Expert Systems With Applications 97 (2018) 70–82

Contents lists available at ScienceDirect

Expert Systems With Applications

journal homepage: www.elsevier.com/locate/eswa

Sequencing of items in personalized recommendations using multiple recommendation techniques

Anand Shanker Tewari[a], Asim Gopal Barman[b,*]

[a] Department of Computer Science and Engineering, National Institute of Technology Patna, Patna 800005, India
[b] Department of Mechanical Engineering, National Institute of Technology Patna, Patna 800005, India

ARTICLE INFO

Article history:
Received 8 August 2017
Revised 20 October 2017
Accepted 7 December 2017
Available online 12 December 2017

Keywords:
Recommendation system
Item sequence in recommendation
Content based filtering
Collaborative filtering
Opinion mining
E-commerce.

ABSTRACT

Recommendation System (RS) is a piece of software that gives suggestions according to the interest of users in many domains like products in e-commerce, tours, hotels, entertainment etc. In any of the established e-commerce website hundreds of products are available under the same category. RS helps buyers to find the right product based on buyer's past buying pattern and item information. Currently many established approaches for item recommendations like content based filtering, collaborative filtering, matrix factorization, etc., exist. All these approaches create a big list of item recommendations for the target user. In general most users prefer to see only top-*n* recommendations, where the value of *n* is small and just ignores remaining recommendations. It means good RS must have high precision value for smaller values of *n* but at present almost all recommendation systems to the best of authors' knowledge are having high recall value and low precision value. It clearly means that top-*n* recommendations generated by these systems have very few items that may be liked by the target user. The proposed approach generates recommendations by combining features of content based filtering, collaborative filtering, matrix factorization and opinion mining. The proposed RS dynamically keeps track of user's inclination towards different types of items with respect to time. It analyzes user's opinions about products and finds the product popularity in the market by its own unique way. In the proposed approach, items are arranged in such a way that almost all preferred items by target user comes under top-*n* recommendations. The experimental results show that top-*n* recommendations generated by the proposed approach for smaller value of *n* have high precision value when compared with other traditional benchmark recommendation methods.

1. Introduction

Availability of Internet and other related technologies have given platform for the expansion of e-commerce websites. Every e-commerce website has its own Recommendation System (RS) that finds user's interest implicitly or explicitly and recommends items to users depending on their interest. RS helps e-commerce websites in retaining their existing customers and attracting new customers. Most extensively used recommendation approaches are content based filtering and collaborative filtering.

Content based filtering recommends items to users based on their past likings. It creates the profile of each user using attributes of items that user has rated high earlier and compare it with other non rated items in order to recommend new interesting items (Claypool et al., 1999). Widely known examples of such systems are InfoFinder and NewsWeeder (Krulwich & Burkey, 1996; Lang, 1995). Major drawbacks of content based filtering is that it does not provide quality assessment and serendipitous recommendations.

Collaborative Filtering (CF) addresses these shortcomings of content based filtering, it is based on human judgments (ratings). Collaborative filtering recommends items to the target user that are already items of interest for other like minded users. Sometimes CF systems are not able to make suitable recommendations to the target user because no ratings or only a small number of ratings are available for new user or new item, this situation is known as Cold Start (CS) problem in recommender system. CS problem occurs when new item or new user is introduced into the system (Balabanovic & Shoham, 1997; Claypool et al., 1999; Sarwar, Konstan, Borchers, Herlocker, & Riedl, 1998). CF system also suffers from the Gray Sheep (GS) problem. In this problem a group of users/ user will not receive correct recommendations because their views are different from any group of people. These users

* Corresponding author.
E-mail addresses: anand@nitp.ac.in (A.S. Tewari), asim@nitp.ac.in (A.G. Barman).

https://doi.org/10.1016/j.eswa.2017.12.019
0957-4174/

rarely get benefit from the collaborative filtering although the recommendation system has enough ratings to make recommendations (Balabanovic & Shoham, 1997; Claypool et al., 1999). Many researchers have tried to solve these problems by the use of hybrid approaches of content based filtering, collaborative filtering and matrix factorization techniques. The success of collaborative filtering relies on finding the similar users of the target users. Collaborative filtering performs well if there is sufficient data available in the rating matrix. In general any single user rates few items in comparison to large number of available items, this situation leads to the sparseness problem in the collaborative filtering. The insufficiency of data in rating matrix hinders CF process in generating good quality recommendations. Matrix factorization is another technique of recommendation that alleviates the sparse rating problem of the collaborative filtering.

Matrix factorization is another recommendation technique that finds latent features of both users and items inferred from their rating pattern (Koren, 2008). This technique is very similar to singular value decomposition method. It uses user-item rating matrix to find the latent features.

With the arrival of Web 2.0, almost all e-commerce websites are giving opportunity to users to write their own experiences about products in the form of reviews. Now users are reading reviews of other users before making any new purchase. All prominent websites are having hundreds of reviews about popular items, so it becomes very difficult for the user to read every review before making any new purchase. Opining mining is alleviating this pain of users by finding the semantic orientation of reviews i.e. positive or negative.

Most users prefer to see only top-n recommendations from e-commerce websites. Almost all buyers prefer smaller value of n. But most established recommendation techniques generate recommendations with high recall and low precision for smaller value of n. It means many recommended items that are of interest of the target user may come at the very end of the recommendation list. The proposed RS tries to keep user's preferred items at the beginning of the recommendation list, which results in high precision value for smaller values of n. The proposed approach creates the profile of each user dynamically using content based filtering. It uses collaborative filtering to generate item recommendations list. It uses collaborative filtering as a *classifier*. The proposed approach finds the popularity of all items in the market and converts reviews into ratings using opinion mining. It uses matrix factorization to predict good quality of approximate missing ratings and generates recommendations (Alami, Nfaoui, & Beqqali, 2015). The proposed recommendation technique also addresses the item- side cold start problem and preserves gray sheep users interests in a fairly decent manner.

1.1. Our contributions

The key contributions of this paper are as follows:

- Most users prefers to see only small size recommendation list with their items of interest in it. In this approach we have proposed a technique that increases the precision value of top-n recommendations, specially for small value of n. It means that the most of the items in small size list recommended by the proposed approach are liked by users.
- With the arrival of Web 2.0, millions of product reviews are available in e-commerce websites.To the best of authors knowledge in comparison to other approaches found in literature our approach uses these reviews to generate high precision top-n recommendations using opinion mining in co-operation with collaborative, content based filtering and matrix factorization technique.
- The proposed approach uses people's opinions to calculate popularity of different items available in e-commerce website and uses it to generate high precision top-n recommendations.
- Now-a- days almost all e-commerce websites are having data about user's ratings, reviews and descriptions of items that they are selling. Our approach uses only ratings, reviews and items description data to generate recommendations. This makes our approach suitable to almost all the e-commerce websites.
- The proposed approach is developed without using user's tags and social network related data in generating user's top-n recommendations because it is almost unavailable to most e-commerce websites.
- Proposed approach has a provision that changes the profile of each user dynamically with the changing interest of each user. Our approach does not use complex calculations and handles item side cold start and gray sheep problems also in fairly simple but robust manner.

The paper is further organized as follows: Section 2 discusses background knowledge and related work, Section 3 throws light on proposed approach, Section 4 explains practical implementation and evaluation. Finally Section 5 concludes the paper.

2. Background and related work

2.1. Content based filtering

When people read newspaper, instead of reading complete newspaper they read only certain pages or sections of the newspaper in which they are interested. It means even though their eyes are exposed to all the data available in the newspaper but their mind naturally does content based filtering and selects certain contents of the newspaper and starts reading it.

Earlier Selective Dissemination of Information (SDI), one of the information filtering technique was used to inform scientists about new articles published in their areas of interest. The scientist's profile was created by list of keywords that describe their interests (Houseman & Kaskela, 1970). Content Based Recommendation (CBR) system recommends items to the user based on the characteristics of items that are liked by the user in past (Adomavicius & Tuzhilin, 2005).

Item profiles are generally stored in the database table. The columns of table represent the attributes of items like size, cost etc. CBR system uses user profiles to store user's interests (Pazzani & Billsus, 2007). In the database, the presence or absence of any attribute of an item is represented by binary values i.e. true or false / 0 or 1 etc. So matrix of item-attribute rarely suffers from sparsity problem because item either posses that attribute or not. Thus computation of item–item similarity on the basis of content is relatively more accurate (Fan, Pan, & Jiang, 2014).The system calculates the user profile similarity with the item profile and recommends those items that best match with the user's taste and interest (Li, Cai, & Liao, 2012).

2.1.1. Vector space model

CBR system often uses Vector Space Model (VSM) to represent user's profile and item's profile in terms of keywords. In VSM every user and item can be represented in n-dimensional space where each dimension corresponds to attribute or keyword. This model is frequently used in fetching relevant documents from the corpus of documents.

Let $D = (doc_1, doc_2, \ldots, doc_N)$ is a corpus of documents and $T = (t_1, t_2 \ldots, t_n)$ be the set of words in the corpus. The relevance of any keyword in the document is calculated using weights (w_{ji}). Each document d_j is represented as vector in n-dimensional space as $dj = (w_{j1}, w_{j2}, \ldots, w_{jn})$. The most common weighting

scheme is Term Frequency - Inverse Document Frequency (TF-IDF) (Lops, De Gemmis, & Semeraro, 2011). It says that keywords that occur many times in one document (TF) but few times in the collection of remaining documents (IDF), are more relevant to the content of the document. The term frequency (TF_{ij}) of keyword t_i in document doc_j is defined in Eq. (1)

$$TF_{i,j} = \frac{f_i}{f_{total,j}} \tag{1}$$

where f_i is the frequency of keyword t_i in document doc_j and $f_{total,\ j}$ is the summation of all the term frequencies that are in document doc_j.

