ISSN:0975-9646
Kumar Abhishek et al, / (IJCSIT) International Journal of Computer Science and Information Technologies, Vol. 2 (3) , 2011, 1272-1278

A Review on Personalized Information Recommendation System Using Collaborative Filtering

Kumar Abhishek, Saurabh Kulkarni, Vipin Kumar.N Archana, P. Kumar
Department of Computer Science and Engineering Manipal Institute of Technology, Manipal-576104

***Abstract*-- The term Collaborative Filtering is used as a backbone in almost all Commercial Recommendation Systems today. This paper attempts to review all major developments in area of Collaborative Filtering. Main application area is outlined and examples of applications of Collaborative Filtering are presented.**

***Keywords*- Collaborative filtering, Recommendation system, Concept drift, Trust, Cold-start, Hybrid model**

I. INTRODUCTION

Much of the information on the Internet today consists of documents made available to many recipients through mailing lists, distribution lists, bulletin boards, asynchronous computer conferences, newsgroups, and the World Wide Web. World Wide Web is a huge source of information. It will give information on almost all topics existing today. Because of this today's age is called information age. But information available to user has many options available which will consume a lot of his time and efforts. This leads to information overload. This issue can also be seen as a quality problem: people want to read the most interesting messages, and want to avoid having to read low-quality or uninteresting messages.

Filtering is tools to help people find the most valuable information, so that the limited time spent on reading / listening/ viewing can be spent on the most interesting and valuable documents. Filters are also used to organize and structure information. Filtering is also needed on the search results from Internet search engines. Future software for the Internet can be expected to employ more advanced and user-friendly filtering functions than today, in order to support less computer-specialist users. Since people download millions of messages and web documents every day, and very often do not immediately get what they would mostly like to get, the gains through better filtering are enormous. Even a filter with a 10 % efficiency gain, the gain would be worth billions of dollar a year.

Collaborative Filtering is a technique used in almost all Recommendation Systems. Recommendation Systems are active information filtering systems that attempt to present to the user information items the user is interested in. These systems add information items to the information flowing towards the user, as opposed to removing information items from the information flow towards the user. Recommendation systems help overcome information overload by providing personalized suggestions based on a history of a user's likes and dislikes.

Recommendation Systems automate the task of "word-of-mouth" by which people recommend products, services to one another [14]. In situations where we don't have any previous experience we will rely on other people's opinion. Same task is done by Recommendation system by collecting opinions in the form of ratings and giving suggestions or recommendations on items for which a particular user has not given any rating.

II. COLLABORATIVE FILTERING

The collaborative filtering problem can be defined as follows:
Given a database D as a tuple
< Ui; Ij ;Oij >,
Where *Ui* identifies the i-th user of the system, *Ij* iden-tifies the j-th item of the system and *Oij* represents the i-th user's opinion on the j-th item, find a list of k recommended items for each user U[1].

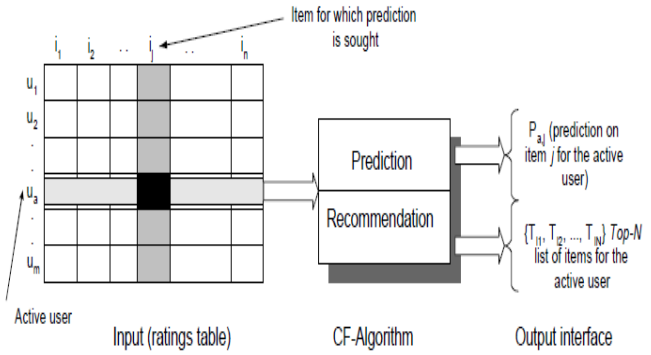

Fig. 1.The Collaborative Filtering process[2]

As shown in Fig. 1. Given Rating table as an input, where u_1, u_2,..,u_m are users of the system and i_1,i_2,...,i_n is a list of items for which prediction is sought, Collaborative Filtering algorithm applies two processes:

a) Prediction is a numerical value $P_{a,j}$ expressing the predicted likeliness of item i_j that does not belong to $I_{u,a}$. This predicted value is in same scale as opinion values provided by user u_a[2].

b) Recommendation is a list of N items that the active user will like the most. This recommended list must be on items not already purchased by the active user. This interface of Collaborative filtering algorithm is called *Top-N recommendation*[2].

