A Trust-Based Collaborative Filtering Approach to Design Recommender Systems

Vineet K. Sejwal[1]
Department of Computer Science
Jamia Millia Islamia, New Delhi, India

Muhammad Abulaish[2], *SMIEEE*
Department of Computer Science
South Asian University, New Delhi, India

***Abstract*—Collaborative Filtering (CF) is one of the most frequently used recommendation techniques to design recommender systems that improve accuracy in terms of recommendation, coverage, and rating prediction. Although CF is a well-established and popular algorithm, it suffers with issues like black-box recommendation, data sparsity, cold-start, and limited content problems that hamper its performance. Moreover, CF is fragile and it is not suitable to find similar users. The existing literatures on CF show that integrating users' social information with a recommender system can handle the above-mentioned issues effectively. Recently, *trustworthiness* among users is considered as one such social information that has been successfully combined with CF to predict ratings of the unrated items. In this paper, we propose a trust-based recommender system, `TrustRER`, which integrates users' trusts into an existing user-based CF algorithm for rating prediction. It uses both ratings and textual information of the items to generate a trust network for users and derives the trust scores. For trust score, we have defined three novel trust statements based on user rating values, emotion values, and review helpfulness votes. To generate a trust network, we have used trust propagation metrics to compute trust scores between those users who are not directly connected. The proposed `TrustRER` is experimentally evaluated over three datasets related to movie, music, and hotel and restaurant domains, and it performs significantly better in comparison to nine standard baselines and one state-of-the-art recommendation method. `TrustRER` is also able to effectively deal with the *cold-start* problem because it improves the rating prediction accuracy for *cold-start* users in comparison to baselines and state-of-the-art method.**

***Keywords*—Recommender system; collaborative filtering; cold-start; trust; rating prediction**

I. Introduction

Personalized recommender systems recommend items based on the users' past experiences, previous ratings, and their preferences. Recommender systems handle *information overload* problem effectively where users find difficult to get right information at right time [1] [3]. Incorporating concepts like machine learning and information filtering with user profiling makes the recommender systems more effective towards product recommendation. Algorithms like content-based, collaborative filtering (CF), and hybrid methods are some well-known approaches for recommender systems [4]. Out of these CF is a frequently used algorithm to design recommender systems. It is very effective and simple to use in comparison to other approaches. CF can be categorized into memory-based (neighborhood) and model-based (latent factor) methods. In memory-based methods, similarities between users or items are used for rating prediction, whereas, model-based methods use machine learning approaches for creating rating prediction and recommendation models. In recent years, matrix factorization (MF) based CF models have gained huge popularity due to their scalability and high accuracy [14]. However, one of the major issues with CF is to find similar users or items that are used to compute user- or item-based similarity in the recommending process. Moreover, CF algorithm alone is not capable to handle issues like *cold-start* and *data sparsity*.

Recently, researchers have shown that integrating users' social information with CF can handle the *cold-start* and *data sparsity* issues effectively with improved recommendations. Social networks play an important role in filtering user information because most of the information is obtained and diffused by the users' acquaintances, such as friends and colleagues [15] [26]. The social network-based recommendation approaches assume the existence of social ties between users, where users in the network are directly or indirectly connected to each other. One such promising social information is *trust* because users like to accept different viewpoints of other users in a trust-based social network. Abbasi et al. [5] defined trust as follows: "trust between two entities refers a situation where the first entity (trustor) rely on the activities of another entity (trustee)". Mayer et al. [2] defined trust as "the willingness of a party to be vulnerable to the actions of another party based on the expectation that the other will perform a particular action important to the trustor, irrespective of the ability to monitor or control that other party".

Since the incorporation of trust information into the recommender systems helps to improve rating prediction accuracy and handles *cold-start* and *data sparsity* problems, in this paper, we propose a trust-based recommender system, `TrustRER`, which predicts the ratings of the unrated items for the users based on their trust scores. We believe that only similar ratings between a user-pair are not ideal and sufficient for the recommendation, instead, they should have similar preferences, tastes, and trustworthiness. The proposed `TrustRER` computes the trust score between users using different trust statements. Unlike other trust-based models, where only user ratings are considered as a trust statement, we introduced three novel trust statements based on ratings, emotions, and review helpfulness votes. The users' trust scores are incorporated with the user-based CF model to enhance its rating prediction and recommendation capability. We generate a *trust-based* network for the users using the trust statements. It might be possible that users in a trust network are not directly connected because they do not rate the same items. To handle this issue, we incorporate one-hop trust propagation for those users who are not directly connected.

For experimental evaluation of TrustRER, we have used publicly available datasets – Amazon dataset (music domain) and Yelp (hotels and restaurants domain), and a movie dataset containing users' reviews and movie-related information and used in one of our previous work [28]. TrustRER is compared with nine standard baselines and one state-of-the-art method using error-based and decision support-based metrics and outperforms these methods. TrustRER also improves rating prediction accuracy in comparison to comparative methods for *cold-start* users.

In short, the contributions of this work can be summarized as follows:

- Developing a recommendation method, TrustRER, combining trust and user-based collaborative filtering (UBCF) to predict the ratings of the unrated items.
- Defining three new trust statements to compute trust scores using both user ratings and reviews.
- Validating the proposed recommendation method using standard error-based and decision support-based metrics and comparing it with nine standard baselines and one state-of-the-art method over three datasets.
- Empirically evaluating the effectiveness of TrustRER to handle *cold-start* users in comparison to standard baselines and state-of-the-art methods.

The rest of the paper is organized as follows. Section II presents a brief discussion on the existing works to design recommender systems using trust statements and collaborative filtering. Section III presents a detailed description of the trust-related concepts, such as trust metrics, trust propagation, and trust aggregation. The work-flow of our proposed TrustRER method is discussed in Section IV. It also presents the computation of trust scores between users using three trust statements and trust models to predict ratings using the trust values and UBCF. Section V discusses the experimental and evaluation results. Finally, Section VI concludes the paper with future directions of research.

II. Literature Review

In this section, we present a brief review of the existing literatures on trust-based recommender systems. We also review the approaches that have utilized both memory-based and model-based CF in trust-based recommender systems.

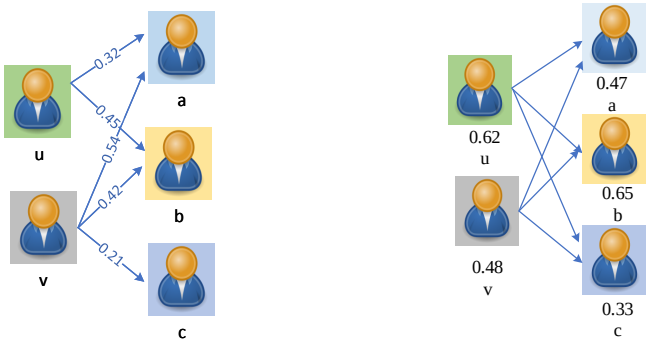
A. Memory-based Trust-Aware Recommender Systems

In online social networks, *trust* represents the social relationships between users. Recently, trust has been used in many applications, such as multi-agents recommender systems, semantic web, and cloud computing. Rahman and Hailes presented a trust contextualization model in [6] that computes an explicit trust between a user-pair using *local trust* metric. This work was further extended by Massa and Bhattacharjee in [8], which proposed a model to compute local and explicit trust between a user-pair using the path (distance) that connects the respective users. The authors believed that only ratings of users are not sufficient for a recommender system to predict items because many users rate a few items. They introduced the concept of *webs of trust* for users where trust propagation algorithms compute trust scores for indirectly connected users. The resultant trust value is binary. Further, Golbeck introduced a trust-based recommender system, *FilmTrust*, in [9] which recommends movies to users based on *Tidal* trust. *FilmTrust* computes an explicit *local trust* between users where trust values are gradual. To compute trust between directly and indirectly connected users, *FilmTrust* uses both trust propagation and trust aggregation techniques, respectively. In contrast to [9], O'Donovan and Smyth presented a *trust-based* recommendation technique [4] which computes an implicit local and global trust values. The trust value in the proposed work is gradual and uses trust metrics to compute the trust for user-pairs. The technique used in this work is based on the similarity between users. Hwang and Chen [27] incorporated both local and global trusts, respectively in a trust network that improves the rating prediction accuracy and coverage for recommendations. Massa and Avesani [10] introduced the *mole* trust algorithm, which computes the trust score for user-pairs using backward exploration. The path used for a user-pair in *mole* trust is based on the *maximum-depth*. Jamali and Ester [13] proposed *TrustWalker*, a coherent framework based on a random walk that combines both user-oriented approaches and user trust. The proposed model uses the ratings of the users that are directly/indirectly connected to the target user for the target item by performing a random walk. The methods discussed so far used rating-based trust statements for recommendations.

