Content-Based Movie Recommendation Using Different Feature Sets

Mahiye Uluyagmur, Zehra Cataltepe and Esengul Tayfur

***Abstract*—Movie recommendation systems aim to recommend movies that users may be interested in. In this paper, we introduce a content-based movie recommendation system which can use different feature sets, namely, actor features, director features, genre features and keyword features. For each user, we assign a weight to each feature in a feature set based on the particular user's past behavior. We produce user's implicit rating for a movie based on the duration of the movie that the user viewed. In order to predict a rating for user and a movie, using a particular feature set, we merge the user specific weights of movie's features. We also produce ratings using all feature sets. We evaluate each recommendation method based on precision, recall and F-measure on ten movie recommendations.**

***Index Terms*— Content-based Movie Recommendation, Feature Weight Calculation, Recommender Systems, Implicit Rating**

I. Introduction

Digital TV broadcasting brought a huge increase in the number of TV channels and the amount of content they provide. It is very hard for users to find and watch the content they actually like among these many options. Users have to zap around the channels or follow the program guides to find the contents that they are likely to prefer. But the programming guide is a very long list, and zapping takes time and it may also not be possible to zap over so many different contents to have an idea on them. So, the suitable content selection for each user becomes an important problem [1]-[2]. Recommendation systems have emerged as a solution to this problem. If explicit ratings (e.g. like/dislike, a rating between 0 and 5) are available, then a recommendation system may use these ratings. On the other hand, for many systems, users do not want to provide such an explicit feedback, therefore implicit ratings need to be produced based on the user viewing history. Another taxonomy of recommendation systems is based on whether content of each movie, or viewing behavior of other users are taken into account. Collaborative filtering methods rely on a user-item matrix which shows whether a user liked an item or not [3]. Usually, the collaborative filtering methods ask the users to give explicit ratings about the contents they watched previously. So, the ratings are obtained explicitly by the system [4]. If ratings are not available, then they may need to be generated implicitly, based on user behavior. It should be emphasized that using direct ratings or calculated implicit ratings may produce different results.

Sparsity of the user-item matrix is a major problem in collaborative filtering [5]. There are usually a large number movies in the system and each user gives ratings to a small number of movies. Since the number of commonly rated movies by two users is mostly zero, it becomes very difficult to find users who are nearby. Moreover, if a movie is not rated by any users at all, this movie cannot be recommended (cold-start problem) [5].

Content-based recommendation uses movie information and users' viewing profile. Music Genome Project is an example music recommendation system [6] which uses a content-based recommendation method. In a content-based method each user is uniquely characterized and the user's interest is not matched some other user as in the collaborative methods [7]. The ability to show content features that causes an item to be recommended also gives users' confidence about the recommendation system and insight into their own preferences [7].

In this paper, we introduce a content-based movie recommendation method. First of all, we convert the viewing history of a user for each movie to an implicit rating. We consider feature sets, such as actor, director, keyword etc. that describe a movie. For each feature in a feature set, based on the user's past viewed movies and the user's rating for each movie, we compute a feature weight. Each feature weight is calculated separately for each user. If a user watched a movie completely or much of it, then, the features extracted from this movie are important, and their weights will be assigned accordingly. When a movie needs to be rated for a user, based on the features of the movie, we produce a different rating for each feature set and compare them. We also produce combined ratings which take into account all different feature sets.

II. Related Work

With the increase in the amount of items users can buy/watch and the ability to keep the history of users' consumed items, recommendation systems have become both necessary and available.

A recommendation method for a Japanese video service provider has been proposed in [8]. They used the actor and keyword information of the users films. They also considered the time of the day the users watch TV. They used the ratio of the number of times a user watched a movie with a certain feature (such as actor, keyword) to the number of times the feature is observed in all the movies. This ratio is calculated for all actor and keyword features. Then for each movie, sum of the movie's ratio features is calculated. They used recall, precision and F-Measure as evaluation measures. Different from our evaluation method,

Manuscript received July 6, 2012; revised August 10, 2012. This work was supported in part by the Turkish Ministry of Science, Industry and Technology, SanTez Program, Project number 00966.STZ.2011-2 and in part by Digiturk.