There are two types of Collaborative Filtering algorithms:

A) Model Based Collaborative Filtering

Models are developed using data mining, machine learning algorithms to find patterns based on training data. These are used to make predictions for real data. There are many model based CF algorithms. These include Bayesian Networks, clustering models, latent semantic models such as singular value decomposition, probabilistic latent semantic analysis, Multiple Multiplicative Factor, Latent Dirichlet allocation and Markov decision process based models.

This approach has a more holistic goal to uncover latent factors that explain observed ratings. Most of the models are based on creating a classification or clustering technique to identify the user based on the test set. The number of the parameters can be reduced based on types of principal component analysis [1, 2].

B) Memory Based Collaborative Filtering

This mechanism uses user rating data to compute similarity between users or items. This is used for making recommendations. This was the earlier mechanism and is used in many commercial systems. It is easy to implement and is effective. Typical examples of this mechanism are neighbourhood based CF and item-based/user-based top N recommendations .

The neighbourhood-based algorithm calculates the similarity between two users or items, produces a prediction for the user taking the weighted average of all the ratings. Similarity computation between items or users is an important part of this approach. Multiple mechanisms such as Pearson correlation and vector cosine based similarity are used for this.

The user based top-N recommendation algorithm identifies the k most similar users to an active user using similarity based vector model. After the k most similar users are found, their corresponding user-item matrices are aggregated to identify the set of items to be recommended. A popular method to find the similar users is the Locality sensitive hashing, which implements the nearest neighbour mechanism in linear time.

The advantages with this approach is the explain ability of the results, which is an important aspect of recommendation systems. It is easy to create and use. New data can be added easily and incrementally. It need not consider the content of the items being recommended. The mechanism scales well with co-rated items.

There are several disadvantages with this approach. First, it depends on human ratings. Second, its performance decreases when data gets sparse, which is frequent with web related items. This prevents the scalability of this approach and has problems with large datasets. Third, it cannot handle new users or new items.

There are two types of Memory Based Collaborative Filtering:

1. User Based Collaborative Filtering

Here similarity between two user vectors is computed using similarity measures and predictions about user's preferences are made according to those measures.

There are two problems with this approach:

a) **Sparsity:** In practice, many commercial recommender systems are used to evaluate large item sets (e.g., Amazon.com recommends books and CDnow.com recommends music albums). In these systems, even active users may have purchased well under 1% of the items (of 2 million books is 20,000 books). Accordingly, a recommender system based on nearest neighbour algorithms may be unable to make any item recommendations for a particular user. As a result the accuracy of recommendations may be poor.

b) **Scalability**: Nearest neighbour algorithms require computation that grows with both the number of users and the number of items. With millions of users and items, a typical web-based recommender system running existing algorithms will suffer serious scalability problems.

Because of these drawbacks it is not implemented in most of the E-commerce websites [1, 2].

2. Item Based Collaborative Filtering

If the algorithm computes the similarity between different items and uses a set of items as nearest neighbours to do recommendation, it is called item-based collaborative filtering. It is also called item-item Collaborative Filtering.

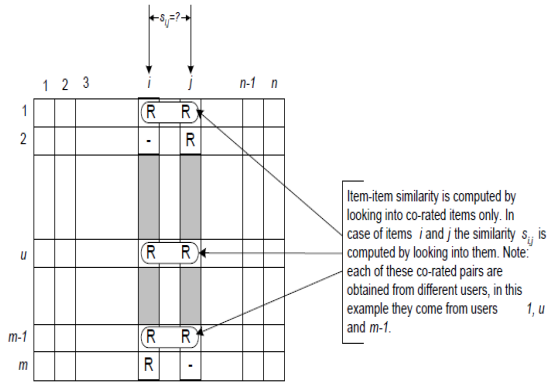

Fig. 2 .Isolation of the co-rated items and similarity computation [2].