B. Model-based Trust-Aware Recommender Systems

To improve the efficacy of recommender systems, there are various approaches where both trust and matrix factorization (MF) are incorporated in existing 2-dimensional recommender systems. Guo et al. [12] proposed *TrustSVD*, a SVD++-based MF method incorporated with users trust for rating prediction and recommendation. The idea behind the design of the *TrustSVD* model was to consider both users' implicit and explicit ratings and trust for item recommendation. Yang et al. [17] proposed a variant of CF by incorporating users with social information and trust. The proposed model used MF technique, users sparse social trust network, and sparse rating data for rating prediction. Further, Jamali and Ester [7] proposed *SocialMF* model, a variant of MF which incorporates a trust propagation algorithm in an MF-based recommender system. The work proposed in [14] and [15] were based on co-factorization methods where the target user shares the same user and trust relation space. The co-factorization method factorizes the user-user trust relation matrix and user-item matrix using latent factors shared by the same user. Ma et al. [15] proposed *SoRec*, a social regularization model, which includes various social constraints of users. The proposed *SoRec* improves the rating prediction accuracy of recommender systems using social network information and also handles *cold-start* issue. Similarly, the works proposed in [16] and [19] were based on regularization methods which consider user preferences and assume that all user preferences should be available in their trust network. In another work, Jamali and Ester proposed a SocialMF model [16] that considered user preferences to generate a trust network.

Although, the approaches using either memory-based or model-based trust-aware recommendations have improved accuracy and coverage of the recommender systems, they



(a) Local trust (b) Global trust

Fig. 1. Trust Metrics for Directly Connected users in a Trust Network.

still have certain limitations. First, the datasets used in these methods are either synthetic datasets or generated using questionnaire-based methods, such as `Epinions` and `FilmTrust` in which trust scores between users are provided. Second, the trust score between users is computed only using their ratings. In this paper, we overcome these limitations and compute the trust score automatically from the underlying datasets.

III. Preliminaries

In this section, we present a brief description of some trust-related concepts that are used to design trust-based recommender systems. We mainly describe trust metrics, which includes local and global trusts, trust propagation, and trust aggregation in the following sub-sections.

A. Trust

In a social network, the computation of a trust value between a user-pair is generally difficult because it is very tough to interpret users' imagination, social behavior, and actual activities. To handle such issues, trust statements are defined and used to compute trust scores using users' attributes and behavior. As discussed in section I, trust between a user-pair *(u,v)* determines the faith that a user (say u) can keep on other user (say v) based on their action and activities. Further, trust keeps various properties, such as *generic*, *asymmetric*, *distributive*, and *transitive* which are used to generate a trust network between users [18]. The trust metrics like *local trust* and *global trust* along with trust propagation and aggregation are useful to construct trust networks and discussed in the following sub-sections.

1) Local Trust: Local trust is always defined between a user-pair, and it is computed independently from every other users present in the trust network. Mathematically, local trust between a user-pair can be defined as $LT : U \times U \rightarrow [0, 1]$, where U is the set of users and [0, 1] is the range of trust score values. The local trust is said to be "local" because it is defined between a user-pair irrespective of the other users in the trust network, and computed using the trust statements. This implies that in a trust network comprising of n users, each user has maximum $n - 1$ local trust scores. An example of local trust is given in figure 1a, where every connected user-pair shares a local trust value.

2) Global Trust: Global trust in a trust network represents a unique trust value for each user of the network. In general, global trust for a user is an aggregate single trust score computed using the trust statements. Mathematically, global trust can be defined as $GT : U \rightarrow [0, 1]$, where U is the set of users. This implies that in a trust network with n users there are n global trust scores, one for each user. An example of global trust is given in Fig. 1b, where every user has a global trust score value.

In comparison to *global trust*, *local trust* can handle various issues like *controversial topics* and *fake profiling* attack effectively. However, the time complexity of computing local trust is high in comparison to the global trust, because each user-pair is evaluated using the trust statements to compute the trust score values. There are various factors like topics, domains, and trust networks that can be taken into account while deciding to generate either local or global trust scores for a network.

B. Trust Propagation and Aggregation

In trust networks, there is a high chance that many users may not directly interact with other users. Therefore, trust propagation and aggregation algorithms are used for such users. A trust propagation algorithm computes both implicit and explicit trust values of the users in a trust network. Trust networks exhibit an atomic direct propagation relation which is also known as transitive relation. For example, if "Bob" trusts on "Alice", and "Alice" trusts on another user "Eve" in a trust network, then using atomic direct propagation property, trust score between "Bob" and "Eve" can be calculated. However, trust networks not always hold the transitive relation property, which proves the subjectivity of trust [11]. In large trust networks where users are not directly connected, there is a chance that more than one path is available to set up connectivity between them. In such situations, propagating trust scores are not capable to compute trust between a user-pair. To handle this issue effectively, the aggregation of trust propagation scores are helpful to calculate trust values. Therefore, the structure of a trust network determines the requirement of both trust propagation and aggregation, as shown in Fig. 2a and 2b. To calculate the trust between a user pair E and J in Fig. 2a, the trust aggregation algorithm is required, because there are multiple paths between users E and J. On the other hand, trust propagation is sufficient in Fig. 2b for user pair (A, C) to estimate the trust score. Popular algorithms that are generally applied in a trust network to compute trust propagation and aggregation are briefly described in the following sub-sections.

1) Mole Trust: In a trust network, *mole* trust between a user-pair can be computed using both local trust and trust propagation metrics [22]. The working of *mole* trust is based on backward exploration, where a walk between a user-pair is initiated using the trust edges, as given in equation (1). In this equation, $\mathcal{T}(v)$ represents trust value for user v for the previous walk, and $\mathcal{T}\_edge(v, u)$ represents trust edge score which connects the users v and u.

$$t_u = \frac{\sum_{v \in precursors}(\mathcal{T}(v) * \mathcal{T}\_edge(v, u))}{\sum_{v \in precursors}(\mathcal{T}(v))} \quad (1)$$

(a) Trust aggregation

(b) Trust propagation

Fig. 2. Trust Metrics for Indirectly Connected users in a Network.

2) Tidal Trust: Tidal trust was first proposed to recommend movies in a movie-based recommender system [20]. For a user-pair, *tidal* trust computes moderate trust in comparison to the *mole* trust which assigns a binary trust value. To compute *tidal* trust, first, all paths that connect a user-pair are identified, then the path having shortest distance is selected. Equation (2) presents a formal way to calculate *tidal* trust.

$$t_{m,n} = \frac{\sum_{p \in adjacent(m) | \mathcal{T}_{m,p} \geq maximum} \mathcal{T}_{m,p} \mathcal{T}_{p,n}}{\sum_{p \in adj(m) | \mathcal{T}_{m,p} \geq maximum} \mathcal{T}_{m,p}} \quad (2)$$

IV. Proposed Approach

In this section, we present a detailed description of the proposed `TrustRER` architecture, which is shown in Fig. 3. `TrustRER` predicts and recommends items to users using their trust values incorporated to the UBCF. First, a trust network is constructed using three novel trust statements based on users' ratings, emotions, and helpfulness votes. Thereafter, the trust score between users is incorporated into the UBCF model to find the top-k users. Finally, ratings are predicted using users' trust scores with other users. A brief detail of different functioning modules of our proposed `TrustRER` architecture is presented in the following sub-sections.

