Effect of Using Additional Movie Information in Netflix Prize Challenge

Babak Bostan
Department of Computing Science
University of Alberta
bostan@cs.ualberta.ca

Mojdeh Jalali-Heravi
Department of Computing Science
University of Alberta
jalalihe@cs.ualberta.ca

Amin Jorati
Department of Computing Science
University of Alberta
jorati@cs.ualberta.ca

Yavar Naddaf
Department of Computing Science
University of Alberta
naddaf@cs.ualberta.ca

Abstract

In this paper we investigate the effect of applying hybrid recommender systems on Netflix dataset, using additional movie information. We experiment with a number of methods to combine collaborative and content-based methods for movie rating prediction. We combine our prediction results with a number of canonical predictions generated by "Reel Ingenuity", and slightly improve their accuracy. This improvement indicates that hybrid methods have the potential to improve the current collaborative approaches to Netflix challenge.

1 Introduction

Netflix, an online movie rental company, is interested in improving its current movie recommendation system. An accurate recommender system will allow Netflix to suggest more relevant movies to its customers. When customers are offered movies closer to their taste, they are more likely to order them. This results in both higher profits and a more satisfied customer base for Netflix. In October 2006, Netflix released a large set of customer ratings, and offered a prize of one million dollars to anyone who can improve their current prediction accuracy by 10%. This has turned out to be a difficult challenge, and after more than a year, no team has yet met their required accuracy.

The current methods used in the Netflix challenge make their predictions solely based on the provided $(user, rating)$ training set [16, 19, 1, 6, 3, 2]. This is partly because additional information about movies is not easily available. Furthermore, it is not clear that additional information, even if it was available, can generate more accurate predictions.

In this paper, we investigate whether adding movie feature data, such as directors, actors, or genre, can increase the accuracy of the current prediction models. We develop a number of hybrid methods that make use of user ratings as well as the movie features. We combine our new predictions with a number of existing predictions. We compare the prediction accuracy of the new combined predictions with the previous predictions, and conclude that using additional data slightly increases the prediction accuracy.

The remaining of this paper is organized as follows: Section 2 provides an overview of recommender systems. Section 3 discusses some of the current approaches used in the Netflix challenge. In Section 4, we introduce the hybrid methods that we have applied on the Netflix dataset. Section 5 presents our experimental results. Section 6 contains our conclusion.

2 Recommender systems

Recommender systems suggest new items to users based on their activity history, such as previous purchases, items viewed, or ratings. Depending on how these suggestions are calculated, recommender systems are categorized into three types: content-based, collaborative, and hybrid [7].

2.1 Content-based recommender systems

A content-based recommender system bases its suggestions for a user solely on the history of that user. It suggests new items that are similar to the previous items that the user has liked before [8]. The similarity between two items is calculated based on the features of the items. For instance, when the items are movies, the similarity between two movies can be based on their release-year, genre, directors, or actors.

One shortcoming of content-based approaches is *over-specialization*, *i.e.,* the items suggested by the recommender system are too similar to the items that are highly rated by the user. Hence, the user does not receive a good exploration of other items that she may like [8, 7]. Another problem with content-based recommenders is *limited content analysis*, *i.e.,* the items that share the same set of features will be indistinguishable [7]. *User cold-start*, *i.e.,* poor recommendations for a new user, is yet another problem with content-based methods.

2.2 Collaborative recommender systems

A collaborative recommender system bases its suggestion on the opinion of other users, and how they have agreed with the current user on previous items [13]. In the case of a movie recommender system, the system will recommend movies that other users with similar taste in movies have rated highly. By using the opinion of other users, collaborative filtering avoids the problems of over-speicilization and limited content analysis in content-based approaches [8, 7]. However, it still suffers from *item cold-start*, that is, it performs poorly when few people have rated an item.

