Combining Collaborative, Diversity and Content Based Filtering for Recommendation System

Jenu Shrestha, Mohammed Nazim Uddin and GeunSik Jo

Intelligent E-Commerce Systems Lab., School of Computer Science & Engineering, Inha University
253 Younghyun-dong, Nam-Gu, Incheon, Korea 402-751
{jenustha, nazim}@eslab.inha.ac.kr, gsjo@inha.ac.kr

Abstract

Combining collaborative filtering with some other technique is most common in hybrid recommender systems. As many recommended items from collaborative filtering seem to be similar with respect to content, the collaborative-content hybrid system suffers in terms of quality recommendation and recommending new items as well. To alleviate such problem, we have developed a novel method that uses a diversity metric to select the dissimilar items among the recommended items from collaborative filtering, which together with the input when fed into content space let us improve and include new items in the recommendation. We present experimental results on movielens dataset that shows how our approach performs better than simple content-based system and naive hybrid system

Keywords: Recommendation system, collaborative and content based filtering, diversity.

Introduction

Recommender system suggests music, movies, books and other products and services to users based on history of likes and dislikes. A variety of techniques have been proposed for performing recommendation including collaborative, content-based, demographic, knowledge-based and other techniques, however the two most popular approaches being-collaborative filtering (CF)[7][10][12] and content-based (CB) recommending [7][10][12].CF systems aggregate ratings or recommendation of objects, recognize commonalities between users on the basis of their ratings, and generate new recommendations based on inter-user comparisons. A typical user profile in collaborative system consists of a vector of items and their ratings, continuously augmented as the user interacts with the system over time [6]. On the other hand, CB methods provide recommendations by comparing representations of content contained in an item to representations of content that interests the user.

Most collaborative filtering techniques falls into one of two categories: memory-based and model-based methods. Memory-based methods are deployed widely at many commercial websites, because not only they are simple and intuitive on a conceptual level, but also they are deemed sufficiently accurate for many real-world applications. Memory-based methods store historic user ratings in a database and identify users with the similar preferences with an active user for whom a recommendation has to be made. In the predicting phase, they would predict the active user's ratings based on the corresponding ratings of these similar or like-minded users. In contrast, model-based algorithms build models that can explain the records of historic ratings well and predict the ratings of active users using estimated models. Both types of approaches have been shown to be effective for collaborative filtering.

In content-based filtering, each user is assumed to operate independently and the system requires a profile of the user's needs or preferences in order to be able to provide a recommendation. Thus, a user entering a site and seeking a recommendation would have to provide information on her personal preference in order for the system to build a profile. The user profile includes information about the content of items of interest, namely web pages, movies, music etc. Using these items as a basis, the technique identifies similar items which are returned as recommendations. These techniques are particularly valuable to users who have specific interests and who are looking for related recommendations [17].

If used independently, both techniques exhibit certain weakness. *Early rater problem* i.e. a new item that has not had many ratings can't be easily recommended, *Sparsity problem* i.e. insufficient item ratings, *Gray sheep problem* i.e. users whose taste is unusually different than others, are most common problems in Collaborative filtering. Similarly, one of the limitations when using content-based techniques is that no new topics are explored. It recommends items closely related to those the user has previously rated. Such systems never reveal novel items that users might enjoy outside their usual set of choices [10] .This leads to over-specialization: one is restricted to seeing items similar to those that have already been rated highly. For example if a user rates comedy movies starring a small set of actors, it's more likely that the majority of content-based system recommendation will also be comedy movies starring those actors. This has been addressed in some cases with the injection of randomness. In the context of information

filtering, for example, the crossover and mutation operations (as part of genetic algorithm) have been proposed as a solution [19].

One common trend in recommender systems research is the need to combine recommendation techniques to achieve peak performance. Hybrid recommender systems combine these techniques to gain better performance with fewer of the drawbacks of any individual one. Many researchers have chosen different ways to combine these techniques. The Fab system [1], Libra [7], Collaborative via Content [2], Grouplens'filterbots' [8], Content boosted Collaborative [12], etc are some of them.

Accuracy being the main focus in collaborative filtering systems suffers in terms of user satisfaction with the products recommended. In recommender system while the k best recommendations are very similar to the target query, they are very similar to each other [4].Thus on Amazon.com, many recommendations seems to be similar with respect to the content [5]. For example, customers that have purchased Hermann Hesse's prose may happen to obtain recommendation lists where all top-5 entries contain books by that respective author only. Considering pure accuracy, all these recommendations appear excellent since the active user clearly appreciate books written by Hesse. But, assuming that the active user has several interests other than Hesse, the recommended set of items appears poor, owing to the lack of diversity [5].