Authors M. Uluyagmur and Z. Cataltepe are with Istanbul Technical University. Author Esengul Tayfur is with Digiturk. Corresponding Author: Z. Cataltepe, Istanbul Technical University, Computer Engineering Department, cataltepe@itu.edu.tr, +90-212-285-3551.

they measured their performance by getting feedback from the user right after making a recommendation.

An approach to support context-aware recommendation for personalized digital TV has been proposed in [1]. In this work they used contextual user profile, which consists of user personal data profile, user contextual information and the genre of the TV program. They got this information from the users by asking them directly. The distinction of our system is that we do not ask any questions to the user to get their demographic information or preferences. Since in our system each customer may consist of a number of family members and the information provided may not always be reliable, we do not use demographic information of users at all. Instead of asking preference questions to the user, we extract this information from the user's watching history.

FIT system recommends TV programs to family members [9]. They constructed a user profile by asking each household about their program genre preferences. The user profile also contained the times of the day they watch television. In the recommendation phase, firstly FIT system guesses which household turn on the television by using the time of the day information. If the guess is wrong then the system may make the wrong recommendations for the household.

TiVo television show collaborative recommendation system uses item-item form of the collaborative filtering [2]. Process starts by a user giving a rating to a movie. There are two types of ratings: explicit and implicit. For explicit feedback, the user presses the remote control button according to how much s/he loves the movie.

III. Methodology

A. Conversion of Movie Viewing Duration to Rating

In our system, users do not rate movies explicitly, so we calculate implicit ratings by using the viewing durations. Assume that user *u* watches the movie *i* for *t(u,i)* minutes during the year and t_i is the total duration of the movie *i*. We define the normalized viewing duration (the implicit rating) of the user *u* for movie *i* as:

$$r(u,i) = \frac{t(u,i)}{t_i} \qquad (1)$$

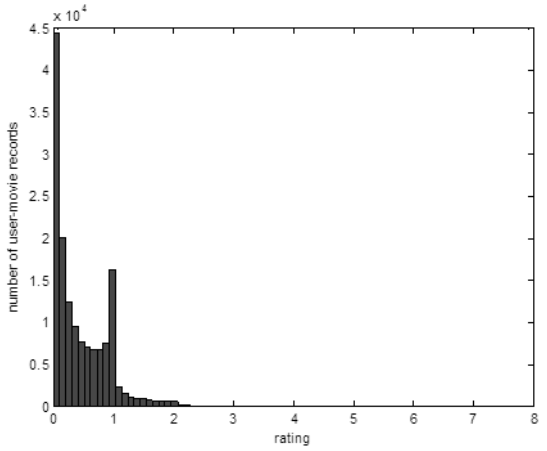


Fig. 1. Distribution of the normalized viewing durations of all users.

This formulation is similar to that of [1]. The difference in our formulation is that $r(u,i)$ can be greater than 1 if a user has watched a content more than once. Also, we work with continuous ratings, we do not use a threshold to determine relevant items.

As it is seen in Fig. 1, $r(u,i)$ values are between 0 and 7.2. Most contents are watched once, but there are also contents which are watched more than once.

B. Feature Based Weight Calculation Method

The aim of the movie recommendation system is to find the contents that the user may actually want to watch. We use the features of the movies the user viewed in the past and the implicit ratings for these movies, to compute a weight for each user and feature. We use the feature sets of type actor, genre, director. Training contents have 4716 actors, 1927 directors, 34 genres. The weight of each feature is assigned according to the implicit rating of user for all training data contents including that feature. The actor feature set may contain features like Brad Pitt, Harrison Ford, Engin Günaydın; the genre feature set may contain dram, action, comedy; the director feature set may contain Woody Allen, James Cameron etc.

In order to determine the weight of each feature for user *u*, we use the training rating data.

