

# A New Temporal Recommendation System based on Users' Similarity Prediction

Nima Joorabloo<sup>1</sup><sup>a</sup>, Mahdi Jalili<sup>1</sup><sup>b</sup> and Yongli Ren<sup>2</sup><sup>c</sup>

<sup>1</sup>*School of Engineering, RMIT University, Swanston, Melbourne, Australia*

<sup>2</sup>*School of Science, RMIT University, Swanston, Melbourne, Australia*

**Keywords:** Recommendation System, Sequential Pattern, Similarity Measure, Time.

**Abstract:** Recommender systems have significant applications in both industry and academia. Neighbourhood-based collaborative Filtering methods are the most widely used recommenders in industrial applications. These algorithms utilize preferences of similar users to provide suggestions for a target user. Users' preferences often vary over time and many traditional collaborative filtering algorithms fail to consider this important issue. In this paper, a novel recommendation method is proposed based on predicting similarity between users in the future and forecasting their similarity trends over time. The proposed method uses the sequence of users' ratings to predict the similarities between users in the future and use the predicted similarities instead of the original ones to detect users' neighbours. Experimental results on benchmark datasets show that the proposed method significantly outperforms classical and state-of-the-art recommendation methods.

## 1 INTRODUCTION

Recommendation systems (RSs) are developed to deal with the exponential growth of information on the web and to provide personalized contents and service delivery to users. RSs have been applied in many areas including e-commerce, advertisement, news, document management, and e-learning to increase the probability of cross-selling, customer needs fulfilment, and customer loyalty by recommending products of possible interests to users (Lihua et al., 2005).


Neighbourhood-based Collaborative Filtering (CF) is a widely used approach in RSs, which is based on similarity values between the users (or items). In other words, the CF approach works based on the assumption that if users have similar accessing behaviours in the past, they are likely to prefer similar items in the future. There are different criteria for the user's preferences including location, time, weather, and device type. These criteria can give us valuable information to improve the performance of RSs (Adomavicius et al., 2011, Baltrunas and Ricci, 2014). However, these valuable criteria are not often


considered in classical CF-based methods to provide recommendations. Using timestamp and sequences of ratings can be useful to improve the accuracy of recommendations. To address these issues, a novel temporal recommendation method is proposed using predicting users' similarities in the future. To this end, the similarity between two users is calculated in different time-windows which leads to having a trend of the similarity between them over time.


The rest of the paper is organized as follows: related studies are reviewed in Section 2. Section 3 introduces the proposed method. In Section 4, the proposed method is compared with the other methods by performing experiments on well-known datasets. Finally, some concluding comments are discussed in Section 5.

## 2 RELATED WORKS

Some studies have considered time information in recommendations. Time information for the ratings is useful criterion, which can be used to track changes in user preferences and behaviour over time (Liu et

<sup>a</sup> <https://orcid.org/0000-0001-8322-7857>

<sup>b</sup> <https://orcid.org/0000-0002-0517-9420>

<sup>c</sup> <https://orcid.org/0000-0002-3137-9653>

al., 2010). Ding used exponential time decay function to assign a weight to each user’s rating (Ding and Li, 2005). However, not all recent data are more always important than old data, while their method does not capture the importance. Zimdars treated CF as a time series problem, and transformed the data into time order information, and then used a decision-tree learning process recommend items (Zimdars et al., 2001). Lathia formalized CF as a time-dependent problem and proposed a method to automatically assign and update neighbourhood sizes per users instead of setting fix size (Lathia et al., 2009). Ricci proposed a recommendation method using long-term users’ preferences by mining past interactions and also by letting users define a set of stable preferences explicitly (Ricci and Nguyen, 2007). Koren predicted movie ratings by modelling the temporal dynamics via a factorization model over Netflix prize dataset (Koren, 2009). Tang et al. used movie production years to improve the accuracy of the collaborative filtering-based recommender systems by scaling down candidate sets (Tang et al., 2003). Lee considered user purchase time and item launch time to improve the accuracy of recommendations (Lee et al., 2008). They further used an empirical study to show that how temporal information, such as user purchase time, item launch time, the time difference between the two and also a combination of them affects the accuracy of a CF system (Lee et al., 2009). A user-based CF algorithm was proposed in (Karahodza et al., 2014) using temporal contextual information to increase the accuracy of RS; where weight function is considered which is based on changes in the group user’s preferences over time. Xia introduced the concept of time decay and the item-based similarity function was redefined based on time decay. Moreover, a dynamic item-based top-N recommendation algorithm was proposed to provide recommendations (Xia et al., 2010).