Though diverse recommendation is not the core hub of this research, basically in collaborative–content system, where the output from first is fed into the second, this effect of similar items recommendation from the collaborative part is more pronounced degrading the performance of the later. Since, the main objective of the collaborative-content system is directing the user towards relevant information and extending prediction based on former to the later; the similar items from the former will squeeze the performance of the later, forcing the recommendation in one direction. Since the most prominent drawback of content-based system is the difficulty in exploring new items outside the usual choice, feeding similar items into it will again stop it from exploring extra items. To alleviate this problem, we developed this novel idea in which we first find the diverse set of items among the recommended items from the collaborative part i.e. dissimilar with each other but similar with respect to the active user, which, together with the active user's ratings when fed into content-based part will improve the recommendation and motivate the content-based filtering to explore items in a wider range which in turn helps to solve the cold start problem in an efficient way.

2. Related Work

Derek Bridge *et al* [9] pointed out the effect of diversity in conversational collaborative filtering. Unlike our technique; it used a conversational collaborative approach in recommendation. Barry Smyth [4] studied the effect of similarity and diversity in recommendation. Malcolm Slaney [22] described a way to measure the diversity of consumer's musical interests and characterized this diversity using published musical playlists. Similarly Cai-Nicolas et al [5] presented a novel method designed to balance and diversify personalized recommendation list in order to reflect the user's complete spectrum of interests.

In hybrid technique, A simple approach is to allow both content-based and collaborative filtering methods to produce separate recommendations, and then to directly combine their predictions [15]. It uses a weighted average of content-based prediction and collaborative prediction. The Fab system [1] combines collaborative and content-based filtering in its recommendations by measuring similarity between users after first computing a profile for each user. It used content analysis to create the user profile and compare these profiles to determine similar users for collaborative recommendation. Melville *et al* [12] followed a two-stage approach: first they applied a naive Bayesian classifier as content-based predictor to complete the rating matrix, and then they re-estimated ratings from this full rating matrix by Collaborative Filtering. In another approach, Basu *et al* [16] treat recommending as a classification task. They combine collaborative and content information, by creating features. The GroupLens research team working with Usenet news filtering also employed feature augmentation [8].They implemented a set of knowledge-based 'filterbots' using specific criteria such as the number of spelling errors and the size of included messages. In Pazzani's approach [2], each user-profile is represented by a vector of weighted words derived from positive training examples using the Winnow algorithm. Predictions are made by applying CF directly to the matrix of user-profiles. LIBRA [7] makes content-based recommendations of books based on data found in Amazon.com using a naïve Bayes text classifier. The text data used by the system includes *related authors* and *related titles*, that Amazon generates using collaborative algorithms. These features were found to make a significant contribution to the quality of recommendations.

3. Background Knowledge

3.1 Collaborative filtering

Collaborative Filtering stresses the concept of community, where every user contributes with her ratings to the overall performances of the system [18].Collaborative-based recommender systems can produce recommendations by identifying overlapping interests among users and computing the similarity between a user's preferences and those of other people. This technique is very similar to the way in which we ask friends or colleagues for recommendations. For instance, when a new movie has just come out and a friend of ours has already seen it, it is natural to ask him his opinion about it and once we have enough information , we can make decision whether it is worth watching the movie or not.

The user's preferences about different items create user-item matrices, such as the one illustrated in Table1.

Each of the cells in the matrix represents a user's rating for a specific item. The empty cells indicate that the user has not rated this item yet. The aim of a recommender system is to be able to fill in the empty cells in the user-item matrix in the most accurate way, as if the user herself had entered the rating in the matrix [17]. This rating we call as predicted rating for the user.

Table1: user-item matrix

	Item1	*Item2*	*Item3*	*Item4*	*Item5*
UserA	4	4	1	3	4
UserB	2	1	4	5	2
UserC	3	1	3	1	2
UserD	5	4	2	3	

In the above table, to predict the rating of *UserD* for *Item5*, collaborative filtering performs three steps.
It compares the current user's ratings against every other user's ratings. CF computes a similarity value for every other user. Usually the similarity measure is the Pearson correlation coefficient given by equation (1), where 1 means totally similar and -1 totally dissimilar, but any other could be used [3]. In this case, the similarity between *UserD* and *UserA*, *UserB*, *UserC* is 0.9, -0.7, and 0.0 respectively.
The second step is to select the neighborhood, which is the set of most similar users to the active user. In this example we take all three, but most recommender systems use only a subset of them. It might consider on the basis of threshold or highest *n* similarity values.
Finally, the weighted average of all the ratings given by the users i.e. neighbors on tem5 is calculated using equation (2), which gives the predicted rating of *UserD* on *Item5*. In this case, it is 4.5.
This prediction value seems reasonable given that *UserD* tends to agree with *UserA* who rated that item high. The process defined above is applied to all the items in the user item matrix and are recommended in descending order of the predicted value.
The main advantage of collaborative filtering is its ability to discover new items of interests because other people (neighbor) liked them. On the other hand, one of the disadvantages of this method is the first rater problem because the recommendations are items that similar users have rated so an item cannot be recommended until a user rates that item. Similarly it suffers from startup and sparsity problem as well.

