Collaborative Filtering-Based Film Recommendation Technique Utilizing Time and Film Genres

Huilin Wang and Jianhua Wu
Institute of Rail Transportation, Jinan University, No.206, Qianshan Road, Zhuhai City, Guangdong Province, China
Email: 552135371@qq.com; tjhwu@jnu.edu.cn

Abstract. Due to the uncertainty of users’ preferences for different movies at different times, movie recommendation poses a huge challenge. We present an improved collaborative filtering algorithm for film recommendation whose design was motivated by a need to extract meaningful information for further recommending from film viewing applications. To this end, the method we propose utilises an improved method for analyzing user interest in film genres, which can then be subsequently used to predict users’ ratings of unwatched movies in combination with the current viewing time. In our new algorithm, we extract genre information considering time decay factor and combine it with matrix factorization collaborative filtering algorithm, and propose a novel algorithm structure, the MFTGICF algorithm. Compared with general traditional collaborative filtering algorithms and some other methods, this algorithm has more precision and higher stability. Both the theoretical analysis and extensive experiments show the improvement of the effectiveness and efficiency of the proposed method.

1. Introduction
Recommendation systems—“a subclass of information filtering system that seeks to predict the ‘rating’ or ‘preference’ that user would give to an item [1]” is widely used in commercial applications. It can for example be recognized as product recommenders in E-commercial platforms like Amazon, or content recommenders for online news and magazines. Usually, fast and accurate recommendation system contributes to the promotion of revenue growth by increasing user interest and interaction.

As the core of the recommendation system, recommendation algorithm, which aims to create a specific method to achieve accurate prediction within a limited time, directly affects the accuracy and practicality of the system. However, due to overload information and the extreme demand for a small amount of time and calculation, most algorithms are faced with challenging situations.

The two most frequently used algorithm models so far are collaborative filtering and content-based filtering. Collaborative filtering uses existing user preferences or tastes to predict unknown items. Content-based filtering methods are based on a description of the item and a profile of the user’s preferences [2-3].

In this paper, we propose an improved method, which is based on matrix factorization - a class of collaborative filtering, and combines time-faded functions as well as movie genres to predict the unknown ratings of films. Our new algorithm MFTGICF is inspired by [4], which employs Ebbinghaus forgetting curve [5-6] in order to present the relationship between time and user preferences and film genre [6-7] to find out recent types that users like for further predictions. The key issue in this paper is to find an improved algorithm that makes the results more accurate within limited computation time.
We perform evaluation of our algorithm by using real world dataset-movielens 100k dataset. In addition, we compare our approach to a few of state-of-the-art algorithms in the field. Our experiments show that MFTGICF produces competitive results.

2. Related Work

2.1. Memory Forgetting Curve and Interest Drift
The notion of forgetting curve was first introduced by Hermann Ebbinghaus [8]. Figure 1 shows how information is lost over time when there is no attempt to retain it. Psychology proves that interests of people affected by external environmental, physiological and various factors is changing over time, and this change is called Interest Drift [9].

2.2. Matrix Factorization
Matrix factorization (MF) is a class of collaborative filtering algorithms. This family of methods became widely known during the Netflix prize challenge due to its effectiveness as reported by Simon Funk in his 2006 blog post. It works by decomposing the user-item interaction matrix into the product of two lower dimensionality rectangular matrices [10], as illustrated in figure 2.

![Figure 1. Forgetting curve.](image1)

![Figure 2. Matrix factorization.](image2)

There is a variety of ways to decompose the matrix such as Funk MF [11], SVD++ [12], Asymmetric SVD [8] and so on. However, the basic ideas of decomposing are the same - finding two matrices whose product are the closest to the original matrix. Since the decomposed matrices are filled with values, unknown values in the original matrix can therefore be predicted using matrix production.