### 3 PROPOSED METHOD

In this section, a novel recommendation method is proposed which is based on sequences of ratings and predicting user’s similarities in the future. To this end, first a formal representation model is considered for sequences of users’ ratings. Then, similarity between users is calculated using Pearson correlation in different time-windows to generate a time-series of similarities between users over time. In the next step, the proposed algorithm predicts the similarity between users in the future and uses predicted values

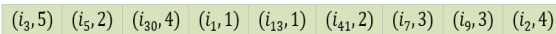
to recommend items. In the following we provide details of these steps.

#### 3.1 Sequential Pattern Representation

A formal representation model for sequential patterns is introduced to efficiently consider time and sequences of ratings. This model is based on the order of items which have been rated by the users. If  $U$  be the set of all users,  $I$  the set of all items and  $R$  the set of all ratings in the system, sequence of rating for target user  $u$  is represented as  $S_u = \langle x_1, x_2, \dots, x_l \rangle$ , where each  $x_i$  is an element of the sequence denoted as  $(i, r)$  with  $i \in I$  being an item rated by user  $u$  and  $r$  is its rating. It should be noted that  $l$  is the number of items rated by user  $u$ .  $x_i$  pairs are sorted in an ascending order based on the time of ratings. Suppose that the rating history for users  $u$  and  $v$  is like Table 1.  $S_u$  and  $S_v$  are sequential patterns of ratings for users  $u$  and  $v$  respectively.  $S_u$  and  $S_v$  are depicted in Figure 1.

Table 1: Rating history for users  $u$  and  $v$ .

User	Item	Rate	Time
$u$	$i_3$	5	$t_1$
	$i_1$	1	$t_4$
	$i_2$	4	$t_9$
	$i_{30}$	4	$t_3$
	$i_5$	2	$t_2$
	$i_9$	3	$t_8$
	$i_7$	3	$t_7$
	$i_{41}$	2	$t_5$
	$i_{13}$	1	$t_5$
	$v$	$i_2$	5
$i_1$		1	$t_4$
$i_6$		2	$t_3$
$i_9$		3	$t_5$
$i_7$		3	$t_1$
$i_8$		2	$t_2$

$S_u$ :  



$S_v$ :  


Figure 1: Sequential patterns for users  $u$  and  $v$  generated based on ratings’ timestamp.

#### 3.2 Obtaining Similarity Trend

In this section, a method is proposed to obtain trend

of similarity between two users. Our aim is to capture similarity changes over time. The classical CF algorithms first use methods like Pearson and Cosine similarity to calculate the similarity between users, and then select users with the highest similarities as neighbours of the target user. Pearson similarity between user  $a$  and  $b$  is denoted by  $PC(a, b)$  and calculated as follow:

$$PC(a, b) = \frac{\sum_{i \in I_{a,b}} (r_{a,i} - \bar{r}_a)(r_{b,i} - \bar{r}_b)}{\sqrt{\sum_{i \in I_{a,b}} (r_{a,i} - \bar{r}_a)^2} \sqrt{\sum_{i \in I_{a,b}} (r_{b,i} - \bar{r}_b)^2}}, \quad (1)$$

