Accurate and novel recommendations: An algorithm based on popularity forecasting

AMIN JAVARI, Sharif University of Technology
MAHDI JALILI, Sharif University of Technology

Recommender systems are in the center of network science and becoming increasingly important in individual businesses for providing efficient personalized services and products to users. The focus of previous research in the field of recommendation systems was on improving the precision of the system through designing more accurate recommendation lists. Recently, the community has been paying attention to diversity and novelty of recommendation list as key characteristics of modern recommender systems. In many cases, novelty and precision do not go in the same direction and the accuracy-novelty dilemma is one of the challenging problems in recommender systems, which needs efforts in making a trade-off between them.

In this work, we propose an algorithm for providing novel and accurate recommendation to users. We consider the standard definition of accuracy and an effective self-information based measure to assess novelty of the recommendation list. The proposed algorithm is based on item popularity, which is defined as the number of votes they receive in a certain time interval. Wavelet transform is used for analyzing popularity time series and forecasting their trend in future time steps. We introduce two filtering algorithms based on the information extracted from analyzing popularity time series of the items. Popularity-based filtering algorithm, gives higher chance to items which are predicted to be popular in future time steps. The other algorithm, denoted as novelty and population based filtering algorithm, is to move towards items with low popularity in past time steps that are predicted to become popular in the future. The introduced filters can be applied as adds-on to any recommendation algorithm. In this paper, we use the proposed algorithms to improve the performance of classic recommenders including item-based collaborative filtering and Markov-based recommender systems. The experiments show that the algorithms could significantly improve both the accuracy and effective novelty of the classic recommenders.

Categories and Subject Descriptors: **G.2.2 [Graph Theory]: Graph algorithms; H.3.3 [Information Storage and Retrieval]**: Information Filtering, Information Search and Retrieval

General Terms: Algorithms; Experimentation; Performance

Additional Key Words and Phrases: Item popularity time series, time aware recommendation systems, collaborative filtering, Item popularity forecasting

ACM Reference Format:

Amin Javari and Mahdi Jalili, 2013. Accurate and novel recommendations: An algorithm based on popularity forecasting. *ACM Transaction on Intelligent Systems and Technology*.

Author's addresses: A. Javari (javari@ce.sharif.edu) and M. Jalili (mjalili@sharif.edu), Department of Computer Engineering, Sharif University of Technology, Tehran, Iran.

DOI:http://dx.doi.org/10.1145/0000000.0000000

1. INTRODUCTION

We have witnessed tremendous progress in mining and analysis of complex networks in the last decade. There has been lots of efforts in understanding structure and dynamics of social networks, which has been shown to be different from other networked structures [Newman and Park 2003]. Since the appearance of WWW and online networks, recommendation systems have become an important research topic in both industry and academic sectors [Adomavicius and Tuzhilin 2005; Deshpande and Karypis 2004; Resnick and Varian 1997]. Recommender systems have enormous practical applications that help users to receive personalized recommendations and choose the best matches for their needs.

Methods used in recommendation systems are usually categorized into three main classes: collaborative, content-based and hybrid recommendation algorithms [Adomavicius and Tuzhilin 2005]. While in content-based recommendation algorithms, the most similar items to those already rated by a particular user is recommended to that user [Basu et al. 1998; Pazzani and Billsus 2007], collaborative recommendation (or collaborative filtering) algorithms recommend the items to a user based on the items rated by those with similar preferences and tastes with that user [Konstan et al. 1997; Sarwar et al. 2001]. There are also hybrid methods where the recommendation list is built based on content-based and collaborative approaches [Balabanović 1997; Burke 2002]. The recommendation algorithms are often based on constructing bipartite networks of users and items. These networks are used to find the connection structure between the users in the one-mode projected network of users and those of items in the one-mode projected networks of items [Cattuto et al. 2007; Maslov and Zhang 2001; Shang et al. 2010; Zhou et al. 2010; Zhou et al. 2007].

Most of previous methods have not considered contextual information in their recommendation algorithms. It has been shown that by employing such information, one can build better recommender systems [Baltrunas and Ricci 2013; Shin et al. 2009]. The contextual information describes the context (e.g. time, location or mood) in which the users interact with the system [Adomavicius and Tuzhilin 2011]. Among different resources for contextual information, time is one of the most informative and important domains [Campos et al. 2013; Xiang and Yang 2009] which can be easily collected. It has been shown that users' preferences evolve over time and time-aware models have been proposed to track this issue in order to increase quality of recommendations [Baltrunas and Ricci 2013; Ricci and Shapira 2011]. Koren proposed a method based on Matrix factorization in which vectors of latent factor are generated in each time step [Koren 2010]. Markov-based recommender systems can also be counted as time-aware systems since they employ time domain to extract order of user ratings [Javari and Jalili 2013; Shani et al. 2002]. Xiong et al. proposed another time-aware recommender system based on incorporation of Bayesian probabilistic model and information obtained from time domain [Xiong et al. 2010]. In their model, the time has been incorporated as a distinct feature vector associated with those of users and items.

Although these time-aware models have been shown to improve performance of classic recommenders in terms of precision, novelty of recommendations has not been considered in them. The users would like to be recommended a list of diverse and novel items to earn more information from recommended lists. Indeed, an important issue affecting the recommendation results is popularity of the items [Steck 2011]. Most of the collaborative filtering algorithms that are designed to provide accurate results often recommend popular items, while users might be interested in non-popular and novel ones. Furthermore, the number of popular items is often much less than others. This usually causes similar recommendations for different users, which is not desired in general. The recommendations should be diverse and novel to satisfy users' willing [Shani and Gunawardana 2011; Zhou, Kuscsik, Liu, Medo, Wakeling and Zhang 2010].

