected communities, the proposed system detects and removes the items ratings of fake's users and suspect items to improve the efficiency of the system.

We explain the detail of proposed system, T&TRS, based on the Fig. 2 as follows:

Step 1: Inject attack

First, we injected many Average and Random attacks using the tools has been introduced by Rezaimehr and Dadkhah in [57] on the Epinion datasets and the details of the

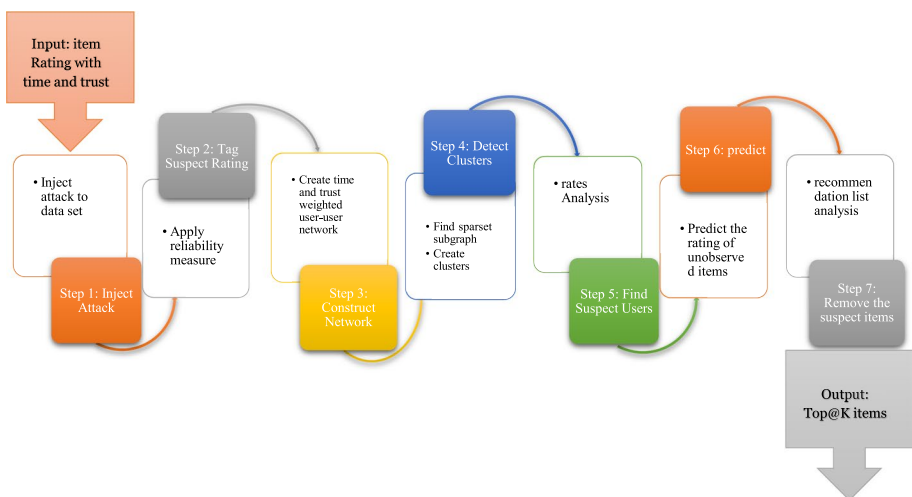


Fig. 2 The structure of proposed method; T&TRS

attacks such as type, strategy and other parameters has explained in the evaluation section. They designed a new user-friendly tool to inject any type of attack into the selected data set for collaborative filtering recommender systems based on Attack strategies.

Step 2: Tag suspect ratings

Each item rating recorded in the dataset must be validated; for this purpose, a reliability value [58] is calculated using Eq. (1) based on positive and negative factors for each entry in the initial rating matrix. T&TRS labeled the item rating with a reliability value less than the threshold as a suspicious rating and the others are considered as the correct and reliable ratings.

$$R_{a,i} = \left(fp(r_{a,i}) \cdot fn(r_{a,i}) \right)^{\frac{1}{1+fn(r_{a,i})}} \quad (1)$$

Where, $f_p(r_{a,i})$ and $f_n(r_{a,i})$ are the positive and negative factors for rating of item; i that is given by the user; a , which are calculated from Eq. (2) to Eq. (6).

$$fp(r_{a,i}) = 1 - \frac{\bar{s}}{\bar{s} + \sum_{u \in N_{a,i}} sim_{a,u}} \quad (2)$$

$$fn(r_{a,i}) = \left(\frac{\max - \min - V_{a,i}}{\max - \min} \right)^\gamma \quad (3)$$

Where, \bar{s} is the mean value of similarity between users ($N_{a,i}$) that rated item i .

$$sim_{a,u} = \frac{\sum_{i=1}^m (r_{a,i} - \bar{r}_a)(r_{u,i} - \bar{r}_u)}{\sqrt{\sum_{i=1}^m (r_{a,i} - \bar{r}_a)^2 \sum_{i=1}^m (r_{u,i} - \bar{r}_u)^2}} \quad (4)$$

$$\gamma = \frac{\ln 0.5}{\ln \frac{\max - \min - \bar{V}}{\max - \min}} \quad (5)$$

\bar{r}_u is the mean item ratings that is given by user u , \max and \min are the maximum and minimum item ratings in the initial rating matrix and $r_{u,i}$ is the rating that is given by user u for item i .

$$V_{a,i} = \frac{\sum_{u \in N_{a,i}} sim_{a,u} \cdot (r_{u,i} - \bar{r}_u - r_{a,i} + \bar{r}_a)^2}{\sum_{u \in N_{a,i}} sim_{a,u}} \quad (6)$$

If the reliability value of item rating is more than a threshold that was tuned by systems, then that rating item is a suspect rating.

Step 3: Construct network

A weighted user-user network is created using the rating matrix and Trust relationships among users. The weight for each item rating in the initial rating matrix with time t is calculated as Eq. (7).

$$W_{a,i}^t = \begin{cases} e^{-(T-t)/\theta} & \text{if } t > 0 \\ 0 & \text{otherwise} \end{cases} \tag{7}$$

Where, T indicates the last time that item; *i* was rated by users.

θ is a parameter specified by the user and the value is between 0.25 and 1.25. It should be noted that the higher value of this parameter indicates less effectiveness of time. Accordingly, the new rating matrix (R_t) is generated and its entries are calculated as Eq. (8).

$$tr_{a,i} = \begin{cases} r_{a,i} \times W_{a,i}^t & \text{if } r_{a,i} > 0 \\ 0 & \text{otherwise} \end{cases} \tag{8}$$

T&TRS calculated the implicit trust between users *u* and *a* as Eqs. (9) and (10).

$$Trust - imp_{a,u} = \frac{d_{max} - d_{a,u} + 1}{d_{max}} \tag{9}$$

Where $d_{a,u}$ is the distance between users *u* and *a* according to the explicit trust in the network.

$$d_{max} = \frac{\ln(n)}{\ln(k)} \tag{10}$$

Where, *n* and *k* indicate the size and average degree of users in the explicit trust network, respectively.

T&TRS calculated the final similarity between users *a* and *u*, $f-sim_{a,u}$, as Eq. 11. This similarity has calculated based on trust between user and time of item ratings using Eq. (12).

$$f-sim_{a,u} = \begin{cases} \frac{2 \times t_sim_{a,u} \times Trust - imp_{a,u}}{t_sim_{a,u} + Trust - imp_{a,u}} & t_sim_{a,u} > 0 \text{ and } Trust - imp_{a,u} > 0 \\ t_sim_{a,u} & t_sim_{a,u} > 0 \text{ and } Trust - imp_{a,u} \leq 0 \\ Trust - imp_{a,u} & t_sim_{a,u} \leq 0 \text{ and } Trust - imp_{a,u} > 0 \\ 0 & t_sim_{a,u} \leq 0 \text{ and } Trust - imp_{a,u} \leq 0 \end{cases} \tag{11}$$

$$t_sim_{a,u} = \frac{\sum_{i \in I_{a,u}} (tr_{a,i} - \bar{tr}_a)(tr_{u,i} - \bar{tr}_u)}{\sqrt{\sum_{i \in I_{a,u}} (tr_{a,i} - \bar{tr}_a)^2} \sqrt{\sum_{i \in I_{a,u}} (tr_{u,i} - \bar{tr}_u)^2}} \tag{12}$$

Where \bar{tr}_a is the average ratings that given by user *a* and $I_{a,u}$ indicates the items which are rated by both users *u* and *a*.