The Inverse Document Frequency (IDF) of keyword t_i is defined in Eq. (2), where N is the total number of documents in the collection and n_t is the count of documents in the collection that have term t_i at least once.

$$IDF_i = \log \frac{N}{n_t} \tag{2}$$

The TF-IDF weight w_{ji} of keyword t_i in document d_j is defined as shown in Eq. (3).

$$w_{ji} = TF_{i,j} \times IDF_i \tag{3}$$

Next the cosine based similarity between two vector documents doc_1 and doc_2 can be calculated using Eq. (4).

$$similarity(doc_1, doc_2) = cos(\vec{doc_1}, \vec{doc_2}) = \frac{\vec{doc_1} \cdot \vec{doc_2}}{\left|\vec{doc_1}\right| \times \left|\vec{doc_2}\right|} \tag{4}$$

2.2. Collaborative filtering

Collaborative Filtering is well established recommendation technique. It addresses the shortcomings of content based filtering by evaluating the quality of recommendations in the form of ratings. CF provides different type of recommendations to the user based on preferences of other like-minded people. CF was initially introduced by Goldberg et al., by creating a system for filtering email named as Tapestry (Goldberg, Nichols, Oki, & Terry, 1992). CF is also referred as social filtering because it recommends items to the target user based on the ratings of other neighboring users. Two users are said to be neighbors if they provide similar ratings to the commonly bought items (Shardanand & Maes, 1995), (Herlocker, Konstan, Borchers, & Riedl, 1999). User based collaborative filtering collects the ratings of the target user in a given domain and matches it with other users in the same domain to generate useful personalized recommendations for the target user (Resnick, Iacovou, Suchak, Bergstrom, & Riedl, 1994). Similarity between two users i.e. target user and neighboring user, can be calculated using Pearsons correlation formula as shown in Eq. (5).

$$Sim_{m,n} = \frac{\Sigma_{a \in I_{m,n}} \left(R_{m,a} - \bar{R_A}\right)\left(R_{n,a} - \bar{R_n}\right)}{\sqrt{\Sigma_{a \in I_{m,n}} \left(R_{m,a} - \bar{R_A}\right)^2 \Sigma_{a \in I_{m,n}} \left(R_{n,a} - \bar{R_n}\right)^2}} \tag{5}$$

where $R_{m,\ a}$ is the rating of the item a by user m, $\bar{R_m}$ is the average rating of user m, and $I_{m,\ n}$ is the common item set rated by both user m and user n.

$$P_{x,k} = A_x + \frac{\sum_{m=1}^{c} (R_{mk} - A_m) * Sim(x, m)}{\sum_{m=1}^{c} Sim(x, m)} \tag{6}$$

Then the rating P_{xk} of the target user x to the target item k is calculated using Eq. (6), where A_x the average rating of the target user x, R_{mk} is the rating of the neighboring user m to item k, A_m is the average rating of user m, $Sim(x,m)$ is the similarity between the target user x and the neighboring user m and c is the total number of the neighbors of the target user x.

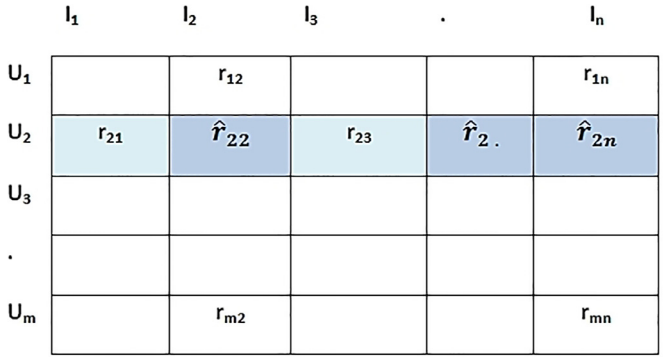

Fig. 1. Sparse Rating matrix (R).

2.3. Opinion mining

The Web 2.0 has given freedom to Internet users, to act as both producer and consumer of data. Now people are not only reading the information about products provided by the manufacturers but also writing their own experiences about various products, services and other things in the form of reviews. Review written by one customer helps another customer in deciding whether to buy or not to buy particular item. It also helps manufacturer to know positive and negative aspects of their product (Homoceanu, Loster, Lofi, & Balke, 2011). Almost every established e-commerce website has hundreds of reviews available for any popular product. So for a customer it becomes very difficult to read every review about the product before buying new item.

Opinion mining helps customers by analyzing each and every review and finding their polarity or semantic orientation i.e. positive, negative or neutral using various machine learning techniques (Pang, Lee, & Vaithyanathan, 2002; Salvetti, Lewis, & Reichenbach, 2004). Opinion mining extracts information from opinions or reviews of other people, about some particular item or topic. In general adjective words are used to describe nouns and pronouns. So for finding a review polarity adjective words are extracted from the review using natural language processing method. The polarity of these adjective words helps in finding the review polarity of the complete review (Hu & Liu, 2004). The semantic orientation of these adjective words can be found using SentiWordNet, which is derived from WordNet (Miller, Beckwith, Fellbaum, Gross, & Miller, 1990). SentiWordNet is a lexical resource that assigns three sentiment scores i.e. *positive, negative and objective* to each word (Esuli & Sebastiani, 2006; Hardeniya & Borikar, 2016).

2.4. Matrix factorization

Matrix factorization recommendation model has evolved after collaborative filtering. In reality most users have rated only few of items in comparison to complete list of items that are available on the website. It results in large sparse user-item ratings matrix. The sparseness problem leads to degradation in the quality of recommendations. Matrix factorization method addresses this issue in a fairly good manner. Matrix factorization is an example of latent factor model. The idea behind this model is to predict the missing entries based on some latent factors (Koren, Bell, & Volinsky, 2009). These models are highly accurate and scalable in many domains (Ilhami & Suharjito, 2014; Takacs, Pilaszy, Nemeth, & Tikk, 2008). Matrix factorization has its root in singular value decomposition (Ott, 2008).

Suppose in an e-commerce website there are m users and n items. This can be represented in matrix form as shown in Fig. 1. This matrix is called as Rating matrix (R) of size $m \times n$. An entry r_{mn} in rating matrix R represent rating of user m of item n. MF

finds latent features space K such that user-item interaction can be modeled as dot product in that space. Suppose U is a set of users and I is a set of items then rating matrix R can be approximated as product of two matrices P and Q, where P is $U \times K$ matrix and Q is $I \times K$ matrix, as shown in Eq. (7).

$$\hat{R} \approx P \times Q^T \tag{7}$$

where $\hat{R}$ is an approximated rating matrix. Any particular row of matrix P signifies how much that user possess those K latent factors. Similarly any particular column of matrix Q^T signifies how much that item possess those K latent factors. The approximated rating of user i for an item j i.e. $\hat{r}_{ij}$ can be calculated by dot product of vectors p_i and q_j^T as shown in Eq. (8).

$$\hat{r}_{ij} = p_i.q_j^T = \sum_{k=1}^{K} p_{ik}q_{kj} \tag{8}$$

Squared error (e_{ij}^2) between the approximated rating and the real rating (r_{ij}) can be calculated using Eq. (9).

$$e_{ij}^2 = (r_{ij} - \hat{r_{ij}})^2 = \left(r_{ij} - \sum_{k=1}^{K} p_{ik}q_{kj}\right)^2 \tag{9}$$

The error shown in Eq. (9) can be minimized iteratively using gradient descent numerical approximation method. This method finds the directions in which the values of p_{ik} and q_{kj} are modified to minimize the error by partially differentiating Eq. (9) with respect to p_{ik} and q_{kj}.

$$\frac{\partial e_{ij}^2}{\partial p_{ik}} = -2e_{ij}q_{kj} \tag{10}$$

$$\frac{\partial e_{ij}^2}{\partial q_{kj}} = -2e_{ij}p_{ik} \tag{11}$$

Once the gradient is obtained, the new additive updated values for both p_{ik} and q_{kj} are shown in Eqs. (12) and13.

$$p_{ik}^{'} = p_{ik} - \alpha\frac{\partial e_{ij}^2}{\partial p_{ik}} = p_{ik} + 2\alpha e_{ij}q_{kj} \tag{12}$$

$$q_{kj}^{'} = q_{kj} - \alpha\frac{\partial e_{ij}^2}{\partial q_{kj}} = q_{kj} + 2\alpha e_{ij}p_{ik} \tag{13}$$

where α is a learning rate, which finds the rate of approaching minimum. Usually its value is very small around 0.0002 (Ott, 2008). Using Eqs. 12 and13 iteratively, updates the values of p_{ik} and q_{jk} until the error converges to its minimum.