Novelty and diversity in recommendation systems has received attention in recent years and some algorithms have been proposed to provide recommendations with high novelty and diversity [Agrawal et al. 2009; Celma and Herrera 2008; Clarke et al. 2008; McGinty and Smyth 2003; Vargas and Castells 2011; Ziegle et al. 2005]. Novelty and accuracy do not often go in the same direction, which means if accuracy is improved, novelty often worsens, and vice versa. Zhou *et al* (2010) introduced a hybrid algorithm taking into account accuracy and novelty of recommendation and demonstrated that it partially solves the accuracy-diversity dilemma of recommender systems [Zhou, Kuscsik, Liu, Medo, Wakeling and Zhang 2010]. Adomavicious *et al* (2011) employed graph theoretic methods based on maximum bipartite matching to maximize aggregate recommendation diversities [Adomavicius and Kwon 2011]. However, most of the algorithms introduced to recommend novel and diverse lists, result in recommending the items that have little chance to be purchased by the target user (i.e., very low accuracy of recommendation) [McNee et al. 2006; Zhou, Kuscsik, Liu, Medo, Wakeling and Zhang 2010]. In other words, they sacrifice precision to achieve higher novelty and diversity. To the best of our knowledge, the contextual information of time-domain has not been previously employed in order to increase quality of recommendations in terms of both novelty and precision. In this work, we exploit dynamics of item popularity in the time-domain to provide novel yet accurate recommendations. We suggest that in each time step, there is a subset of items which has higher chance to receive attentions of the users in near future. By focusing our recommendations to this subset, we build algorithms supporting both novelty and precision of recommendations.

It is possible to predict items which have higher chance to be purchased by the users in the future through analyzing their popularity in the past time steps. Our proposed recommendation algorithm takes into account this issue to increase novelty and precision of recommendations. To this end, we introduce two filtering algorithms based on items' past and future popularity (i.e., the number of votes an item receives in a certain time interval). The future trends of popularity values are predicted through regression analysis and wavelet decomposition. Based on the information obtained from popularity prediction, the filtering algorithms select a set of candidate items from item space. These filters act as adds-on to classic recommendations algorithms such as collaborative filtering and its variants. In general, the proposed method for recommendation has two steps. In the first step, the filtering algorithm selects a subset of item set based on analyzing popularity time series of items. In the second phase, any recommendation algorithm can be applied to recommend personalized list of items to the target user from the selected subset of items.

The proposed recommenders pay special attention to unpopular items that are predicted to be popular in the future. Since such items are currently unpopular ones, recommending them to the users will result in high novelty. Furthermore, they are predicted to be popular in the future time steps, and thus, recommending them will provide high accuracy. We add the proposed filters to two classic recommender systems (markov-based recommendation and item-based collaborative filtering) and evaluate the performance of the recommendations in terms of precision and novelty on a random subset of huge Netflix dataset. Our key contributions in this manuscript are as follows:

- Employing time-domain to intensify performance of any recommendation algorithm in terms of both novelty and precision
- Proposing two filtering algorithms to restrict the number of candidate items for recommendation in each time step
- Predicting and analyzing main trends of items' popularity time series through a method based on wavelet transform

2. PROBLEM FORMULATION AND BACKGROUND

In this section we formulate the recommendation problem and briefly introduce two classic recommendation algorithms: memory-based collaborative filtering and Markov-based recommender.

2.1 Recommendation systems

A recommendation system aims at finding the best-matching items for a particular user. Recommender systems are often based on users-items interaction networks. A recommendation system consists of users and items where there are a number of ratings from users to items. They can be modeled as bipartite networks with users in one side and items in the other side. Let us denote items set as $I = [i_1, i_2,\ldots, i_n]$ and users set as $U = [u_1, u_2,\ldots, u_m]$; the network can be fully described by a bipartite $n \times m$ adjacency matrix $A = [a_{pq}]$, where a_{pq} takes a value if i_q is rated (used or collected) by u_p, and $a_{pq} = 0$ otherwise. The rating values can be positive or negative indicating like (trust) or dislike (distrust), respectively.

Recommendation task is similar to link prediction problem in networked structures [Liben-Nowelly and Kleinberg 2003; Lü and Zhou 2011]. In link prediction problems, particular (global and local) similarity measures are defined in order to infer the potential future links to come. The recommendation problem can be formally defined as follows. Consider a bipartite network of users U and items I, where there are links (or rates) from each user to some of the items. Many of the potential links between users and items are missing. For any user u_p, we would like to find a certain number of items such that u_p is likely to rate (or create a link) with these items. One can also sort the items (considering their predicted value) for each user and chose top-N ones.

2.2 Memory-based Collaborative filtering

Collaborative filtering (CF) and its variants are the most widely used algorithms in recommender systems [J. Ben Schafer 2007]. CF is rather simple to compute – as compared to other recommendation algorithms – which makes it suitable for practical applications where the number of users and items might go over millions. It is based on proper similarity measures between the users, i.e., users with similar preferences are likely to have similar tastes, and thus, will be treated similarly with CF algorithms. In other words, the users that are similarly rating the target items or the items that are rated in a similar way, are first identified, and then the items are recommended to the users based on these similarity measures. CF can be performed in either user-based (using users' similarity scores) or item-based (using items' similarity scores) fashions [Manos Papagelis 2005].