Step 4: Detect Clusters

In this phase, we proposed a novel community detection method that consists of three phases and find the sparsest sub graph of weighted user-user network. The pseudo-code of our proposed graph-based community detection as algorithm 1 is shown in Fig. 3. In the first phase, the initial centers of communities are obtained using a sparsest sub graph of weighted user-user network. After selecting the initial center of communities, our proposed community detection algorithm calculates the fitness function using Eq. 15 for

Algorithm 1. The proposed graph-based community detection method.

Inputs: $G = (V, E, H)$, $iter$.
 Output: List of communities: $\{C\}$.
 1: Find initial centers of communities $\{Init_C\}$ using Algorithm 2;
 2: Set $k' = |\{Init_C\}|$;
 3: Set $C_j = Init_{C_j}, \forall j = 1, \dots, k'$ be the initial center associated with j -th community C_j ;
 4: $v \leftarrow v - \{Init_C\}$;
 5: while $iter$ do
 6: for $N \in v$ Calculate fitness function f_N for $C_j, \forall j = 1, \dots, k'$ (using Eq. (15));
 7: $j = \text{Community with } \text{argmax } f_N$;
 8: **Add N To C_j** ;
 9: update initial center of C_j ;
 10: end for
 11: end while
 12: Calculate Overlapping community using Algorithm 3;
 13: Return $\{C\}$;

Fig. 3 Our proposed community detection algorithm

each node. The nodes add to the community with the maximum value of fitness function and the algorithm identifies the new initial centers. These steps will continue until all nodes have been attached to the communities. In the next step, our proposed algorithm merges the overlapped communities.

Figure 4 shows the pseud-code of algorithm 2 for selecting the initial centers of communities. It should be noted that the initial centers must have the maximum dissimilarities with each other based on the general concept of clustering and community detection algorithms. Consider the graph $G = (V, E, H)$ as a weighted user-user network, V, E

Algorithm 2. Finding initial centers of the graph.

Inputs: $G = (V, E, H), K > 0$, and $\varepsilon > 0$.
 Output: $\{Init_C\}$: Initial centers set of communities.
 1: $\{Init_C\} \leftarrow V, \{Init_C\} \leftarrow \emptyset$;
 2: while $\{Init_C\} \neq \emptyset$ do
 3: $\{\bar{A}(Init_C)\} \leftarrow \{\forall i \in \{Init_C\} | wd_{Init_C}(i) \geq (2 + 2\varepsilon) * \rho(\{Init_C\})\}$;
 4: Sort all $i \in \{\bar{A}(Init_C)\}$ descending based on their $wd_{Init_C}(i)$;
 5: Set $r = \frac{\varepsilon}{1+\varepsilon} \times |\{\bar{A}(Init_C)\}|$;
 6: Select **top_r** nodes from $\{\bar{A}(Init_C)\}$ as $\{A(Init_C)\}$;
 7: $\{Init_C\} \leftarrow \{Init_C\} - \{A(Init_C)\}$;
 8: if $|\{A(Init_C)\}| == k$ then
 9: $\{Init_C\} \leftarrow \{A(Init_C)\}$;
 10: break;
 11: else if $|\{A(Init_C)\}| > k$ and $\rho(\{\forall i \in \{A(Init_C)\} | \rho(i)\}) \leq AVG_\rho(\{Init_C\})$ then
 12: $\{Init_C\} \leftarrow i$;
 13: if $|\{A(Init_C)\}| == k$ then break;
 14: else
 15: $\{Init_C\} \leftarrow \{A(Init_C)\}$;
 16: end if
 17: end while
 18: return $\{Init_C\}$;

Fig. 4 Finding initial centers algorithm

and H indicate users, the relationship between users, and the weights of the relationship ($f_{sim_{a,u}}$), respectively. The density of a sub-graph $S \subseteq V$ is defined as Eq. 13.

$$\rho(S) = \frac{\sum_{e \in E(S)} H_e}{|S|} \tag{13}$$

Where $H_e \in H$ is the weight of the edge e and $E(S)$ denotes the edges set of the sub-graph S .

The weighted degree of each node a ($user\ a$) $\in S$ is calculated as Eq. 14.

$$wd_s(a) = \sum_{e_{au} \in E(S)} H_{e_{au}} \tag{14}$$

The candidate nodes set, $\tilde{A}(S)$ consists of the nodes which weighted degrees are higher than the threshold value $((2 + 2\epsilon) * \rho(S))$. The algorithm sorts the members of this set based on their weighted degrees and removes a portion of them with the highest weighted degree. The algorithm will be continued on the remaining nodes in an iterative process until the resulted sub-graph is non-empty. It should be noted that the output of the algorithm is a sub-graph with k nodes. These nodes can be considered as the initial centers of the graph.

After selecting the initial centers of communities, the users can be assigned to their nearest communities. For each user, T&TRS calculated the fitness function as Eq. 15. User has associated to the community, C_j which has a high value of fitness function. Then the centers of communities were updated in order to maximize a fitness function.

$$f_{v_a} = \frac{\sum_{v_i \in C_j, v_i \neq v_a} f_{sim_{a,u}}}{\sum_{v_u \in C_j, v_u \neq v_a} f_{sim_{a,u}} + \sum_{C_k \in C, C_k \neq C_j} \sum_{v_u \in C_k, v_u \neq v_a} f_{sim_{a,u}}} \tag{15}$$

Where C indicates the set of communities, $f_{sim_{a,u}}$ is the weight between the nodes v_a and v_u which is calculated as Eq. 11. This process is iteratively repeated until members of communities do not change and a steady state is achieved. A set of communities are identified where the users are assigned to their corresponding communities. Some of the communities may have overlap and they can be merged. The pseudo-code of the overlapping community algorithm [59] is shown in Fig. 5. If two communities have too many overlapping nodes, they should be merged into a single community according to the different sizes of communities. First, our propose system calculated the overlap ratio and the combined parameter as Eqs. 16 and 17 respectively. Then according to these values, our proposed system decides how to combine the communities based on the two rules as follow:

$$r_{pq} = \frac{\max(|C_p|, |C_q|)}{\min(|C_p|, |C_q|)} \tag{16}$$

Where C_p and C_q are the p th and q th overlapping communities.

$$\zeta = \frac{\rho_2 + \rho_1 \times \rho_2 - \rho_1}{\rho_1} \tag{17}$$

$\rho_1, \rho_2 \in [0, 1]$ are pre-designed thresholds.

Algorithm 3. Overlapping Community Algorithm.

Inputs: List of communities $\{C\}$, ρ_1, ρ_2 .