3.2 Content-based filtering

The content-based approach to recommendation has its roots in the information retrieval (IR) community, and employs many of the same techniques. Text documents are recommended based on a comparison between their content and a user profile. For example, if a user profile contains the words "discovery", "knowledge" and "rules" a new paper about data mining is very likely to be recommended to her because the paper and user profile have words in common.
Content-based methods provide recommendations by comparing representations of content contained in an item to representations of content that interests the user. In content-based filtering items are matched either to a user's interest profile or query on the basis of content rather than opinion. One strength of this approach over collaborative filtering is that as long as the system has some information about each item, recommendations can be made even if the system has received a small number of ratings, or none at all.
A variety of algorithms have been proposed for analyzing the content of text documents and finding regularities in this content that can serve as the basis for making recommendations. Many approaches are a specialized versions of classification learners, in which the goal is to learn a function that predicts which class a document belongs to (i.e., either liked or not-liked). Other algorithms would treat this as a regression problem in which the goal is to learn a function that predicts a numeric value (i.e., the rating of the document) [2]. Different methods have been implemented in content-based filtering, the most famous being the bag-of- words naïve Bayesian text classifier [12][13]extended to handle a vector of bags of words where each bag-of words corresponds to feature. The classifier learns a user profile from a set of labeled documents and later uses it to predict the label (rating) of unrated documents.
Similarly Salter [10] proposed a simple content-based system that takes rating as input to the process and uses a simple scoring mechanism to find other similar items.
The main advantage of the content-based system is that it doesn't possess first rater problem. This is due to the fact that it recommends item to a user if the user's profile and the text of the item shares word in common. The disadvantage of this system is its deficiency in exploring new items or topics beside's the items similar in the user's profile.

3.3 Diversity

In music recommender system, it is easy for a collaborative recommendation system to say which song has the highest rating, but these systems do not say anything about the range of songs a particular user might want to listen to [22]. Users do not want to listen to the highest rated song over and over again. Instead there are various a methods to broaden the playlist—increasing the diversity of the results, exposing the user to new music, and hopefully increasing customer satisfaction [21].
In recommender system while the *k* best recommendations are very similar to the target query, they are very similar to each other [4].Thus on Amazon.com, many recommendations seems to be similar with respect to the content [5]. For example, customers that have purchased Hermann Hesse's prose may happen to obtain recommendation lists where all top-5 entries contain books by that respective author only. Considering pure accuracy, all these recommendations appear excellent since the active user clearly appreciate books written by Hesse. But,

assuming that the active user has several interests other than Hesse, the recommended set of items appears poor, owing to the lack of diversity [5].Improving the diversity characteristics of a fixed-size recommendation list means sacrificing similarity, but the goal is to develop a strategy that optimizes this similarity-diversity trade-off, delivering recommendation sets that are diverse without compromising their similarity to the target query [4].

4. System Description

The general overview of our system is shown in figure 1.