Item-based collaborative filtering predicts rates of a target user on a particular item i, based on the items that the user has rated and similarity of these items with item i. To this end, proper statistical methods are often used to associate a similarity value for each pair of items. To do that, the similarity between two items can be measured by computing correlation coefficient [Adomavicius and Tuzhilin 2005]. The Pearson similarity between items i_i and i_j is obtained as

$$S_{Pearson}(i,j)=\frac{\sum_{h=1}^{n}(R_{h,i}-\overline{R_i})(R_{h,j}-\overline{R_j})}{\sqrt{\sum_{h=1}^{n}(R_{h,i}-\overline{R_i})^2}\sqrt{\sum_{h=1}^{n}(R_{h,j}-\overline{R_j})^2}}, \tag{1}$$

where $R_{h,i}$ denotes the rating of user u_h on item i_i and $\overline{R_i}$ is the average rating on item i_i. n is the total number of the users in the system.

As the Pearson similarity coefficients between items are obtained, the weighted sum of item-based similarities is computed. Then, the prediction value of the rating of user x to item y, $P(x,y)$, is obtained as

$$P(x,y)=\overline{R_y}+\frac{\sum_{h=1}^{m}(R_{x,v}-\overline{R_v})S_{Pearson}(y,v)}{\sum_{h=1}^{m}\left|S_{Pearson}(y,v)\right|}, \tag{2}$$

where $R_{x,v}$ is the rating item v receives from user x and $\overline{R_v}$ is average rating of user v. $P(x,y)$ is used for recommending items to users, i.e., among the items that has not yet been rated by a particular user, N items with highest value are recommended to that user.

2.3 Markov-based recommender

Recommender systems based on Markov chain are model-based systems. Markov models use not only information on users' rates on items, but also information about the order in which the users rate the items. In order to use Markov models in recommendation problem, state space and state transition function should be defined. In recommendation problem, state of a user can be defined as the last vote given by the user. Considering this definition for the state, new rating of a user can be modeled as his/her transition from one state to another one. Let us consider the vector $S_u = <I_m, I_{m-1},...,I_1>$ as state of user u, which denotes the user's last-m rated items in a sequential manner. The transition function describes the probability that a target user that has rated items $I_m, I_{m-1},...,I_1$ will rate (or selects) item I_{m+1} in the next step. The transition function between the states based on the training data and maximum likelihood optimization method can be estimated as [Javari and Jalili 2013; Shani et al. 2006]

$$TF(<I_m,I_{m-1},...,I_1>,<I_{m+1},I_m,...,I_2>)=\frac{N(<I_{m+1},I_m,...,I_1>)}{N(<I_m,I_{m-1},...,I_1>)}, \tag{3}$$

where $N(<I_m,I_{m-1},\ldots,I_1>)$ indicates the number of users visiting the state $<I_m,I_{m-1},\ldots,I_1>$ in their rating sequences in the training dataset.

Once the transition functions are estimated, the recommendation can be performed for each user based on its last-m rated items. To this end, first, state of the target user is determined based on her/his last-m purchased items, and then the algorithm recommends items belonging to the states which have the highest transition probability from the state of the target user. As mentioned, the transition probability values can be extracted from the transition function. For example, let us consider that states of the target user is <I_m, I_{m-1} ,...,I_1>, i.e., her/his last-m purchased items are I_m, I_{m-1} ,...,I_1. According to the transition function, top-N states with the highest transition from S_u can be extracted. Suppose that <I_{m+1}, I_m ,...,I_1> is the state which has the highest transition probability from S_u; I_{m+1} becomes the first item to be included in the recommended list to user u.

3. REPUTATION-BASED FILTERS FOR RECOMMENDER SYSTEMS

Regarding accuracy and novelty of recommendations, it is desired to recommend items which have low popularity in the past time steps while predicted to become popular items in the future. Recommending unpopular items in the past times will provide a novel list to the users, and having items with high future popularity guarantees good accuracy for the algorithm. This is the main contributions of this work. We consider the temporal properties of votings made by users, study the evolution of items' popularity and predict the popular items in future times. To this end, (daily) ratings for the items are collected and a wavelet-based method is used for predicting future popularity trends (e.g., in 10 days). Items with lower popularity in future time steps will get less weight in the recommender system for recommendation. In other words, even-though an item was a popular one, if its popularity shows a decreasing trend, the system should not recommend that item. We would like to recommend the items that are likely to be popular in future time steps.

Information extracted from time series can be used to construct a filter to focus the recommendations to items with above characteristics. We introduce two types of filters in this work. In the first one, depending on the time the recommendation is going to be made, we identify a list of items that will be the most popular items in the future time steps. Then, all or some of the recommendation list is devoted to these (going to be) popular items. Here, we use classic collaborative filtering and Markov-based recommender to choose among these items. Any other algorithms such as variants of collaborative filtering can also be used. In the second model, not only the predicted popularity is considered, but also its previous values are taken into account. In order to improve the novelty of recommendation, among the items predicted to be popular, we give more weight to the items with less popularity in the past time intervals.

The algorithms are evaluated on Netflix dataset [James Bennett 2007]. This dataset have been frequently used as a benchmark in recommender systems [Cacheda et al. 2011] and consists of ratings to a number of movies on a scale 1 – 5, with 5 being excellent and 1 being terrible. The dataset includes information of the timings (year, month, and day) at which the ratings have been made and contains votes in years 2000 to 2005. In our experiments, we use a portion of huge Netflix dataset with 60000 users, 4000 items and 6600000 ratings from users to items.