Output: List of final communities: $\{C'\}$.

$$\text{Calculate } \zeta = \frac{\rho_2 + \rho_2 \times \rho_1 - \rho_1}{\rho_1}$$

For $p, q \in C$ and $p \neq q$ Calculate $r_{pq} = \frac{\max(|C_p|, |C_q|)}{\min(|C_p|, |C_q|)}$

If $r_{pq} < \zeta$ then

$$\{ \text{Calculate } \delta_{pq} = \beta \times \frac{|C_p \cap C_q|}{|C_p \cup C_q|} + (1 - \beta) \times \frac{|NC_p \cap NC_q|}{|NC_p \cup NC_q|} \text{ (the size of } C_p \text{ and } C_q \text{ is equal)}$$

if $\delta_{pq} > \rho_1$, then combine p and q }

Else if $r_{pq} \geq \zeta$ then

$$\{ \text{Calculate } \delta_{pq} = \frac{|C_p \cap C_q|}{\min(|C_p|, |C_q|)} \text{ (size of one community is much smaller than the other)}$$

if $\delta_{pq} > \rho_2$, then combine p and q }

End For

Return $\{C'\}$;

Fig. 5 Overlapping community algorithm

If $r_{pq} < \zeta$ It means that the size of the two communities is different.

Rule 1: If the size of two communities is equal, then we calculated the overlapping proportion using Eq. (18) and if this value is greater than ρ_1 , our propose system will combine the two communities.

$$\delta_{pq} = \beta \times \frac{|C_p \cap C_q|}{|C_p \cup C_q|} + (1 - \beta) \times \frac{|NC_p \cap NC_q|}{|NC_p \cup NC_q|} \tag{18}$$

Where NC_p and NC_q are the set of neighbor nodes that directly connected to the nodes in C_p and C_q , respectively. $\beta \in [0, 1]$ is a parameter to determine the influence of internal nodes of the neighbors list.

Rule 2: If the size of two communities is different, then we calculated the overlapping proportion using Eq. 19 and if this value is greater than ρ_2 , our propose system will combine the two communities

$$\delta_{pq} = \frac{|C_p \cap C_q|}{\min(|C_p|, |C_q|)} \tag{19}$$

The final communities were used as the nearest neighbors set of the active user in the same community for items recommendation.

Figure 6 shows an example of our proposed graph-based community detection method. The weighted user-user network is shown in Fig. 6a. The initial centers of communities are identified in Fig. 6b where four nodes are considered as the centers set. Based on the general concept of clustering and community detection algorithms, the obtained centers should be dissimilar to each other as much as possible. Finally, the final communities are shown in Fig. 6c.

The most important problem to identify communities is the initial number of clusters and also the centers of the clusters which are selected randomly. In this method, the

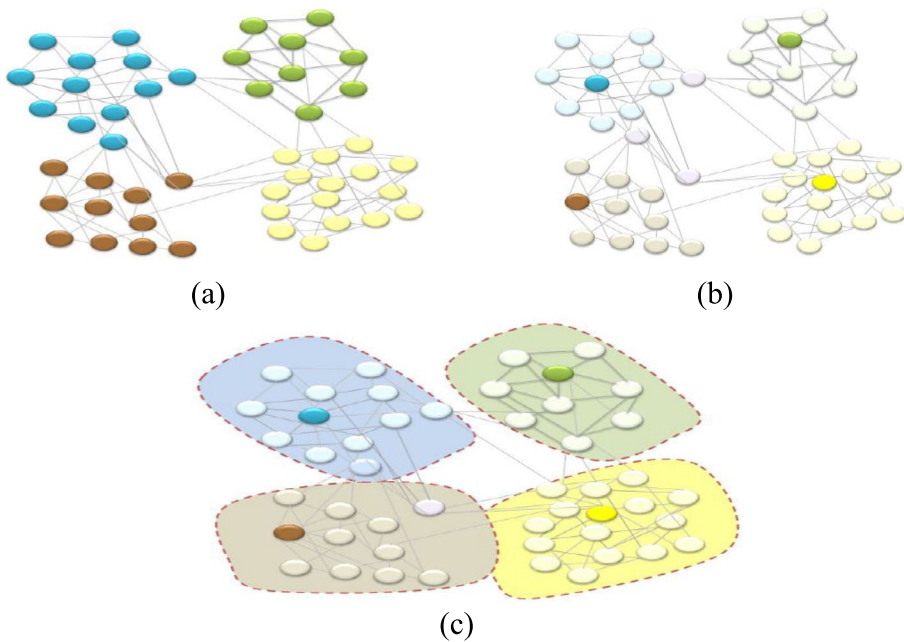


Fig. 6 **a** Example of trusted user-user network, **b** community centers, and **c** final communities

center of the clusters should be the nodes that have the maximum links with other nodes in the network.

Step 5: Identify the suspicious users

The similarity of these users with the center of the cluster is calculated Using Eq. (11) described in step 3 of proposed system. If this value was less than the threshold, that user is considered as a fake user and if all item rates given by this user are not labeled as fake, the user will not be removed from the collection, otherwise the user will not be considered in the prediction process.

Step 6: Predict the rating of unobserved items

In this step, our propose system recommends the items that relevant to the active user. The users that belongs to the same cluster which contains the active user have considered as the neighbors of active user. We proposed the novel formula as Eq. (20) to predict the rating of unobserved items for active user.

$$P_{a,i} = \bar{r}_a + \frac{\sum_{u \in N_{a,i}} f_sim_{a,u} (r_{u,i} - \bar{r}_u)}{\sum_{u \in N_{a,i}} f_sim_{a,u}} \tag{20}$$

Where, \bar{r}_a is the average rating given by user a , and $N_{a,i}$ is the set of neighbors of user a who rated item i , and $f_sim_{a,u}$ is the similarity between user a and user u , that was calculated as Eq. 11 described in the step 3 of proposed system. After ranking the unobserved

items based on the prediction values, the Top@N recommendation generated for active user.

Step 7: Remove the suspect items

In this step, the propose system check the Top@N recommendation for active users. Any item of the list which has been rated for more than 80% of fake users will delete and the remained items recommend to the active user.

4 Evaluation of T&TRS

We evaluated the proposed T&TRS system on the Epinions dataset contains 943 users, 1682 items, 3212 ratings, 3325 trust and 99.97% sparsity in two phases: Recommendation list and Attack detection. The Epinions dataset has been collected over several years. We grouped items into different months to better assess the impact of time on user tastes. Item ratings are 1 to 5, that 5 represent the highest rate. This dataset also contains the rating time and trust between users

We evaluated T&TRS based on Precision, Recall, F-measure, MAE, RMSE and RC measures, which can be calculated using Eqs. 21 to 26, respectively. In this paper, in addition to evaluate the Top@N items recommendations to the target user, the accuracy of detecting and deleting attack profiles was also calculated.

$$Precision = \frac{1}{n} \sum_{i=1}^n \frac{TP}{TP + FP} \quad (21)$$

$$Recall = \frac{1}{n} \sum_{i=1}^n \frac{TP}{TP + FN} \quad (22)$$

$$F1 = \frac{2 \times precision \times recall}{precision + recall} \quad (23)$$

In the above equations, True Positive (TP) represents the items that are attractive to users and are recommended to users, False Positive (FP) represents the items that are not attractive to users but are recommended to them, False Negative (FN) represents the items that are not recommended but users are interested in them and (TN) presents real negative.

$$MAE = \frac{\sum_{i=1}^N |r_i - p_i|}{N} \quad (24)$$

$$RMSE = \sqrt{\frac{1}{N} \sum_{i=1}^N (r_i - p_i)^2} \quad (25)$$

Where, p_i and r_i are the predicted and actual rates of item i , respectively and N is the number of all predicted rates.