A. Rating-based Deviation

In rating-based deviation, trust for a user-pair (a, u) is computed using the rating values that represent their rating behavior. To calculate the trust score, we define a global trust function for both users a and u, as given in equations (3) and (4), where global trust shows the overall trust score for users a and u. In these equations, N represents the rating range for an item i, $\mathcal{R}$ represents the average rating value, and r_{ai} represents the rating value r given by user a on item i.

$$\mathcal{T}^{\delta}_{ai} = \begin{cases} \frac{-1}{N} |\mathcal{R} - r_{ai}| + 1, & if\ r_{ai} \neq 0 \\ 0, & otherwise \end{cases} \quad (3)$$

$$\mathcal{T}^{\delta}_{ui} = \begin{cases} \frac{-1}{N} |\mathcal{R} - r_{ui}| + 1, & if\ r_{ui} \neq 0 \\ 0, & otherwise \end{cases} \quad (4)$$

The deviation between the average rating $\mathcal{R}$ and the ratings provided by users a and u on item i as r_{ai} and r_{ui} show the trust values of user a and u, respectively. Equation (5) computes the trust score for a user-pair (a, u) using rating-based deviations. It can be observed from equation (5) that the trust value for a user pair (a, u) is maximum only when both users rate nearly similar on identical items.

$$\mathcal{T}^{\delta}_{a,u,n} = \frac{\sum_{j=1}^{n} (1 - |\mathcal{T}^{\delta}_{aj} - \mathcal{T}^{\delta}_{uj}|)}{n} \quad (5)$$

B. Emotions-based Information

Emotions like *sadness*, *happiness*, *joy*, and *anger* between a user-pair (a, u) can help to identify the trust/distrust relations between them. In [24] and [25], it has been proven by sociologists and psychologists that emotions are important parameters to compute trust/distrust values between users. Further, it has also been analyzed and observed that incorporation of emotions can reduce the data sparsity issue [23] [24]. Emotion values like *satisfaction*, *happiness*, and *joy* represent the positive emotions of users, whereas *sadness*, *anger*, and *fear* emotion values represent negative emotions [23]. The ratings and reviews provided by users on various e-commerce sites and platforms are highly correlated to their emotion values extracted from the reviews. The high ratings from users signify their positive emotions, whereas low ratings from users represent their negative emotions towards the consumption of the items. To calculate trust score for a user pair (a, u), we define an emotion vector, $\varepsilon = [\varepsilon_1, \varepsilon_2, \varepsilon_3, \varepsilon_4, \varepsilon_5]$ where, ε_1, ε_2, ε_3, ε_4, and ε_5 represent *disgust*, *fear*, *sadness*, *joy*, and *anger*, respectively. Formally, (6) is used to compute the trust score using emotion vectors, where $\mathcal{D}(a_j(\varepsilon), u_j(\varepsilon))$ is the Euclidean distance between the emotion vectors $a_j(\varepsilon)$ and $u_j(\varepsilon)$ for the user-pair (a, u), $\mathcal{T}_{a,u,n}$ is the aggregate trust score, and n represents the total number of similar items. The Euclidean

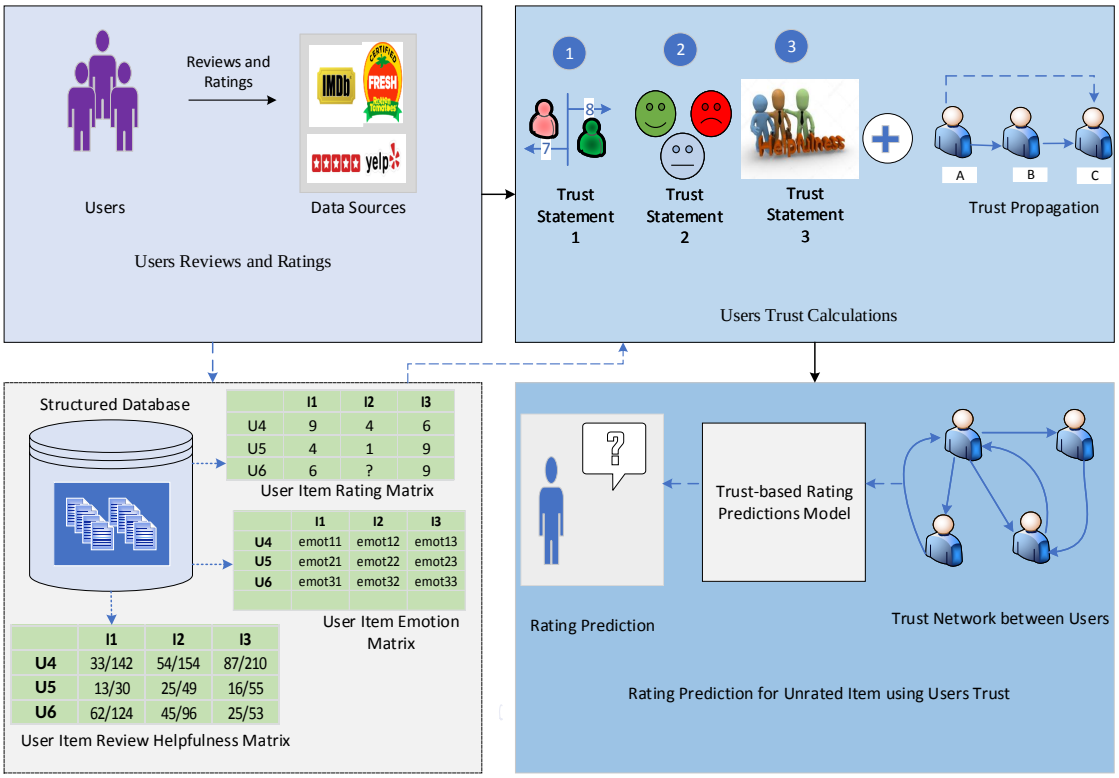


Fig. 3. Architecture of the Proposed `TrustRER` for Trust-based Recommendation.

distance between a user-pair (a, u) determines the correlation between them.

$$\mathcal{T}^{\varepsilon}_{a,u,n} = \frac{\sum_{j=1}^{n}(1 - \mathcal{D}(a_j(\varepsilon), u_j(\varepsilon))}{n} \tag{6}$$

C. Review Helpfulness

A user can rate and write reviews on various e-commerce sites and platforms. With ratings and reviews, users try to share their experiences and opinions on different aspects of the items. The e-commerce sites and platforms also provide voting facilities, where other users can cast votes to show whether they liked or disliked a particular review. The experiences and opinions are evaluated by users with a score of 0 or 1, in the form of *helpful* or *not helpful*. Review helpfulness can determine and infer trust score for users representing the reputation, reliability, and honesty score using their reviewed items. A number of websites, such as `Yelp`, `Flipkart`, and `Amazon` provide users' vote summary on reviews in the form, "17 out of 23 people find this review helpful". Ghose and Ipeirotis [29] described that the correlation between *review helpfulness* and reviewer reputation can be used to compute *trust*, *honesty*, and *reliability* values. In this study, we have used a variant of the trust statements given in [30] to compute trust score using review helpfulness. Equation (7) formally presents the way to compute trust score using review helpfulness votes. In equation (7), K shows the upper bound on trust value, and V is the *voting skewness*, which represents the number of positive and negative votes a reviewer has received on her reviews from different users. Voting skewness is formally defined in equation (8), where f_{ip} and f_{in} represent the frequency of positive and negative votes, respectively, and N is the total number of reviews. Finally, the trust score using the *review helpfulness* trust statement can be computed using equation (9), where $\mathcal{T}^{\mathcal{H}}_{a}$ and $\mathcal{T}^{\mathcal{H}}_{u}$ are the trust score for users a and u, respectively calculated using the *review helpfulness* votes, and n is the number of common-rated items.