3.1 Popularity of items as time series

Popularity of items and voting patterns of users over time could be influenced by temporal effects, which can be categorized into four groups: user preference shifting, time bias, user bias shifting and item bias shifting [Xiang and Yang 2009]. User preference shifting and time bias indicate changes in the taste of users and society, respectively. User bias shifting demonstrates pattern of changes in the average ratings made by a specific user in a time interval. For example, a user might be not optimistic at certain times, and on average, gives low ratings for the items. Item bias shifting indicates popularity changes of items in a time interval. For example, when a movie wins the academy award, its popularity is likely to increase for a certain time. In this manuscript, we use item bias shifting to take into account the changes observed in the popularity of the items and enhance the recommendation precision and novelty. Popularity of an item in a certain time interval (daily in this work) is defined by the number of votes it receives in that time interval.

Figure 1 shows an example of daily popularity values for three items selected from the dataset. It is often observed that items which have received many ratings show an increased popularity in a certain time interval after which their popularity starts to decrease (such as upper and middle panels in Figure 1). Indeed, this is a similar pattern that can be seen in popularity time series of different movies. There are also some items with small number of votes received in all times. In other words, these items have not been able to attract many attentions from users. Popularity of such items is low and does not undergo considerable increase or decrease over time. However, many other factors may influence popularity of items. For example, advertisement on a movie or its success in a festival may increase its popularity.

Time series of popularity values are not often smooth, which makes them to look noisy. Users might have some behaviors influencing the popularity levels of the items. For example, the number of votes to an item could be high in a certain day due to environmental reasons. However, in many cases, it is possible to extract a smooth trend out of such noisy time series. Here we use wavelet transforms in order to predict main popularity trends. Based on the predicted future popularity trends, we introduce two filtering algorithms in order to limit the number of candidate items for recommendation.

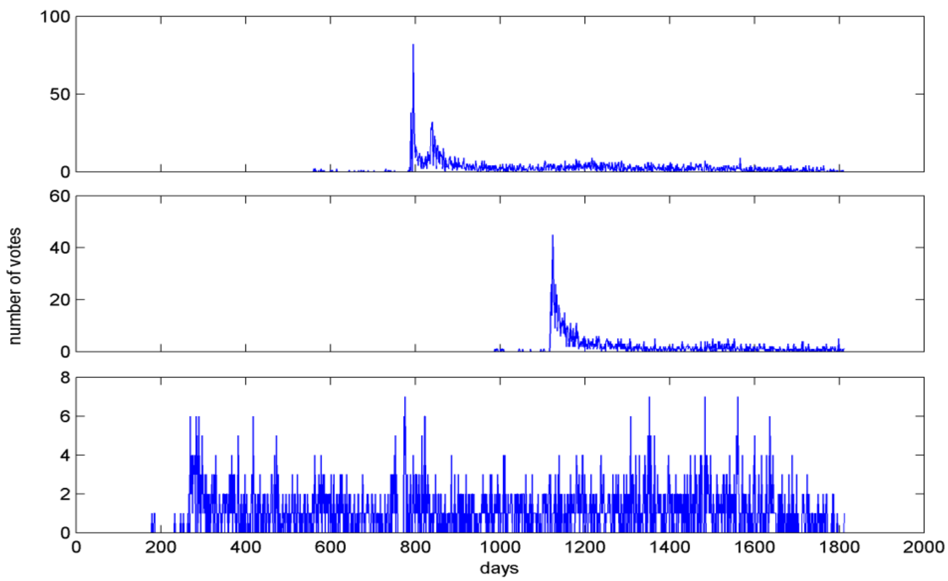

Figure 1: Popularity (i.e., number of votes) time series for three sample items (i.e., movies) from Netflix dataset during years 2000 to 2005.

3.2 Forecasting popularity time series based on wavelet decomposition

In this section, we introduce a method to predict the popularity trends using a wavelet transform. One of the most important factors in predicting the future values of time series is to have long enough history of the data. In recommendation systems, we do not often have enough samples to perform a reliable prediction based on classic machine learning methods such as neural networks. Among methods for time series forecasting, regression methods can be used to forecast time series with short histories [Hamilton 1994]. However, due to high complexity and nonlinearity of popularity time series, the predicted values by regression method are not often accurate. Indeed, these time series contains many high frequency noise-like patters (Figure 1), which make many prediction algorithms not to work well. Considering the complexity of popularity time series of items, we employ a method based on wavelet transform to do the prediction task [Soltani 2002]. The main idea behind signal prediction using wavelet transform is to decompose the signal into some elements with low complexity. Since the original signal is complex and cannot be predicted using conventional methods (e.g., regression), this approach first decomposes the signal into some elements with lower complexity. Then, the future values of the original signal can be predicted by combining the extended signals.

In general, time series prediction based on wavelet transforms is performed in three steps. In the first step, wavelet coefficients in multiple levels are obtained through wavelet decomposition. Then, the obtained signals go through signal extinction. Finally, the coefficients obtained in the signal extinction step are used in wavelet reconstruction stage. Wavelet decomposition of a time series results in signals with lower complexity as compared to the original ones. Wavelet decomposition can be performed in different levels. Number of levels in decomposition step depends on the complexity of the time series and can vary for different applications; generally, the higher the complexity of the time series, the higher the decomposition level. For example, a 2-level wavelet decomposition of a time series a results in two new time series: a_1 and a_2, where a_1 describes details of a and a_2 represents its slow dynamics. Similarly, by N-level decomposition of a time series, we will have N distinct time series ($a1$, $\ldots a_N$), where a_N and a_I represent the highest and the lowest dynamics of a, respectively. Each of these time series is less complex compared to the original time series, and hence more predictable. The next step is to separately predict each of these time series (resulted from decomposition of the original time series). Finally, the predicted time series are combined using wavelet reconstruction to obtain the predicted future values of the original time series. This method for time series forecasting has been previously used for prediction of oil prices [Yousefi et al. 2005] and stock market prices [Hsieh et al. 2011].