Table 2 The precision of different algorithms

Algorithm	Top5	Top10	Top15	Top20	Top30
TRACCF [60]	0.9462	0.8981	0.8981	0.8981	0.5319
KMCF-U	0.7857	0.7857	0.7857	0.7857	0.7367
KMCF-I	0.7472	0.7361	0.7361	0.7361	0.7361
TOTAR[59]	0.8550	0.8521	0.8416	0.8281	0.8237
T&TRS	0.8502	0.8527	0.8543	0.8541	0.8543

Table 3 The Recall of different algorithms

Algorithm	Top5	Top10	Top15	Top20	Top30
TRACCF [60]	0.5239	0.5399	0.5399	0.5399	0.5481
KMCF-U	0.4158	0.4158	0.4158	0.4158	0.4257
KMCF-I	0.6481	0.6481	0.6481	0.6481	0.6481
TOTAR[59]	0.4356	0.5467	0.7322	0.7985	0.8134
T&TRS	0.8152	0.8509	0.8635	0.8657	0.8679

Table 4 The F1-measure of different algorithms

Algorithm	Top5	Top10	Top15	Top20	Top30
TRACCF [60]	0.6744	0.6744	0.6744	0.6744	0.5399
KMCF-U	0.5438	0.5438	0.5438	0.5438	0.5395
KMCF-I	0.6941	0.6893	0.6893	0.6893	0.6693
TOTAR [59]	0.5636	0.6660	0.7830	0.8130	0.8185
T&TRS	0.8323	0.8518	0.8588	0.8599	0.8610

Fig. 7 The comparison of methods based on RMSE, MAE and RC measures

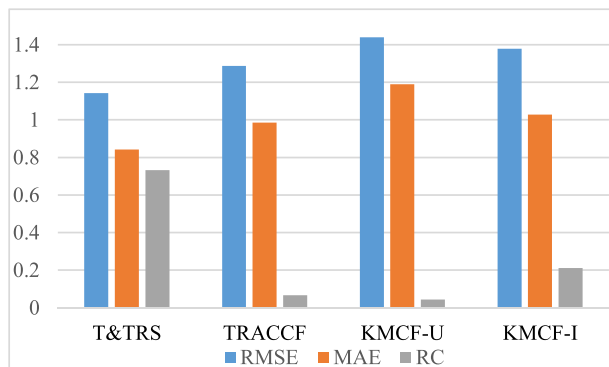


Table 5 Impact of θ parameter of proposed community detection algorithm

$\theta = 0.5$					
Top_N	5	10	15	20	30
Precision:	0.8502	0.8527	0.8534	0.8543	0.8543
Recall:	0.8152	0.8509	0.8588	0.8635	0.8679
F1:	0.8323	0.8518	0.8561	0.8588	0.8610
MAE:	0.8547				
RMSE:	1.1709				
RC:	0.7332				
$\theta = 0.25$					
Top_N	5	10	15	20	30
Precision:	0.8489	0.8514	0.8520	0.8529	0.8529
Recall:	0.8135	0.8496	0.8575	0.8622	0.8667
F1:	0.8308	0.8505	0.8548	0.8576	0.8598
MAE:	0.8424				
RMSE:	1.1420				
RC:	0.7322				
$\theta = 0.75$					
Top_N	5	10	15	20	30
Precision:	0.8502	0.8527	0.8534	0.8543	0.8543
Recall:	0.8152	0.8509	0.8588	0.8635	0.8679
F1:	0.8323	0.8518	0.8561	0.8588	0.8610
MAE:	0.8547				
RMSE:	1.1709				
RC:	0.7332				
$\theta = 1$					
Top_N	5	10	15	20	30
Precision:	0.8502	0.8527	0.8543	0.8541	0.8543
Recall:	0.815	0.8509	0.8635	0.8657	0.8679
F1:	0.8323	0.8518	0.8588	0.8599	0.8610
MAE:	0.8547				
RMSE:	1.1709				
RC:	0.7332				

$$RC = \frac{\text{\#number of predicted ratings}}{\text{\#number of all ratings}} \quad (26)$$

4.1 Community detection

We evaluated our proposed algorithm on the Epinions data set and compared the results with a time-based recommender system. Tables 2, 3 and 4, show precision, recall, and F1 measures of different algorithms for $N = 5, 10, 15, 20, 25$ and 30 for Top@N recommendation respectively. As shown in Table 3, TRACCF [60] method has the best

Table 6 Impact of φ parameter of proposed community detection algorithm

$\varphi = 0.5$					
Top_N	5	10	15	20	30
Precision	0.8614	0.8637	0.8643	0.8651	0.8651
Recall	0.7965	0.8296	0.8369	0.8412	0.8453
F1	0.8277	0.8463	0.8503	0.8530	0.8551
MAE	0.8846				
RMSE	1.2307				
RC	0.7425				
$\varphi = 0.8$					
Top_N	5	10	15	20	30
Precision	0.8502	0.8527	0.8543	0.8541	0.8543
Recall	0.8152	0.8509	0.8635	0.8657	0.8679
F1	0.8323	0.8518	0.8588	0.8599	0.8610
MAE	0.8547				
RMSE	1.1709				
RC	0.7332				
$\varphi = 1$					
Top_N	5	10	15	20	30
Precision	0.8475	0.85	0.85	0.8516	0.8516
Recall	0.8274	0.8637	0.87	0.8765	0.8810
F1	0.8373	0.8568	0.86	0.8639	0.8661
MAE	0.8362				
RMSE	1.1344				
RC	0.7312				

performance for all values of N except the value 30 and the precision decreases when the value of N increases. As shown in Tables 3 and 4, our proposed method, T&TRS, outperforms the two K-means methods for User-based CF (KMCF-U) and Item-Based CF(KMCF-I), TOTAR [59] algorithms and TRACCF algorithm for N=30. The performance of T&TRS is better than TRACCF according to the recall and F1-measure.

Figure 7 illustrates the performance of different recommender systems in terms of MAE, RMSE, and RC measures. Our proposed method has the lowest MAE and RMSE values in comparison to the other methods.