$$\mathcal{T}^{\mathcal{H}}_{u} = \frac{K}{1 + e^{-KV_u}} \tag{7}$$

$$V_u = \frac{\sum_{i=1}^{m} f_{ip} - \sum_{i=1}^{m} f_{in}}{N} \tag{8}$$

$$\mathcal{T}^{\mathcal{H}}_{a,u,n} = \frac{1 - |\mathcal{T}^{\mathcal{H}}_{a} - \mathcal{T}^{\mathcal{H}}_{u}|}{n} \tag{9}$$

Once the individual trust scores based on different trust statements are calculated for a user-pair (a, u), the final trust score of (a, u) is calculated as their mean value, as given in equation (10).

$$\mathcal{T}_{a,u,n} = \frac{\mathcal{T}^{\delta}_{a,u,n} + \mathcal{T}^{\varepsilon}_{a,u,n} + \mathcal{T}^{\mathcal{H}}_{a,u,n}}{3} \tag{10}$$

D. Trust Model for Rating Prediction

In this section, we discuss the rating prediction of the unrated items jointly using UBCF and trust scores in line to the proposed approach in [37]. Equation (11) formally describes

TABLE I. STATISTICS OF THE DATASETS

Category	Movie	Yelp	Amazon Music
#Users	49080	45981	5542
#Items	1300	11537	3569
#Reviews	250882	229907	64719
Data Sparsity	99.60%	99.95%	99.67%
#Reviews per User	5.11	5.00	11.67
#Reviews per Item	192.98	19.92	18.13

the computation of rating prediction on item i by user u, where $sim\mathcal{T}(u,m)$ represents the trust score between users u and m, r_{um} represents the rating score for user m on overlapping items, and U_u^n represents the top-k users, similar to users u that have rated the same items.

$$\hat{r}_{ui} = \frac{\sum_{m \in U_u^n} sim\mathcal{T}(u,m) r_{um}}{\sum_{m \in U_u^n} |sim\mathcal{T}(u,m)|} \tag{11}$$

However, in a user-item interaction matrix, various users knowingly provide low or high ratings to certain items. They are bias users and their biasness or critical nature affects the rating prediction. In line to [37], we have used first-order approximation to handle users' biasness to improve rating predictions, as given in equation (12). In this equation, $b_{ui} = b_u + \mu + b_i$, where b_u and b_i are user and item deviations with respect to ratings, and μ is the mean rating of the items.

$$\hat{r}_{ui} = b_{ui} + \frac{\sum_{m \in U_u^n} sim\mathcal{T}(u,m)(r_{um} - b_{um})}{\sum_{m \in U_u^n} |sim\mathcal{T}(u,m)|} \tag{12}$$

Similarly, to avoid the overfitting issue, we have used a regularized model used in [31] [32], which is formally described in equation (13). In this equation, the first term represents the mean square error (MSE), which helps to learn the best fit rating values for the user (b_u) and item deviation (b_i). The second term in equation (13) includes the regularization terms to avoid overfitting by controlling the size of parameters, $\mathcal{K}$ is the set of ratings given by user u on item i, and $||.||$ denotes the Frobenius norm.

$$min \sum_{\mathcal{K}} (r_{ui} - \widehat{r}_{ui})^2 + \lambda(||b_u^2|| + ||b_i^2)|| \tag{13}$$

Korean in [31] and Hu et al. in [32] proposed an approach to compute the values of bias terms b_u and b_i using rating deviation. However, the major issue with their proposed solution is that they are not accurate methods to predict the ratings. In order to handle this issue and to compute regularizing parameters, *Alternating Least Squares* (ALS) or *Stochastic Gradient Descent* (SGD) [33] methods can be applied on equation (13). In this work, we have used ALS to compute the regularizing parameters because it performs better on sparse data, it is scalable over large datasets, and it can perform parallel execution in comparison to other approaches.

V. EXPERIMENT SETUP AND RESULTS

In this section, we present the experimental evaluation of TrutRER over three real-world datasets. It starts with a detailed description of the datasets used to perform experiments. Thereafter, it explains the evaluation metrics, baseline methods, and performance estimation results of TrustRER with baselines and state-of-the-art method. Finally, this section presents an empirical evaluation for *cold-start* users using TrustRER, baselines, and state-of-the-art method.

A. Dataset Description

To demonstrate the effectiveness of TrustRER, we performed experiments on two real-world datasets – Yelp and Amazon, and one Movie dataset. The first dataset is associated to Yelp, a crowd-sourced review forum, which allows users to write reviews on hotels and restaurants. It contains review information related to *restaurants*, *shopping*, *nightlife*, *automotive*, *home services*, *beauty and spa*, and *active life*. In this paper, we have used restaurant datasets for experiments and evaluations. The *restaurants* review dataset contain information related to *user check ins*, *business*, *user information*, *tip*, and *user reviews*. The second dataset, Amazon, is used in [21], and contains user and item meta-data on various items like books, automotive, digital music, movies, sports and outdoors, video games, etc. In this paper, we have used Amazon digital music dataset that includes users' ratings, reviews, and helpfulness votes on the reviews provided by different users on different items. The third dataset is based on the movie domain and used in one of our previous works [28]. It contains meta-data, ratings, and reviews information for the movies. The statistics of these datasets are presented in table I.

B. Evaluation Metrics

In this section, we briefly describe various metrics used to evaluate TrustRER and its comparison with the baselines and state-of-the-art method. We have used two types of standard evaluation metrics viz. error-based and decision support-based which are explained in the following paragraphs.

- *Error-based metrics*: In these metrics, the accuracy of filtering algorithms are evaluated by measuring the deviation between the real and estimated ratings. The absolute difference between the real and estimated ratings is used to compute the rating prediction accuracy. We have used two error-based metrics viz. MAE and RMSE for evaluations. MAE represents the average of absolute errors over a set of items, where absolute error is the deviation of the real and estimated ratings [34], as given in equation (14). RMSE computes the standard deviation of the predicted errors as given in equation (15). RMSE penalizes large errors because it gives a comparatively high weight to large errors. In equations (14) and (15), r_{ui} and $\hat{r}_{ui}$ represent the real and estimated ratings for user u and item i, and $\mathcal{T}$ is the test dataset.

$$MAE = \frac{\sum_{(ui) \in \mathcal{T}} |\hat{r}_{ui} - r_{ui}|}{|\mathcal{T}|} \tag{14}$$

$$RMSE = \sqrt{\frac{\sum_{(ui)\in\mathcal{T}}(\hat{r}_{ui} - r_{ui})^2}{|\mathcal{T}|}} \quad (15)$$

- *Decision support-based metrics*: In these metrics, accuracy of a recommender system is evaluated in terms of how well a recommender system facilitates its users to make good decisions. The term "good decisions" means recommending relevant items and filtering irrelevant items. In this work, we have used Precision, Recall, and F-score decision support-based metrics for evaluations. In recommender systems, relevant and recommended items are used to compute the values of Precision and Recall. Relevant items contain the real ratings on items provided by the users in past. On the other hand, recommended items contain the predicted ratings on the predicted items. Precision is defined as the fraction of the number of items that are both relevant and recommended to the number of items that are recommended, as given in equation (16). Recall is defined as the fraction of the number of items that are both relevant and recommended to the number of items that are relevant, as given in equation (17). In equations (16) and (17), $\mathcal{R}_{rel}$ is the relevant items and $\mathcal{R}_{rel,rec}$ is the relevant recommend items. The harmonic mean of Precision and Recall computes the F-score value.

$$Precision(P) = \frac{|\mathcal{R}_{rel,rec}|}{|\mathcal{R}_{rec}|} \quad (16)$$

$$Recall(R) = \frac{|\mathcal{R}_{rel,rec}|}{|\mathcal{R}_{rel}|} \quad (17)$$

$$F - score(F) = \frac{2 \times \frac{|\mathcal{R}_{rel,rec}|}{|\mathcal{R}_{rec}|} \times \frac{|\mathcal{R}_{rel,rec}|}{|\mathcal{R}_{rel}|}}{\frac{|\mathcal{R}_{rel,rec}|}{|\mathcal{R}_{rec}|} + \frac{|\mathcal{R}_{rel,rec}|}{|\mathcal{R}_{rel}|}} \quad (18)$$

C. Baseline Methods

In order to show the effectiveness of `TrustRER`, we compare it with 9 standard baseline methods viz. normal predictor, k-nearest neighbors (KNN), baselines, co-clustering, non-negative matrix factorization (NMF), Singular Value Decomposition (SVD), slope one, and SVD++. The baseline methods are briefly described in the following paragraphs.