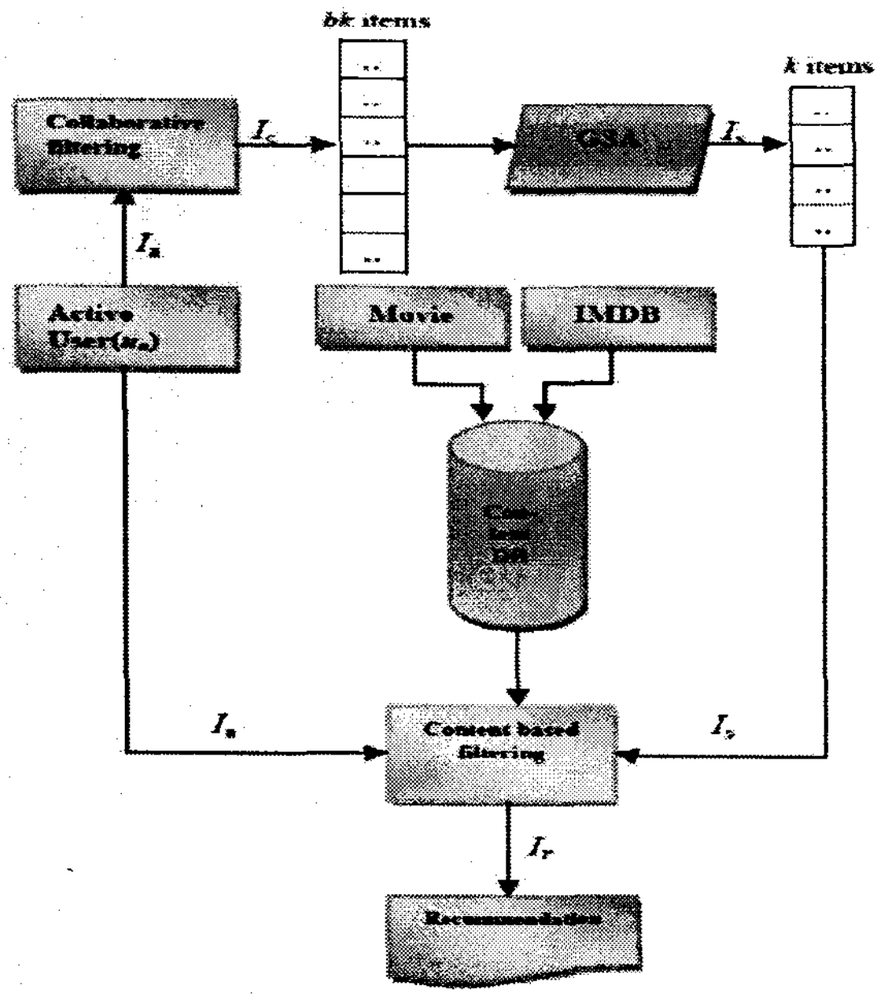

Figure 1: System Overview

The content database contains the content information about the movies, particularly about actors, directors and genre. Collaborative filtering takes active user's rating as input and calculates the similarity with various other users. It then gives *bk* items together with their predicted ratings based on the suggestion of the neighbors i.e. the items the neighbors liked the most. Greedy selection algorithm (GSA) is used to select *k* items among the *bk* items recommended by collaborative filtering. It uses item dissimilarity as a metric to select these items. Among the *bk* items, it takes one item as an active item with highest predicted rating and calculates dissimilarity with the remaining *bk*-1 items. The item with highest score will now be an active item and is removed from the remaining item. This process is repeated until we get *k* dissimilar item. The content-based filtering takes the active user's rating and the *k* selected items rating from greedy selection process (GSA) as input. For every rated input to the process, it queries the content databases for relevant information and follows a simple scoring mechanism in the content attributes. Now prediction for a new item is made using content-based filtering in the active user's rating and pseudo user rating we get from the selection process. It can briefly be explained as follows.

Let, there be a set of m users $U= \{u_1, u_2, u_3 \ldots\ldots u_m\}$ and n items $I = \{i_1, i_2, i_3\ldots\ldots i_n\}$. Let I_a be a set of items the active user u_a has rated. Now, from collaborative filtering, we get I_c i.e. a list of bk items together with their predicted ratings, the active user will like the most such that ,$I_c \cap I_a = \phi$ and $I_c \subset I$. The Greedy Selection Algorithm (GSA) with items dissimilarity as cost function selects k most dissimilar items i.e I_s from bk similar items I_c i.e. $I_s \subset I_c$. The ratings for these items are taken from the previous. Since, $I_s \subset I_c, I_c \cap I_a = \phi$ and $I_c \subset I$, therefore $I_s \subset I$ and $I_s \cap I_a = \phi$. This I_s together with the active user's rating I_a is fed into content-based filtering and prediction are made for new items based on their content attributes.

The following sections describe our implementation of collaborative filtering; greedy selection algorithm and content-based filtering followed by experimental results to support our idea.

4.1 Collaborative Filtering

We implemented a pure collaborative filtering algorithm that uses a neighborhood-based algorithm [3]. In this method, subsets of users are chosen based on the similarity to the active user, and a weighted combination of their ratings is used to produce predictions for the active user. The algorithm can be summarized as follows.

1. Calculate the similarity between active user and each other user. Similarity is computed using Pearson Correlation between their rating vectors.

$$sim(a,u) = \frac{\sum_{i \in I}(R_{a,i} - \overline{R_a})(R_{u,i} - \overline{R_u})}{\sqrt{\sum_{i \in I}(R_{a,i} - \overline{R_a})^2}\sqrt{\sum_{i \in I}(R_{u,i} - \overline{R_u})^2}} \quad (1)$$

Here, I denote a set of items, users a and u both rated, $R_{a,i}$ is the rating given to item i by user a; $\overline{R_a}$ is the mean rating given by user a

2. Find the *nearest neighbors* of the active user i.e. Select n users that have the highest similarity with the active user. For our experiments we used a neighborhood size of 40.