Our proposed community detection algorithm includes two adjustable parameters (φ and θ), and its performance depends on assigning proper values for these parameters. Tables 5 and 6 show the experimental results considering the different values of these parameters. According to these results, it can be seen the best value for $\theta = 0.25$ and $\varphi =$

Table 7 Time complexity of the different algorithms

Algorithm	Time	Trust	Graph-based	Time complexity
TRACCF [60]	No	Yes	No	$O(inmc + nm)$
KMCF-U	No	No	No	$O(inmc + nm)$
KMCF-I	No	No	No	$O(inmc + nm)$
TOTAR [59]	Yes	No	Yes	$O(n^2m)$
T&TRS	Yes	Yes	Yes	$O(n^2m)$

Table 8 Experimental results of the T&TRS

Algorithm		Top5	Top10	Top15	Top20	Top30	MAE	RC	RMSE
T&TRS(without attack)	Precision	0.8543	0.8541	0.8543	0.8527	0.8502	0.8424	0.73223	1.14208
	Recall	0.8679	0.8657	0.8635	0.8509	0.8152			
	F1	0.8610	0.8599	0.8588	0.8518	0.8323			
T&TRS(with Average Attack- Push)	Precision	0.6547	0.7666	0.7727	0.8392	0.9428	3.88	0.5	4.0704
	Recall	0.8269	0.5316	0.5153	0.3628	0.1923			
	F1	0.7308	0.6278	0.6183	0.5066	0.3194			
T&TRS(With Average Attack- Nuke)	Precision	0.8134	0.765	0.5765	0.56	0.5401	3.75	0.62	4.0094
	Recall	0.7535	0.7806	0.654	0.7887	0.827			
	F1	0.7823	0.7727	0.6128	0.6549	0.6534			
T&TRS(With Random Attack- Nuke)	Precision	0.8087	0.6654	0.5765	0.7456	0.6301	2.465	0.67	3.2874
	Recall	0.7976	0.6645	0.6764	0.7987	0.7707			
	F1	0.8031	0.6649	0.6224	0.7712	0.6933			
T&TRS(With Random Attack- Push)	Precision	0.7934	0.7	0.7654	0.6543	0.6745	2.844	0.58	3.576
	Recall	0.6544	0.6544	0.6755	0.8765	0.7654			
	F1	0.7172	0.6764	0.7176	0.7492	0.7170			



Fig. 8 Precision for Top@N item recommendation

1 for MAE, RMSE and RC measures. So, we considered these values for evaluating our proposed method.

Table 7 shows the time complexity of the proposed system compared to the other systems. n is a number of users and m is a number of items. In cluster-based methods, the number of clusters must be predefined. For example, in TRACCF [60], KMCF-U and KMCF-I, C denotes the number of clusters and k denotes the maximum number of



Fig. 9 Recall for Top@N item recommendation

iterations. In real-world cases, k and c are often small ($k, c < n$), and thus can be ignored in the time complexity analysis. Also in the clustering method, the number of iterations is considered as a fixed number, so the time complexity of TRACCF [60] can be reduced to $O(nm)$. On the other hand, graph-based methods such as T&TRS and TOTAR [59] need to create a graph of users and items in their processes and their time complexity is $O(n^2m)$. Moreover, the graph-based methods require $O(n^2m + nm)$ memory, while the memory complexity of the other methods is $O(nm)$.

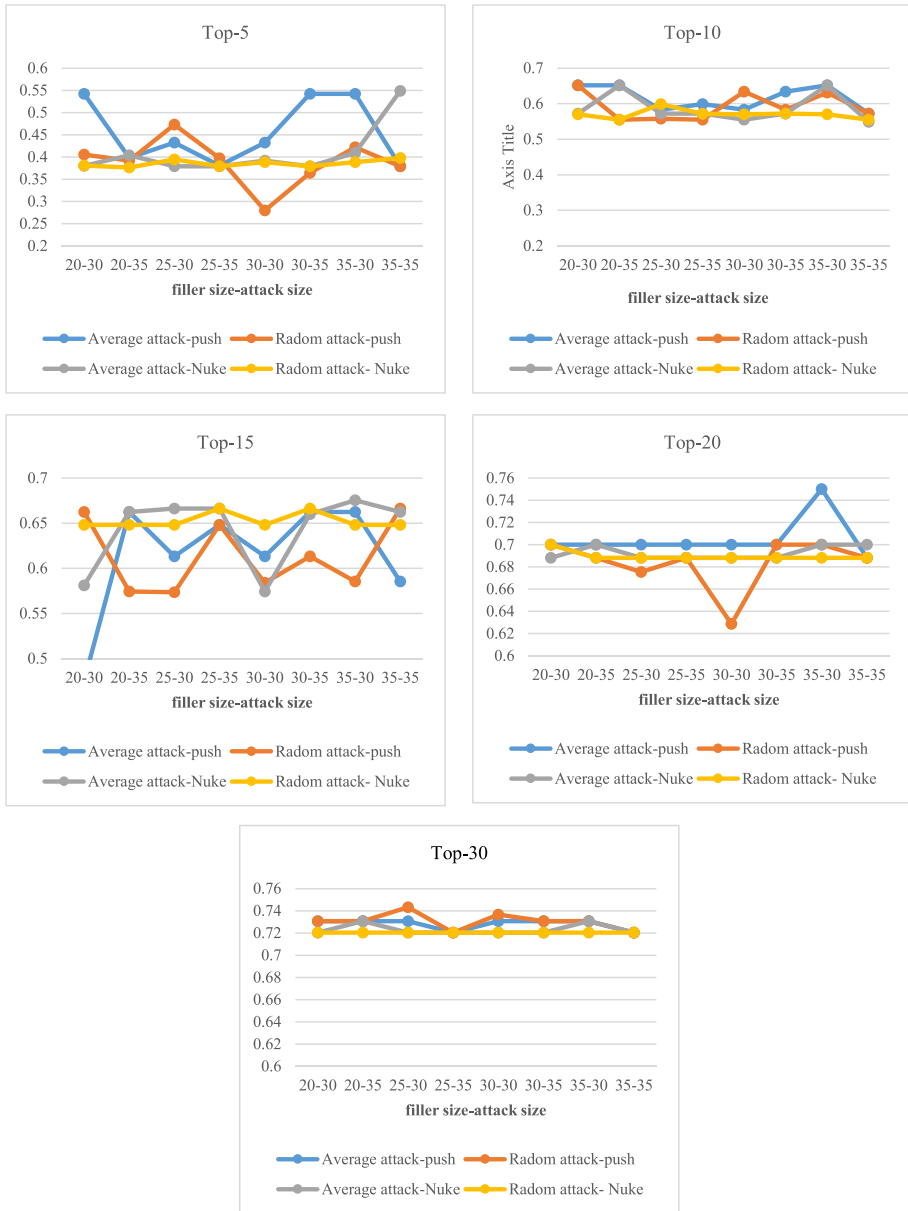


Fig. 10 F1 measure for Top-N item recommendation

However, the time and memory complexities of the proposed method, T&TRS, are the same as graph-based methods such as TOTAR [59]. On the other side, the time complexity of TRACCF [60] is better than T&TRS, but it does not consider the Time concept in the recommendation process, and their MAE, and RMSE values are more than T&TRS.