TABLE I
ITEM-FEATURE MATRIX FOR USER *U*

user u	j_0	j_1	j_2	j_3	j_4	Puan
i_0	1	1	0	0	0	0.5
i_1	0	1	0	0	0	0.3
i_2	1	1	1	0	0	0.9
i_3	1	0	0	1	0	0.7
i_4	0	0	0	1	0	0.2
i_5	1	0	0	1	0	1.0
i_6	1	0	0	1	0	0.44
i_7	0	1	0	0	0	0.67
i_8	1	0	0	1	0	0.2
$w_k(u,j)$	0.42	0.26	0.1	0.26	0	-

Let user *u* watch items $i_0,\ldots,i_8$, which have features $j_0,\ldots,j_3\ldots$ In Table I, we show the features for the feature set *k* for user *u*. Note that $j_0,\ldots,j_3\ldots$ contain the features that appear on items all users watched in the training set. Rating column shows the rating of user *u* for movies $i_0,\ldots i_8$.

The weight of feature j in feature set *k* for user *u* is calculated as:

$$w_k(u,j) = \frac{1}{\left|I_u^{train}\right|} \sum_{i \in I_u^{train}} x_{k,u}(i,j) r(u,i) \qquad (2)$$

In equation (2), *k* represents the type of the feature set, $k \in$ {actor, genre, director}. $r(u,i) \in \Re$ is the implicit rating of user *u* for item *i* and $x_{k,u}(i,j) \in \{0,1\}$ *jth* feature of item *i*. I_u^{train} is the set of movies watched by user *u* in the

ISBN: 978-988-19251-6-9
ISSN: 2078-0958 (Print); ISSN: 2078-0966 (Online)

training period. If movie *i* has feature j then $x_{k,u}(i, j)$ will be 1 and user rating for movie *i* will contribute to the sum.

C. Rating Prediction Method

After calculating the weight of each feature for each user, we use it to predict the recommendation ratings of contents. We calculate the rating for each feature set separately using the weights $w_k(u, j)$ obtained by using the training data.

We use two methods to generate ratings. As it is seen in (3) the first one is to sum up the feature weights of the contents. Equation (4) represents the second one and it is to normalize this sum, by dividing it to the number of its features.

$$r_k(u,i) = \sum_{j \in D_{k,i}} w_k(u, j) \tag{3}$$

$$r'_k(u,i) = \frac{1}{\left|D_{k,i}\right|} \sum_{j \in D_{k,i}} w_k(u, j) \tag{4}$$

$r_k(u,i)$ represents the rating that user *u* gives the movie *i* according to the *k* feature set. $r'_k(u,i)$ is the rating form of the normalization of the feature weights summation with $D_{k,i}$. $D_{k,i}$ is the features that appear in movie *i* from feature set *k*.

D. Combining the Ratings of Different Feature Sets

Each feature set may contain a different number of items; therefore $r_k(u,i)$ values may be in different ranges.

In order to determine the total effect of all the features, the rating of each feature which is calculated according to (3) needs to be normalized. In this work min-max normalization method is used it is conducted as follows:

$$r_k^N(u,i) = (r_k(u,i) - mR_{u,k}) / (MR_{u,k} - mR_{u,k}) \tag{5}$$

$mR_{u,k}$ indicates the minimum predicted rating of the user *u* calculated according to feature set *k* in the training set and $MR_{u,k}$ indicates the maximum predicted rating of the user *u* calculated according to feature set *k* in the training set.

As in Equation (5), we also normalize users' actual ratings:

$$r^N(u,i) = (r(u,i) - mR_u) / (MR_u - mR_u) \tag{6}$$

mR_u indicates the minimum actual rating of the user *u* and MR_u indicates the maximum actual rating of the user *u*.

Ratings are brought together with the use of two different methods. Actor, genre, director, time zone, channel, keyword, release year normalized ratings are summed in the first method.

$$r_{\text{sum}}^N(u,i) = \sum_{k=1}^{K} r_k^N(u,i) \tag{7}$$

Equation (7) is used to generate ratings. In this equation, *K* is the number of the feature sets. $r_k^N(u,i)$ is the normalized rating. After the combined ratings are generated, the number of correct recommendations is evaluated.