3. For each item i that has not been rated by active user u_a but has been rated by at least one of the neighbors, u_a's rating for i is predicted, $P_{a,i}$ essentially as a weighted average of the neighbor's rating for item i.

$$P_{a,i} = \overline{R_a} + \frac{\sum_{u=1}^{n}(R_{u,i} - \overline{R_u})sim(a,u)}{\sum_{u=1}^{n}|sim(a,u)|} \quad (2)$$

Where $P_{a,i}$ is the prediction for the active user a for item i; $sim(a,u)$ is the similarity between users a and u; and n is the number of users in the neighborhood.

4. These items are then sorted in descending order of $P_{a,i}$;

5. Top bk items along with their predicted ratings are recommended as output

4.2 Greedy selection Algorithm (GSA)

In our system, Greedy selection algorithm works as selecting dissimilar items among the similar ones recommended by collaborative filtering. Since the items recommend by collaborative filtering are more similar with each other [4][5], these items when fed into content-based system will not perform well as ideally expected. As the most prominent drawback of content-based system is the difficulty in exploring new items outside the usual choice, feeding similar items into it will again stop it from exploring extra items.

Our intension of including this metric is to find the dissimilar items among the similar ones recommended by the former, irrespective of their predicted ratings. These items (dissimilar with each other but similar with respect to the target query) when fed into the content-based system will broader the path for recommending new items. For example, we get five movie recommendations A, B, C, D, and E from the collaborative part which is similar with each other. When these items are fed into content-based system, the content-based system will not include new movies as ideally expected. But, if those five movies are dissimilar with each other, then the content-based part will have much higher possibility than the previous one to recommend newly released, unrated movie to the active user.

In order to do so, we implemented Greedy selection algorithm from [4]. In [4] there are *bk* items that are most similar to the query, but here we modified it according to our requirements, so here, these are *bk* items from collaborative filtering with highest prediction values $P_{a,i}$.

We implemented GSA to select items among the recommended items. Our intuition behind this approach is improving diversity among the items without sacrificing similarity with the target user.

Greedy Selection Algorithm

R = stack to place k items
R_T = item at the top of the stack
1. Select one item form *bk* items with highest predicted ratings and put it in stack
2. *Candidates* ← *bk*-1 items recommended by collaborative filtering
3. $R = \{R_T\}$
4. **for** j← 1 to k-1 **do**
   best← the i ∈ *Candidates* for which $Div(R_T,i)$ is highest
   Insert best into R
   remove best from *Candidates*
end for
return R

Figure 2: Greedy Selection Algorithm

Here, *Div (i, j)* gives the dissimilarity between the items *i* and *j* and is calculated from the user item rating matrix as,

$$Div(i,j) = 1 - sim(i,j)$$

Where,

sim (i, j) is the similarity between items i and j and is calculated by using Pearson correlation between items from the user item rating matrix and is given by,

$$sim(i,j) = \frac{\sum_{u \in U}(R_{u,i} - \overline{R_i})(R_{u,j} - \overline{R_j})}{\sqrt{\sum_{u \in U}(R_{u,i} - \overline{R_i})^2}\sqrt{\sum_{u \in U}(R_{u,j} - \overline{R_j})^2}} \quad (3)$$

Here, U denotes a set of users who both rated i and j. $R_{u,i}$ denote the rating of user u on item i. $\overline{R_i}$ is the average rating of i-th item. We computed the Pearson correlation between the item columns because it has advantage of being sensitive to actual ratings.

Since, the similarity, *sim (i, j)* between two items ranges from -1 to 1, the dissimilarity, *Div (i, j)*, ranges from 0 to 2.Hence, higher the score, the more dissimilar are the items with each other.

The intuition behind our approach is that the community of users who have rated item *i* have a certain set of tastes. The more the membership of the community who rated item *i* differs from the membership of the community who rated item *j*, the more likely *i* and *j* satisfy different tastes and are different kinds of items

Table2: Dissimilarity calculation

	ItemA	*ItemC*
User1	5	5
User2	1	1
User3	2	2

Correlation between *ItemA* and *ItemB* is 1
i.e. *sim(ItemA,ItemB)*=1
Div(ItemA,ItemB)=1-1=0

	ItemA	*ItemC*
User1	5	1
User2	1	3
User3	2	4

Correlation between *ItemA* and *ItemC* is-0.75
i.e. *sim(ItemA,ItemC)*=-0.75
Div(ItemA,ItemB)=1-(-0.75) =1.75

From table2, we can see that *ItemA* and *ItemB* seem to be more similar. On the other hand, *ItemA* and *ItemC* are more dissimilar compared to *ItemB*. In dissimilarity calculation, we used Pearson correlation between the item columns because it has advantage of being sensitive to actual rating.

From the above algorithm, we can see that once the execution is complete we get *k* dissimilar elements in R which is our goal. Since we now find the most dissimilar items among the similar ones we take their respective predicted rating of the neighbors from previous i.e. collaborative filtering and feed it into the content-based system.