2.3 Hybrid recommender systems

Hybrid recommender systems attempt to avoid the drawbacks of content-based and collaborative methods by combining them. Hybrid systems use both the rating history of a user and the opinion of other users. Below is a list of four main categories of hybrid methods followed by an example for each category [7]:

1. *Combining separate recommenders:* The prediction of each recommender system can be combined with the predictions of other systems. The easiest way to combine collaborative and content-based recommender systems is to use a linear combination of their predictions.
2. *Adding content-based characteristics to collaborative models:* Several systems such as Fab [8] are basically traditional collaborative systems, but use user features to compute similarities between users.
3. *Adding collaborative characteristics to content-based methods:* One popular approach of this category is to create a collaborative view of user profiles; Soboroff and Nicholos [18] use *latent semantic indexing* to create such view. Dimensionality reduction is then performed on groups of feature profiles of users.
4. *Single unified approach:* The last approach is to use both content-based and collaborative characteristics in a single unified model. Basilico and Hofmann [9] suggest using features extracted from user-item pairs (u, x) rather than u and x individually. Feature maps $\Psi(u, x)$ are thus over the product of user and item spaces. These joint feature maps can be constructed by taking the tensor product of user and item features. Various kernels are then defined to capture concepts ranging from attributes to correlation between ratings.

3 Current approaches to the Netflix Prize challenge

The current approaches used in the Netflix challenge focus on collaborative methods. Bell and Kore, the members of the Bellkor team who won the Progress Prize of 2007, used a combination of

107 different predictions, based on neighborhood-base models, factorization, restricted Boltzmann machines, and asymmetric factor model ([12, 10, 11, 15, 17] as cited in [16]). Their justification for focusing on collaborative filtering strategies is that extracting external information for movies is not trivial [12]. The other teams currently at the top of the Netflix board, who report their approach, also employ collaborative methods [19, 1, 6, 3, 2].

It is not clear whether using additional information about movies, even if they were easily available, can be helpful for generating more accurate predictions. To investigate whether the additional data can be helpful, we have developed a number of hybrid methods, and study the effect of combining them with the current collaborative methods.

4 Hybrid recommender systems for the Netflix Prize challenge

This section describes a number of hybrid approaches that we have applied on the Netflix dataset. Based on the classification described in section 2, our methods can be categorized into: adding collaborative characteristics to content-based methods (section 4.1), and adding content-based characteristics to collaborative methods (section 4.2). In section 4.3 we explain how the predictions from different methods can be combined to make a final prediction.

4.1 Adding collaborative characteristics to a content-based method

Our approach to add collaborative characteristics to a content-based method is to subtract the average ratings of movies from the $(user, movie)$ ratings in the training set, before applying the content-based method.

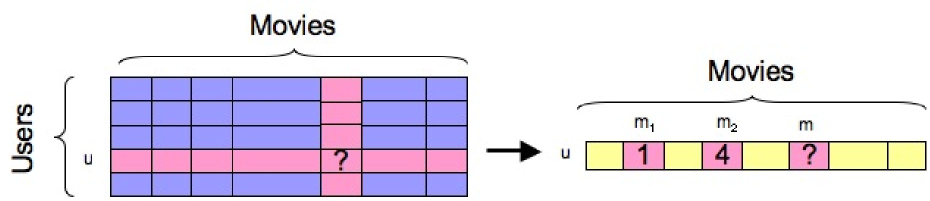

Figure 1: A content-based method: predicting the rating of user u for movie m

As illustrated in figure 1, to make a prediction for the rating of user u for movie m, a simple content-based method considers all the previous movies rated by this user, and calculates a weighted sum of their ratings based on how similar they are to m:

$$rating(u, m_i) = \sum_{m_j \in\, movies\, rated\, by\, u} rating(u, m_j) \cdot sim_{movie}(m_i, m_j)$$

The similarity between two movies is defined as a weighted sum of their similarities based on a number of features:

$$\begin{aligned} sim_{movie}(m_i, m_j) \quad = \quad & w_{director} \cdot sim_{director}(m_i, m_j) \\ & + w_{actors} \cdot sim_{actors}(m_i, m_j) \\ & + w_{genre} \cdot sim_{genre}(m_i, m_j) \\ & + w_{year} \cdot sim_{year}(m_i, m_j) \end{aligned}$$

The feature similarity functions can be very simple. For instance, if m_i and m_j have the same director, $sim_{director}(m_i, m_j) = 1$, and otherwise $sim_{director}(m_i, m_j) = 0$. Likewise, the similarity based on actors depends on the number of actors that the two movie share. The weights are calculated by doing a grid or hill-climbing search on the training set.