2.4.1. Regularization parameter

Overfitting is one of the frequent problems in prediction modeling. In this the prediction model learns and shows its peak performance on training data while the performance on real or testing data become worse. To avoid overfitting, regularization parameter with a factor β is added to the squared error as shown in Eq. (14).

$$e_{ij}^2 = (r_{ij} - \sum_{k=1}^{K} p_{ik}q_{kj})^2 + \frac{\beta}{2}\sum_{k=1}^{K}(\|P\|^2 + \|Q\|^2) \tag{14}$$

where $\|P\|$ and $\|Q\|$ are *Frobenius norms* of matrix P and Q, which are defined as follows:

$$\|P\| = \sqrt{\sum_{k=1}^{K} p_{ik}^2}$$

$$\|Q\| = \sqrt{\sum_{k=1}^{K} q_{kj}^2}$$

Regularization parameter controls the weights of user-feature and item-feature vectors. It scales down large values of one vector, so that they cannot dominate the possible smaller number values of the other vector. So that P and Q would give better approximation of R. Practically β is set to some values in the range of 0.015 (Ott, 2008). Again to minimize the extended error as shown in Eq. (14), once more the partial derivatives of extended error is computed with respect to p_{ik} and q_{kj} as follows:

$$\frac{\partial e_{ij}^2}{\partial p_{ik}} = 2(r_{ij} - \hat{r_{ij}})(-q_{kj}) + \beta p_{ik}$$

$$\frac{\partial e_{ij}^2}{\partial p_{ik}} = -2e_{ij}q_{kj} + \beta p_{ik} \tag{15}$$

$$\frac{\partial e_{ij}^2}{\partial q_{kj}} = 2(r_{ij} - \hat{r_{ij}})(-p_{ik}) + \beta q_{kj}$$

$$\frac{\partial e_{ij}^2}{\partial q_{kj}} = -2e_{ij}p_{ik} + \beta q_{kj} \tag{16}$$

The new update rules of p_{ik} and q_{kj} are shown in Eqs. (17) and(18).

$$p_{ik}^{'} = p_{ik} - \alpha\frac{\partial e_{ij}^2}{\partial p_{ik}} = p_{ik} + \alpha(2e_{ij}q_{kj} - \beta p_{ik}) \tag{17}$$

$$q_{kj}^{'} = q_{kj} - \alpha\frac{\partial e_{ij}^2}{\partial q_{kj}} = q_{kj} + \alpha(2e_{ij}p_{ik} - \beta q_{kj}) \tag{18}$$

2.5. Related work

Mostly recommendation systems are using hybrid approaches involving content based filtering, collaborative filtering and matrix factorization techniques to solve the shortcomings of collaborative filtering and generating top-n recommendations to the users.

2.5.1. Top-n recommendation systems-related approaches

Sarwar, Karypis, Konstan, and Riedl (2000) have presented seminal recommendation algorithm suited for e-commerce applications. They have used association rules and collaborative filtering to generate top-n recommendations. Their approach has considered only those association rules for generating recommendations that are formed using only target user's neighbors. In their approach all items purchased by every customer in past framed as one transaction for generating association rules. If there are no sufficient number of rules available to recommend items to the target user then remaining items are extracted by scanning transactional dataset of each neighbors of the target user. The frequency count of each product is performed in the transactional dataset of each neighbor of the target user and n most frequent items are appended to the target user's recommendation list. The main problem with this approach is that very few number association rules are generated from transactional dataset of target user's neighbors. Their approach does not judge the quality of items that are recommended to the target user based on association rules and cannot differentiate between target user's past and present interest. Their approach does not address cold start problem. The proposed approach presented in this paper uses ratings and opinions of other users to judge the quality of every recommendation. The proposed approach focuses on finding the user's present interest and uses

this information while choosing target user's neighbors. The proposed RS addresses the item based cold start problem using attribute information of items provided by the seller or e-commerce website.

The work of Ziegler, Lausen, and Schmidt-Thieme (2004) is similar to the current proposed work. They have used the hybrid approach involving content based filtering, collaborative filtering and item's taxonomic information to generate top-*n* recommendations. Ziegler et al. (2004) approach creates the profile of each user based on different taxonomic classes of products rated by them. They have found the neighbors of the target user based on content without using items' ratings. Their approach gives equal importance to user's past and current interests during user's profile construction phase. The approach presented in this paper is based on the assumption that user's interest is not static. It changes from time to time. Generally users prefer to get recommendations based on their present interests. The proposed approach creates the profile of each user using unique dynamic keyword vector which uses product's taxonomic information. User's dynamic keyword vector changes itself from time to time to store user's current interest. The proposed approach finds neighbors of the target user based on his/her present interest and works with both user's ratings and reviews to generate positive top-*n* recommendations.

The work of Singh and Mehrotra (2015) has realized that items in top-*n* recommendations have high chances to be converted into sales. Their proposed approach uses implicit behavior of user to generate recommendations. They have captured the preferences of users according to seasons i.e. winter, summer, spring and autumn. The final top-*n* recommendations to the target user are generated by using collaborative filtering and items in recommendation list are sorted based on target user's past preferences in different seasons. This approach does not give good recommendations for many domains like movies. Example: the user can like any type of movie in any season of the year. Contrary to this, the current proposed approach is designed in such a way that it can work in multi domain e-commerce websites irrespective of seasons.

Kwon (2008) has proposed a classical approach to improve top-*n* recommendations using rating variance of items. He has asserted that items with small rating variance must be included in top-*n* recommendation list. He has included all the items in the recommendation list of the target user, which have satisfied the minimum rating threshold criteria and have low rating variance. He has used the common rating threshold for all users. Opposite to Kwon (2008) technique, the proposed approach uses different ratings thresholds for different users based on individual user's rating pattern. In addition to this, our RS uses reviews and considers the popularity of different items among e-commerce website users to generate top-*n* recommendations.

With the acceptance of Web 2.0 people are writing their experiences about items, tourist places, services, etc., in the form of reviews. Reviews written by one customer helps another customer to gain insight about the product. Now-a-days almost all users are reading reviews about items before making purchase. Only few researches have incorporated the reviews i.e. opinion mining in the process of generating recommendations to the users.

The work of Li and Sun (2015) have tried to improve the performance of collaborative filtering using opinion mining. They have derived the consumer ratings of different items from items reviews. They have used the static procedure for computation of ratings from reviews. In their procedure, every positive review is equivalent to rating value 5 and every negative review is equivalent to rating value 4, in the rating scale of 1–5. These static review ratings are finally used to generate item recommendations. Their approach does not handle other issues like sparseness in the derived rating matrix and cold start problem. Our proposed approach converts reviews into its equivalent ratings counterpart dynamically. The ratings in our approach are not static, it depends on the degree of positivity or negativity of reviews. Review rating is calculated using SentiWordNet. In addition to this our proposed approach handles issues of rating sparseness using matrix factorization and cold start.

Li, Wang, and Yan (2015) have proposed a recommendation system based on opinion mining. In their proposal they have extracted features of the product from reviews in chinese language and calculated sentiment orientation of users about those features. The review score calculated by opinion mining reflects the quality of product. Their recommendation process is based on features of products the target user has mentioned earlier in their reviews. Their approach recommends other products that have similar features as target user has mentioned in his past reviews with good opinion based score. So by this way recommendation's quality completely depends on how meticulously user has written his past item reviews. Their approach does not work well for short reviews because short reviews generally have either very few features or no features.

Dong, O'mahony, Schaal, Mccarthy, and Smyth (2016) have suggested the approach that uses product features similarities and sentiments to recommend products that are similar to features and superior to sentiments of the queried product. In their approach they have extracted the product description from the reviews and calculated the popularity of individual feature of the product. They have recommended products to the user that have more similar features and better sentiment score than the queried product. The approaches of Li et al. (2015) and Dong et al. (2016) does not handle the issues related to sparseness and cold start. In comparison to the approach of Dong et al. (2016) our approach calculates complete product popularity in the e-commerce website rather than calculating popularity of the individual feature of the product.

A recent survey has been conducted at our institute among 300 people regarding people's preferences about the length of product reviews they prefer to write and number of recommendations they want to receive. It has been observed that 84% participants prefer to give very short reviews. Examples: *Its good, waste of time and money, Fabulous handset in this range, Good looking, etc*. Keeping this fact in mind our proposed approach is designed in such a way that it works perfectly well for all size reviews and is able to handle the problems like data sparseness, cold start and preferences for gray sheep users.

2.5.2. Cold start and gray sheep problems-related approaches

Many researchers tried to solve the major shortcomings of collaborative filtering like cold start and gray sheep users' problem using hybrid approaches involving content based filtering, collaborative filtering and matrix factorization techniques. *Fab* is among the earliest hybrid recommendation approach that solves these shortcomings of CF by combining it with the content based filtering to recommend web pages (Balabanovic & Shoham, 1997). GroupLens research approach has addressed the rating sparsity problem for Usenet news using semi-intelligent filtering agents called filterbots. These are rating robots that evaluates every new article and provide ratings for those articles (Sarwar et al., 1998). Claypool et al. (1999) has also developed the RS for online newspaper their approach generates the recommendations for users using content based and collaborative filtering separately without any interdependency to address the problem of gray sheep and cold start. Ahn (2008) has given the solution of new user cold start problem by using new similarity measure PIP (Proximity, Impact, and Popularity) between the two users. However this system does not work for pure cold start users. Zhang, Tang, Zhang, and Xue (2014) have addressed the cold start problem using the demographic information of both users and items in combination with matrix factorization technique . Lika, Kolomvatsos, and Had-

Table 1
User-Keywords table.