We use discrete wavelet transform (with Coefmann as mother wavelet) available in wavelet toolbox (MatLab package) for wavelet decomposition. We use 6-level wavelet decomposition. As mentioned, the number of decomposition levels depends on the complexity of the signal. In this work, we examined different decomposition levels and fixed the level such that the signal describing the lowest dynamics of the original time series does not contain any high frequency patterns. Since we are mainly interested in predicting the future popularity trends (and not the exact value), we use only the decomposed signal with the slowest dynamics for the prediction. Thus, to predict main trends of input signal, after decomposition in 6 levels we perform the extinction on a_6 signal (wavelets coefficients of level 6). Then, we reconstruct the signal based on the extended a_6 signal. In order to perform the extinction task and regression analysis, we use spline fit tools in MatLab. Figure 2, the upper panel, shows the popularity time series of a sample item and the middle panel represents the a_6 signal obtained from decomposition in 6 levels. By doing extinction on a_6 and reconstruction from this signal, we can obtain prediction for main trends of the input signal. Figure 2 (bottom panel), shows the predicted trends of the input signal based on reconstruction from extended a_6 signal. Indeed, by decomposing the signal into its slow dynamics, we filter out noise-like high frequency patterns from the signal. As it is seen, the reconstructed signal demonstrates the main trend of changes in the popularity with much less high frequency noise-like patterns as compared to the original signal (Figure 2; the upper panel). Predicted trends of popularity are used in the recommender systems.

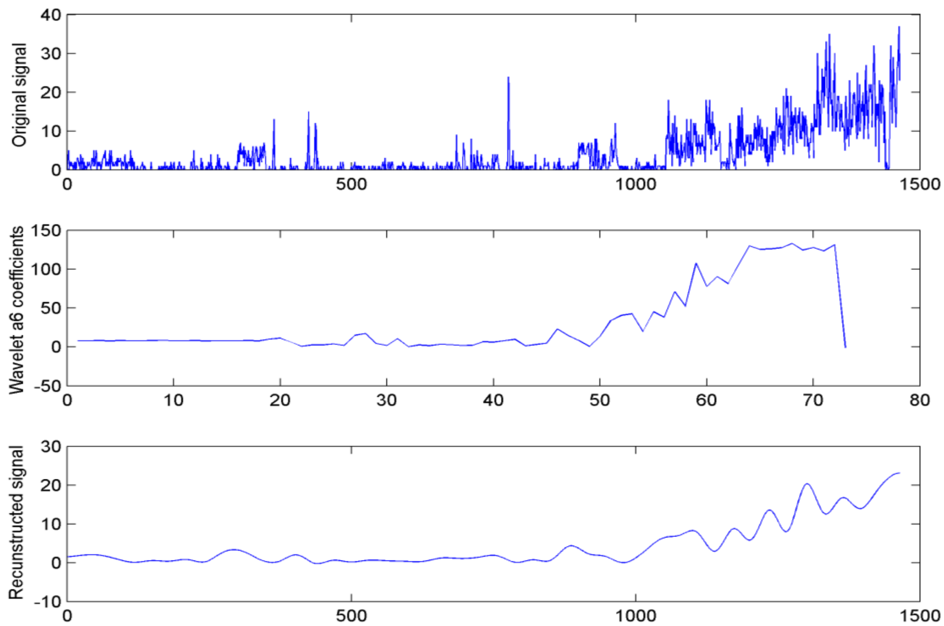

Figure 2: Popularity time series for a sample item (upper panel), decomposed wavelet coefficient at level 6 (middle panel), and reconstructed signal from this coefficient (bottom panel).

3.3 Item selection for top-N recommendation based on information extracted from popularity time series

In this work, we propose to improve the performance of recommender systems by focusing the recommendations to a subset of items in each time step. Selection of candidate items can be done by analyzing their popularity time series. Thus, the proposed recommendation algorithm has two steps. In the first step, a filtering is performed independent of users' preferences (based on only item popularity time series) to select M items, and in the second step of the algorithm, top-N appropriate items (among the items that have passed the first step) are recommended taking into account taste of users. Although any recommendation algorithm can be used to choose top-N from the list of M items, we use standard CF and markov-based algorithms in this paper. One can also run the algorithm in a way to have more freedom in choosing the final recommendation list. For example, P of N items can be chosen from the whole items and N–P from the top-M selected items.

Considering the popularity trends over time, we introduce two filtering algorithms for selection of candidate items. While in the first algorithm, we use only the predicted popularity values, in the second algorithm both predicted and past values are used. In the first algorithm, denoted as Popularity Score (PS) based filtering, a short term (e.g., 10-day) prediction is performed on the main popularity trends. Then, we determine a set of *M* items with the highest predicted popularity values. Indeed, using this filter, we aim to pay extra attention to items that are predicted to attract users in future time steps. As mentioned, slow dynamics of a time series is more stable than its high frequency dynamics. It is possible that an item becomes a popular one in a short period of time affected by some non-stable properties. For example, popularity of an item may increase in s short term period, in spite of its decreasing main trend. Thus, it is not wise to select the candidate items based on their high frequency dynamics. Indeed, one expects that items selected based on their main trends are more likely to stay popular for longer period of time. *M* items with the highest predicted popularities are selected based on Popularity Scores (*PS*) which can be defined as

$$PS_i(t) = APP_i(t, t+m)\,, \tag{4}$$

where $APP_i(t,t+m)$ is the average predicted popularity value for item i in m future time steps. According to this definition for PS value, the PS filter, selects top-M items that, in average, have the highest main trends in m future time steps.