Most of the E-Commerce websites make use of item based Collaborative Filtering to predict preferences of their users and increase their profit by giving recommendations to their users regarding items they like. Examples of such sites are amezon.com, CDNow.com etc. Our further discussion will be on different methods used in item-based collaborative filtering and how to improve algorithms currently used for collaborative filtering to overcome their limitations.

There are two phases of item based Collaborative Filtering:

Phase 1: Similarity computation

Here we compute similarity between two items i and j using some similarity measure as discussed below:

1. Cosine Similarity

The similarity between different items is measured by computing the cosine of the angle between different vectors as:

$$sim(I_a, I_b) = \cos(\vec{a}, \vec{b}) = \frac{\vec{a} \cdot \vec{b}}{|\vec{a}||\vec{b}|} = \frac{\Sigma_i^m O_{ia} \times O_{ib}}{\sqrt{\Sigma_i^m O_{ia}^2}\sqrt{\Sigma_i^m O_{ib}^2}} \quad (1)$$

where I_a identifies the a-th item of the system.

O_{ia} represents the i-th user rating on the a-th item[1, 2].

2. Pearson Correlation Coefficient

The similarity between different items is measured as:

$$sim(I_a, I_b) = \frac{\Sigma_i^m (O_{ia} - \overline{O_i}) \times (O_{ib} - \overline{O_i})}{\sqrt{\Sigma_i^m (O_{ia} - \overline{O_i})^2} \sqrt{\Sigma_i^m (O_{ib} - \overline{O_i})^2}} \quad (2)$$

Where O_i is the average of the i-th user's ratings [1, 2].

Phase 2: Preference Prediction

The prediction on an item *i* for a user *j* can be computed by using the sum of the ratings of the user to items weighted by similarity between different items as:

$$O_{ij} = \frac{\Sigma_{c=1}^k O_{ic} \cdot sim(I_j, I_c)}{\Sigma_{c=1}^k sim(I_j, I_c)} \quad (3)$$

where I_j identifies the j-th item I_c identifies nearest neighbours of the j-th item.

O_{ij} represents the i-th user's rating on the j-th item [1].

Apart from this method Amazon suggested search based algorithm. Search- or content-based methods treat the recommendations problem as a search for related items. Given the user's purchased and rated items, the algorithm constructs a search query to find other popular items by the same author, artist, or director, or with similar keywords or subjects. If a customer buys the Godfather DVD Collection, for example, the system might recommend other crime drama titles, other titles starring Marlon Brando, or other movies directed by Francis Ford Coppola.

If the user has few purchases or ratings, search based recommendation algorithms scale and performs well. For users with thousands of purchases, however, it's impractical to base a query on all the items. The algorithm must use a subset or summary of the data, reducing quality. In all cases, recommendation quality is relatively poor. The recommendations are often either too general (such as best-selling drama DVD titles) or too narrow (such as all books by the same author). Recommendations should help a customer find and discover new, relevant, and interesting items. Popular items by the same author or in the same subject category fail to achieve this goal.

Rather than matching the user to similar customers, item-to-item collaborative filtering matches each of the user's purchased and rated items to similar items, then combines those similar items into a recommendation list.9 To determine the most-similar match for a given item, the algorithm builds a similar-items table by finding items that customers tend to purchase together. We could build a product-to-product matrix by iterating through all item pairs and computing a similarity metric for each pair. However, many product pairs have no common customers, and thus the approach is inefficient in terms of processing time and memory usage.

It's possible to compute the similarity between two items in various ways, but a common method is to use the cosine measure we described earlier, in which each vector corresponds to an item rather than a customer, and the vector's *M* dimensions correspond to customers who have purchased that item. This offline computation of the similar-items table is extremely time intensive, with *O* (N^2M) as worst case. In practice, however, it's closer to *O*(*NM*), as most customers have very few purchases. Sampling customers who purchase best-selling titles reduces runtime even further, with little reduction in quality. Given a similar-items table, the algorithm finds items similar to each of the user's purchases and ratings, aggregates those items, and then recommends the most popular or correlated items. This computation is very quick, depending only on the number of items the user purchased or rated [3].