- Normal Predictor method is based on the concept of maximum likelihood estimation. The formulation of this method requires the values of mean (μ) and variance (σ) to compute normal distribution as presented in equations (19) and (20), where R_{train} is the training dataset and r_{ab} shows user a rating on item b.

$$\mu = \frac{1}{|R_{train}|} \sum_{r_{ab}\in R_{train}} r_{ab} \quad (19)$$

$$\sigma = \sum_{r_{ab}\in R_{train}} \frac{(r_{ab} - \mu)}{|R_{train}|} \quad (20)$$

- K-nearest neighbors (KNN) is a machine learning approach that uses either users or items as nearest neighbors for rating prediction. It needs a user-item interaction matrix and a similarity method to formulate a user or an item neighbors. Equation (21) shows the rating estimation of unrated items using KNN, where k is top-k similar users and r_{vj} represents user v rating on item j.

$$\hat{r}_{vj} = \frac{\sum_{i\in N_v^k j} sim(j,i).r_{vi}}{\sum_{i\in N_v^k j} sim(j,i)} \quad (21)$$

- Co-clustering method uses the concept of pair-wise interactions of 2 types of concurrent entities. Equation (22) presents the rating estimation for co-clustering method, where $\overline{Clust_{vj}}$ is the mean rating of the co-cluster $\overline{Clust_{vj}}$, and $\overline{Clust_v}$ and $\overline{Clust_j}$ represent the mean ratings of users' and items' clusters [35].

$$\hat{r}_{vj} = \overline{Clust_{vj}} + (\mu_v - \overline{Clust_v}) + (\mu_v - \overline{Clust_j}) \quad (22)$$

- Baseline method uses the biasses of users and items to predict the ratings, as shown in equation (23), where μ represents the mean ratings of all items in the dataset and $bias_i$ and $bias_u$ are the item and user rating deviations.

$$\hat{r}_{ui} = \mu + bias_i + bias_u \quad (23)$$

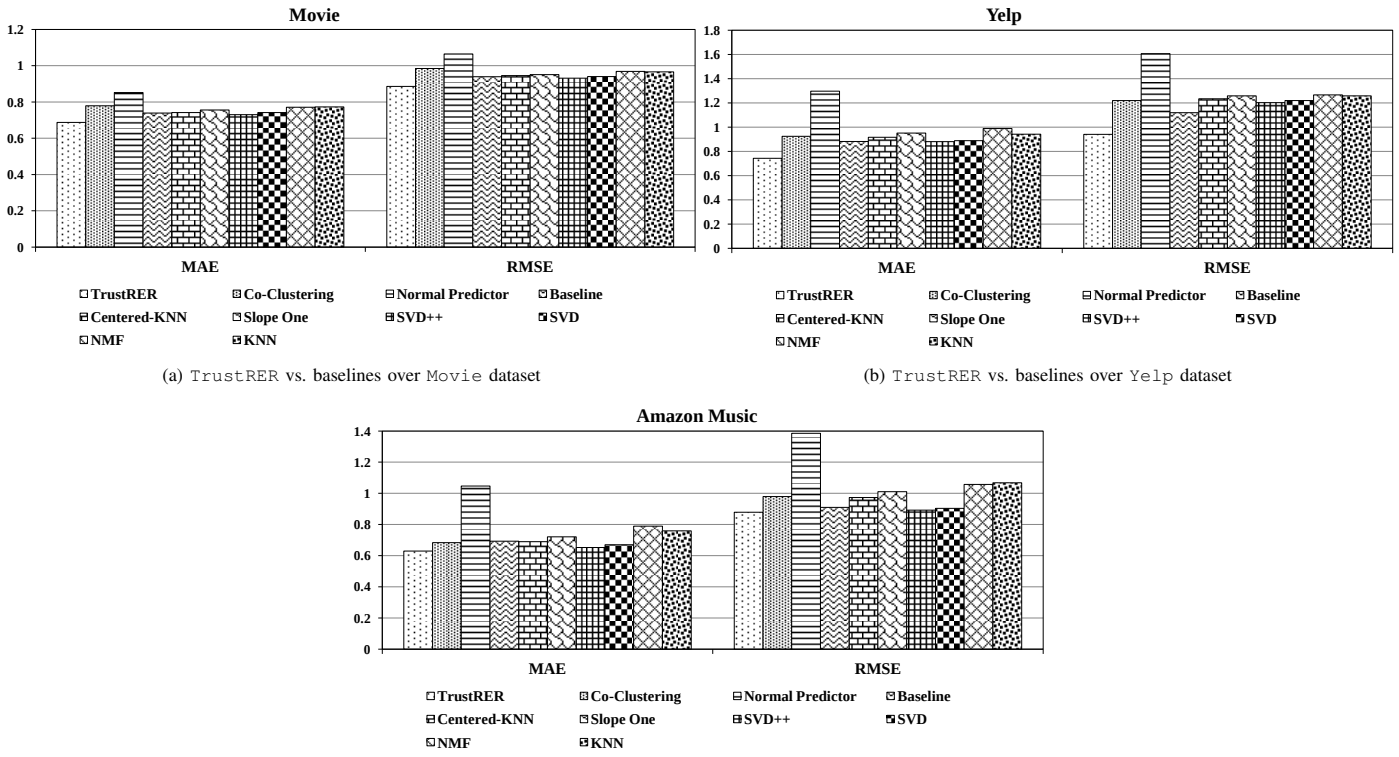
- SVD++, SVD, and NMF methods are used in collaborative filtering based recommendation techniques where a user-item rating matrix is decomposed into two low dimensional matrices viz. *user interest* and *item features*. The decomposed matrices help to compute the rating prediction for unrated items.

- Slope One method is designed for CF-based recommendation techniques, especially for item-based CF. It uses users' and items' mean ratings to predict the ratings of unrated items. Equation (24) presents the computation of slope one, where $|R_j|$ represents a set of pertinent items, $\overline{\mu_v}$ is mean rating of user v, and $dev_{j,i}$ represents the difference in the ratings of items j and i [36].

$$\hat{r}_{vj} = \overline{\mu} + \frac{1}{|(R_j)|} \sum_{i\in R_j} dev_{j,i} \quad (24)$$

D. Comparative Evaluation-1: `TrustRER` vs. Baseline Methods

This section presents a comparative analysis of `TrustRER` with nine baseline methods using error-based and decision support-based metrics. We have used SurPRISE (Simple Python Recommendation System Engine) [39] and `RecQ` Python libraries to implement the baseline methods. It can be observed from Fig. 4 that `Yelp` dataset has high error values in comparison to `Movie` and `Amazon Music` datasets because `Yelp` dataset contains minimum reviews for both users and items in comparison to the `Movie` and `Amazon` music datasets. This shows that `Yelp` dataset contains more *cold-start* users in comparison to the other two datasets. It can also be observed from Fig. 4 that SVD++ has minimum MAE and RMSE values in comparison to other baseline methods.