In the second filtering algorithm, denoted as Popularity and Novelty Score (PNS) based filtering, we try to find the items with low popularity values in the past times showing increased popularity trend in the future. In this way, the items showing decreased popularity will be less likely to be recommended than those with increasing popularity trend. Let us define Popularity and Novelty Score (*PNS*) as

$$PNS_i(t) = PS_i(t) \times NS_i(t)\,. \tag{5}$$

Novelty Score (*NS*) for item i can be defined as

$$NS_i(t) = \frac{\log\left(\dfrac{HP(t)}{NV_i(t)}\right)}{\log\left(\dfrac{HP(t)}{LP(t)}\right)}\,, \tag{6}$$

where $NV_i(t)$ is the total number of votes to item i up to time step t. $HP(t)$ represents the total number of votes to the item with the highest popularity among the items up to time t and LP indicates the total number of votes to the item with the lowest popularity up to time t.

According to the definition of PNS, it depends on two factors: popularity of items in future time steps and their popularity in the past. The higher PS value gives more chance to items with higher popularity in the future and high NS value intensifies the chance of novel items to be selected as candidate ones. As the novelty and popularity of the item increases in future time steps, the value of *PNS* will also increase. The NS value introduced here represents a similar concept as self-information based novelty; as an item is more novel, it has higher NS value. After extraction of *PNS* value for all items, PNS filter selects top-M items as candidates with the highest *PNS* value. Let us denote the CF algorithm with PS and PNS filters as PS-CF and PNS-CF, respectively. Also we denote conjugation of Markov recommender with PS and PNS filters as PS-MR and PNS-MR, respectively.

3.4 Evaluating performance of the algorithms

The main idea behind the proposed algorithms is to monitor the item popularity and recommend those showing increasing trend in their popularity. In this section, we evaluate the algorithms in terms of their precision in recommended items as well as novelty of recommendation. Since in applying filters, we first filter out the items to identify top-M popular items, and then, run a recommendation algorithm such as CF, the computations is less than the case when the recommendation algorithm is performed on the whole items. The filtering step includes only wavelet decomposition and reconstruction, which does not need heavy computations and is scalable for large systems. Let us consider the time steps on a daily fashion (same as the Netflix dataset used in this work). Since selection of the candidate items are performed daily, wavelet decomposition and reconstruction on all items is necessary. Computational complexity of discrete wavelet decomposition and reconstruction on each item is $O(L)$, where L is length of the popularity time series. Thus, applying the proposed filters will cost $O(LN)$, which is performed offline and daily. Indeed, this part of the algorithm will not increase real-time computational complexity of the recommendation process. As M candidate items are selected, the recommendation algorithms will use only these items in their recommendations, resulting in much lower computational complexity as compared to conventional methods. For example, real-time computational complexity of item-based CF and Markov recommender are $O(Nd)$, where d is the average degree of item (considering that similarity extraction in item-based CF and estimation of transition function can be done in offline manner). By restricting the number of candidate items to M, the complexity becomes $O(Md)$ and we have $M << N$.

The main objective of recommender systems is to recommend those items to users that are likely to be used (or voted) by them while giving them the most satisfaction. On one hand, since we consider the popularity of the items and recommend those with high levels of popularity in future time steps, it is somehow expected that such items are likely to be used by the users in future times. On the other hand, the CF filtering or Markov algorithm used for choosing the final recommendation list among the most popular items, take into account the individual taste of the users and assigns proper items to any of them. Applying PS-based filter on CF or Markov recommender puts the stress on improving the precision of the system and does not take into account the novelty of the recommendation list. Let us consider an item that has received many positive ratings and was a popular item in the last times; however, its current popularity (and also the forecasted value) shows a decreasing trend. Applying CF or Markov recommender alone will recommend such an item to the users; however, this item is less likely to be used by the users in future times (which is predicted through analysis of its time series). Furthermore, this item is a well-known one and the users are likely to be aware of that; recommending such an item does not provide novel options for the users. In other words, as the items are more popular, recommendation of them for the target user is less novel. In PNS-based filter, the items get higher chance to be recommended, if the prediction shows their votes will increase in future times; although their votes might not be high in the time the recommendation is made. Furthermore, this algorithm does not recommend the items that are not popular at the present time, while being popular in the past times. Such items have low novelty, and thus, PNS-based filters targets increasing novelty in the recommendation list.

Using these filters, we first select some candidate items in a non-personalized manner, and then limit ourselves to these items which have high chance to be attended by the users in near future. However, according to the coverage principle, we are interested in recommendation methods which can recommend large proportion of items in their lists [Ge et al. 2010; Shani and Gunawardana 2011]. Item space coverage based on Shannon Entropy can be defined as

$$C(N)=\frac{\sum_{u\in u_{TestSet}}\left(-\sum_{i=1}^{N}p(i)\log p(i))\right)}{\left|u_{TestSet}\right|}, \tag{7}$$

where p_i is the percentage of recommendation lists that contains item i.