where  $r_{a,i}$  is the given rating by the user  $a$  to item  $i$  and  $\bar{r}_a$  is the average rating of user  $a$ . In our proposed method, we use Pearson measure to calculate the similarity between two users in different time periods to obtain the trend of similarity between two users from the past to current time. To obtain the similarity trend for user  $u$  with user  $v$ , first we generate  $S_u$  and  $S_v$  sequences. Then, we calculate the similarity of  $S_u$  partially with  $S_v$  to see how the behavior of  $u$  changes over time. To this goal, we define a time-window of dynamic size with an initial size of  $w$ . We apply this time-window on  $S_u$  and calculate the Pearson similarity between items in the time-window with all items in  $S_v$ , which is denoted by  $Sim_1(u, v)$ . In the next step, we do not move the time-window, but increase its size by  $w$  and again calculate the similarity between  $S_u$  items in time-window and  $S_v$  items; this is denoted it by  $Sim_2(u, v)$ . We continue increasing the size of the time-window until it covers all items in  $S_u$ . In general, the similarity between  $S_u$  and  $S_v$  in different time-windows is calculated as follow:

$$Sim_i(u, v) = PC(S_u(1, ws_i), S_v) \quad (2)$$

where  $i$  shows the time-window indices that starts from 1.  $S_u(1, ws_i)$  indicates a subsequence of  $S_u$  from the first item  $S_u$  to the  $ws_i$ -th item and  $ws_i$  is the size of  $i$ -th time-window, which is calculated as follow:

$$ws_i = \begin{cases} i \times w & \text{if } i \times w \leq |S_u| \\ |S_u| & \text{if } i \times w > |S_u| \end{cases} \quad (3)$$

After calculating similarities, we make a time-series from obtained similarities that shows the similarity trend between user  $u$  and  $v$  and is denoted by  $ST(u, v)$ . Figure 2 shows the process of comparing  $S_u$  with  $S_v$  over different time-windows for our example of Figure 1. We set the initial size of the time-window to  $w=3$ . Thus, we can have three different time-windows over  $S_u$ , and as a result, we have three different similarities. The calculated

similarities are  $Sim_1(u, v) = 0$ ,  $Sim_2(u, v) = 0.11$ , and  $Sim_3(u, v) = 0.56$ . We use them to create a time-series of similarities that is  $ST(u, v) = \langle 0, 0.11, 0.56 \rangle$ . The goal is to predict the next item in this time-series, which is the similarity of users  $u$  and  $v$  in the future.

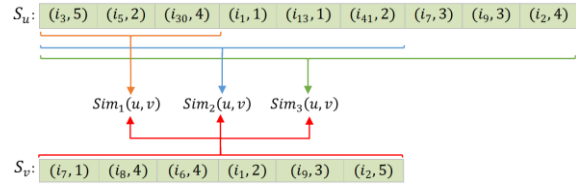


Figure 2: Rating history for users  $u_1$  and  $u_2$ .

In the next section we predict the similarity between user  $u$  and user  $v$  using  $ST$  time-series.

### 3.3 Predicting Users' Similarity

Time series forecasting has always been a hot topic in various fields with many applications in natural science, engineering, and economic management. It is about predicting the future as accurately as possible using historical data (De Gooijer and Hyndman, 2006). Two prediction strategies are usually employed in time series prediction models. One is the single-step prediction or short-term and immediate prediction. The other one is multi-step prediction, which shows long-term change over time (Xiong et al., 2013, Bao et al., 2014). In our work, we want to predict only the next step in our time-series, thus we need a method for short-term prediction. The only issue is that we have only a few data points in our time-series that limit our options to choose a proper model to fit to the data. With short time series, often there is not enough data to use for testing and validation purposes. Hyndman suggested simple models for such cases, because anything with more than one or two parameters will produce poor forecasts due to the estimation error (Hyndman and Athanasopoulos, 2018).