There are two common ways to optimize the loss function in MF method - Alternating Least Squares (ALS) and Stochastic Gradient Descent (SGD). The core of ALS algorithm is fixing one matrix at first and then the non-convex optimization problem in equation (1) is turned into an “easy” quadratic problem.

\[
\text{argmin}_{p, q} \sum_{u,i} (r_{u,i} - p_u^T q_i)^2
\]

(1)

\( r_{u,i} \) is the rating of user \( u \) and item \( i \). \( p_u \) and \( q_i \) represent the values in the corresponding submatrices. By applying this step alternately to the User Matrix and Item Matrix, we can iteratively improve the matrix factorization and get a guaranteed solution.

Different from ALS method, SGD approach optimizes the loss function by calculation the derivation of equation (1) and then get to the next step by choosing the opposite direction of the gradient.

Comparing these two methods, ALS algorithm is more practical because it always reaches the order of billions when using SGD method for implicit feedback datasets.
2.3. Time and Genre Based Recommendation Algorithm

In film recommendation, user interest tends to be extremely sensitive to time. A popular approach to apply forgetting curve based on time to collaborative filtering is represented in Ref. [13], which states that the longer the time, the lower the importance of the movie to the user, and the less weight it takes compared to movies that are recently watched. However, this algorithm uses a linear decreasing function to calculate the time weight, which has a certain gap in describing people’s actual interest drift [14]. To make up for this deficiency, Ref. [14] utilizes Ebbinghaus’ forgetting rule to fit the nonlinear forgetting curve to the greatest extent and integrates it into the recommendation algorithm. However, due to the fact that the user’s forgetting rule will be affected by various factors in real life, the algorithm is less flexible. To address this issue, Ding et al. [15] introduced the concept of information half-life, which is, a parameter that is used to flexibly defines a nonlinear forgetting curve for adapting to different recommended scenarios. Based on this method, an improved algorithm is represented in Ref. [16], which put forward the concept of information retention period. Ref. [16] considers the impact of information as a constant for a period of time and it also combines Neighbor-based collaborative filtering in order to get a more ideal result. However, although the accuracy is improved, the calculation of large-scale data sets is extremely complicated, and the algorithm is unable to show the superiority in time. To make up for this, Ref. [14] utilize SVD matrix factorization to simplify the calculation complexity, and increase the accuracy and time superiority at the same time. Combined with these two methods in Refs. [14, 17], an improved matrix factorization algorithm in Ref. [4] was presented. This method introduces the genre influence factor. If users repeatedly watch films with the same genre, the weight of this genre will be positively or negatively enhanced [4]. In addition, the algorithm also considers the impact of time and incorporates matrix factorization model for recommendation. However, although this algorithm improves the accuracy, it is sensitive to extreme data points. In the actual recommended scenario, it is very likely to be affected by extreme data points, which leads to deviations in the overall prediction.

3. Preliminaries

Ref. [4] has introduced genre weight and time weight two parts to get final recommendation. Some related definitions and algorithm details are shown as below.

3.1. Definitions

Definition 3.1: (Impact Enhancement Cycle $T_1$)

The time it takes to double the influence of a movie genre.

According to the “mere-exposure effect” [18], a psychological phenomenon by which people tend to develop a preference for things merely because they are familiar with them. The more often a thing appear, the stronger their love for the thing. Therefore, the more times a user is exposed to films of the same genre while watching, the greater the influence of the genre. In other words, the smaller $T_1$, the shorter the time required for the user to enhance the impression of the movie genre. Conversely, the larger $T_1$, the more time the user needs to enhance the impression of the movie genre [4].

Definition 3.2: (Viewing frequency freq)

A viewing frequency is defined as

$$freq(u, a_{vl}) = \frac{T_{avl last} - T_{avl first}}{t_{now} - T_{avl last}}$$  (2)

freq($u, a_{vl}$) is the frequency with which user $u$ watches movies with genre type $a_{vl}$. $T_{avl last}$ is the time when user $u$ last watched $a_{vl}$ type movies. $T_{avl first}$ represents the time when user $u$ first watched $a_{vl}$ type movies. $t_{now}$ indicates the time when the system recommends the movie $v$ to the user. Overall, the value of freq determines how often the user watches the movie. The larger the values, the more frequently the user watches the movie of corresponding genres, and vice versa.