To convert this to a hybrid method, instead of doing a weighted sum over the rating of other movies, we perform a weighted sum over how much the user rating is different from the average rating of each movie:

$$rating(u, m_i) = \overline{m_i} + \frac{\sum_{m_j \in \, movies\, rated\, by\, u} (rating(u, m_j) - \overline{m_j}) \cdot sim_{movie}(m_i, m_j)}{\sum_{m_j \in \, movies\, rated\, by\, u} \cdot sim_{movie}(m_i, m_j)}$$

Where $\overline{m_i}$ stands for average rating of users for the ith movie. Subtracting the averages from the user ratings will help to better capture specific user interests and remove the effect of movie quality.

4.2 Adding content-based characteristics to collaborative methods

In this section we introduce another approach to combine content-based and collaborative recommender systems. This method either partitions or transforms the entire training set based on a feature of movies, and applies a regular collaborative system. The following subsections explain each method in detail.

4.2.1 Hybrid method by partitioning the training set

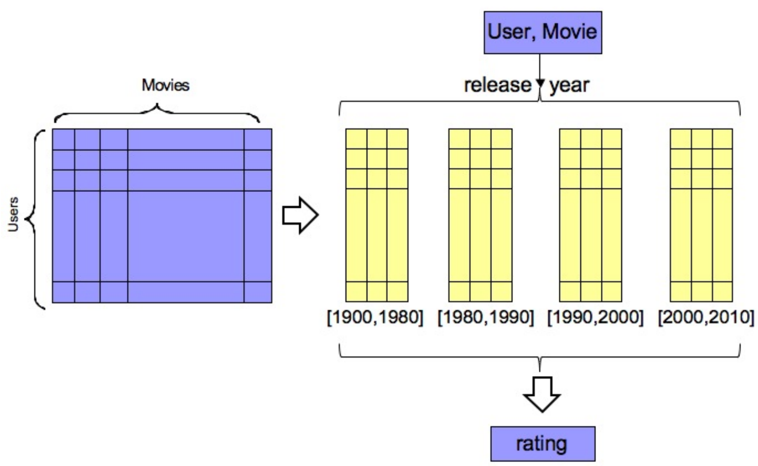

Figure 2: Partitioning the training set based on release year and then applying a collaborative method

For features that have small number of categories, such as genre or release year, we partition the dataset into a number of subsets. A collaborative method, in our case a simple Movie K-Nearest-Neighbor algorithm, is then applied on the partitioned dataset. Figure 2 illustrates this process using the release year feature. The entire training set is first partitioned into a number of subsets, based on the release year of the movies. Each subset contains the rating of all users for movies that were released in a particular decade. To predict a rating for a new $(user, movie)$, the K-NN algorithm is applied on the subset of the dataset that corresponds to the release year of the movie.

4.2.2 Hybrid method by transforming the training set

When the number of categories of a feature is large, and it is not trivial to group them into smaller number of buckets, the method introduced in 4.2.1 cannot be applied. In these cases, instead of partitioning the training-set, we transform the user ratings into a new training-set. Instead of $(user, movie)$ ratings, the transformed dataset contains $(user, feature)$ ratings. Figure 3 shows how this transformation can be used to predict ratings based on movie directors. First, the user-movie rating matrix R_{um} is transformed to a user-director rating matrix R_{ud}. To calculate the $R_{ud}[u, d]$ element of this matrix, *i.e.,* how user u rates director d, we look at all the movies directed

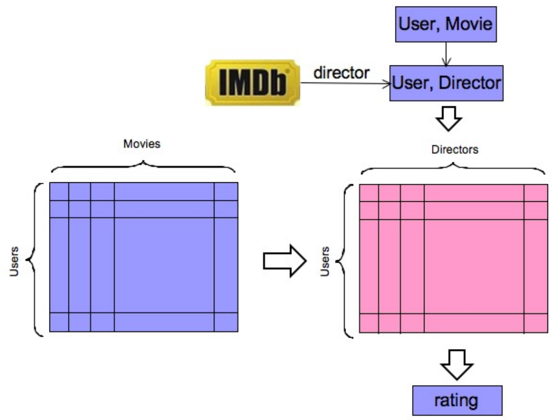

Figure 3: Transforming the $(user, movie)$ matrix to a $(user, director)$ matrix and predicting the rating based on the director of the given movie

by director d, and calculate the average ratings of user u to these movies. To make a prediction for a new $(user, movie)$, we look up the director of the movie and make a K-NN prediction on R_{ud} matrix. A similar approach can be used to predict ratings based on the actors of a movie[1].