III VARIOUS ASPECTS OF COLLABORATIVE FILTERING

There are many aspects of Collaborative Filtering including change in user's interest with respect to time, sparsity problem, trust on users, and development of hybrid model for recommendation. Paper discusses these issues and gives overview of different approaches for handling them.

A . Change in user's interest

The item-based Collaborative Filtering algorithm discussed above doesn't take into consideration the fact that user's interests keep on changing with respect to time and latest ratings given by the user reflect their current interest which should be given more importance over old ratings. Traditional algorithm gives equal importance for all ratings.

Change in user's interest with time is called *concept drift* [1].

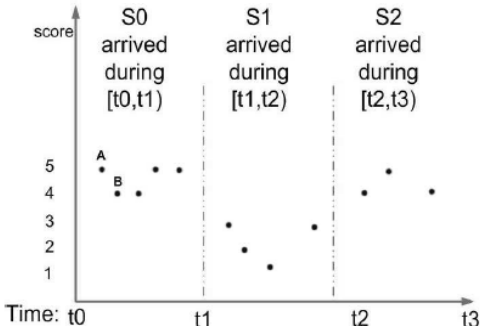

Fig. 3. Concept drift in Collaborative Filtering [1].

Consider a simple example to illustrate the problem of concept drift in collaborative filtering. One data point represents the user's rating on one item. For example, Figure 3 is used to represent the user Alice's preference for thriller movies. Data point A represents Alice's rating on one thriller: *The 39Steps*. Data point B represents Alice's rating on another thriller: *The Bourne Identity*. For every data point, the value of x means the rating's produced time and the value of y means the score. Here, Alice's rating on *The 39 Steps* is 5 and Alice's rating on *The Bourne Identity* is 4. *S*0 is the oldest data and *S*2 is the newest data. Let *Si* be the data that came in between time *ti* and *ti*+1. From Figure 3, we can see that Alice's preferences for thrillers are always changing. In *S*0 Alice's scores on thrillers are relatively high, while in *S*1 the scores are low. This means Alice's preference for thrillers went down at this period of time. However, in the time interval [*t2; t*3), Alice recovered the preference for thrillers. In summary, the challenge is: How do we deal with the changing data in collaborative filtering so

that the most accurate prediction can be obtained? On the one hand, in order to reduce the influence of old data that may represent old preferences, we should use the most recent data. For example, we should only use the data in *S*2. However, because of insufficient amount of data, the prediction precision is likely to be degraded. On the other hand, using all the historical data simply may also reduce the prediction precision. From Fig. 3, we can see that the discrepancy between the underlying trends of *S*1 and *S*2 becomes the cause of the problem. Most recent ratings reflect user's future preferences most. . Hence, in Figure 3, we should assign a high weight to *S*0 and *S*2 and a low weight to *S*1. This is because *S*2 represents the newest trend and, at the same time, *S*0 and *S*2 have similar data distribution [1].

Y.Ding, X.Li [1] suggested Recency based Collaborative Filtering algorithm. Algorithm suggests new similarity measure. It also has weights included for ratings. Most recent rating will get highest weight i.e. 1 and rest of the ratings will get less weight depending on recency. The prediction equation also incorporates weights of ratings.

Yehuda Koren[4] suggested Collaborative Filtering with Temporal Dynamics. This approach creates model which can track time changing behaviour throughout lifespan of data exploiting relevant components of all data instances and discarding irrelevant data instances. It uses time changing baseline predictors and time changing factor model. It also discusses about temporal dynamics at neighbourhood model.