Definition 3.3: (Genre influence weight $w_{increase}$)

A genre influence weight, abbr. genre-weight, for user $u$ at time $T_{now}$ is defined as
\[ w_{\text{increase}}(u, a_v, \text{freq}, T_1) = \text{freq} \times T_1 \]  

(freq is the viewing frequency of user \( u \) for genre \( a_v \). \( T_1 \) is Impact Enhancement Cycle which is a constant.)

Definition 3.4: (Information half-life \( T_0 \))
The time from the release of the information until the influence on the user drops by 50 percent [16].)

Definition 3.5: (Information retention period \( T' \))
The time period for keeping the influence of the movies on the users unchanged [16].)

Definition 3.6: (Time influence weight \( w_{\text{time}} \))
Time influence weight, abbr. time-weight is defined as

\[ w_{\text{time}} = \frac{1}{T_0} \times T' \times \text{floor} \left( \frac{t_{\text{now}} - t_0}{T'} \right) \]  

(4)

\( T_0 \) is information half-life and \( T' \) is the information retention period. \( \text{floor} \left( \frac{t_{\text{now}} - t_0}{T'} \right) \) is round down function.

Definition 3.7: (Time and genre influence weight)
Time and genre influence weight, abbr. t-g-weight is defined as

\[ f_{u,v}(t_{\text{now}}) = \frac{1}{n} \sum_{i=1}^{n} \ln 0.5 + (\alpha \times w_{\text{time}} - \beta \times w_{\text{increase}}) \]  

(5)

\( w_{\text{time}} \) is the time-weight formula, \( w_{\text{increase}} \) is the genre-weight, and \( \alpha \) and \( \beta \) are the weight balance parameters, which are used to adjust the weight of \( w_{\text{time}} \) and \( w_{\text{increase}} \).

3.2. NMFTCG Algorithm
The pseudocode of the NMFTG algorithm is shown in figure 3.

Algorithm 1: NMFTCG Algorithm

Require: User-item rating matrix \( R \), Information half-life \( T_0 \), Information retention period \( T' \), decay parameter \( \varepsilon \), Number of recommended movies \( N \).

Result: List of \( N \) recommended films for a specific user.

for each user \( i \) in matrix \( R \) do

Determine a candidate list \( Q \) of movies to be predicted for user \( i \);
Pre-process the rating of the user-item matrix to obtain new matrix \( R' \) using equation (5);
Use maximum-minimum normalization to map the scoring matrix to the [1-5] scoring interval to obtain a new weighted scoring matrix \( R'' \);
end
Use matrix \( R'' \) to perform ALS matrix factorization and get user characteristic matrix \( U \) and item characteristic matrix \( V \);

for each user \( i \) in matrix \( R \) do

for each item \( j \) in matrix \( R \) do

Predict user \( i \)'s rating of movie \( j \) according to \( P_{ij} = U_i \times V_j \);
end
Get the top \( N \) highest rated films as final recommendation list for user \( i \);
end

Figure 3. Pseudocode for NMFTCG algorithm.

4. MFTGICF Algorithm
In this section, we introduce MFTGICF algorithm. This algorithm combines concepts from NMFTCS algorithm to create a new method in order to improve accuracy and make up for the shortcomings of NMFTCG algorithm.
The method for calculating genre-weight in equation (2) has certain limitations. If a user watches a movie containing certain genres for a long period of time and continues until the current recommended time, the formula denominator approaches 0, and the calculation result in equation (2) is extremely large. In addition, if a user watches a movie containing some certain genres for a really short time, the calculation result tends to 0. These two situations are very likely to appear in real data, but it will inevitably lead to large deviations in the normalization step, resulting in extreme errors.

Our approach utilizes a new way to improve viewing frequency from equations (2)-(6).