Let $E_{u,k}$ represent the Mean Absolute Error (MAE) of ratings for user *u* according to feature set *k*. The MAE is calculated as the absolute value of the difference of the predicted and actual ratings for user *u* according to feature set *k* on the training set:

$$E_{u,k} = \frac{1}{\left|I_u^{train}\right|} \sum_{i=1}^{\left|I_u^{train}\right|} \left| r^N(u,i) - r_k^N(u,i) \right| \tag{8}$$

The ratings for feature sets which have less MAE should have larger weights in the combination. Therefore, the second rating combination method uses the MAE as follows:

$$r_{\text{expsum}}^N(u,i) = \sum_{k=1}^{K} r_k^N(u,i) * \exp(-E_{u,k}) \tag{9}$$

E. Evaluation of Recommendations

In order to evaluate the performance of our recommendation methods we used precision, recall and F-Measure metrics.

We use the topN hit counts to measure the accuracy of our recommendation system. We recommend movies to the user according to the predicted ratings as we generated before. We sort all the test movies according to their generated ratings and recommend the top N=10 movies to the user. After that we count the number of movies watched in the test set by the user out of top 10 recommendations and name this quantity as the #hitCounts.

Precision is the measure of the how many movies the system hits in the top 10 movies:

$$\text{Precision} = \frac{\#\text{hitCounts}}{\text{N}} \tag{10}$$

Recall is the ratio of the number of hits out of 10 recommendations and the size of I_u^{test} which is the number of movies that user *u* watched in the test set:

$$\text{Recall} = \frac{\#\text{hitCounts}}{\left|\text{I}_\text{u}^{\text{test}}\right|} \tag{11}$$

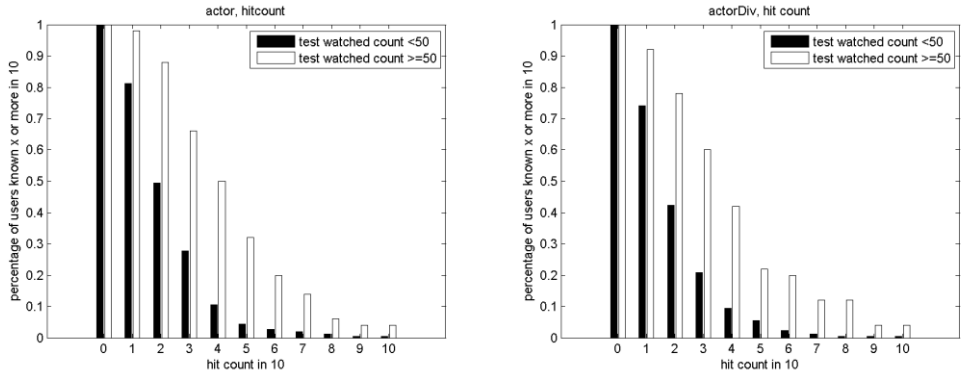
For example if the recommendation system hits 5 movies which have been watched out of the 10 recommended movies, the precision value will be 0.5. If the user watched 30 movies in the test set, then the recall value will be 0.16. F-measure uses both precision and recall as follows:

$$\text{F} - \text{measure} = 2 \cdot \frac{\text{Precision} \cdot \text{Recall}}{\text{Precision} + \text{Recall}} \tag{12}$$

IV. EXPERIMENTS

In our experiments, we used 13 months of log data, the first 12 months for training phase, the last 1 month for the test phase. There are 621 users and 3700 contents.

ISBN: 978-988-19251-6-9
ISSN: 2078-0958 (Print); ISSN: 2078-0966 (Online)