Collaborative Filtering using Time Period partitioning is another way of handling *concept drift* .In this method user's rating's history is divided into many periods. Analysis of user's interest is done in each period. At the same time user's recent interest is found by setting a time window. By combining these two methods, algorithm tries to achieve the same goal. Algorithm is called TPPCF [5].

B. Sparsity and cold-start problem in Collaborative Filtering

The numbers of users and items in major e-commerce recommendation systems is very large. Even users that are very active result in rating just a few of the total number of items available in a database and respectively, even very popular item result in having been rated by only a few of the total number of users available in the database. This problem is called sparsity problem. This has a major negative impact on the effectiveness of a collaborative filtering approach. Because of sparsity, it is possible that the similarity between two users cannot be defined, rendering collaborative filtering useless. Even when the evaluation of similarity is possible, it may not be very reliable, because of insufficient information processed. The cold-start problem emphasizes the importance of sparsity problem. Cold-start refers to the situation in which an item cannot be recommended unless it has been rated by a substantial number of users. This problem applies to new and obscure items and is particularly detrimental to users with eclectic taste. Likewise, a new user has to rate a sufficient number of items before the recommendation algorithm be able to provide reliable and accurate recommendations [6].

Manos Papagelis [6] suggested trust-inference based approach to handle sparsity and cold-start problems. First phase talks about trust through user-user similarity. The next phase talks about finding trust inferences typically in case of sparse data. Here suppose user S and N has rated item i_1 and user N and T has rated item i_2 then this phase establishes trust between users S and T through user N. This propagation of trust indicates presence of trust path in the network. Evaluation and selection of paths is done using Maximum path confidence and Minimum Mean Absolute Deviation.

Hyung Jun Ahn [7] talks about new similarity measure for reducing impact of cold-start problem. It first talks about limitations of traditional similarity measures.

Those limitations include

(1) Very limited number of co-ratings under data sparsity.

(2) If the number of co-rated items is 1, correlation cannot be calculated.

(3) If all the available ratings of a given user are flat (for example <1, 1, 1>), then correlation cannot be calculated.

To overcome these limitations, a new similarity measure called PIP is designed. The measure includes Proximity, Impact, and Popularity as its components. Hence the name is PIP. For any two ratings r_1, r_2 PIP can be measured as:

PIP (r_1, r_2) =proximity (r_1, r_2)

*impact (r_1, r_2*popularity (r_1, r_2) (4)

Computation of proximity, impact and popularity are given in [7].

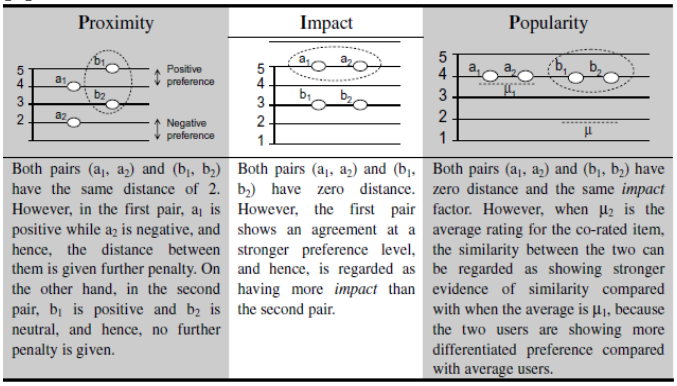

Fig.4.Description of factors of PIP using example rating [7]

Fig.4 shows three components of PIP using sample example. Pair (a_1, a_2) and pair (b_1, b_2) have same distance of 2 between them. However in first pair a_1 is positive and a_2 is negative. Therefore this pair is given further penalty. But in second pair b1 is positive and b2 is neutral hence there is no further penalty. However in case of impact, fig.4 shows that both pairs have zero distance but first pair (a_1, a_2) lies at high preference level hence has more impact than other pair. As far as popularity is concerned, as per fig.4, when two ratings a1and a2 are close to the average rating of the item, this agreement between the two might not provide much information about the similarity between the two users, because the similar ratings can be just a result of being close to the average alike. In contrast, if two ratings are close to each other but are very far from the average rating of the item as with the ratings b1 and b2 in the example when the average is l2, this can be signalling stronger similarity between the two users.