4.3 Content Based Filtering

Different methods have been implemented in content-based filtering, the most famous being the bag-of- words naïve Bayesian classifier [12][13]extended to handle a vector of bags of words which is best suited for systems with many content attributes . However, in our system we only used three content attributes mainly actors, director and genre. Since the main objective of our research is enhancing recommendation by extending prediction based on collaborative filtering into content-based system and further we want our system for prediction on items rather than classify in various class labels, we implemented a simple

content-based filtering approach from [10]. This method uses information about each movie with content-based rating as input to the process and uses a simple scoring mechanism to find other similar movies.
Once we obtain a diverse set of movies along with the predicted ratings i.e. similar with respect to active user but dissimilar with respect to each other, it is fed into the content-based system together with the input i.e. active user's rating.
The recommendation process may be summarized as below.
For every rated movie input to the process
1. We find the information about it in the movie content database (actors, directors, and genre).
2. Add the movie rating to the score of each movie attributes .i.e. actors, directors and genre. For example, if the rating to a movie is 4, then each actor, director and genre associated with the movie will have 4 added to his score.
3. Calculate the average score of each actor, director and genre. This score gives the user's likeliness towards each element i.e. actor, director and genre.
4. For each actor, we add the average score to current score for each movie it is associated with.
5. Similar process is done for each director and genre.
6. Lastly, we calculate the average score i.e. predicted score for each movie.

5. Experimental Evaluation

In this section, we describe the experimental methodology and metrics we use to compare with different algorithms and present the results of our experiments.

5.1. Dataset

To test the system's performance, we used *MovieLens* data set containing 100,000movie ratings by 943 users in 1682 movies. The University of Minnesota's Grouplens Research centre makes this dataset publicly available. User ratings ranges on a scale of one star (poor) and five stars (excellent). Each user has rated at least 20 movies and each movie has been rated at least once. The content information for the movies was used from the Internet Movie Database (www.imdb.com). Three content attributes used were -actors, directors and genre (available in dataset) as the information about each movie

5.2 Methodology

We compare our system's performance with naïve hybrid approach and content-based filtering. Naïve hybrid approach takes the average of the predicted ratings generated by collaborative filtering and content-based filtering. The value of b and k for our system is kept as 2 and 40 respectively.

For evaluating the quality of prediction we divided the data set into two groups resulting 80% of the data as training set and 20% of the data as test set. Further, 10 ratings of the test data were withheld and the remaining were used to train the system. Predictions were computed for the withheld items and the quality of predictions were measured by comparing it with the actual ratings.

5.3 Metrics

In order to measure the accuracy of the predictions, *mean absolute error* (MAE)[12][20] metric-defined as the absolute difference between the predicted and actual ratings is used. The mean absolute error is defined as,

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

Where, p_i is the predicted rating and r_i is the actual rating. N is the number of items for which prediction is calculated.
Since, a recommender system might be highly efficient but give prediction for a small number of items, so in order to indicate the number of movies our system can produce prediction we used coverage metrics [20]. We used Prediction coverage that shows how many of the films removed from the user's rating set in our experiment to test for unrated (new) films could be predicted by each recommendation technique.

5.4 Experimental Result

i) Comparing MAE with other systems

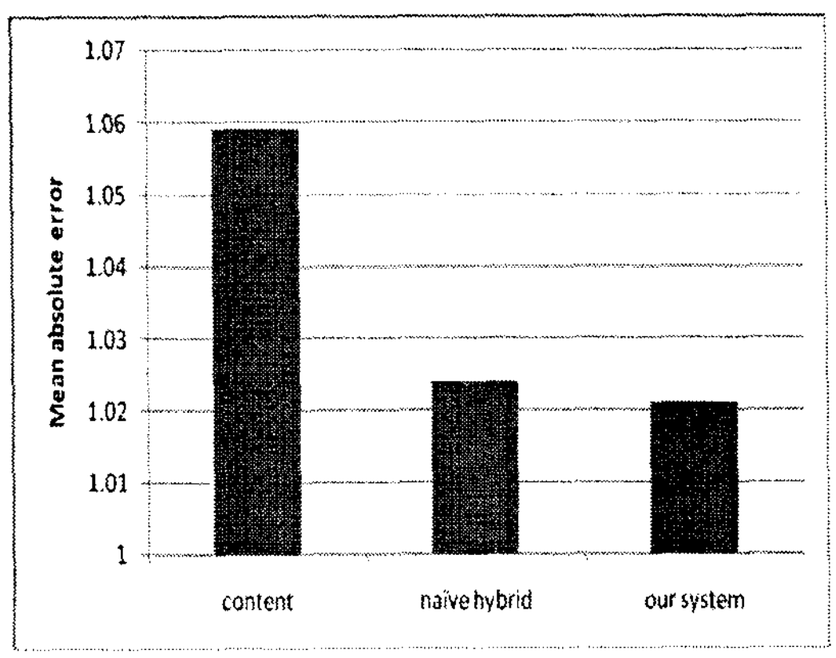

Figure3: Comparing MAE with other systems

From the above graph we can see that, our system performs better than content-based filtering and naïve hybrid approach by 3.58 % and 0.29% respectively. This is due to the fact that our system has enough information and is able to deduce a better relationship among the attributes than the content-based system.