User_Id	Keyword	Count	Timestamp

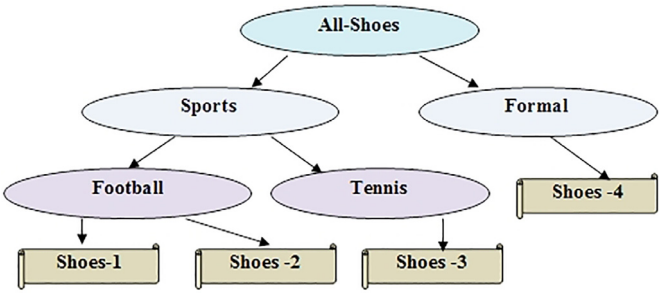

Fig. 2. Hierarchical arrangement of items (shoes) in e-commerce website.

jiefthymiades (2014) have solved the new user cold start problem in very feasible manner, they have used customers' demographic information like age, occupation, gender, etc., to solve it. The recent interesting approach of Wei, He, Chen, Zhou, and Tanga (2017) to address CS problem is similar to Lika et al. (2014) technique but in place of user's demographic information they have used item information along with the matrix factorization technique. To find gray sheep users, Ghazanfar and Prugel-Bennett (2011) have used K-Means++ clustering algorithm, once GS users are detected they used clustering based collaborative filtering to generate recommendations for those users.

3. Proposed approach

The proposed approach generates recommendations in the sequence that best suits the target user's taste using its seven specialized building blocks. These blocks are Dynamic Content Builder, User Similarity Finder, Item Similarity Finder, Collaborative Classifier, Dual purpose Opinion Miner, Matrix Factorizer and Collective Recommender block. The role of each block in the course of final recommendation generations is described as follows:

3.1. Dynamic content builder (DCB)

The proposed approach uses its own DCB block, which creates the profile of every user using content based filtering in a different manner. It has been observed that user's interest is not static. It is dynamic in nature and varies from time to time. For example: At certain stage of life user may be interested in cartoon networks, afterward his interest may be switched towards action movies, later on romantic movies and so on.

The proposed DCB block creates the profile of each user taking into consideration the changing moods of users. It stores user's browsing history and shopping pattern dynamically to find user's present and past areas of interests. DCB constructs user's profiles in the form of Dynamic Keywords Vectors (DKV) and stores it in User-Keywords (U-K) table as shown in Table 1.

Almost all e-commerce websites store items in the hierarchical order as shown in Fig. 2., where root shows collection of all items and leafs are individual items in the website (Shambour, Hourani, & Fraihat, 2016).

Whenever any user buys or browses any item, the DCB block finds the branch of that item to which it is directly connected inside item-hierarchy in e-commerce website and stores all the keywords that are in branch inside user's DKV, excluding the root and leaf nodes in the hierarchy. The DKVs of all the users are stored in the U-K table. The DCB block also creates *item description vector* for each item, which contains all the detail information about the product, provided by the seller.

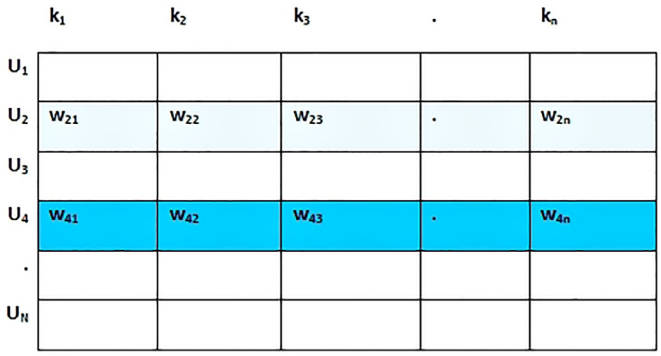

Fig. 3. User-Keywords weighting matrix.

Example: Consider an e-commerce website of shoes, the snapshot of hierarchical arrangement of items is shown in Fig. 2. In Fig. 2 item description vector of *Shoes-1* contains keywords like sports and football. Whenever any user buys *Shoes-3*, the DCB block stores the keywords i.e. *Tennis* and *Sports* in the DKV of that user inside U-K table and sets the *count* value as 1 and *timestamp* as current date for these keywords in U-K table. For any user when the same keywords come again in the DKV of that user then *count* value is incremented by 1 and *timestamp* field of these keywords in the U-K table is updated by current date. A DKV of any user can have maximum of n keywords in it.

When the DKV of any user already have n keywords in it and some more new keyword comes for insertion in the DKV then keyword having oldest *timestamp* value in the DKV of the user is deleted and new keyword is inserted with current date. This way proposed approach finds user's present interest and also retain his old likings. The pseudo code for storing keywords in user's DKV is shown in Algorithm 1.

3.2. User similarity finder (USF)

Once the DCB block creates the profile of every user, it can be arranged in the vector form as shown in Fig. 3, where rows are DKVs of the users and columns are keywords. Now USF block calculates the weights (w_{ji}) of every keyword in user's DKV using Eq. (3). Afterwards USF finds the similarity between any two users U_1 and U_2 using Eq. (4), this equation can be rewritten as shown in Eq. (19), where U_1 and U_2 are user vectors.

$$similarity\,(\vec{U_1},\vec{U_2}) = cos(\vec{U_1},\vec{U_2}) = \frac{\vec{U_1}\cdot\vec{U_2}}{\left|\vec{U_1}\right|\times\left|\vec{U_2}\right|} \tag{19}$$

3.3. Item similarity finder (ISF)

This block helps in addressing item side cold start problem. It finds the similarity of each item with every other items by using *item description vectors*.

The ISF finds the similarity ($ItemSim_{i_1,i_2}$) between two items I_1 and I_2 using Eq. (4). This equation can be rewritten as shown in Eq. (20), where I_1 and I_2 are *item description vectors* created by the DCB block.

$$ItemSim_{i_1,i_2} = cos(\vec{I_1},\vec{I_2}) = \frac{\vec{I_1}\cdot\vec{I_2}}{\left|\vec{I_1}\right|\times\left|\vec{I_2}\right|} \tag{20}$$

3.4. Collaborative classifier (CC)

This block generates recommendations for users using collaborative filtering. It works like a classifier. It selects k most nearest

Algorithm 1: Pseudo code for storing keywords in DKV of user.

```
Search_Path (Item_Name/ _Category/ _SubCategory)
{/* This function will find all the intermediatenodes in between root node and the argument of function. */ }
Count(Item_Name/ _Category/ _SubCategory)
{/* This function will count the number ofintermediate nodes in between the rootnode and the argument of function. */ }

StoreKeywordsInDKV (User_Id, Item_Name / _Category / _SubCategory, _Date)
{ IntermediateNodes = Search_Path(Item_Name/ _Category/ _SubCategory)
if Intermediate nodes found in Search_Path already exist in the DKV of user (User_Id) then
    Update Timestamp field of the U-K Table with the current date.
    Update  Count field in the U-K Table.
    SET Count = Count + 1;
else if Number of keywords in DKV of user >= n then
    /* Remove older keywords from the U-KTable based on the Timestamp field of theU-K Table */
    Number of keywords removed from the U-K Table = COUNT (Item_Name/ _Category/ _SubCategory)
    SET Keyword = IntermediateNodes;
    SET Timestamp = Date;
    Count = 1;
else
    /* Insert intermediate nodes along with theDate of Purchase in the U-K Table. */
    SET Keyword = IntermediateNodes;
    SET Timestamp = Date;
    Count = 1;
}
```

Table 2
Opinion rating table.

User_Id	Item_Id	Review_Sentiment_Score	Rating
5	1	0.26	4
5	3	0.40	5
5	7	−0.25	2
5	4	−0.75	1
5	5	−0.43	1

neighbors of the target user using USF block. This block does not generate ratings of items that may be liked by the target user, instead it classify items that may be *liked* or *disliked* by the target user (Basu, Hirsh, & Cohen, 1998). It does so using following steps:

- It partitions the item ratings of the target user into quartiles.
- Afterwards it performs the collaborative filtering and predicts the ratings of items that may be suggested to the target user using Eq. (6).
- Subsequently CC block creates the list of all items with label *like*, whose predicted ratings are greater than or equal to the target user's third quartile ratings and remaining suggested items are considered as *dislike* by the target user.

The output of the CC block is list of items that may be liked by the target user.

3.5. Dual purpose opinion miner (DOM)

This block works on items' reviews. It realizes two purposes:

- DOM calculates the *Sentiment Score* of each review of every item. Afterward it uses this *sentiment score* in the conversion of every review into its rating counterpart and stores it in *Opinion Rating* table shown in Table 2.
- Second, it calculates the popularity of different items in the market by calculating *Item Weight* of each item.