(a) TrustRER vs. baselines over Movie dataset

(b) TrustRER vs. baselines over Yelp dataset

(c) TrustRER vs. baselines over Amazon Music dataset

Fig. 4. Performance Comparison between TrustRER and Baselines in Terms of MAE and RMSE over different Datasets.

TABLE II. PERFORMANCE COMPARISON BETWEEN TRUSTRER AND SLOPE-ONE$_{Trust}$ IN TERMS OF MAE AND RMSE OVER DIFFERENT DATASETS

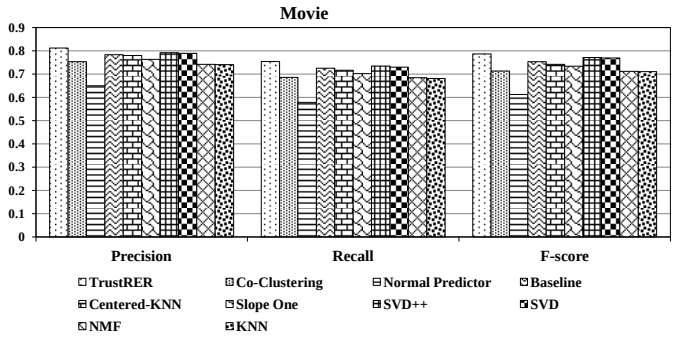
		k=20		k=40		k=60		k=80	
Dataset		MAE	RMSE	MAE	RMSE	MAE	RMSE	MAE	RMSE
Movie	TrustRER	0.7198	0.9215	0.6738	0.8591	0.6625	0.8421	0.6799	0.8608
	Slope-One$_{Trust}$ [38]	0.7616	0.9753	0.7329	0.9492	0.7017	0.8982	0.7293	0.9283
Yelp	CRecSys	0.7916	0.9811	0.7537	0.9602	0.7233	0.9209	0.7319	0.9322
	Slope-One$_{Trust}$ [38]	0.8428	1.0572	0.8028	1.0168	0.7759	0.9682	0.7822	0.9806
Amazon Music	CRecSys	0.6692	0.8624	0.6420	0.8382	0.6173	0.8082	0.6209	0.8120
	Slope-One$_{Trust}$ [38]	0.6972	0.9023	0.6749	0.8825	0.6527	0.8528	0.6673	0.8603

TABLE III. PERFORMANCE COMPARISON BETWEEN TRUSTRER AND SLOPE-ONE$_{Trust}$ FOR DIFFERENT VALUES OF k IN TERMS OF PRECISION (P), RECALL (R), AND F-SCORE (F) OVER DIFFERENT DATASETS

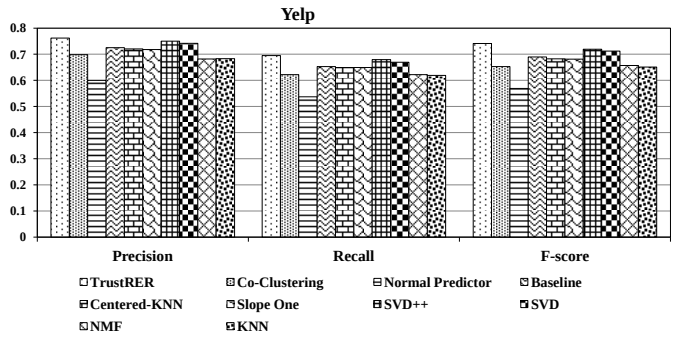
		k=20			k=40			k=60			k=80		
Dataset		P	R	F	P	R	F	P	R	F	P	R	F
Movie	TrustRER	0.7863	0.7171	0.7501	0.8001	0.7315	0.7642	0.8267	0.7527	0.7879	0.8191	0.7459	0.7807
	Slope-One$_{Trust}$ [38]	0.7336	0.6717	0.7012	0.7510	0.6805	0.7140	0.7718	0.6981	0.7331	0.7657	0.7015	0.7321
Yelp	TrustRER	0.7351	0.6524	0.6912	0.7410	0.6631	0.6998	0.7599	0.6994	0.7283	0.7514	0.6873	0.7179
	Slope-One$_{Trust}$ [38]	0.6715	0.6027	0.6353	0.6855	0.6135	0.6475	0.7019	0.6250	0.6612	0.6950	0.6212	0.6560
Amazon Music	TrustRER	0.8011	0.7063	0.7507	0.8457	0.7367	0.7874	0.8632	0.7691	0.8134	0.8492	0.7552	0.7994
	Slope-One$_{Trust}$ [38]	0.7764	0.6782	0.7239	0.8105	0.7019	0.7523	0.8327	0.7339	0.7801	0.8193	0.7250	0.7633

TABLE IV. PERFORMANCE COMPARISON BETWEEN TRUSTRER AND SLOPE-ONE$_{Trust}$ IN TERMS OF MAE AND RMSE FOR *cold-start* USERS OVER DIFFERENT DATASETS

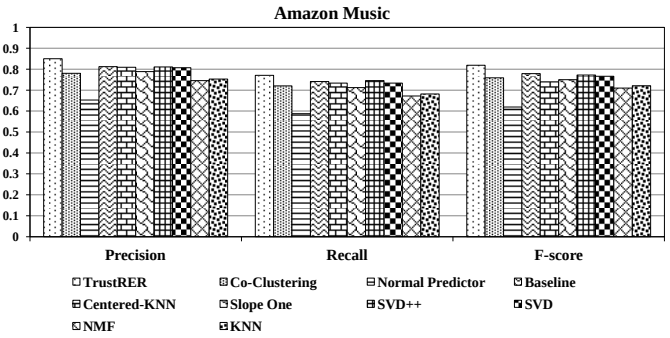
Dataset		MAE (k@80)	RMSE (k@80)
Movie	TrustRER	0.8012	1.014
	Slope-One$_{Trust}$ [38]	0.9182	1.1125
Yelp	TrustRER	0.9214	1.1253
	Slope-One$_{Trust}$ [38]	1.0421	1.2874
Amazon Music	TrustRER	0.7742	0.9861
	Slope-One$_{Trust}$ [38]	0.8281	1.0375



(a) TrustRER vs. baselines over Movie dataset

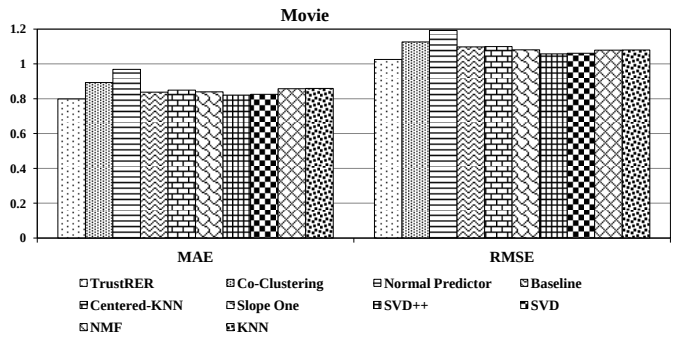


(b) TrustRER vs. baselines over Yelp dataset

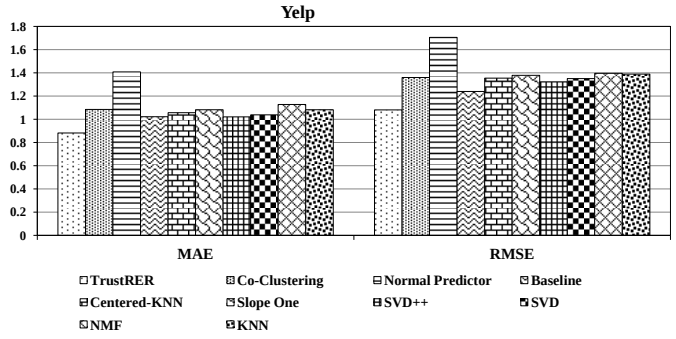


(c) TrustRER vs. baselines over Amazon Music dataset

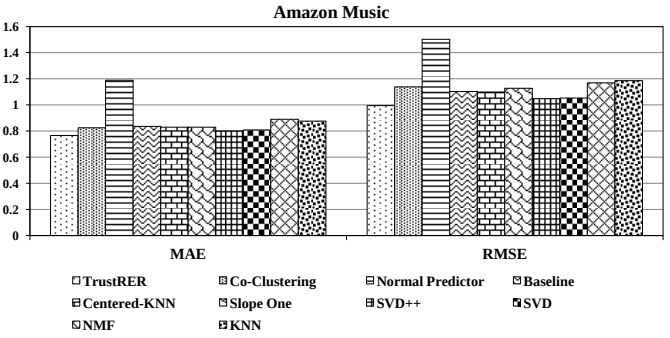
Fig. 5. Performance Comparison between TrustRER and Baselines in Terms of Precision, Recall and F-score over different Datasets.