We propose a method based on linear regression and threshold approach to predict the next value in the time-series. For each user  $u$  and  $v$ , first we use linear regression to fit a line to the data points in  $ST(u, v)$ . The fitted line shows whether the similarity trend is upward or downward. If the trend is downward, it means that the similarity will likely be decreased in the future and the upward trend is proposing to have a higher similarity in the future. We use this idea to predict  $Sim_f(u, v)$ , which is indeed the similarity between  $u$  and  $v$  in the future by adding a positive value to the last value in  $ST(u, v)$  if the trend is

upward or deducting a positive value if the trend is downward.

$$Sim_f(u, v) = \begin{cases} ST_{last}(u, v) + p & \text{if } slop_{u,v} \geq 0 \\ ST_{last}(u, v) - n & \text{if } slop_{u,v} < 0 \end{cases} \quad (4)$$

where  $p$  and  $n$  are fix values between 0 and 1,  $ST_{last}(u, v)$  is the last items in the time-series and,  $slop_{u,v}$  is the slope of the fitted line to  $ST(u, v)$  points. We cap the predicted similarities that are more than 1 and less than -1 to 1 and -1 respectively to keep the correlation value between 1 and -1.

We do this process for all pair of users in the system and use them to recommend items to users. Figure 3 shows how the proposed method predicts  $Sim_f(u, v)$  for users  $u$  and  $v$  in our example. As can be seen, the trend of similarity is upward. Thus, based on our proposed method, we add a positive vale  $p$  to the last value in the time-series that is 0.56. We set  $p = 0.3$  leading to  $Sim_f(u, v) = 0.86$ .

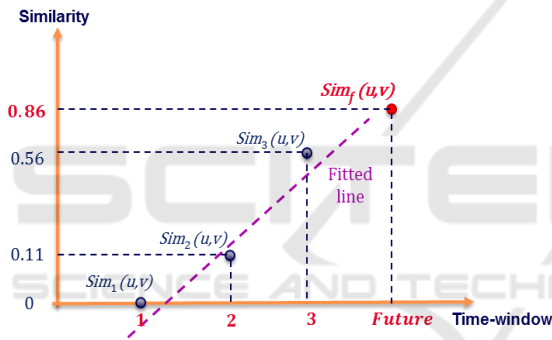


Figure 3: Rating history for users  $u_1$  and  $u_2$ .

### 3.4 Recommendation

After calculating the similarity values between the users, a subset of users with the highest similarities is formed as the set of nearest neighbors for the active user. In the next step, we use these neighbors to predict ratings for items in the dataset that have not been rated by the target user yet. The predicted rating of item  $j$  for the active user  $a$  can be calculated as follows:

$$P_{a,j} = \bar{r}_a + \frac{\sum_{u \in K_{a,j}} Sim_f(a,u) \cdot (r_{u,j} - \bar{r}_u)}{\sum_{u \in K_{a,j}} Sim_f(a,u)} \quad (5)$$

where  $\bar{r}_a$  is the average of the ratings provided by the active user  $a$ ,  $K_{a,j}$  is a subset of neighbors for the active user  $a$  that have rated item  $j$ , and  $Sim_f(a, u)$  is the predicted similarity value between the users  $a$  and

$u$ .

## 4 EXPERIMENTAL RESULTS

In this section, the proposed method is compared with the other recommendation methods to evaluate the quality of recommendation based on evaluation metrics. To this end, methods including user-based collaborative filtering (UCF), item-based collaborative filtering (ICF) (Sarwar et al., 2001), multi-level collaborative filtering (MLCF) (Polatidis and Georgiadis, 2016), GBF (Joorabloo et al., 2019), RankSGD (Töscher and Jahrer, 2012), EALS (He et al., 2016), AspectModel (Hofmann and Puzicha, 1999), IMULT (Ranjbar et al., 2015), and ULPE (Formoso et al., 2013) are used to evaluate the effectiveness of the proposed method. The details of the experiments are presented in the following subsections.

### 4.1 Dataset

Two benchmark datasets (MovieLens<sup>4</sup> and Netflix) are used in the experiments to verify the effectiveness of the proposed method. MovieLens dataset was collected by the GroupLens research group which contains 1682 movies, 943 users, and 100,000 ratings. Netflix dataset contains about 100 million ratings from over 480 thousand randomly-chosen, anonymous subscribers on nearly 18 thousand movie titles. The data were collected between October 1998 and December 2005 and reflect the distribution of all ratings received by Netflix during this period. We randomly select 1000 users for Netflix dataset for our experiment. All of the rating values are integer numbers in the range of 1 (bad) to 5 (excellent). Moreover, each rating has a timestamp which indicates the time of providing this rating by a user for a specific item.