As it can be seen from the definition, the algorithm has the best coverage that gives the same chance to all items to appear in the recommendation lists. In our proposed method, we propose to modify the chance of items to be recommended based on their popularity time series. Thus, our approach is in contrast with traditional coverage principle. Since item popularity is a dynamic phenomenon, in each time interval, some items lose their popularity and are unlikely to be purchased in the future time steps. Obviously, recommending such items increases the coverage; however, they often do not satisfy the users. Indeed, one of the major problems for diversification algorithms is that they recommend items with such characteristic in order to provide diverse recommendation lists. Alternatively, one can only consider items that are likely to be purchased in the future time steps and consider coverage of these items, i.e., partial coverage. According to the partial coverage, we are interested in recommendation methods which can recommend large proportion of items that are likely to be popular in the future time steps.

4. RESULTS

In real-world applications of recommender systems, only past data are used in order to recommend proper items to target users. To evaluate the proposed algorithms, we first select five test points uniformly distributed on the data time interval, and then, take the ratings up to the that time as training data and those after that point up to the next 20 days as test data. In this way, we generate five different test and train datasets. Based on train data, we recommend each test user a list of 10 items (top-10 recommendation).

The performance of the proposed methods (PS-CF, PNS-CF, PS-MR and PNS-MR) is compared with four other algorithms including item-based CF, Markov recommender, time decayed item-based CF and popularity-based recommendation. Since time decay algorithm is one of the time-aware recommenders which improves precision and novelty of recommendations, we consider it in our experiments. The time decayed strategy is such that the ratings are weighted based on the time they have been registered; the older is a rating, the less its significance [Ding and Li 2005]. Item-based time decayed collaborative filtering can be defined as

$$P(x,y)=\overline{R_y}+\frac{\sum_{h=1}^{m}(R_{x,v}-\overline{R_v})S_{Pearson}(y,v)e^{-\lambda(t-t(R_{x,v}))}}{\sum_{h=1}^{m}\left|S_{Pearson}(y,v)\right|}, \tag{8}$$

where t represents time of recommendation and $t(R_{x,v})$ is the time user x rates on item v and λ is a constant value. In our experiments we set $\lambda = 200$.

In popularity-based recommendation algorithms, the most popular items that have not yet been rated by the user are recommended to him/her.

4.1 Evaluation method

Various metrics have been proposed for performance evaluation of recommender systems. In this work, we study the performance in terms of both precision and novelty. In order to assess how novel are the recommendation list, we use self-information based novelty measure.

4.1.1 Precision

We use P@N for assessing how precise are the prediction results [Ding and Li 2005]. P@N for a user indicates to how much extent the recommendation list (of N items) coincides with the real user's taste, as in the test data. P@N metric can be calculated as

$$P@N = \frac{\sum_{u \in u_{Testset}} P@N(\Gamma_N(u))}{|u_{Testset}|}, \tag{9}$$

where P@$N(\Gamma_N(u))$ indicates the P@N of recommended list of size N for user u and $|u_{TestSet}|$ indicates the whole number of users in test dataset.

4.1.2 Novelty and Effective Novelty

In general, the novel items for a user are those that the user is not yet aware of them. Indeed, novelty of a recommendation list indicates the amount of information the list provides for the target user. The self-information based novelty is a metric to evaluate novelty and works based on this idea that as an item is popular, the target user is more probable to know about the existence of the item. It is computed as [Zhou, Kuscsik, Liu, Medo, Wakeling and Zhang 2010]

$$NOV_u(\Gamma_N(u)) = \frac{\sum_{i \in \Gamma_N(u)} \log_2\left(\frac{|u|}{Rels_i}\right)}{N}, \tag{10}$$

where $\Gamma_N(u)$ indicates the list of recommended items to user u, N is the size of recommendation list, and $Rels_i$ is the users that have used (or rated) item i, – which is in $\Gamma_N(u)$.

Due to some limitations in self-information based novelty, we use a special form of relevance-aware metric [Vargas and Castells 2011] to evaluate novelty of recommendations. In the following, we describe some scenarios in which self-information based novelty is misleading. In order to evaluate the performance of self-information based novelty measure, let us assume that the system randomly recommends a number of items with low popularity in the past times. For such a recommendation list, the accuracy will be very low, while the novelty will be high. However, such a novel recommendation is not functional in the view of the users, since it does not have any relation with users' preference. A good recommendation system is the one providing novel items for the users, while also satisfying their taste. Let us consider top-5 recommendation system with precision as P and novelty as N. We can replace one of the five items in the list with an item with high novelty (although this item might not be in the preference list of the user). This will result in dramatic increase in N, while P will not change significantly. However, this system is not favored for the users, since the recommendation list includes an irrelevant item. In order to explain this issue in more details, let us consider CF and PS-CF algorithms. CF results in better self-information based novelty than PS-CF algorithm. Gray boxes in the table 1 show the items that are in the users' preference (i.e., those which have received ratings from the users). CF has larger average novelty (averaged over the items) than PS-CF algorithm; however, this is mainly due to the items that are not in the preference of the users. One may argue that from view point of a user, the information he/she gathers from an irrelevant item is almost near to zero. Indeed, in reality, the information obtained from a list could significantly vary between the users. However, according to the self-information based novelty, the amount of information that different users obtain from a recommendation list is similar to each other.

Considering the above facts, one may argue that novelty of an item is more valuable when it is more relevant to the target users' taste. However, according to equation (10), self-information based novelty obtains the novelty using the whole recommendation list without considering relevancy of the items to the target user. In this work, to consider relevancy of items to the target user, we use NOV_{RA}, as a relevance-aware metric scheme for novelty [Vargas and Castells 2011], as

$$NOV_{RA} = C \sum_{i_k \in \Gamma_N(u)} disc(k)\, p(rel \mid i_k, u)\, nov(i_k)\,, \tag{11}$$

where C is a constant for normalization against biases, i_k is the item at position k of the recommendation list $\Gamma_N(u)$ and $disc(k)$ is a decreasing rank discounting function. According to this function, novelty of items is more valuable as they are ranked in the top of the recommended list. $p(rel \mid i_k,u)$ is the probability of the target user to see item i_k when she/he is browsing the list and $nov(i_k)$ is the probability that item i_k has not yet been observed.