ii) Comparing MAE of new movie with other systems
In order to compare the MAE for new movie we deleted the all the ratings of the withheld items from the rating table so as to treat it as new items and attempted to generate the prediction of them from the test items. Under this condition, pure content-based system is unaffected, since its prediction is based solely on the content attributes. The experimental results are as shown.

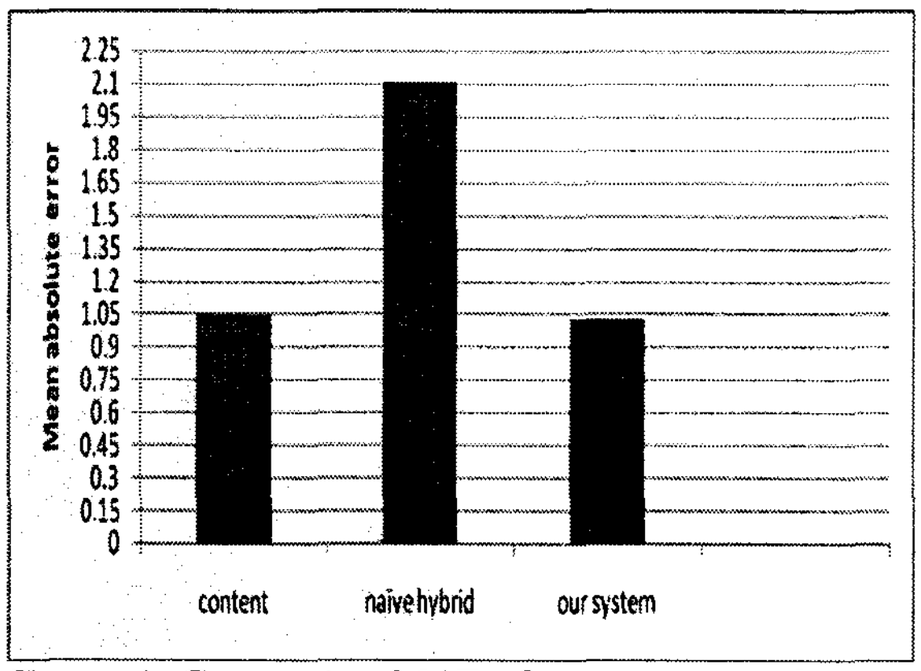

Figure4: Comparing MAE of new movie with other systems

Since naïve hybrid system takes the average of the ratings generated by collaborative and content-based filtering, in this case, as collaborative filtering can't generate any ratings, so we assumed it to be zero. In other word, for new items, its predicted ratings of naïve hybrid approach will be half of the content-based system.
In this condition, we can see that our system outperforms content-based filtering by 2.92%.
From Figure3 and Figure4 we can conclude that our system performs better than naïve hybrid and content-based approach under both the conditions. Though under normal condition i.e. when the system has enough ratings, our system outperforms naïve hybrid approach with a small margin, but under cold-start condition we can see that our system seems more effective compared to others.

iii) Comparing coverage with other systems

Table3: comparing coverage with other systems

Technique	Coverage
Our system	95.04
Naïve hybrid	93.81
Content-based only	79.86

iv) Comparing coverage for new movies with other systems: -
For this, we deleted all the ratings of the item in the rating matrix and asked the system to generated prediction. The results are as shown.

Table4: Comparing coverage of new movie with other systems

Technique	Coverage
Our system	94.6
Naïve hybrid	93.01
Content-based only	79.86

From Table3 and Table4 we can see that the coverage of our system is high. This is due to the fact that we take into account the active user's rating and additional diverse items which expand the coverage of the system. The coverage of content-based filtering is low because it has limited scope to deduce the relationship between content attributes. The coverage of naïve hybrid system is better than content-based system because it takes the average of collaborative and content-based system.