C. Trust in Collaborative Filtering

Recommender systems have proven to be an important response to the information overload problem, by providing users with more proactive and personalized information services. And collaborative filtering techniques have proven to be a vital component of many such recommender systems as they facilitate the generation of high-quality recommendations by leveraging the preferences of communities of similar users. John O'Donovan [8] suggests that the traditional emphasis on user similarity may be overstated. Additional factors have an important role to play in guiding recommendation. Specifically [8] proposes that the trustworthiness of users must be an important consideration. It presents two computational models of trust and show how they can be readily incorporated into standard collaborative filtering frameworks in a variety of ways. It also shows how these trust models can lead to improved predictive accuracy during recommendation. In addition to profile-profile similarity—the standard basis for partner selection—we argue that the trustworthiness of a partner should also be considered. A recommendation partner may have similar ratings to a target user but they may not be a reliable predictor for a given item or set of items. For example, when looking for movie recommendations we will often turn to our friends, on the basis that we have similar movie preferences overall. However, a particular friend may not be reliable when it comes to recommending a particular type of movie. The point is that partner similarity alone is not ideal. Our recommendation partners should have similar tastes and preferences and they should be trustworthy in the sense that they have a history of making reliable recommendations. John O'Donovan [8] proposes a number of computational models of trust based on the past rating behaviour of individual profiles. These models operate at the profile-level (average trust for the profile overall) and at the profile-item-level (average trust for a particular profile when it comes to making recommendations for a specific item). One of the models is based on trust based weighting. In this model, trust and similarity are combined to form compound weighting to be used in prediction formula.

Trust as one's expectation of another peer's competence in providing recommendations to reduce its uncertainty in predicting new items' ratings. A trust metric is defined which will incorporate trust in similarity computation. A trust propagation graph can be drawn based on it and recommendation can be made which will be closer to more accurate one [9].

D. Hybrid Models of Collaborative Filtering

In order to improve quality of recommendation, hybrid approach of Collaborative Filtering is suggested. Traditional memory based Collaborative Filtering algorithms work reasonably well in practice especially when an active user has rated significant number of items. David M. Pennock, Eric Horvitz [10] suggests Personality Diagnosis algorithm to generate predictions. One benefit of this approach is that the modelling assumptions are made explicit and are thus amenable to scrutiny, modification and even empirical validation. The algorithm has time and space complexity O (nm) as do most of the memory-based methods have. The model is depicted as a naive Bayesian network and has same structure of classical diagnostic model. So this approach combines both memory and model based methods to give more accurate prediction [10].

Thomas Tran and Robin Cohen [11] tells about hybrid recommender system that combines the collaborative filtering and knowledge-based approaches. Architecture consists of the major components like The Interactive Interface Agent, the Knowledge-Based Engine, the Knowledge Base of the product domain, the Collaborative Filtering Engine, the Database of Users' Ratings for Items and The Product Database.

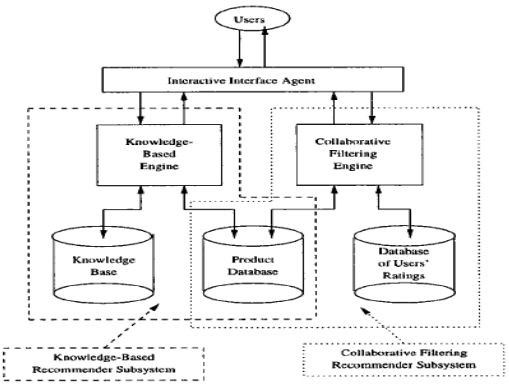

Fig.5.Architecture for Integrating Knowledge-Based and Collaborative Filtering Approaches [11]

ClustKNN is another algorithm which uses hybrid model i.e. model based and memory based approaches to recommend items. So it has advantages of both, memory based and model based approaches. This approach is simple and intuitive. For this purpose it uses partitional clustering algorithm for modeling users. To generate recommendations from the learned model, we use a nearest-neighbour algorithm. However, since the data is greatly compressed after the model is built, recommendations can be computed quickly, which solves the scalability challenge discussed previously [12].