To fulfill first objective DOM collects all negation words like not, never, no etc, in the *Negation Bag*. Afterwards it uses SentiWordNet to find the sentiment score of each adjective word in review, which helps in finding the overall Review Sentiment Score (RSS) (Khan, Baharudin, & Khan, 2011). It also finds the polarity of each review. The combination of Review Sentiment Score and polarity helps in finding the equivalent rating of the review. The pseudo code for calculating sentiment score and polarity of each review shown in Algorithm 2. DOM gives higher ratings to every positive review and lower ratings to every negative review in the given rating scale (Li & Sun, 2015). If the overall polarity of review is positive then its equivalent rating is calculated using Eq. (21), where $R_{\max}$ is maximum rating in the given rating scale and $RSS_{positive\_\max}$ is highest RSS of any positive review.

$$Rating = \left\lceil R_{max} \times \frac{RSS}{RSS_{positive\_max}} \right\rceil \tag{21}$$

The rating equivalent of review with negative polarity is calculated using Eq. (22), where $RSS_{negative\_\min}$ is lowest RSS of any negative review and γ is a variable known as *adjustment parameter*, which prevents the zero valued ratings.

$$Rating = \left\lceil \frac{R_{max}}{2} \right\rceil - \left\lceil \frac{R_{max}}{2} \times \frac{RSS}{RSS_{negative\_min} + \gamma} \right\rceil \tag{22}$$

Example: Consider a case in which item ratings are given in the rating scale of 1–5. Table 2 shows a small instance of *Opinion Rating* table. It shows the RSS of different items calculated from the reviews given by User 5 to different items like 1, 3, 7 etc.

In the given data instance, the value of rating for item 1 is calculated by putting the values in Eq. (21). The value of $R_{\max}$ is 5, RSS is 0.26 and $RSS_{poisitve\_\max}$ is 0.40. Similarly the rating for item 5 is calculated using Eq. (22). In given data instance the value of RSS for *item 5* is −0.43, value of $RSS_{negative\_\min}$ is −0.75 and the value of *adjustment parameter* γ is taken as 0.20.

DOM's second objective is to calculate the *Item Weight (IW)*. To realize this objective DOM uses ratings derived from reviews as shown in Table 2. For every item it calculates the Average Review Rating (ARR) and the number of users who have given their review about that item. The weight of an item is calculated using Eq. (23), where $R_{\max}$ is the highest rating in the given rating scale, $Count_i$ is number of users who have given their reviews about that item

Algorithm 2: Pseudo code for calculating review sentiment score.

```
Procedure Review_Sentiment_Score ()
foreach review_sentence senten do
    review_sentiment_score = 0;polarity = 0;
    foreach adjective_word aw in review_sentence senten do
        review_sentiment_score + = ScoreCalculator(aw; senten);
        /* Positive =1 , Negative = −1*/
    if review_sentiment_score > 0 then
        SET senten polarity = Positive ;
    else if review_sentiment_score < 0 then
        SET senten polarity = Negative ;

Procedure ScoreCalculator (adjective_word, review_sentence)
review_sentiment_score= Sentiment Score of adjective_word from SentiWordNet
if there is any NEGATIVE WORD from Negation_Bag found near adjective_word in review_sentence then
    review_sentiment_score = reverse polarity of (review_sentiment_score)
```

Table 3
Item weight table.

Item_Id	ARR	Review count
3	4	21
5	2	10
6	3	18
9	5	16

and $Count_{\max}$ is highest value of count of any item.

$$IW_i = \frac{ARR}{R_{max}} \times \frac{Count_i}{Count_{max}} \tag{23}$$

Example: Consider data instance shown in Table 3, where *Review Count* shows number of users who have given their reviews. The *IW* of an item_id 5 is calculated by putting the values in Eq. (23). For item_id 5 the value of ARR is 2, $R_{\max}$ is 5, $Count_i$ is 10 and $Count_{\max}$ is 21.

3.6. Matrix factorizer (MF)

Matrix Factorizer block takes input from DOM and uses opinion rating table to create rating matrix as shown in Fig. 1. It predicts approximate ratings of unknown items using Eq. (8). Then MF minimizes the squared error between the approximate ratings and opinion ratings using Eqs. (12) and 13.

To avoid over-training MF block adds regularization parameter using Eqs. 17 and 18. Finally MF block predicts all missing review ratings of all items for the target user.

3.7. Collective recommender (CR)

CR block finally generates recommendations for the target user in proper sequence. Here sequencing means items in recommendation list are arranged in order that suits target user's interest (Agarwal, Chakraborty, & Chowdary, 2017). CR block takes input from the CC, DOM and MF blocks and generates the final recommendations to the target user in proper sequence using Eq. (24), where w_1, w_2 and w_3 are website dependent *weights*, their value ranges in between 0 and 1 i.e. $0 < w_{1,2,3} \leq 1$. This block calculates the Final Recommendation Value (FRV) of each item (i) for every user (u), which helps in arranging items in proper sequence. Items are recommended to the target user in decreasing order of their FRV's values. Block diagram of the proposed approach is shown in Fig. 4.

$$FRV_{ui} = w_1 CC_i + w_2 IW_i + w_3 MF_i \tag{24}$$

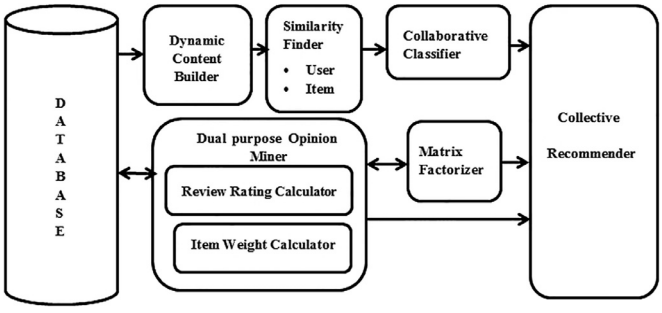

Fig. 4. Block diagram of the proposed recommendation system.

In general for any e-commerce websites it is very difficult to get demographic information from users. Most of the times users hesitate to share their personal information to e-commerce website like age, occupation, gender, etc. But contrary to this sellers are providing complete details of their products to e-commerce website to increase their sale. Every time when sellers are adding new products in the website they are also including details about products in the proper hierarchical order. The proposed approach has used this new product information to addresses item side cold start problem with the help of DCB and ISF blocks in the following manner:

- Whenever any new item is added to the website, the ISF block calculates its similarity with all other existing items in the website.
- Afterwards proposed approach creates the list (l_u) of all the existing items that are rated or reviewed by the target user. Next it calculates the FRV_{ui_1} of new item (i_1) for the target user (u) from the FRVs of items in list l_u using Eq. (25).

$$FRV_{ui1} = \frac{\sum_{i_2 \in l_u} FRV_i \; ItemSim_{i_1,i_2}}{\sum_{i_2 \in l_u} ItemSim_{i_1,i_2}} \tag{25}$$

Once FRV value of newly added item is calculated, it is placed in the recommendation list of the target user based on its FRV value. The proposed approach gives a fair chance to the newly added item to be included into the recommendation list of the target user over a period of time. Items are placed in recommendation list of the target user in decreasing order of their final recommendation values. Our approach exploits content based filtering and collaborative filtering and takes motivation from the research works of Balabanovic and Shoham (1997), Claypool et al. (1999) and Ghazanfar and Prugel-Bennett (2011) to

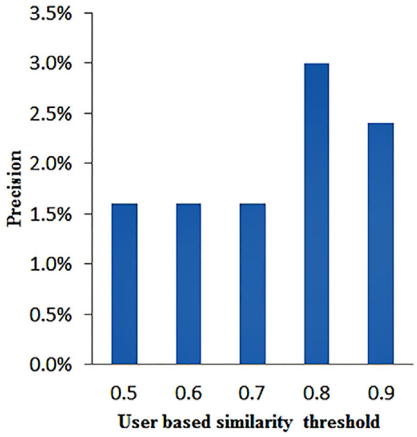

Fig. 5. Precision graph of user based collaborative recommendations with different similarity values.

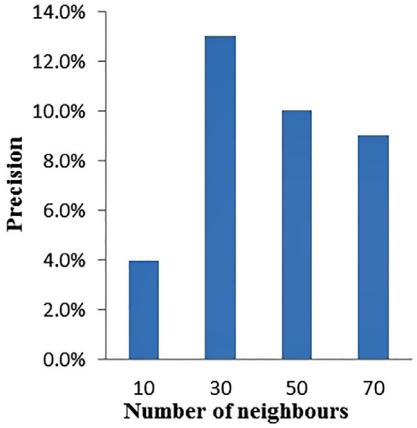

Fig. 6. Precision graph of user based collaborative recommendations using different number of neighbors.

address the gray sheep users problem. To handle GS users problem, the DCB block of the proposed approach in association with USF block congregates users having different mindset from the rest of the community. Afterwards these gray sheep users help each other within their group, in generating quality recommendations by taking the advantage of CC, DOM and MF blocks of the proposed approach.

4. Experimental results

The performance of the proposed system is evaluated by comparing it with standard recommendation approaches using live dataset.

4.1. Experimental setup and dataset

The proposed system is completely implemented in Java7 using Netbeans 8.0 framework. Reviews, ratings and other related information is stored in MySql database. The complete system runs over Intel Core i5 machine with 4 GB RAM.