(a) TrustRER vs. baselines over Movie dataset



(b) TrustRER vs. baselines over Yelp dataset



(c) TrustRER vs. baselines over Amazon Music dataset

Fig. 6. Performance Comparison between TrustRER and Baselines in Terms of MAE and RMSE for the *cold-start* users on different Datasets.

SVD++ is a variant of MF and uses model-based collaborative filtering for rating predictions. SVD++ includes implicit ratings (implicit feedback information) which help to improvise the predictions and recommendations. TrustRER outperforms SVD++ methods by 6.86% and 6.88% over Movie dataset, 15.70% and 21.68% over Yelp dataset, and 3.52% and 2.54% over Amazon music dataset for MAE and RMSE values. Similarly, Fig. 5 presents a comparative analysis of TrustRER with baseline methods using decision support metrics. It can be observed from Fig. 5 that TrustRER beats SVD++ with respect to Precision, Recall, and F-score by 5.07%, 6.44%, and 5.79% over Movie dataset, 7.27%, 9.76%, and 8.82% over Yelp dataset, and 5.55%, 4.76%, and 4.94% over Amazon Music dataset, respectively.

E. Comparative Evaluation-2: TrustRER vs. Slope-One$_{Trust}$ [38]

In this evaluation, TrustRER is compared with the state-of-the-art method, Slope-One$_{Trust}$ [38], which has used *slope one* algorithm to compute trust score between the users. Slope-One$_{Trust}$ approach proposed a *slope one* based trust algorithm that comprises the fusion of users' trust scores and their similarities for rating prediction and recommendation. The proposed approach first selects the trusted data of users and then computes similarities between them. The computed similarity scores are then added to the weight factor of the *slope one* algorithm to determine the ratings for the unrated items. Slope-One$_{Trust}$ method only uses rating deviations of users to generate the trust score, whereas TrustRER uses three novel trust statements, which seem very important to compute trust scores for the users.

Table II presents a comparative analysis of TrustRER and Slope-One$_{Trust}$ in terms of error-based metrics for k= 20, 40, 60, and 80. The different k values are the top-k similar users for a target user. Similarly, Table III presents a comparative analysis of TrustRER and Slope-One$_{Trust}$ in terms of decision support-based metrics for different k values. It can be observed from table II that both TrustRER and Slope-One$_{Trust}$ received minimum MAE and RMSE values at $k = 60$. On analysis, we found that TrustRER beats Slope-One$_{Trust}$ by 6.77% and 4.88% over Yelp dataset, 5.42% and 5.22% over Amazon Music dataset, and 5.58% and 6.24% over Movie dataset in terms of MAE and RMSE values, respectively. It can also be analyzed from table III that TrustRER beats Slope-One$_{Trust}$ and improved Precision, Recall, and F-score by 4.64%, 5.25%, and 4.95% over Movie dataset, 6.63%, 9.63%, and 8.21% over Yelp dataset, and 3.53%, 4.57%, and 4.09% over Amazon Music dataset.

F. Dealing with Cold-Start Users

In a user-item interaction matrix, some users provide very few ratings to items, and they are termed as cold-start users. With the availability of few ratings, it is very difficult to generate profiles for cold-start users and rating prediction for such users is a challenging task. Similarly, in the user-item interaction matrix, some items receive very few ratings from users, and such items are termed as *cold-start* items. However, in this work, we are predicting ratings for only *cold-start* users because we have used the UBCF model for rating prediction.

1) TrustRER vs. Baseline Methods for Cold-Start Users: To perform an empirical evaluation of TrustRER in comparison to the baseline methods for *cold-start* users, we conducted the same experiments as performed in Section V-D on Movie, Yelp, and Amazon Music datasets. In these experiments, the users who have estimated at most 5 items are considered as *cold-start* users, as used in [22]. The evaluated results are presented in Fig. 6a, 6b, and 6c. It can be observed from these figures that SVD++ outperformed all baseline methods on all datasets because of the availability of implicit feedback information. However, TrustRER beats the SVD++ method on all datasets because the user-based collaborative network in our approach considers both user ratings and reviews to generate the trust network and compute trust scores accordingly. Therefore, TrustRER computes trust for the users who consumed items but not directly interacted with other users using trust propagation metrics. In comparison to SVD++, TrustRER shows an improvement of MAE and RMSE values by 6.29% and 4.22% over Movie dataset, 15.02% and 19.9% over Yelp dataset, and 7.15% and 6.40% over Amazon Music dataset.

2) TrustRER vs. Slope-One$_{Trust}$ [38] for Cold-Start Users: TrustRER is compared with Slope-One$_{Trust}$ [38] for *cold-start* users on Movie, Yelp, and Amazon Music datasets. As discussed in Section V-F1, users who have estimated at most 5 items are considered as *cold-start* users. Table IV presents the comparative performance evaluation of TrustRER and Slope-One$_{Trust}$ with respect to MAE and RMSE values. It can be observed from Table IV that both TrustRER and Slope-One$_{Trust}$ have minimum MAE and RMSE values at $k = 80$. Similarly, it can also be observed from Table IV that TrustRER beats Slope-One$_{Trust}$ by 12.74% and 8.85% over Movie dataset, 11.58% and 12.59% over Yelp dataset, and 6.62% and 5.01% over Amazon Music dataset in terms of MAE and RMSE values. This is because TrustRER considers both users' ratings and reviews to define trust statements. As a result, the *cold-start* users who have provided either ratings or reviews are able to find similar users. On the other hand, Slope-One$_{Trust}$ only considers rating deviations as trust statement, and accordingly *cold-start* users who provide only reviews are not considered.

VI. Conclusion and Future Work

In this paper, we have proposed TrustRER, a trust-based recommender system, which uses trust statements to calculate trust scores between the users based on their ratings and reviews for rating prediction and recommendation. We have incorporated trust scores of the users into a UBCF model for rating prediction and recommendation. The novelty of the proposed method lies in generating a trust network for users using their trust scores that are computed using both ratings and reviews of the users. To this end, we have proposed three trust statements based on users' *ratings*, *emotions*, and *helpfulness votes*. TrustRER is compared with nine standard baselines and one state-of-the-art recommendation method, Slope-One$_{Trust}$ [38] using both error-based and decision support metrics over three different datasets. TrustRER is able to resolve the *cold-start* user problem and the comparative performance results show that TrustRER beats both baselines methods and Slope-One$_{Trust}$ and provides better ratings. In the future, trust statements can be used to generate *trust-based*

embeddings for the users in the form of graph embeddings to construct trust networks, and they can be incorporated into a model-based CF to improve rating prediction and recommendation.

REFERENCES

[1] J. S. Breese, D. Heckerman and C. Kadie, "Empirical Analysis of Predictive Algorithms for Collaborative Filtering" Gregory F. Cooper and Serafin Moral, editors, in *Proc. UAI*, San Francisco, USA, 1998, pp. 43–52.

[2] R. C. Mayer, J. H. Davis and F D. Schoorman, "An Integrative Model of Organizational Trust", in *Academy of Management Review*, vol. 20, no. 3, 1995, pp. 709–734.