One interesting property of ClustKNN is its tunable nature. Therefore ClustKNN adaptable to systems of different sizes and allows it to be useful throughout the life of a system as it grows.

Algorithm consists of two phases namely model building and prediction generation [12].

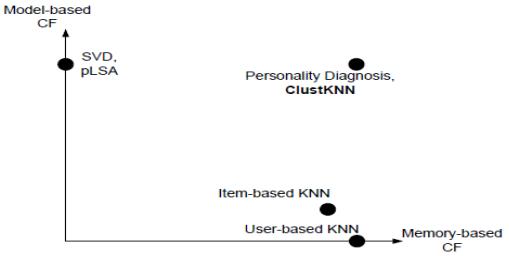

Fig.6. Space encompassed by different Collaborative Filtering Algorithms [12]

Fig.6 shows some selected algorithms and where they belong to when we construct space for model and memory based models.

TABLE 1[12]COMPARISON OF TIME COMPLEXITIES OF SELECTED ALGORITHMS[A]

CF algorithm	Offline	Online
pLSA	$O(mn)$	$O(m)$
SVD	$O(n^2m + m^2n)$	$O(m)$
Personality Diagnosis	-	$O(mn)$
CLUSTKNN	$O(mn)$	$O(m)$
User-based KNN	-	$O(mn)$
Item-based KNN	-	$O(mn)$

As shown in table 1, offline time complexity of ClustKNN is O (mn) and online time complexity is O (m) where m is number of users and n is number of items.
Another approach can be combination of Content Filtering with Collaborative Filtering. One such approach is presented in [13].

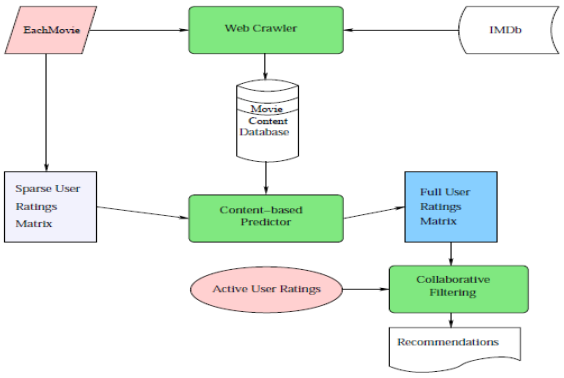

Fig.7. Content boosted Collaborative System Overview [13]

The web crawler uses the URLs provided in the EachMovie dataset to download movie content from IMDb. After appropriate pre-processing, the downloaded content is stored in the Movie Content Database. The EachMovie dataset also provides the user-ratings matrix, which is a matrix of users versus items, where each cell is the rating given by a user to an item. Each row of this matrix is called user rating vector. The user-ratings matrix is very sparse, since most items have not been rated by most users. The content based predictor is trained on each user-ratings vector and a pseudo user-ratings vector is created. A pseudo user-ratings vector contains the user's actual ratings and content-based predictions for the unrated items. All pseudo user-rating vectors put together form the pseudo ratings matrix, which is a full matrix. Now given an active user's ratings, predictions are made for a new item using Collaborative Filtering on the full pseudo ratings matrix. Reference [13] also describes algorithm for the approach specified.