4.2. Performance evaluation

The proposed system is evaluated over live dataset of the website myopinions.in, having 610 genuine users, 1612 reviews and 3600 ratings of 1200 books and movies. The results are compared

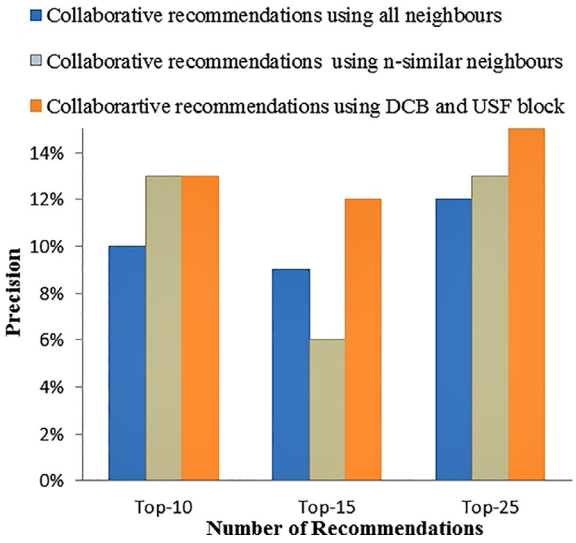

Fig. 7. Precision comparison of different collaborative recommendations techniques.

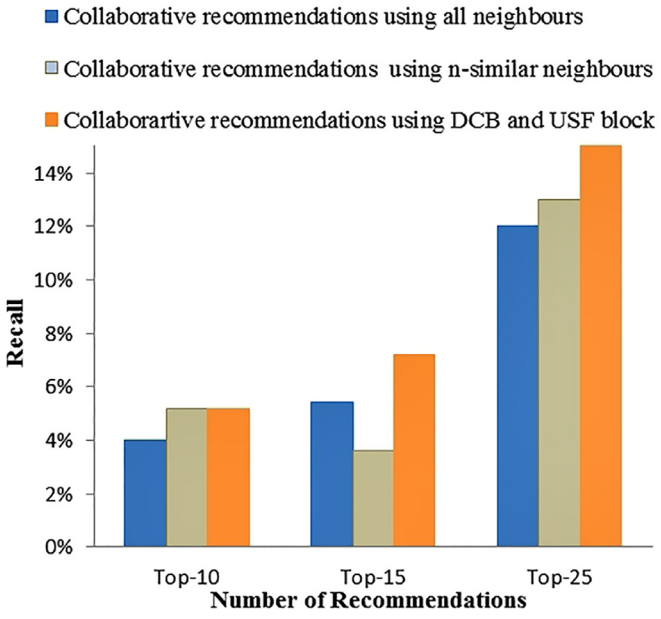

Fig. 8. Recall comparison of different collaborative recommendations techniques.

with existing benchmark methods of recommendations like collaborative filtering, matrix factorization, taxonomy based item recommendations, opinion based recommendation system, recommendation techniques using rating variance. In our experiments movies are considered as items, rated in the scale of 1–5. The main idea of any e-commerce website is to sell items, keeping this fact in light, the proposed approach does not predict the ratings of recommended items, instead it focuses whether items suggested by it are bought by the user or not.

4.2.1. Evaluation metrics

The proposed system is evaluated using precision, recall and F-measure. *Precision* in the context of RS is the fraction of relevant recommendations generated to the total number of recommendations generated to the user, whereas *recall* is the fraction of relevant recommendations generated to the total number of relevant recommendations to the user. They are computed as follows:

$$Precision = \frac{Relevant\ recommendations\ generated}{Total\ number\ of\ recommendations\ generated} \quad (26)$$

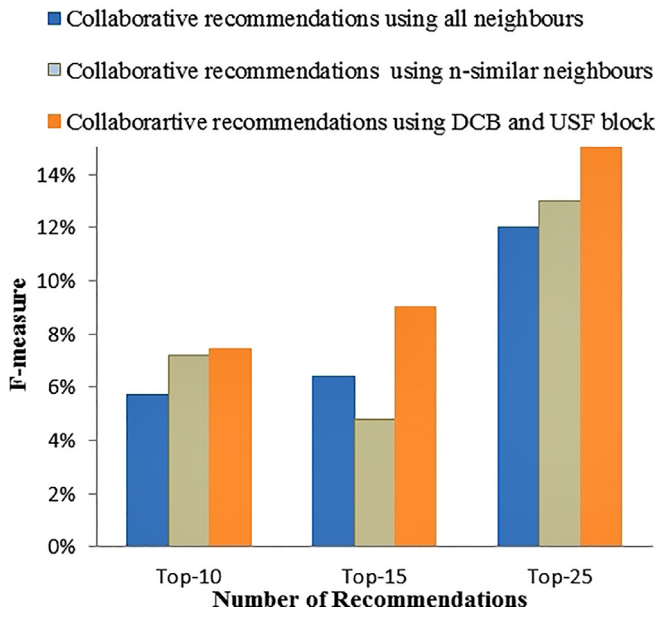

Fig. 9. F-measure comparison of different collaborative recommendations techniques.

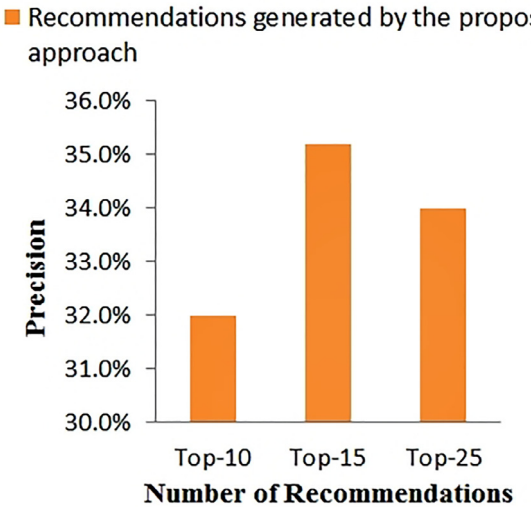

Fig. 10. Precision value graph of the proposed approach.

$$Recall = \frac{Relevant\ recommendations\ generated}{Total\ number\ of\ relevant\ recommendations} \quad (27)$$

F-measure combines the precision and recall into a single measure. F-measure makes comparison among algorithms and across datasets uncomplicated. It is defined as follows:

$$F - measure = \frac{2 \times Precision \times Recall}{Precision + Recall} \quad (28)$$

The value of all the parameters and results are obtained by doing five-fold cross validation on the dataset.

4.2.2. Results and discussion

The purpose of our experiments is to evaluate the quality of top-n recommendations for small value of n, generated by the proposed approach.

For choosing the value of n, we have conducted a survey among 300 people. Based on the outcome of the survey we have concluded that user generally prefers to see less than 20–25 recommendations at a time. So we have chosen the value of n as 25 and

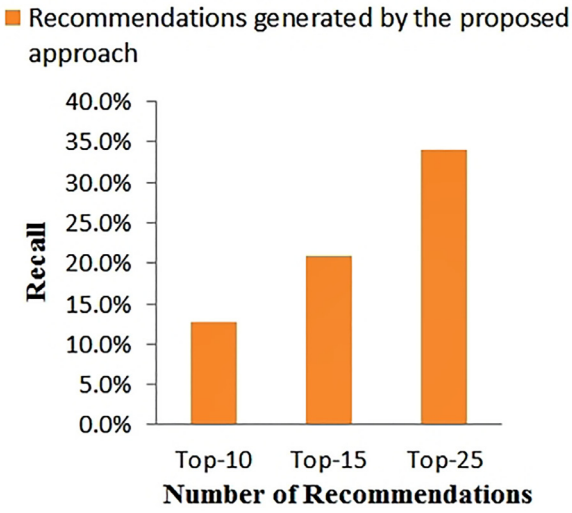

Fig. 11. Recall value graph of the proposed approach.

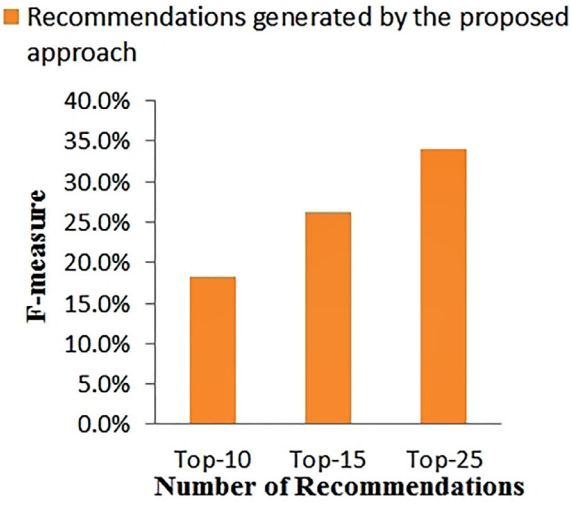

Fig. 12. F-measure value graph of the proposed approach.

evaluated the performance of our system for top-25 recommendations.

The experiment is initiated by generating user based collaborative recommendations. Firstly the system has calculated the similarity among users using Eq. (5). Next we have generated recommendations for the target users by using *all* their neighbors having similarity with the target user greater than 0.5. The precision graph of recommendations with different values of similarity threshold is shown in Fig. 5.

It is visible in the graph shown in Fig. 5 that recommendations generated by user similarity value 0.8 have high precision value for used dataset. So for further experimentations, the user similarity value is set to 0.8.

In the next step, recommendations are generated by varying the number of neighbors having user similarity value equal to 0.8 with the target user is shown in the precision graph of top-25 recommendations in Fig. 6. It is clear from graph that 30 neighbors of target user are sufficient to get better results. So we have set the value of k as 30 for the collaborative classifier block to generate recommendations.