[3] D. O Sullivan, D. C. Wilson and B. Smyth, "Improving case-based Recommendation: A Collaborative Filtering Approach" in *Proc. ECCBR*, Aberdeen, Scotland, 2002, pp. 278–291.

[4] J. O Donovan and J. Dunnion, "A Framework for Evaluation of Collaborative Recommendation Algorithms in An Adaptive Recommender System" in *Proc. CICLING*, Seoul, Korea, 2004, pp. 502–506.

[5] M. A. Abbasi, J. Tang and H. Liu, "Trust-aware recommender systems," in *Machine Learning Book on Computational Trust,* Chapman and Hall/CRC Press, Boca Raton, FL, USA, 2014.

[6] A. Abdul-Rahman and S. Hailes, "Supporting Trust in Virtual Communities", in *Proc. ICSS*, Maui, USA, 2000, pp. 1–9.

[7] M. Jamali and M. Ester, "A Matrix Factorization Technique with Trust Propagation for Recommendation in Social Networks", in *Proc. RecSys*, Barcelona, Spain, 2010, pp. 135–142.

[8] P. Massa and B. Bhattacharjee, "Using Trust in Recommender Systems: An Experimental Analysis", in *Proc. iTrust*, Oxford, United Kingdom, 2004, pp. 221–235.

[9] J. A. Golbeck, "Computing and Applying Trust in Web-based Social Networks", in *Ph.D. dissertation, Dept. of Computer Science, Univ. of Maryland*, Maryland, USA, 2005.

[10] P. Massa and P. Avesani, "Trust metrics on controversial users: Balancing between tyranny of the majority", in *IJSWIS*, vol. 3 no. 1, 2007, pp. 39–64.

[11] A. Jøsang, R. Hayward and S. Pope, "Trust Network Analysis with Subjective Logic", in *Proc. ASCS*, Hobart, Australia, 2006, pp. 85–94.

[12] G. Guo, J. Zhang and N. Yorke-Smith, "TrustSVD: Collaborative Filtering with Both the Explicit and Implicit Influence of User Trust and of Item Ratings", in *Proc. AAAI*, Austin Texas, USA, 2013, pp. 123–129.

[13] M. Jamali and M. Ester, "Trustwalker: A Random Walk Model for Combining Trust-based and Item-based Recommendation", in *Proc. KDD*, San Diego, USA, 2009, pp. 397–406.

[14] J. Tang, H. Gao, X. Hu and H. Liu, "Exploiting Homophily Effect for Trust Prediction", in *Proc. WSDM*, Rome, Italy, 2013, pages 53–62.

[15] H. Ma, H. Yang, M. R. Lyu and I. King, "Sorec: Social Recommendation using Probabilistic Matrix Factorization", in *Proc. CIKM*, Galway, Ireland, 2008, pages 931–940.

[16] M. Jamali and M. Ester, "A Matrix Factorization Technique with Trust Propagation for Recommendation in Social Networks", in *Proc. RecSys*, Barcelona, Spain, 2010, pp. 135–142.

[17] B. Yang, Y. Lei, J. Liu and W. Li, "Social Collaborative Filtering by Trust", in *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 39, no. 8, 2017, pp. 1633-1647.

[18] T. Bhuiyan, A. Josang and Y. Xu, "Managing Trust in online Social Networks", in *Handbook of Social Network Technologies and Applications*, 2010, pp. 471–496.

[19] H. Ma, D. Zhou, C. Liu, M. R Lyu and I. King, "Recommender Systems with Social Regularization", in *Proc. WSDM*, Hong Kong, China, 2011, pp. 287–296.

[20] J. Golbeck, "Generating Predictive Movie Recommendations from Trust in Social Networks", in *Proc. iTrust*, Pisa, Italy, 2006, pp. 93–104.

[21] J. Ni, J. Li and J. McAuley, "Justifying Recommendations Using Distantly-labeled Reviews and Fined-grained Aspects" in *Proc. EMNLP*, Hong Kong, China, 2019, pp. 188–197.

[22] P. Massa and P. Avesani, "Trust Metrics on Controversial Users: Balancing Between Tyranny of the Majority", in *IJSWIS*, vol. 3, no. 1, 2007, pp. 39–64.

[23] G. Beigi, J. Tang, S. Wang and H. Liu, "Exploiting Emotional Information for Trust/Distrust Prediction", in *Proc. SDM*, Miami, USA, 2016, pp. 81–89.

[24] G. R. Bewsell, "Distrust, Fear and Emotional Learning: An Online Auction Perspective", in *Journal of Theoretical and Applied Electronic Commerce Research*, vol. 7, no. 2, 2012, pp. 1–12.

[25] J. R. Dunn and M. E. Schweitzer, "Feeling and Believing: The Influence of Emotion on Trust", in *Journal of Personality and Social Psychology*, vol. 88, no. 1, 2005, pp. 736–748.

[26] C. Chen, J. Zeng, X. Zheng and D. Chen, "Recommender System based on Social Trust Relationships", in *Proc. ICEBE*, Coventry, UK, 2013, pp. 32–37.

[27] C. Hwang and Y. Chen, "Using Trust in Collaborative Filtering Recommendation", in *Proc. IEA/AIE*, Kyoto, Japan, 2007, pp. 1052–1060.

[28] V. K. Sejwal and M. Abulaish, "Context-Based Rating Prediction using Collaborative Filtering and Linked Open Data", in *Proc. WIMS*, Seoul, Korea, 2019, pp. 1–9.

[29] A. Ghose and P. G. Ipeirotis, "Estimating the Helpfulness and Economic Impact of Product Reviews: Mining Text and Reviewer Characteristics",in *IEEE Transactions on Knowledge and Data Engineering*, vol. 23, no. 10, 2011, pp. 1498–1512.

[30] G. Wang, S. Xie, B. Liu and S. Yu-Philip, "Review Graph Based Online Store Review Spammer Detection", in *Proc. ICDM*, Vancouver, Canada, 2011, pp. 1242–1247.

[31] Y. Koren, "Factor in the neighbors: Scalable and accurate collaborative filtering", in *ACM Transactions on Knowledge Discovery from Data*, vol. 4, no. 1, 2010, pp. 1–24.

[32] Y. Hu, Y. Koren and C. Volinsky, "Collaborative Filtering for Implicit Feedback Datasets", in *Proc. ICDM*, Pisa, Italy, 2008, pp. 263–272.

[33] Y. Zhou, D. Wilkinson, R. Schreiber and R. Pan, "Large-Scale Parallel Collaborative Filtering for the Netflix Prize", in *Proc. AAIM*, Shanghai, China, 2008, pp. 337–348.

[34] J. L. Herlocker, J. A. Konstan, L. G. Terveen and J. T. Riedl, "Evaluating Collaborative Filtering Recommender Systems", in *ACM Transactions on Information Systems (TOIS)*, vol. 22, no. 1, 2004, pp. 5–53.

[35] T. George and S. Merugu "A Scalable Collaborative Filtering Framework Based on Co-Clustering", in *Proc. ICDM*, Houston, USA, 2005, pp. 625–628.

[36] D. Lemire and A. Maclachlan "Slope One Predictors for Online Rating-Based Collaborative Filtering", in *Proc. SDM*, California, USA, 2005, pp. 1–5.

[37] B. J. Schafer, D. Frankowski, J. L. Herlocker, and S. Sen, "Collaborative Filtering Recommender Systems", in *The Adaptive Web*, vol. 4321, 2007, pp. 291–324.

[38] J. Liaoliang, C. Yuting, Y. Li, Li. Jing, Yan. Hongyang and W. Xiaoqin, "A Trust-based Collaborative Filtering Algorithm for E-commerce Recommendation System", in *Journal of Ambient Intelligence and Humanized Computing*, vol. 10, no. 8, 2019, pp. 3023–3034.

[39] N. Hug, "Surprise, a Python Library for Recommender Systems", 2017.