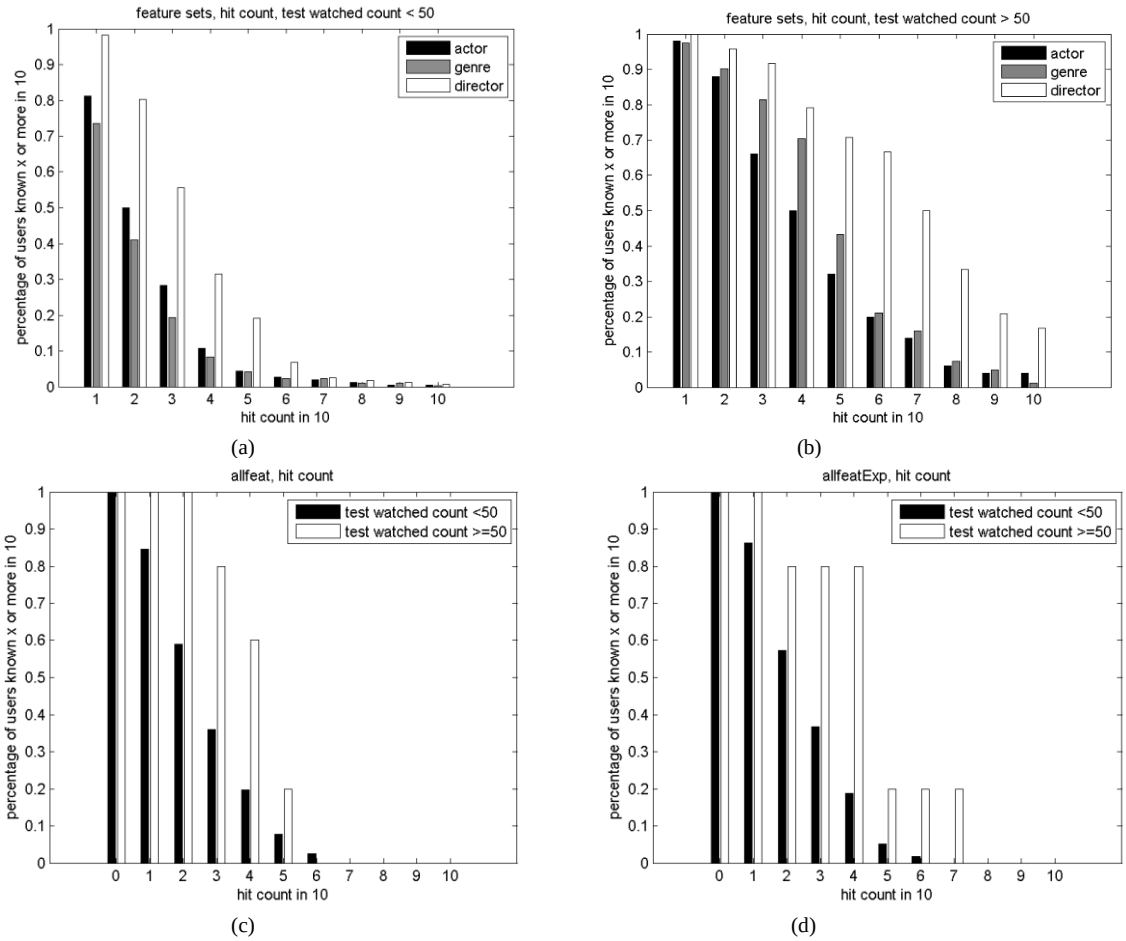
(a) (b)

Fig. 2a-2b. Percentage of the users for whom we recommend 10 contents using respectively (3) and (4) with actor feature set and calculate the hit counts.

The challenge of the system is that there are lots of contents available to watch, but users have watched very few of them and we try to catch their interest according to those small number of watched contents.

In Fig. 2a and 2b, we get results by using only the actor features (3) and (4) respectively. It shows the percentage of the recommendations which have at least given number of successful recommendations. In Fig. 2a, 98% of the users have watched at least one of the recommended contents and 88% of them have watched at least 2 recommended contents. On the other hand, in Fig. 2b these ratios are less. Therefore, we use (3) for the rest of our experiments.

The number of the contents available for the users is about thousands, but the number of contents they actually watch mostly changes from 1 to 40. In Fig. 2a and 2b, we can see that the success rate of the system for the users who watch more than 50 contents are better. As they watch more and more contents, we can get more information from their watching behaviors and predict better for future recommendations.



(a) (b)

(c) (d)

Fig. 3a-3b. Using actor, genre, director feature sets to calculate hit counts .
Fig. 3c-3d. Combining the all feature sets according to respectively (7) and (9).