### 4.2 Evaluation Metrics

In the experiments, precision metric is used to evaluate the performance of the compared methods.  $P_u(N)$  is precision for a list of recommended items to user  $u$  and is defined as the percentage of relevant items to user  $u$  in their list of recommendation. Precision of a system with  $N$  users,  $P(N)$ , is calculated as:

$$P(N) = \frac{\sum_{u \in testSet} P_u(N)}{N} \quad (6)$$

<sup>4</sup> <http://grouplens.org/datasets/movielens/>

### 4.3 Results

In this section, the results of experiments are reported to compare the proposed method with other recommendation methods. To perform the experiments, 80% of the ratings for each user are selected as the training set and the remaining data is used as a test set. To generate the result for the proposed algorithm, we set both  $p$  and  $n$  parameters to 0.3 and set  $w$ , the initial size of time-window to 10. The results of the experiments are reported in Tables 2.

Table 2: Performance of algorithms on MovieLens and Netflix dataset. The best result for each metric is shown in boldface.

Methods	Precision	
	MovieLens	Netflix
Proposed Method	<b>0.3903</b>	<b>0.4856</b>
GBP	0.3478	0.1127
AspectModel	0.2288	0.0810
RankSGD	0.2611	0.0535
UCF	0.3056	0.4780
IMULT	0.1842	0.1298
ICF	0.1096	0.1684
MLCF	0.2584	0.3910
ULPE	0.3722	0.1686
EALS	0.1741	0.0679

As you can see from these results, the proposed method significantly outperforms other methods over MovieLens dataset and the results even far better over Netflix dataset. It shows that the most of algorithms in our experiment are dataset-dependent and don't have a predictable behavior over all datasets. In addition, the results indicate that the classical methods not considering the time in their methodology have often lower precision than the temporal recommendation algorithms. Therefore, it can be concluded that considering timestamps and predicting users' similarities leads to improving the performance of recommendation in terms of accuracy.

## 5 CONCLUSIONS

In this paper, a novel recommendation approach is proposed to consider the effects of ratings' timestamps provided by the users into the recommendation process. To this end, a representation model of sequential patterns is first

introduced for the ratings provided by the users. Then, time-series of similarities between users over time is generated to predict the similarity between users in the future. In the next step, a subset of users with the highest similarities is formed as the set of nearest neighbors for each user in the dataset and then use it to recommend items to users. Experimental results on benchmark datasets indicate that the proposed method outperforms some traditional and state-of-the-art algorithms in terms of accuracy of predictions.

## REFERENCES

- Adomavicius, G., Tuzhilin, A. & Zheng, R. 2011. REQUEST: A query language for customizing recommendations. *Information Systems Research*, 22, 99-117.
- Baltrunas, L. & Ricci, F. 2014. Experimental evaluation of context-dependent collaborative filtering using item splitting. *User Modeling and User-Adapted Interaction*, 24, 7-34.
- Bao, Y., Xiong, T. & Hu, Z. 2014. Multi-step-ahead time series prediction using multiple-output support vector regression. *Neurocomputing*, 129, 482-493.
- De Gooijer, J. G. & Hyndman, R. J. 2006. 25 years of time series forecasting. *International journal of forecasting*, 22, 443-473.
- Ding, Y. & Li, X. Time weight collaborative filtering. Proceedings of the 14th ACM international conference on Information and knowledge management, 2005. ACM, 485-492.
- Formoso, V., Fernández, D., CACHEDA, F. & Carneiro, V. 2013. Using profile expansion techniques to alleviate the new user problem. *Information Processing & Management*, 49, 659-672.
- He, X., Zhang, H., Kan, M.-Y. & Chua, T.-S. Fast matrix factorization for online recommendation with implicit feedback. Proceedings of the 39th International ACM SIGIR conference on Research and Development in Information Retrieval, 2016. ACM, 549-558.
- Hofmann, T. & Puzicha, J. Latent class models for collaborative filtering. IJCAI, 1999.
- Hyndman, R. J. & Athanasopoulos, G. 2018. *Forecasting: principles and practice*, OTexts.
- Joorabloo, N., Jalili, M. & Ren, Y. A Probabilistic Graph-Based Method to Improve Recommender System Accuracy. International Conference on Engineering Applications of Neural Networks, 2019. Springer, 151-163.
- Karahodza, B., Supic, H. & Donko, D. An Approach to design of time-aware recommender system based on changes in group user's preferences. 2014 X International Symposium on Telecommunications (BIHTEL), 2014. IEEE, 1-4.
- Koren, Y. Collaborative filtering with temporal dynamics. Proceedings of the 15th ACM SIGKDD international