Table 9 Evaluation of T&TRS after removing attacks

filler-size	Attack-size	Shilling Attack	Attack Type	Precision	Recall	F1	Accuracy
20	30	Push	Average	0.2396	0.0483	0.0804	0.9352
			Random	0.2396	0.0483	0.0804	0.891
		Nuke	Average	0.2116	0.0483	0.0786	0.9188
			Random	0.153	0.025	0.0429	0.9301
	35	Push	Average	0.2411	0.0485	0.0808	0.9253
			Random	0.2348	0.05	0.0824	0.9192
		Nuke	Average	0.224	0.04	0.0678	0.9294
			Random	0.1818	0.0285	0.0493	0.9284
25	30	Push	Average	0.248	0.048	0.081	0.939
			Random	0.24	0.05	0.0827	0.9331
		Nuke	Average	0.2211	0.0383	0.0653	0.9403
			Random	0.2211	0.0383	0.0653	0.8941
	35	Push	Average	0.2556	0.0485	0.0816	0.9335
			Random	0.2755	0.05	0.0846	0.9417
		Nuke	Average	0.0769	0.01	0.0176	0.9212
			Random	0.2966	0.05	0.0855	0.9509
30	30	Push	Average	0.24	0.048	0.08	0.935
			Random	0.2396	0.0483	0.0804	0.9352
		Nuke	Average	0.2116	0.0483	0.0786	0.9188
			Random	0.153	0.025	0.0429	0.9301
	35	Push	Average	0.2411	0.0485	0.0808	0.9253
			Random	0.224	0.04	0.0678	0.9294
		Nuke	Average	0.2348	0.05	0.0824	0.9192
			Random	0.1818	0.0285	0.0493	0.9284
35	30	Push	Average	0.248	0.048	0.081	0.939
			Random	0.2478	0.0483	0.0808	0.9208
		Nuke	Average	0.2248	0.0483	0.0796	0.927
			Random	0.2326	0.05	0.0825	0.9311
	35	Push	Average	0.2727	0.0471	0.0803	0.9437
			Random	0.2554	0.05	0.0836	0.9314
		Nuke	Average	0.2809	0.0485	0.0828	0.9458
			Random	0.2881	0.0485	0.0831	0.9488

4.2 Attack detection

To evaluate the proposed system, a number of fake profiles as attack were first added to Epinion dataset, including 20 fake profile sizes, each of which had 20 random and average attacks in a compressive and declining manner through tools introduced in [57] by adding the ratings between 1 to 5. The results obtained after applying the proposed method to the dataset with and without attack for $TOP@N$ recommendation can be seen in Table 8. These results show the negative impact of attacks on the performance of the recommender system.

Table 10 Parameter values of θ to evaluate T&TRS

		$\theta = 0.5$				
Top_N	5	10	15	20	30	
Precision	0.8502	0.8527	0.8534	0.8543	0.8543	
Recall	0.81523	0.8509	0.8588	0.8635	0.8679	
F1	0.832395	0.8518	0.8561	0.8588	0.8610	
MAE			0.8547			
RMSE			1.17098			
RC			0.7332			
		$\theta = 0.25$				
Top_N	5	10	15	20	30	
Precision	0.8489	0.8514	0.8520	0.8529	0.8529	
Recall	0.8135	0.8496	0.8575	0.8622	0.8667	
F1	0.8308	0.8505	0.8548	0.8576	0.8598	
MAE			0.8424			
RMSE			1.1420			
RC			0.7322			
		$\theta = 0.75$				
Top_N	5	10	15	20	30	
Precision	0.8489	0.8514	0.8520	0.8529	0.8529	
Recall	0.8135	0.8496	0.8575	0.8622	0.8667	
F1	0.83088	0.8505	0.85482	0.8576	0.8598	
MAE			0.8424			
RMSE			1.1420			
RC			0.7322			

Figures 8, 9 and 10 show the results of Precision, Recall and F1 measures for random and average attack in deference filler and attack size conditions.

Table 9 shows the evaluation results of proposed method after detecting fake users and items and removing them from the data set.

Table 11 Parameter values of *threshold* to evaluate T&TRS

		<i>threshold</i> = 0.5				
Top_N	5	10	15	20	30	
Precision	0.8614	0.8637	0.8643	0.8651	0.8651	
Recall	0.7965	0.8296	0.8369	0.8412	0.8453	
F1	0.8277	0.8463	0.8503	0.8530	0.8551	
MAE	0.8846					
RMSE	1.2307					
RC	0.7425					
		<i>threshold</i> = 0.8				
Top_N	5	10	15	20	30	
Precision	0.8501	0.8507	0.8316	0.8516	0.8516	
Recall	0.8637	0.87179	0.8724	0.8765	0.8810	
F1	0.8568	0.8611	0.8628	0.8639	0.8661	
MAE	0.8362					
RMSE	1.1344					
RC	0.7312					

In order to set the optimal value of the parameter θ and threshold different values of 0.25,0.5,0.75 and values of 0.25,0.8 were examined, respectively. For the final evaluation of the proposed system according to Tables 10 and 11, the values of 0.5 and 0.1 are considered the best values for the parameters of θ and threshold, respectively.

5 Conclusion and future work

Attack detection is one of the major challenges of collaborative filtering recommender systems. If recommender systems able to detect attacks, then they can increase the precision of Top@K items recommendations. Our proposed collaborative filtering recommender system, T&TRS, reduced the impact of Average and Random attacks by identifying and eliminating attack profiles. In our proposed system, a new method of community detection has been introduced, that has increased the precision of clustering for finding fake users. The most important problem to identify communities is the initial number of clusters and also the centers of the clusters which are selected randomly. In T&TRS, the nodes with high links in the generated user-user network are selected as the center of the clusters. We evaluated our proposed community detection algorithm on the Epinions data set and compared the results with the similar methods. The experiential results showed that the precision of our community detection method was superior to theirs. The precision of the algorithm for Top-30 items recommendation was 0.8543 that shows a 0.03 improvement by comparison with the other algorithm. The F1-measure value of our proposed community detection algorithm was also better than the other algorithms. However, the time and memory complexities of the proposed community detection method are the same as graph-based methods such as TOTAR. The overall performance of T&TRS with this community detection algorithm is better than TRACCF according to F1-measure and has the lowest MAE and RMSE values in comparison to the other models

The experimental results of T&TRS evaluation on the Epinion data set by injecting the attacks have shown that the precision of recommendations has been increased after the attacks have been detected and the rating of fake users in the rating matrix has been removed.

Unfortunately, the strategies of attack often mix together to carry out the actual attacks. In future work, the proposed system will be evaluated on these attacks called group attacks and we will introduce a framework to design a robust collaborative filtering Recommender system against all types of attacks.

Declarations

Conflicts of interest We declare that this manuscript is original, has not been published before, and is not currently being considered for publication elsewhere. We know of no conflicts of interest associated with this publication, and there has been no significant financial support for this work that could have influenced its outcome.