In the next step USF block selects 30 similar users to the target user based on their DKVs using Eq. (19). Now again collaborative recommendations for target user are generated using these simi-

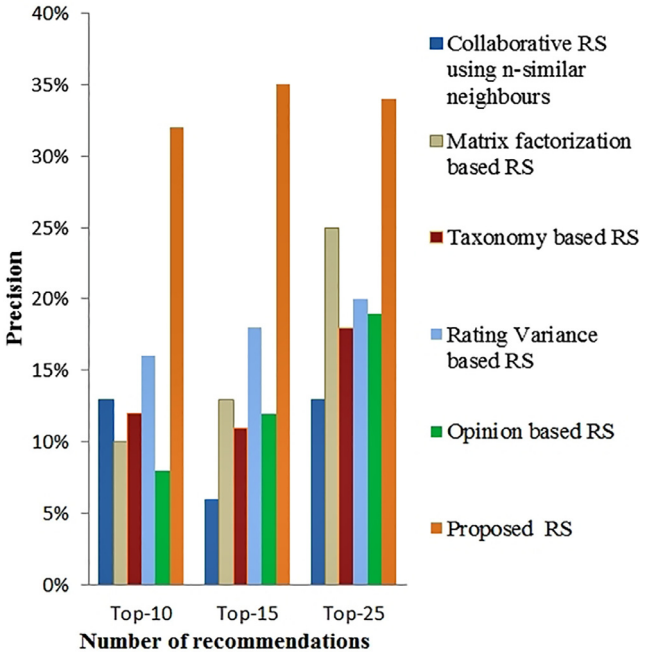

Fig. 13. Precision comparison graph of different recommendation methods.

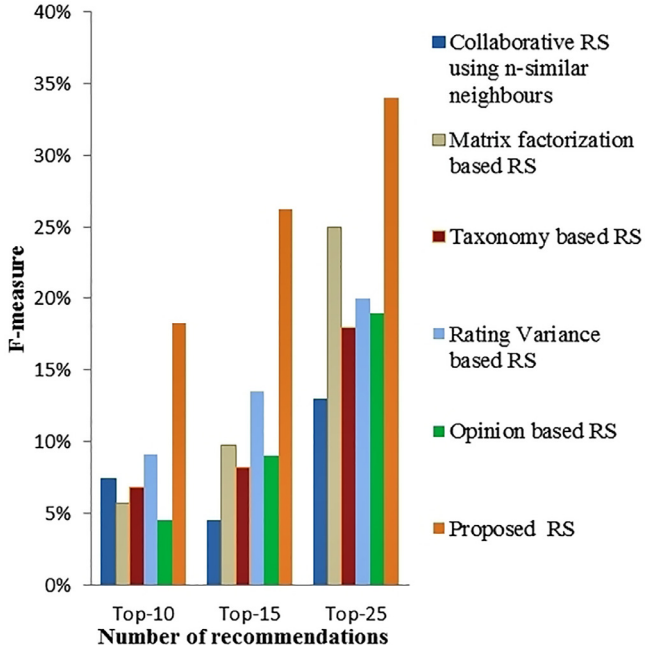

Fig. 15. F-measure comparison graph of different recommendation methods.

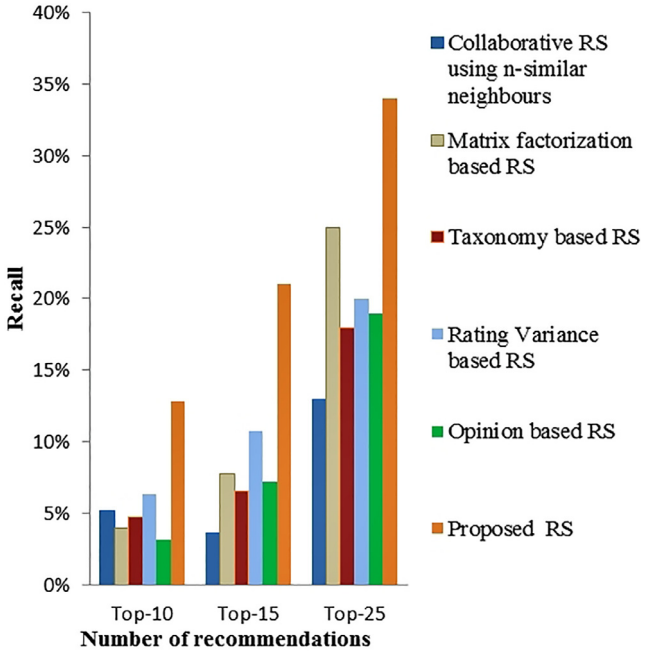

Fig. 14. Recall comparison graph of different recommendation methods.

lar users. The precision, recall and F-measure comparison graphs of user based collaborative recommendations for the target user, generated from all neighbors with similarity greater than or equal to 0.8, k-most similar users based on Pearson similarity and k-most similar users generated by proposed USF block are shown in Figs. 7–9.

Next CC block creates two item list for the target user as *like* and *dislike*. All the items having predicted ratings greater than equal to the target user's third quartile ratings are kept in the *like* list and remaining items are kept in *dislike* list.

Next the system calculates sentiment score of each review using Algorithm 2 and then uses this RSS to find its equivalent rating using Eqs. (21) and (22). These ratings are stored in Opinion Rating table. Now the DOM block finds popularity of every item in form of weights using Eq. (23). Afterwards MF block finds the approximate missing ratings of items using Opinion Rating table.

Finally the CR block collects the input from Matrix Factorizer, Collaborative Classifier and Dual purpose Opinion Miner blocks. Next website dependent weights (w_1, w_2 and w_3) are applied to each outcome of Matrix factorizer, Collaborative Classifier and Dual purpose Opinion Miner blocks. Weights are calculated as follows:

- During the process of 5-fold cross validation, initially we have assigned equal importance to the output of all three blocks (CC, DOM and MF) by setting the values of w1=w2=w3=0.1 and observed the precision values of top- 25 recommendations for each user from training data.
- Afterwards for each user we have incremented the value of w_1 by the incremental factor of 0.1, keeping the other two weights constant until further increment in the value of weight w_1, gives no significant improvement in the precision value of top-25 recommendations for that user.
- The above process is repeated for other two weights w_2 and w_3 to find the optimum values of w1, w2 and w3 for each user in the training data.
- The above three steps of weight calculation are repeated for each user and each iteration produces different values.
- At the end of all iterations final values of weights (w_1, w_2 and w_3) are obtained by calculating the mean value of each weight of different users.

In the used dataset final calculated values of weights (w_1, w_2 and w_3) are 0.8, 0.33 and 0.9. These values are used to generate final recommendations to different users in the test data. The precision, recall and F-measure graphs of the proposed recommendation approach are shown in Figs. 10–12.

The proposed RS is inspired by many state-of-the-art recommendation systems like collaborative recommendations system, matrix factorization based recommendations, taxonomy based item recommendations (Ziegler et al., 2004), recommendation techniques using rating variance (Kwon, 2008) and opinion based recommendation system (Dong et al., 2016). The comparison graphs of precision, recall and F-measure values of the proposed recom-

mendation method with other standard recommendation methods shown in Figs. 13–15.

5. Conclusions and future work

Almost all existing e-commerce recommendation systems have put all their efforts in augmenting all interesting items of target user to their recommendation list, without any priority to the sequence of items in the recommendation list. So these recommendation systems have high recall value but for smaller values of top-*n* recommendations they have very low precision value. It means they have very few relevant items in small *n* item recommendation list. In the proposed approach items are arranged in recommendation list in such a way that items of user's interest come at the beginning of the recommendation list even for smaller top-*n* recommendations. The proposed approach significantly has shown around 34% precision for top-*n* recommendations. It has its own unique DCB block which creates the profile of each user based on its past and current interest. The DCB dynamically changes its content of user profile with time, which indirectly helps in generating good quality recommendations. The proposed RS generates recommendations for the target user in collaboration with other users and uses matrix factorization technique to find the approximate ratings of missing ratings in database. The proposed approach has its unique feature that finds the popularity of items in the market using opinion mining. All these unique features collectively help the proposed RS in creating relevant smaller top-*n* recommendations list for the target user and also help in alleviating item side cold start and gray sheep problems. The proposed RS works on data related to item information, ratings and reviews that are mostly available in all e-commerce websites and it can be easily incorporated in any e-commerce website without putting much efforts. The experimental results show that the proposed RS considerably outperformed the other benchmark recommendation methods like collaborative filtering, matrix factorization, taxonomy based recommendations, rating variance based RS and opinion based recommendation system separately. In future it can be extended by including tag related information of items and trust of one user on another user while generating recommendations.

References

Adomavicius, G., & Tuzhilin, A. (2005). Toward the next generation of recommender systems: A survey of the state-of-the-art and possible extensions. *IEEE Transactions on Knowledge and Data Engineering, 17(6)*, 734–749.. doi:10.1109/TKDE.2005.99.

Agarwal, A., Chakraborty, M., & Chowdary, C. (2017). Does order matter? effect of order in group recommendation. *Expert Systems with Applications, 82*, 115–127. doi:10.1016/j.eswa.2017.03.069.

Ahn, H. (2008). A new similarity measure for collaborative filtering to alleviate the new user cold-starting problem. *Information Sciences, 178(1)*, 37–51. doi:10.1016/j.ins.2007.07.024.

Alami, Y., Nfaoui, E., & Beqqali, O. (2015). Toward an effective hybrid collaborative filtering: A new approach based on matrix factorization and heuristic-based neighborhood. *In proceedings of the IEEE conference on intelligent systems and computer vision*. doi:10.1109/ISACV.2015.7105543.