ISBN: 978-988-19251-6-9
ISSN: 2078-0958 (Print); ISSN: 2078-0966 (Online)

In this work, we also analyze performance of ratings which combine different feature sets. Thanks to normalization (Equation 5), we can combine the ratings for each feature set given by the same user to the same item (7). In Fig. 3c and Fig. 3d we combine actor, genre, director, keyword, time zone, release year feature sets. Equation (7) results are shown in the Fig. 3c. For 60% of the user we make 4 or more successful recommendations out of 10 movies using all the feature sets when the number of user watched movies is more than 50 in the test set, (see Fig 3c). On the other hand this ratio decrease to 50% when using only actor feature set (see Fig. 3b). We could make the following inference: When users watch more than 50 contents, 4 or more successful recommendation ratio is high when using all features sets together. But 7, 8, 9 or more hit counts obtained when using feature sets separately.

The other combination of normalized rating method is given by (9). For each user, MAE is calculated in the training set. Equation 9 uses this MAE value. This rating calculation provides more accurate recommendation as it is seen in Fig. 3d.

Table II shows the average of the performance measures over all users in the test set. Precision, recall and F-Measure metrics are used. For all of these measures, higher values are better. Director gives the best recommendation performance, while combined recommendations have reduced performance. We are in the process of finding better combination methods.

TABLE II
EVALUATION METRIC RESULTS

	Prec	Recall	F-Measure
Actor_Hit10	0.175	0.078	0.108
Genre_Hit10	0.193	0.086	0.12
Director_Hit10	0.213	0.095	0.13
AllFeat_Hit10	0.136	0.06	0.084
AllFeatExp_Hit10	0.135	0.06	0.083

V. CONCLUSION

In this paper, we developed a content-based recommendation system that makes recommendations according to users' past watching behavior. We produce recommendations based on actor, genre, director feature sets separately and also a combination of actor, genre, director, keyword, time_zone, release_year feature sets. In the future, we plan to produce a hybrid recommendation system.

REFERENCES

[1] Santos da Silva, F., Alves, L. G. P., and Bressan, G. 2009. PersonalTVware: A proposal of architecture to support the context-aware personalized recommendation of TV programs. In *Proceedings of the 7th European Conference on Interactive TV and Video*.

[2] K. Ali and W. van Stam. TiVo: making show recommendations using distributed collaborative filtering architecture. In KDD '04: Proceedings of the tenth ACM SIGKDD international conference on Knowledge discovery and data mining, pages 394–401. ACM, 2004

[3] Wang, J., de Vries, A.P., Reinders, M.J.T., 2006: Unifying User-based and Item-based Collaborative Filtering Approaches by Similarity Fusion, SIGIR'06, August 6-11, 2006, Seattle, Washington, USA.

[4] Koren, Y., Bell, R., Volinsky, C., "Matrix Factorization Techniques for Recommender Systems", Computer Journal, IEEE Press, 42-49, 2009.

[5] P. Melville, R. J. Mooney, and R. Nagarajan. Content-boosted collaborative filtering. In Proceedings of the 2001 SIGIR Workshop on Recommender Systems, 2001.

[6] Westergren. T., The music genome project. http://www.pandora.com/, Accessed on 9th of July ,2011.

[7] Raymond J. Mooney , Loriene Roy, Content-based book recommending using learning for text categorization, Proceedings of the fifth ACM conference on Digital libraries, p.195-204, June 02-07, 2000, San Antonio, Texas, United States.

[8] K. Ikawa, T. Fukuhara, H. Fujii and H. Takeda: Evaluation of a TV Programs Recommendation using the EPG and Viewer's Log Data, in C. Peng, P. Vuorimaa, P. Naranen, C. Quico, G. Harboe and A. Lugmayr eds., Adjunct Proceedings EuroITV 2010, pp. 182–185, Tampre, Finland (2010), Tampere University of Technology.

[9] Goren-Bar, D., Glinansky, O.: FIT-recommending TV programs to family members. Computers & Graphics 28, 149-156 (2004).

ISBN: 978-988-19251-6-9
ISSN: 2078-0958 (Print); ISSN: 2078-0966 (Online)