References

1. Aggarwal CC (2016) Recommender systems. Springer

2. Ormel I, Onu CC, Magalhaes M, Tang T, Hughes JB, Law S (2021) Using a mobile app-based video recommender system of patient narratives to prepare women for breast cancer surgery: development and usability study informed by qualitative data. *JMIR Form Res* 5(6):e22970. <https://doi.org/10.2196/22970>
3. Khalaji M, Dadkhah C, Gharibshah J (2021) Hybrid movie recommender system based on resource allocation. *The CSI Journal on Computer Science and Engineering*. 10.48550/arXiv.2105.11678
4. Chawla S (2021) Web page recommender system using hybrid of genetic algorithm and trust for personalized web search. In: *Research Anthology on Multi-Industry Uses of Genetic Programming and Algorithms*. IGI Global, pp 656–675
5. Piao J, Zhang G, Xu F, Chen Z, Zheng Y, Gao C et al (2021) Bringing friends into the loop of recommender systems: An exploratory study. *Proc ACM Human-Comput Int* 5(CSCW2):1–26. <https://doi.org/10.1145/3479583>
6. Beg S, Anjum A, Ahmad M, Hussain S, Ahmad G, Khan S et al (2021) A privacy-preserving protocol for continuous and dynamic data collection in IoT enabled mobile app recommendation system (MARS). *J Netw Comput Appl* 174:102874. <https://doi.org/10.1016/j.jnca.2020.102874>
7. Francia M, Gallinucci E, Golfarelli M (2022) COOL: A framework for conversational OLAP. *Inf Syst* 104:101752. <https://doi.org/10.1016/j.is.2021.101752>
8. Alone V, Gangawane M, Barahate S, Shintre A, Bagewadi S (2022) Travel recommender system for social media. Available at SSRN 4114101. 3 <https://doi.org/10.2139/ssrn.4114101>
9. Alone V, Gangawane M, Barahate S, Shintre A, Bagewadi S (2022) Travel recommender system for social media. Available at SSRN 4114101. <https://doi.org/10.2139/ssrn.4114101>
10. Alenezi T, Hirtle S (2022) Normalized attraction travel personality representation for improving travel recommender systems. *IEEE Access*. 1 - <https://doi.org/10.1109/ACCESS.2022.3178439>
11. Forouzandeh S, Rostami M, Berahmand K (2022) A hybrid method for recommendation systems based on tourism with an evolutionary algorithm and topsis model. *Fuzzy Information and Engineering*, pp. 1–25 <https://doi.org/10.1080/16168658.2021.2019430>
12. Alamoodi A, Mohammed R, Albahri O, Qahtan S, Zaidan A, Alsattar H, et al (2022) Based on neutrosophic fuzzy environment: a new development of FWZC and FDOSM for benchmarking smart e-tourism applications. *Complex & Intelligent Systems*, pp 1–25. <https://doi.org/10.1007/s40747-022-00689-7>
13. Islek I, Oguducu SG (2022) A hierarchical recommendation system for E-commerce using online user reviews. *Electron Commer Res Appl* 52:101131. <https://doi.org/10.1016/j.elerap.2022.101131>
14. Lee S (2010) Using data envelopment analysis and decision trees for efficiency analysis and recommendation of B2C controls. *Decis Support Syst* 49(4):486–497. <https://doi.org/10.1016/j.dss.2010.06.002>
15. Li Y, Cao B, Xu L, Yin J, Deng S, Yin Y et al (2014) An efficient recommendation method for improving business process modeling. *IEEE Trans Indust Inform* 10(1):502–513. <https://doi.org/10.1109/TII.2013.2258677>
16. Reusens M, Lemahieu W, Baesens B, Sels L (2017) A note on explicit versus implicit information for job recommendation. *Decis Support Syst* 98:26–35. <https://doi.org/10.1016/j.dss.2017.04.002>
17. Khan MTR, Jembre YZ, Saad MM, Shah SHA, Kim D (n.d.) Pop-Vndn: Proactive on-path content prefetching in vehicular named data networks. Available at SSRN 4058929. 1–26. <https://doi.org/10.2139/ssrn.4058929>
18. Salehi M, Kamalabadi IN, Ghouschi MBG (2013) An effective recommendation framework for personal learning environments using a learner preference tree and a GA. *IEEE Trans Learn Technol* 6(4):350–363. <https://doi.org/10.1109/TLT.2013.28>
19. Tohidi N, Dadkhah C (2020) Improving the performance of video collaborative filtering recommender systems using optimization algorithm. *Int J Nonlin Anal Appl* 11(1):483–495. <https://doi.org/10.22075/ijnaa.2020.19127.2058>
20. Ricci F (2010) Mobile recommender systems. *Inform Technol Tour* 12(3):205–231. <https://doi.org/10.3727/109830511X12978702284390>
21. Tahmasebi F, Meghdadi M, Ahmadian S, Valiollahi K (2021) A hybrid recommendation system based on profile expansion technique to alleviate cold start problem. *Multimed Tools Appl* 80(2):2339–2354. <https://doi.org/10.1007/s11042-020-09768-8>
22. Massa P, Avesani P (2004) Trust-aware collaborative filtering for recommender systems. *OTM Confederated International Conferences "On the Move to Meaningful Internet Systems"*: Springer, p 492–508 https://doi.org/10.1007/978-3-540-30468-5_31
23. Adomavicius G, Tuzhilin A (2005) Toward the next generation of recommender systems: A survey of the state-of-the-art and possible extensions. *IEEE Trans Knowl Data Eng* 17(6):734–749. <https://doi.org/10.1109/TKDE.2005.99>