Balabanovic, M., & Shoham, Y. (1997). Fab: Content-based, collaborative recommendation. *In Communications of the ACM, 40(3)*, 66–72. doi:10.1145/245108.245124.

Basu, C., Hirsh, H., & Cohen, W. (1998). Recommendation as classification: Using social and content-based information in recommendation. In *In proceedings of 15th national conference on artificial intelligence* (pp. 714–720.).

Claypool, M., Gokhale, A., Miranda, T., Murnikov, P., Netes, D., & Sartin, M. (1999). Combining content-based and collaborative filters in an online newspaper. *In proceedings of ACM SIGIR workshop on recommender systems*. doi: WPI-CS-TR-99-16

Dong, R., O'mahony, M., Schaal, M., Mccarthy, K., & Smyth, B. (2016). Combining similarity and sentiment in opinion mining for product recommendation. *Journal of Intelligent Information Systems, 46(2)*, 285–312. doi:10.1007/s10844-015-0379-y.

Esuli, A., & Sebastiani, F. (2006). Sentiwordnet: A publicly available lexical resource for opinion mining. In *In proceedings of 5th conference on language resources and evaluation* (pp. 417–422.).

Fan, J., Pan, W., & Jiang, L. (2014). An improved collaborative filtering algorithm combining content-based algorithm and user activity,. (pp. 88–91). 10.1109/BIGCOMP.2014.6741413.

Ghazanfar, M. A., & Prugel-Bennett, A. (2011). Fulfilling the needs of gray-sheep users in recommender systems, a clustering solution. *In proceedings of international conference on information systems and computational intelligence*.

Goldberg, D., Nichols, D., Oki, B. M., & Terry, D. (1992). Using collaborative filtering to weave an information tapestry. *Magazine Communications of the ACM, 35(12)*, 61–70..

Hardeniya, T., & Borikar, D. A. (2016). An approach to sentiment analysis using lexicons with comparative analysis of different techniques. *IOSR Journal of Computer Engineering, 18(3)*, 53–57.. doi:10.9790/0661-1803015357.

Herlocker, J. L., Konstan, J. A., Borchers, A., & Riedl, J. (1999). An algorithmic framework for performing collaborative filtering. In *In proceedings of 22nd international ACM SIGIR conference on research and development in information retrieval* (pp. 230–237.). doi:10.1145/312624.312682.

Homoceanu, S., Loster, M., Lofi, C., & Balke, W. (2011). Will i like it? providing product overview based on opinion excerpts. In *In proceedings of 13th IEEE conference on commerce and enterprise computing* (pp. 26–33.). doi:10.1109/CEC.2011.12.

Houseman, E. M., & Kaskela, D. E. (1970). State of the art of selective dissemination of information. *IEEE Transactions on Engineering Writing Speech III, 2*, 78–83.

Hu, M., & Liu, B. (2004). Mining and summarizing customer reviews. In *In proceedings of the 10th international conference on knowledge discovery and data mining* (pp. 168–177.). doi:10.1145/1014052.1014073.

Ilhami, M., & Suharjito (2014). Film recommendation systems using matrix factorization and collaborative filtering. In *In proceedings of IEEE international conference on information technology systems and innovation* (pp. 1–6.). doi:10.1109/ICITSI.2014.7048228.

Khan, A., Baharudin, B., & Khan, K. (2011). Sentiment classification from online customer reviews using lexical contextual sentence structure. In *In proceedings of international conference on software engineering and computer systems- Springer: 179* (pp. 317–331.). doi:10.1007/978-3-642-22170-5-28.

Koren, Y. (2008). Factorization meets the neighborhood: A multifaceted collaborative filtering model. In *In proceedings of the 14th ACM SIGKDD international conference on knowledge discovery and data mining* (pp. 426–434). doi:10.1145/1401890.1401944.

Koren, Y., Bell, R., & Volinsky, C. (2009). Matrix factorization techniques for recommender systems. *IEEE Computer, 42(8)*, 30–37.. doi:10.1109/MC.2009.263.

Krulwich, B., & Burkey, C. (1996). Learning user information interests through extraction of semantically significant phrases. In *In proceedings of AAAI Spring symposium on machine learning in information access* (pp. 110–112).

Kwon, Y. (2008). Improving top-n recommendation techniques using rating variance. In *In proceedings of ACM conference on recommender systems* (pp. 307–310). doi:10.1145/1454008.1454059.

Lang, K. (1995). Newsweeder: Learning to filter netnews. In *In proceedings of 12th international conference on machine learning* (pp. 331–339).

Li, H., Cai, F., & Liao, Z. (2012). Content-based filtering recommendation algorithm using HMM. In *In proceedings of the 4th international conference on computational and information sciences* (pp. 275–277.). doi:10.1109/ICCIS.2012.112.

Li, W., & Sun, B. (2015). An improved collaborative filtering recommendation algorithm incorporating opinions analysis. In *In proceedings of IEEE 7th international conference on intelligent human-machine systems and cybernetics* (pp. 171–173.). doi:10.1109/IHMSC.2015.127.

Li, X., Wang, H., & Yan, X. (2015). Accurate recommendation based on opinion mining. In *Genetic and evolutionary computing: 329* (pp. 399–408). Springer. Advances in intelligent systems and computing. doi:10.1007/978-3-319-12286-1-41.

Lika, B., Kolomvatsos, K., & Hadjiefthymiades, S. (2014). Facing the cold start problem in recommender systems. *Expert Systems with Applications, 41(4)*, 2065–2073. doi:10.1016/j.eswa.2013.09.005.

Lops, P., De Gemmis, M., & Semeraro, G. (2011). Content-based recommender systems: State of the art and trends. In *Recommender systems handbook* (pp. 73–105.). Springer.

Miller, G. A., Beckwith, R., Fellbaum, C., Gross, D., & Miller, K. J. (1990). Introduction to wordnet: An on-line lexical database. *International Journal of Lexicography, 3(4)*, 235–244. doi:10.1093/ijl/3.4.235.

Ott, P. (2008). *Incremental matrix factorization for collaborative filtering.* Anhalt University of Applied Sciences.

Pang, B., Lee, L., & Vaithyanathan, S. (2002). Thumbs up? sentiment classification using machine learning techniques. In *In proceedings of the ACL-02 conference on empirical methods in natural language processing: 10* (pp. 79–86.). doi:10.3115/1118693.1118704.

Pazzani, M., & Billsus, D. (2007). Content–based recommendation systems. In *The adaptive web: 4321* (pp. 325–341.). Springer–Verlag LNCS.

Resnick, P., Iacovou, N., Suchak, M., Bergstrom, P., & Riedl, J. (1994). Grouplens: An open architecture for collaborative filtering of netnews. In *In proceedings of ACM conference on computer supported cooperative work* (pp. 175–186.). doi:10.1145/192844.192905.

Salvetti, F., Lewis, S., & Reichenbach, C. (2004). Automatic opinion polarity classification of movie reviews. In *In proceedings of colorado research in linguistics: 17(1)* (pp. 1–15.).

Sarwar, B., Karypis, G., Konstan, J., & Riedl, J. (2000). Analysis of recommendation algorithms for e-commerce. In proceedings of the 2nd ACM conference on electronic commerce, (pp. 158–167). doi:10.1145/352871.352887.

Sarwar, B. M., Konstan, J. A., Borchers, A., Herlocker, M. B. J., & Riedl, J. (1998). Using filtering agents to improve prediction quality in the grouplens research collab-

orative filtering system. In *In proceedings of ACM conference on computer supported cooperative work* (pp. 345–354). doi:10.1145/289444.289509.

Shambour, Q., Hourani, M., & Fraihat, S. (2016). An item-based multi-criteria collaborative filtering algorithm for personalized recommender systems. *International Journal of Advanced Computer Science and Applications, 7(8)*, 274–279.

Shardanand, U., & Maes, P. (1995). Social information filtering: Algorithms for automating word of mouth. In *In proceedings of the SIGCHI conference on human factors in computing systems* (pp. 210–217.). doi:10.1145/223904.223931.

Singh, M., & Mehrotra, M. (2015). Implicit behavior addition for improving recommendations. *In proceedings of IEEE international conference on computing, communication and automation*. doi:10.1109/CCAA.2015.7148350.

Takacs, G., Pilaszy, I., Nemeth, B., & Tikk, D. (2008). Investigation of various matrix factorization methods for large recommender systems. In *In proceedings of IEEE international conference on data mining workshops* (pp. 553–562.). doi:10.1109/ICDMW.2008.86.

Wei, J., He, J., Chen, K., Zhou, Y., & Tanga, Z. (2017). Collaborative filtering and deep learning based recommendation system for cold start items. *International Journal Expert Systems with Applications, 69*, 29–39. doi:10.1016/j.eswa.2016.09.040.

Zhang, M., Tang, J., Zhang, X., & Xue, X. (2014). Addressing cold start in recommender systems: A semi-supervised co-training algorithm. In *In proceedings of the 37th international ACM SIGIR conference on research and development in information retrieval* (pp. 73–82). doi:10.1145/2600428.2609599.

Ziegler, C., Lausen, G., & Schmidt-Thieme, L. (2004). Taxonomy driven computation of product recommendations. In *In proceedings of the 13th ACM international conference on information and knowledge management* (pp. 406–415). doi:10.1145/1031171.1031252.