- conference on Knowledge discovery and data mining, 2009. ACM, 447-456.
- Lathia, N., Hailes, S. & Capra, L. Temporal collaborative filtering with adaptive neighbourhoods. Proceedings of the 32nd international ACM SIGIR conference on Research and development in information retrieval, 2009. ACM, 796-797.
- Lee, T. Q., Park, Y. & Park, Y.-T. 2008. A time-based approach to effective recommender systems using implicit feedback. *Expert systems with applications*, 34, 3055-3062.
- Lee, T. Q., Park, Y. & Park, Y.-T. 2009. An empirical study on effectiveness of temporal information as implicit ratings. *Expert systems with Applications*, 36, 1315-1321.
- Lihua, W., Lu, L., Jing, L. & Zongyong, L. 2005. Modeling user multiple interests by an improved GCS approach. *Expert Systems with Applications*, 29, 757-767.
- Liu, N. N., Zhao, M., Xiang, E. & Yang, Q. Online evolutionary collaborative filtering. Proceedings of the fourth ACM conference on Recommender systems, 2010. ACM, 95-102.
- Polatidis, N. & Georgiadis, C. K. 2016. A multi-level collaborative filtering method that improves recommendations. *Expert Systems with Applications*, 48, 100-110.
- Ranjbar, M., Moradi, P., Azami, M. & Jalili, M. 2015. An imputation-based matrix factorization method for improving accuracy of collaborative filtering systems. *Engineering Applications of Artificial Intelligence*, 46, 58-66.
- Ricci, F. & Nguyen, Q. N. 2007. Acquiring and revising preferences in a critique-based mobile recommender system. *IEEE Intelligent systems*, 22, 22-29.
- Sarwar, B. M., Karypis, G., Konstan, J. A. & Riedl, J. 2001. Item-based collaborative filtering recommendation algorithms. *WWW*, 1, 285-295.
- Tang, T. Y., Winoto, P. & Chan, K. C. Scaling down candidate sets based on the temporal feature of items for improved hybrid recommendations. IJCAI Workshop on Intelligent Techniques for Web Personalization, 2003. Springer, 169-186.
- Töscher, A. & Jahrer, M. 2012. Collaborative filtering ensemble for ranking. *Journal of Machine Learning Research W&CP*, 18, 61-74.
- Xia, C., Jiang, X., Liu, S., Luo, Z. & Yu, Z. Dynamic item-based recommendation algorithm with time decay. 2010 Sixth International Conference on Natural Computation, 2010. IEEE, 242-247.
- Xiong, T., Bao, Y. & Hu, Z. 2013. Beyond one-step-ahead forecasting: evaluation of alternative multi-step-ahead forecasting models for crude oil prices. *Energy Economics*, 40, 405-415.
- Zimdars, A., Chickering, D. M. & Meek, C. Using temporal data for making recommendations. Proceedings of the Seventeenth conference on Uncertainty in artificial intelligence, 2001. Morgan Kaufmann Publishers Inc., 580-588.