4.3 Combining predictions

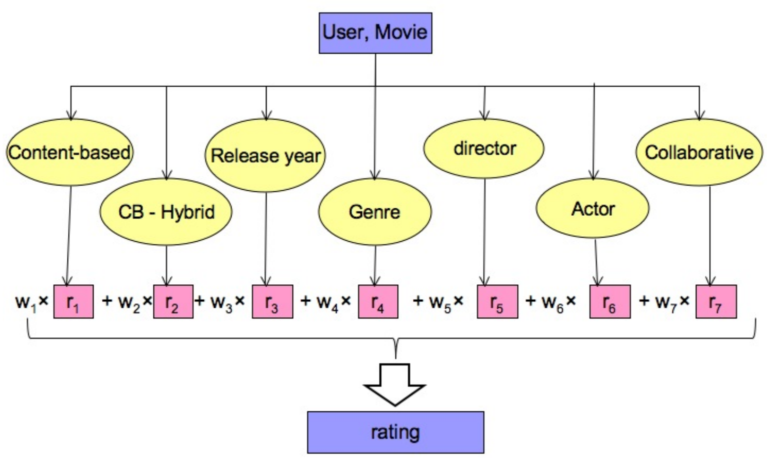

Figure 4: Combining predictions of different methods

Using the methods described in previous sections, as well as other existing collaborative methods, a number of different predictions will be generated for a given $(user, movie)$ pair. These predictions can then be combined, using an ensemble method such as regression, to make a final prediction. The combined prediction can potentially be more accurate than any of the individual predictions, because each of these predictions is based on a subset of features of the movies and is capturing a different aspect of the data. Bell and Kore indicate that in ensemble methods, the accuracy of individual predictors is not as important as their ability to expose different characteristics of the data [16].

Furthermore, the rating habits of different types of users may be better captured by different prediction methods. In our experiments, we also employ an enhancing technique, developed by the "Reel

[1]While attempting to build the user-actor matrix, the size of the resulting training set exceeded 6GB, and we were unable to run it on any of our servers.

Method	RMSE
Collaborative: Movie K-NN	0.948599
Hybrid: Adding collaborative to content-based	0.978943
Hybrid: Dataset partitioned by year	0.9999199
Hybrid: Dataset partitioned by genre	1.0007305
Hybrid: Dataset transformed based on director	1.0183212
Combination of predictions	**0.944979**

Table 1: Prediction accuracy of the implemented hybrid methods and the combination of them.

Ingenuity" team from the University of Alberta [2], that groups the users into a number of categories and calculates a different set of weights for each category.

5 Experimental results

To investigate the effect of using additional movie information in the new hybrid methods, we extracted the feature information for a subset of Netflix movies, and investigated whether adding our new hybrid methods to the existing collaborative methods improves their prediction accuracy.

5.1 Netflix dataset

The Netflix Prize [5] training set contains one hundred million ratings of more than half a million users rating 18000 movies. The data was collected between 1998 and 2005, and was released in October 2006. An additional set, containing only $(user, movie)$ pairs with no ratings, is also released. This set, also known as the *qualifying set*, is used to evaluate the submitted predictions. A subset of the training set, known as the *probe set*, is also offered. This subset contains about 1.4 million ratings that are similar to those in the qualifying set.

5.2 Extracting movie features

IMDb [4] is an easily accessible source of movie features. However, the information provided by IMDb is licensed and using it is prohibited by the rules of the Netflix Prize. On the other hand, extracting the features from license-free sources, such as Wikipedia, is challenging. Since we are only interested in investigating the effect of additional data, and do not plan to submit our predictions to Netflix, we can still use the data provided by IMDb. Once it is demonstrated that the additional data is indeed useful, more resources can be assigned to extract the movie features from license-free sources.

We extracted different features such as genre, actor and director for each movie from IMDb. The release year of each movie is also available in the Netflix dataset. Among more than 430,000 media titles in downloadable text files from IMDb, we were able to match about 8800 movies in the Netflix dataset using title and release year. After removing the unmatched movies from the dataset, the training set was reduced to 80 million ratings and the probe set was reduced to 1 million ratings.