6. Conclusion

Recommender systems help to overcome information overload by providing personalized suggestions based on a history of a user's like and dislike. Collaborative filtering system is one of the recommender system technologies that have achieved widespread success. It has higher prediction accuracy compared to the content-based system, but possesses some serious limitations such as cold-start problem being one. Content-based filtering, on the other hand addresses this issue, but has problem of its own.
Combining components from both methods, we adopted a new approach by incorporating diverse collaborative prediction into content-based filtering. The main goal of this system is successful recommendation of items under both conditions i.e. normal and cold-start condition. Our approach addresses this problem efficiently by boosting the performance of content-based filtering. We incorporated greedy selection algorithm with item dissimilarity as cost function which selects dissimilar items among the similar items recommended by collaborative filtering. Our intuition behind this approach is improving diversity among the items without sacrificing similarity with the target user.
Experimental results on *movielens* dataset show that our system performs better than content-based filtering and naïve hybrid approach. This is due to the

fact that our system has enough information and is able to deduce a better relationship among the attributes than the content-based system. Since the coverage of our system is high, we can deduce that it is least prone to cold start problems because it can at least generate predictions to user.

In the future, we plan to extend this work and implement in various other domains. Further, as the performance of content-based filtering depends on the number of items the active user has rated, we plan to build a model such that the additional dissimilar items that we feed in content-based part is a function of the number of items the active user has rated. Doing so, we believe that the performance of the system will be even better than it is now.

6. References

[1] Balabanovic, M., and Shoham, Y. 1997. Fab: Contentbased, collaborative recommendation. *Communications of the Association of Computing Machinery* 40(3):66–72

[2] Pazzani, M. J. 1999. A framework for collaborative,content-based and demographic filtering. *Artificial Intelligence Review* 13(5-6):393–408

[3] Herlocker, J.; Konstan, J.; Borchers, A.; and Riedl, J. 1999.An algorithmic framework for performing collaborative filtering. In *SIGIR '99: Proceedings of the 22nd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, 230–237

[4] Barry Smyth, Paul McClave; Similarity vs Diversity.ICCBR2001;347-361

[5] Cai-Nicolas Ziegler,Sean M McNee, Joseph A. Konstan Georg Lausen; Improving recommendation lists through topic diversification. WWW2005; 22-32

[6] Robin D. Burke: Hybrid Systems for Personalized Recommendations. ITWP2003: 133-152

[7] Raymond J. Mooney,Loriene Roy: Content-Based Book Recommending Using Learning for Text Categorization CoRRcsDL/9902011: (1999)

[8] Badrul M. Sarwar, Joseph A. Konstan , et al ,Using Filtering Agents to Improve Prediction Quality in. the GroupLens Research Collaborative Filtering System.CSCW1998: 345-354

[9] Derek G. Bridge, John Paul Kelly : Ways of Computing Diverse Collaborative Recommendations. AH2006: 41-50

[10] James Salter, Nick Antonopoulos: CinemaScreen Recommender Agent: Combining Collaborative and Content-Based Filtering.IEEE Intelligent Systems21 (1): 35-41 (2006)

[11] George Karypis: Evaluation of Item-Based Top-N Recommendation Algorithms. CIKM2001; 247-254

[12] Prem Melville, Raymond J Mooney, Ramadas Nagarajan;Content-Boosted Collaborative Filtering for Improved Recommendations. AAAI/IAAI2002; 187-192

[13] Mitchell, T. 1997. *Machine Learning*. New York, NY:McGraw-Hill.

[14] Cotter, P., and Smyth, B. 2000. PTV: Intelligent personalized TV guides. In *Twelfth Conference on Innovative Applications of Artificial Intelligence*, 957–964.

[15] Claypool, M.; Gokhale, A.; and Miranda, T. 1999. Combining content-based and collaborative filters in an onlinenewspaper. In *Proceedings of the SIGIR-99 Workshop on Recommender Systems: Algorithms and Evaluation*

[16] Basu, C.; Hirsh, H.; and Cohen, W. 1998. Recommendation as classification: Using social and content-based information in recommendation. In *Proceedings of the Fifteenth National Conference on Artificial Intelligence (AAAI-98)*,714–720

[17] Maria Fasli ,Agent Technology for E-commerce,2007

[18] Paolo Massa, Pa olo Avesani: Trust-Aware Collaborative Filtering for Recommender Systems. CoopIS/DOA/ODBASE (1) 2004: 492-508

[19] Sheth, B., Maes, P.: Evolving agents for personalized information filtering. In: Proceedings of the Ninth Conference on AI for Applications. IEEE Computer Society Press (1993)

[20] Jonathan L. Herlocker, Joseph A. Konstan, Loren G. Terveen, John Riedl: Evaluating collaborative filtering recommender systems. ACM Trans. Inf. Syst. 22(1): 5-53 (2004)

[21] John C. Platt, Christopher J. C. Burges, Steven Swenson, Christopher Weare, Alice Zheng. Learning a Gaussian Process Prior for Automatically Generating Music Playlists.Advances in Neural Information Processing Systems 14, pp.1425–1432, 2002.

[22] Malcolm Slaney, William White: Measuring Playlist Diversity for Recommendation Systems, *MCMM'06,* October 27, 2006,