III. COMMERCIALIZATION OF COLLABORATIVE FILTERING

Major application which makes use of Collaborative Filtering is Amazon. When a user buys an item, Amazon gives recommendations of items purchased together with the given item along with rating for those items. Here it makes use of Collaborative Filtering to suggest a list of items. Based on ratings given by other users, it will compute similarity among item purchased by a user and other items purchased and rated by other users of system. After that it will predict the rating which a user can give to that item and based on that rating it will suggest that item to the user.
Another application MovieLens is a movie recommender website which will suggest movies which the user will like based on predicted rating computed by it. It also uses Collaborative Filtering for the purpose of recommendation. The dataset of this website is available for research purpose. Similar kind of movie recommendation system is IMDb. IMDb is a huge source of movies, TV series dataset. It will give ratings for movies based on Collaborative Filtering.
For music lovers, rateyourmusic is a website which gives recommendations about albums, EPs, singles, videos, bootlegs and movies. This also uses Collaborative Filtering to suggest items. Before rateyourmusic, cdnow was a website which was giving recommendations on all albums, CDs, music etc.
Netflix is another website which uses Cinematch as recommendation algorithm for predicting user preferences. Like these many recommendation systems use Collaborative Filtering as underlying principle for recommending items to users based on predicted ratings.

V. FUTURE TRENDS IN COLLABORATIVE FILTERING

The major challenge in any Collaborative Filtering technique is time complexity as algorithm has to work on very large dataset. The number of items in database and number of users in database who have rated items are the major components involved in complexity. If offline computations are increased then resources required can be reduced as online computation requires more resources. The aim should be towards reducing complexity of algorithm.
On the semantic web, trust and reputation can be expressed using domain knowledge and Ontologies that provide a method for modelling the trust relationships that exist between entities and the content of information sources. Trust scores in System can be calculated through inference and propagation, of the form (A => B => C) => (A => C), where A, B and C are users with interpersonal trust scores. So using inference engine can the recommendation be improved? , is an area of research [8].

VI. CONCLUSION

During last 10-15 years Collaborative Filtering is developed as a very useful technology with various applications in recommendation system. This paper has reviewed this development and discussed various issues in Collaborative Filtering and solutions proposed for solving those issues. We hope that development in Collaborative Filtering will improve ratings predicted for users and will give more accurate recommendations suitable for user. This development will further extend the potential of Collaborative Filtering technology and allow Collaborative Filtering based recommendation systems to be used far more widely.

REFERENCES

[1] Y. Ding, X. Li., M.E. Orlowska. " Recency-Based Collaborative Filtering".ADC.2006

[2] B. Sarwar, G..Karypis, J. Konstan, and J.Riedl.Item-Based Collaborative Filtering Recommendation Algorithms. In International World Wide Web Conference, pp. 285-95 (2001)

[3] Greg Linden, Brent Smith, and Jeremy York(2003), Amazon.com recommendations, item-to-item collaborative filtering. IEEE Computer Society, February 2003.

[4] Yehuda Koren "Collaborative Filtering with Temporal Dynamics" 2009 ACM 978-1-60558-495-9/09/06

[5] Yuchuan Zhang, Yuzhao Liu(2010) A Collaborative Filtering Algorithm based on Time Period Partition IEEE Computer Society.

[6] Manos Papagelis, Dimitris Plexousakis, Themistoklis Kutsuras "Alleviating the Sparsity Problem of Collaborativ Filtering Using Trust Inferences"

[7] Hyung Jun Ahn, "A new similarity measure for collaborative filtering to alleviate the new user cold-starting problem", Information Sciences 178 (2008) 37–51

[8] John O'Donovan, Barry Smyth "Trust in Recommender Systems", *IUI'05*, January 9–12, 2005, San Diego, California, USA, ACM 1581138946/05/0001

[9] Jianshu Weng, Chunyan Miao, Angela Goh "Improving Collaborative Filtering with Trustbased Metrics", *SAC'06*, April, 2327, 2006, Dijon, France, ACM 1595931082/ 06/0004

[10] David M. Pennock, Eric Horvitz, Steve Lawrence and C. Lee Giles, "Collaborative Filtering by Personality Diagnosis: A Hybrid Memory- and Model-Based Approach", Proceedings of the Sixteenth Conference on Uncertainty in Artificial Intelligence (UAI-2000),pp. 473-480, Morgan Kaufmann, San Francisco, 2000.

[11] Thomas Tran and Robin Cohen, Hybrid Recommender Systems for Electronic Commerce, AAAI Technical Report WS-00-04.