5.3 Evaluation method

Since we are unable to submit our predictions of the qualifying set for evaluation, we randomly chose two third of the probe set as our validating set and the other third as the test set. The validating set was used to find optimal parameters for our predictors and regression weights for combining predictions, while the test set was used to evaluate the accuracy of our models.

A simple movie K-NN collaborative method was developed to evaluate our models. We then combined the predictions of this method and the hybrid methods discussed in section 4. Table 1 summarizes the root mean square error (RMSE) of the individual methods, as well as the RMSE of the combined predictions. Combining the predictions of the hybrid methods has resulted in a small improvement over the accuracy of the collaborative method.

Method	RMSE
Combination of Reel Ingenuity predictions	0.896186
Combination of our predictions	0.944979
Ensemble of both combinations	**0.895751**

Table 2: Result of combining our predictions and Reel Ingenuity predictions.

We also combined our predictions with eleven canonical predictions developed by "Reel Ingenuity". Table 2 presents how combining the result of our predictions and "Reel Ingenuity" predictions can slightly improve the accuracy over the combination of each set.

6 Conclusion

Our results indicate that combining hybrid methods based on extra movie information has the potential to improve the accuracy of current collaborative methods in the Netflix challenge. While we achieve a small improvement in our current experiments, our implementation of the hybrid methods have room for improvement, and can be further fine-tuned.

A simple method that may improve the current results is to partition the movies in the training set based on the number of ratings for each movie, and find a different set of weights for each partition. This may result in better predictions, because collaborative and content based methods perform differently for movies that have large or small number of ratings.

7 Acknowledgments

We wish to thank Dr. Russ Greiner for his leading and supervision during the project and also thanks to the Reel Ingenuity team for all their guidance and sharing the code base.

References

[1] http://acmehill.blogspot.com/.

[2] https://www-s.cs.ualberta.ca/intranet/wiki/bin/view/netflix/currentapproaches.

[3] http://tomfohr.blogspot.com.

[4] http://www.imdb.com.

[5] http://www.netflixprize.com.

[6] http://www.tillberg.us/netflixprize.

[7] Gediminas Adomavicius and Alexander Tuzhilin. Toward the next generation of recommender systems: A survey of the state-of-the-art and possible extensions. *IEEE Trans. on Knowl. and Data Eng.*, 17(6):734–749, 2005.

[8] Marko Balabanović and Yoav Shoham. Fab: content-based, collaborative recommendation. *Commun. ACM*, 40(3):66–72, 1997.

[9] J. Basilico and T. Hofmann. Unifying collaborative and content-based filtering, 2004.

[10] R. Bell and Y. Koren. Improved neighborhood-based collaborative filtering. KDD-Cup and Workshop, 2007.

[11] Robert Bell, Yehuda Koren, and Chris Volinsky. Modeling relationships at multiple scales to improve accuracy of large recommender systems. In *KDD '07: Proceedings of the 13th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 95–104, New York, NY, USA, 2007. ACM.

[12] Robert M. Bell and Yehuda Koren. Scalable collaborative filtering with jointly derived neighborhood interpolation weights. In *ICDM*, pages 43–52, 2007.

[13] Kyung-Yong Jung, Dong-Hyun Park, and Jung-Hyun Lee. Hybrid collaborative filtering and content-based filtering for improved recommender system. In *International Conference on Computational Science*, pages 295–302, 2004.

[14] Kunaver M., Pozrl T., Pogacnik M., and Tasic J. Optimisation of combined collaborative recommender systems. *AEU - International Journal of Electronics and Communications*, 61:433–443, July 2007.

[15] Arkadiusz Paterek. Improving regularized singular value decomposition for collaborative filtering. *Proceedings of KDD Cup and Workshop*, 2007.

[16] Yehuda Koren Robert M. Bell and Chris Volinsky. The bellkor solution to the netflix prize. Technical report, AT&T Labs - Research, 2007.

[17] Ruslan Salakhutdinov, Andriy Mnih, and Geoffrey Hinton. Restricted Boltzmann machines for collaborative filtering. In *Proceedings of the International Conference on Machine Learning*, volume 24, pages 791–798, 2007.

[18] I. Soboroff and C. Nicholas. Combining content and collaboration in text filtering, 1999.

[19] Gábor Takács, István Pilászy, Bottyán Németh, and Domonkos Tikk. Major components of the gravity recommendation system. *SIGKDD Explor. Newsl.*, 9(2):80–83, 2007.