24. Lam SK, Riedl J (2004) Shilling recommender systems for fun and profit. Proceedings of the 13th international conference on World Wide Web. p 393–402. <https://doi.org/10.1145/988672.988726>
25. Caruccio L, Desiato D, Polese G (2018) Fake account identification in social networks. 2018 IEEE international conference on big data (big data): IEEE p. 5078–85. <https://doi.org/10.1109/BigData.2018.8622011>
26. Cerruto F, Cirillo S, Desiato D, Gambardella SM, Polese G (2022) Social network data analysis to highlight privacy threats in sharing data. *J Big Data* 9(1):19. <https://doi.org/10.1186/s40537-022-00566-7>
27. Cirillo S, Desiato D, Scalera M, Solimando G (2023) A visual privacy tool to help users in preserving social network data <https://doi.org/10.1007/s10462-020-09898-3>
28. Rezaimehr F, Dadkhah C (2021) A survey of attack detection approaches in collaborative filtering recommender systems. *Artif Intell Rev* 54(3):2011–2066. <https://doi.org/10.1007/s10462-020-09898-3>
29. Narayanan P, Vivekanandan K (2022) Hybrid CNN and RNN-based shilling attack framework in social recommender networks. *EAI Endorsed Transactions on Scalable. Inf Syst* 9(35):e6-e. <https://doi.org/10.4108/eai.2-11-2021.171754>
30. Ran X, Wang Y, Zhang LY, Ma J (2022) A differentially private matrix factorization based on vector perturbation for recommender system. *Neurocomputing*. 483:32–41. <https://doi.org/10.1016/j.neucom.2022.01.079>
31. Ovaisi Z, Heinecke S, Li J, Zhang Y, Zheleva E, Xiong C (n.d.) RGRecSys: A toolkit for robustness evaluation of recommender systems. In Proceedings of the Fifteenth ACM International Conference on Web Search and Data Mining2022. p 4. <https://doi.org/10.1145/3488560.3502192>
32. Ahmadian M, Ahmadi M, Ahmadian S (2022) A reliable deep representation learning to improve trust-aware recommendation systems. *Expert Syst Appl* 197:116697. <https://doi.org/10.1016/j.eswa.2022.116697>
33. Ahmadian S, Joorabloo N, Jalili M, Ren Y, Meghdadi M, Afsharchi M (2020) A social recommender system based on reliable implicit relationships. *Knowl-Based Syst* 192:105371. <https://doi.org/10.1016/j.knosys.2019.105371>
34. Wahab OA, Rjoub G, Bentahar J, Cohen R (2022) Federated against the cold: A trust-based federated learning approach to counter the cold start problem in recommendation systems. *Inf Sci* 601:189–206. <https://doi.org/10.1016/j.ins.2022.04.027>
35. Rezaimehr F, Moradi P, Ahmadian S, Qader NN, Jalili M (2018) TCARS: Time-and community-aware recommendation system. *Futur Gener Comput Syst* 78:419–429. <https://doi.org/10.1016/j.future.2017.04.003>
36. Koren Y, Rendle S, Bell R (2021) Advances in collaborative filtering. Recommender systems handbook, pp 91–142 https://doi.org/10.1007/978-1-0716-2197-4_3
37. Daneshmand SH, Javari A, Abtahi SE, Jalili M (2015) A time-aware recommender system based on dependency network of items. *Comput J* 58(9):1255–1266. <https://doi.org/10.1093/comjnl/bxu115>
38. Moradi P, Rezaimehr F, Ahmadian S, Jalili M (2016) A trust-aware recommender algorithm based on users overlapping community structure. 2016 sixteenth international conference on advances in ICT for emerging regions (ICTer): IEEE, p 162–7. <https://doi.org/10.1109/ICTER.2016.7829914>
39. Gasparetti F, Sansonetti G, Micarelli A (2021) Community detection in social recommender systems: a survey. *Appl Intell* 51(6):3975–3995. <https://doi.org/10.1007/s10489-020-01962-3>
40. Jiang L, Shi L, Liu L, Yao J, Ali ME. User interest community detection on social media using collaborative filtering. *Wirel Netw* 2022:1–7. <https://doi.org/10.1007/s11276-021-02826-5>
41. Al-Ghobari M, Muneer A, Fati SM (2021) Location-aware personalized traveler recommender system (lapta) using collaborative filtering KNN. *Comput, Mat Cont* 69(2):1553–1570. <https://doi.org/10.32604/cmc.2021.016348>
42. Kumar S, Kumar K (2018) LSR: Lexicon star rating system over cloud. 2018 4th International Conference on Recent Advances in Information Technology (RAIT): IEEE, p 1–6
43. Negi A, Kumar K, Chaudhari NS, Singh N, Chauhan P (2021) Predictive analytics for recognizing human activities using residual network and fine-tuning. *Big Data Analytics: 9th International Conference, BDA 2021, Virtual Event, December 15–18, 2021, Proceedings 9*: Springer, p 296–310. https://doi.org/10.1007/978-3-030-93620-4_21
44. Kumar K, Kurhekar M (2017) Sentimentalizer: Docker container utility over Cloud. 2017 ninth international conference on advances in pattern recognition (ICAPR): IEEE, p 1–6. <https://doi.org/10.1109/ICAPR.2017.8593104>
45. Negi A, Kumar K (2021) Classification and detection of citrus diseases using deep learning. *Data science and its applications*. Chapman and Hall/CRC, p 63–85

46. Negi A, Kumar K (2021) Face mask detection in real-time video stream using deep learning. *Computational intelligence and healthcare informatics*, pp 255–68 <https://doi.org/10.1002/9781119818717.ch14>
47. Sharma S, Kumar P, Kumar K (2017) LEXER: Lexicon based emotion analyzer. *International Conference on Pattern Recognition and Machine Intelligence*: Springer, p 373–9. https://doi.org/10.1007/978-3-319-69900-4_47
48. Sharma S, Kumar K, Singh N (2017) D-FES: Deep facial expression recognition system. 2017 conference on information and communication technology (CICT): IEEE, p 1–6. <https://doi.org/10.1109/INFOCOMTECH.2017.8340635>
49. Kumar K, Shrimankar DD (2017) F-DES: Fast and deep event summarization. *IEEE Trans Multimed* 20(2):323–334. <https://doi.org/10.1109/TMM.2017.2741423>
50. Vijayvergia A, Kumar K (2018) STAR: rating of reviews by exploiting variation in emotions using transfer learning framework. 2018 conference on information and communication technology (CICT): IEEE, p 1–6 <https://doi.org/10.1109/INFOCOMTECH.2018.8722356>
51. Kumar A, Purohit K, Kumar K (2021) Stock price prediction using recurrent neural network and long short-term memory. *Conference proceedings of ICDLAIR2019*: Springer, p 153–60. https://doi.org/10.1007/978-3-030-67187-7_17
52. Negi A, Kumar K, Chauhan P (2021) Deep neural network-based multi-class image classification for plant diseases. *Agricultural informatics: automation using the IoT and machine learning*, pp 117–29. <https://doi.org/10.1002/9781119769231.ch6>
53. Alok N, Krishan K, Chauhan P (2021) Deep learning-Based image classifier for malaria cell detection. *Machine learning for healthcare applications*, pp 187–97 <https://doi.org/10.1002/9781119792611.ch12>
54. Kumari S, Singh M, Kumar K (2021) Prediction of liver disease using grouping of machine learning classifiers. *Conference Proceedings of ICDLAIR2019*: Springer, p. 339–49 [10.1007/978-3-030-67187-7_35](https://doi.org/10.1007/978-3-030-67187-7_35)
55. Negi A, Chauhan P, Kumar K, Rajput R (2020) Face mask detection classifier and model pruning with keras-surgeon. 2020 5th IEEE international conference on recent advances and innovations in engineering (ICRAIE): IEEE, p 1–6 <https://doi.org/10.1109/ICRAIE51050.2020.9358337>
56. Kumar K, Shrimankar DD (2018) Deep event learning boost-up approach: Delta. *Multimed Tools Appl* 77:26635–26655. <https://doi.org/10.1007/s11042-018-5882-z>
57. Rezaimehr F, Dadkhah C (2021) Injection Shilling attack tool for recommender systems. 2021 26th International Computer Conference, Computer Society of Iran (CSICC): IEEE, p 1–4. <https://doi.org/10.1109/CSICC52343.2021.9420553>
58. Moradi P, Ahmadian S (2015) A reliability-based recommendation method to improve trust-aware recommender systems. *Expert Syst Appl* 42(21):7386–7398. <https://doi.org/10.1016/j.eswa.2015.05.027>
59. Feng H, Tian J, Wang HJ, Li M (2015) Personalized recommendations based on time-weighted overlapping community detection. *Inf Manag* 52(7):789–800. <https://doi.org/10.1016/j.im.2015.02.004>
60. Birtolo C, Ronca D (2013) Advances in clustering collaborative filtering by means of fuzzy C-means and trust. *Expert Syst Appl* 40(17):6997–7009. <https://doi.org/10.1016/j.eswa.2013.06.022>

Publisher's note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.

Springer Nature or its licensor (e.g. a society or other partner) holds exclusive rights to this article under a publishing agreement with the author(s) or other rightsholder(s); author self-archiving of the accepted manuscript version of this article is solely governed by the terms of such publishing agreement and applicable law.