Definition 4.1: (Viewing frequency $freq(u, v_i)$)

$$freq(u, v_i) = \frac{n_{u,v_i}}{\sum_{j=1}^{k} n_{u,v_j}}$$

Equation (6) $freq(u, v_i)$ represents the viewing frequency of user $u$ for film $v$ with genre $i$. $n_{u,v_i}$ represents the number of movies user $u$ watched with genre $v_i$. $\sum_{j=1}^{k} n_{u,v_j}$ is the sum of the number of genres watched by the user $u$ during the time period of watching movie $v$ containing genre $j$. $k$ represents the number of genres contained within film $v$. This method can effectively avoid the problems of infinity and infinitesimal, and can reasonably represent the user’s viewing frequency.

Therefore, an improved equation for calculating the genre-weight in equation (3) is defined as

Definition 4.2: (Genre-weight $w_{\text{genre}}$)

$$w_{\text{genre}} = \frac{1}{k} \sum_{i=1}^{k} freq(u, v_i)$$

Equation (7) $w_{\text{genre}}$ represents the genre-weight of user $u$ about the movie $v$.

As for time-weight, this algorithm utilizes an approach from Ref. [16].

Definition 4.3: (Time-weight $w_{\text{time}}$)

$$w_{\text{time}} = e^{\lambda \times \tau'}$$

Equation (8) $w_{\text{time}}$ represents the time-weight of user $u$ about the movie $v$.

In equation (8), $\lambda$ is time decay factor introduced in Ref. [16]. $\tau'$ is information retention period in definition “3.6”.

The final weight function combining time and genre two parts is defined as

Definition 4.4: (Total weight $F_{u,v}(t_{\text{now}})$)

$$F_{u,v}(t_{\text{now}}) = \alpha \times w_{\text{time}} + \beta \times w_{\text{genre}}$$

Equation (9) $F_{u,v}(t_{\text{now}})$ is final total weight of user $u$ about the movie $v$ at the current recommended time. $\alpha, \beta$ are parameters for adjusting the weight of $w_{\text{time}}$ and $w_{\text{genre}}$.

The pseudocode of the MFTGICF algorithm is shown in figure 4.
5. Evaluation

To evaluate our algorithm, we choose a 100k Movielens dataset from GroupLens which contains 100,000 users’ 100,000 rating records for 1,682 movies, with each user rating at least 20 movies and corresponding rating time. In addition, the data set records in detail the genre each movie belongs to. There are 19 movie genres in total, such as action movies, horror movies, thrillers, etc.

In order to restore the actual recommended scene as much as possible, the ranking behavior of each user is sorted according to time, and the first 80% of each user’s data is used as the training set, and the last 20% as the test set.

We adopt Root Mean Square Error (RMSE) as evaluation metric.

\[
RMSE = \left[ \frac{1}{n} \sum_{i=1}^{n} (P_i - R_i)^2 \right]^{-1}
\]

(10)

\(P_i\) and \(R_i\) in equation (13) are the predicted value and actual value of user \(u\)’s rating for film \(v\), respectively. \(n\) represents the number of ratings of all movies by all users. RMSE is the standard deviation of the residuals, and is more sensitive to outliers compared to Mean Absolute Error (MAE) [19].

In order to get best quality of the algorithm, we use different values to adjust the initialization parameters, and iteratively substitute the best parameters to find the optimal values of the next unknown parameters.

Figure 5 indicates that when weight parameter \(\alpha = \beta = 0.5\) in equation (9) with \(T_0 = 30, T' = 3\), the algorithm represents best RMSE result. In figure 6, we adopt the best result \(T_0 = 30, T' = 2\).
Figure 5. RMSE results for different $(\alpha, \beta)$ when $T_0 = 30, T' = 3$

Figure 6. RMSE results for different $T_0, T'$ values when $\alpha = \beta = 0.5$.

For further evaluation, we substitute the previously adjusted parameter values into the algorithm and compare it with memory-based collaborative filtering and matrix factorization algorithm.

Figure 7 contains results of two memory-based collaborative filtering algorithms—traditional neighbor-based collaborative algorithm (NCF) and NTWCF algorithm from Ref. [4].

For matrix factorization algorithm, we adopt traditional Alternative Least Squares algorithm (ALS) and an improved method based on ALS from Ref. [7] to compare with our new algorithm. The results are shown in figure 8.

It is obvious that our algorithm embodies the greatest superiority in RMSE result whether compared to memory-based collaborative filtering or model-based collaborative filtering algorithm.

Figure 7. RMSE results of NCF, NTWCF, MFTGICF algorithms.

Figure 8. RMSE results of ALS, ALSMFCF, MFTGICF algorithms.

6. Conclusion
In this paper, we have proposed a new algorithm MFTGICF, an effective and efficient method for recommending films. This algorithm is able to recommend according to time and users’ interests in film genres. The improved structure of user interest in genres reveals more stability and accuracy. Our experimental performance evaluation over a real data sets demonstrates the effectiveness and stability of MFTGICF in predicting the ratings of film for recommending.

The future work includes the following topics: the analysis of time and space complexity of this algorithm for big data, dynamic adaption of the parameters in data streams, and investigation of our framework for outlier detection.

References
[1] Karypis G 2001 Evaluation of item-based top-N recommendation algorithms Proc. of CIKM 2001 pp 247-254.
[2] Linden G, Smith B and York J 2003 Amazon.com recommendations: Item-to-Item collaborative filtering IEEE Internet Computing 7 (1) 76-80.

[3] Junran C 2019 Application of collaborative filtering technology in movie recommendation Computer Programming Skills and Maintenance (01) 3-94+97.

[4] Shi H Y, Sun T H, Li S Q and Hou X 2019 A matrix factorization recommendation algorithm with time and type weight Journal of Chongqing University 42 (01) 79-87.

[5] Bokde D, Girase S and Mukhopadhyay D 2015 Matrix factorization model in collaborative filtering algorithms: A survey Procedia Computer Science 49 136-146.

[6] Erxin W 2017 Research and Implementation of Personalized E-commerce Recommendation System Based on Implicit Semantic Model and Clustering Algorithm (Beijing University of Posts and Telecommunications).

[7] Wei S Y, Ning Y, Zhu J, Huang X and Zhang S 2012 Collaborative filtering recommendation algorithm based on item clustering and global similarity Computer Science 39 (12) 149-152.

[8] Pu L and Faltings B 2013 Understanding and improving relational matrix factorization in recommender systems Proceedings of the 7th ACM Conference on Recommender Systems-Rec. Sys. pp 41-48.

[9] Jameson A 2005 User modeling meets usability goals Lecture Notes in Computer Science (3538) 1-3.

[10] Jia Y 2014 Users’ brands preference based on SVD++ in recommender systems 2014 IEEE Workshop on Advanced Research and Technology in Industry Applications (WARTIA) pp 1175-1178.

[11] Agarwal D and Chen B-C 2009 Regression-based latent factor models Proceedings of the 15th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining-KDD pp 19-28.

[12] Xing X C, Gao F R, Zhan S N and Zhou L Z 2007 A collaborative filtering recommendation algorithm incorporated with user interest change Journal of Computer Research and Development 296-301.

[13] Koychev I and Schwab I 2000 Adaption to drifting user’s interests Proceedings of ECMI 2000 Workshop: Machine Learning in New Information Age pp 39-46.

[14] Dong L Y, Wang Y Q, He J N, Sun M H and Li Y L 2017 Collaborative filtering recommendation algorithm based on time decay Journal of Jilin University Engineering and Technology Edition 47 (4) 1268-1272.

[15] Ding Y and Li X 2005 Time weight collaborative filtering Proceedings of the 14th ACM International Conference on Information and Knowledge Management pp 485-492.

[16] Lan Y and Cao F F 2017 Research on time weighted collaborative filtering algorithm for movie recommendation Computer Science 44 (4) 295-301, 322.

[17] Sun B S and Dong L Y 2017 Dynamic model adaptive to user interests drift based on cluster and nearest neighbours IEEE Access 5 1682-1691.

[18] By Bornstein, Robert F, D’Agostino and Paul R 1992 Stimulus recognition and the mere exposure effect Journal of Personality and Social Psychology 63 (4) 545-552.

[19] Willmott C J and Matsuura K 2005 Advantages of the mean absolute error (MAE) over the root mean square error (RMSE) in assessing average model performance Climate Research 30 79-82.