In this manuscript, we consider no rank discount (i.e., $disc(k) = 1$), and set $p(rel \mid i_{k,u}) = 1$ for the items to which the target user gives positive ratings in the test dataset and $p(rel \mid i_{k,u}) = 0$ for those that have not received positive ratings. Also, we set $C = \frac{1}{|\Gamma N(u)|}$, and similar to the self-information based novelty (defined in equation (10)), we define $nov(i_k)$ based on popularity of i_k, as

$$nov(i_k) = \log_2\left(\frac{|u|}{Rels_{i_k}}\right). \tag{12}$$

In the rest of the manuscript, we refer to the above configurations for NOV_{RA} as effective self-information based novelty (ESIBN) metric. Consequently ESIBN can be defined as:

$$ESIBN = \frac{\sum_{i \in \Gamma'_N(u)} \log_2\left(\frac{|u|}{Rels_i}\right)}{|\Gamma_N(u)|}\,, \tag{13}$$

where $\Gamma'_N(u)$ is a subset of $\Gamma_N(u)$ that have received positive ratings from user u in the test dataset. In this way, those items that are not in preference of the users will not affect the novelty.

Table 1: Novelty of four recommended items to three sample users by CF and PS-CF algorithms.

		Item 1	Item 2	Item 3	Item 4	Average novelty for the whole list	Average novelty for the items rated by the user
CF	User 1	3.2	1.2	1.8	2.3	2.12	1.8
	User 2	4.1	3.1	1.4	2.1	2.67	1.4
	User 3	2.1	1.6	3.8	1.2	2.17	1.4
PS-CF	User 1	1.8	2.1	1.9	1.6	1.85	1.7
	User 2	2.3	1.9	2.2	1.3	1.97	1.6
	User 3	2	1.9	0.9	1.3	1.52	1.4

4.2 Precision and novelty as a function of basket size in PS- and PNS-based methods

One of the parameters which may largely influence performance of PS- and PNS-based recommenders is the basket size M in the filtering phase (i.e., identifying top-M popular items). We assess precision and novelty of PS- and PNS-based algorithms as a function of basket size on fifth test point of the Netflix dataset. We select the fifth test point since the training set of this test point is the largest one among others, and thus, the results obtained from this test point are more reliable as compared to the other test points.

As basket size decreases, the computational load of recommender systems decreases; however, the performance in terms of precision and novelty may have different behaviors. Figures 3 and 4, respectively, show precision and Self-Information based Novelty (SIBN) of PS- and PNS-based recommendation algorithms as a function of the basket size. As the basket size increases, the precision of PS-based methods decreases (PS-MR and PS-CF) and their SIBN increases. Indeed, by increasing the basket size, the items that are predicted to have low popularity in future time steps are likely to appear in recommendation lists. Although it results in more user-specific recommendations, it decreases precision of recommendations. For many of the items, their popularity does not change swiftly, i.e., many items predicted to be popular in near future have also been popular in the past times. Thus, as the recommendation lists include items with low predicted popularity, SIBN of the lists increases. Figure 5 shows that ESIBN of PS-based methods is not significantly influenced by changing the basket size. In contrast to PS-based algorithms, PNS-based algorithms do not show dramatic change in precision, SIBN, and ESIBN as the basket size varies. Since in PNS-based algorithms the selection of candidate items depends on the predicted popularity and popularity of items in past time steps, one may not expect any significant change in precision or novelty by increasing the basket size.

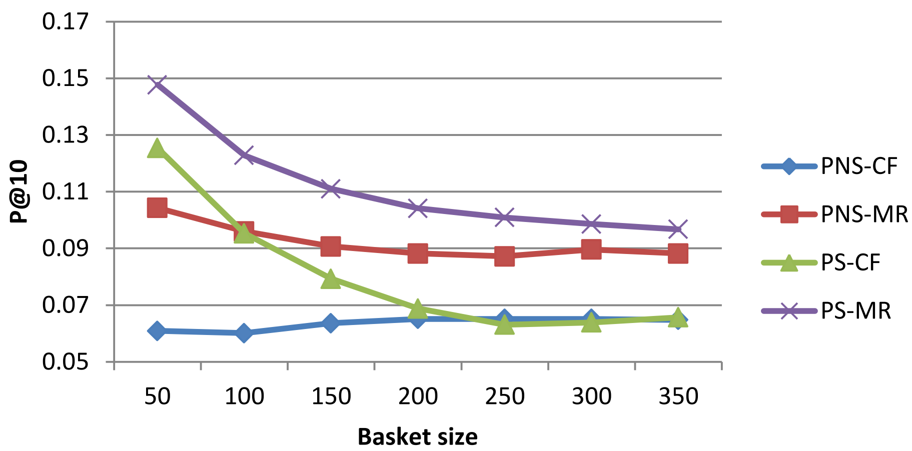

Figure 3: Precision of four recommendation algorithm (Popularity and Novelty Score based filter on Collaborative Filtering (PNS-CF), Popularity and Novelty Score based filter on Markov Recommender (PNS-MR), Popularity Score based filter on Collaborative filtering (PS-CF), Popularity Score based filter on Markov Recommender (PS-MR)) as a function of basket size.

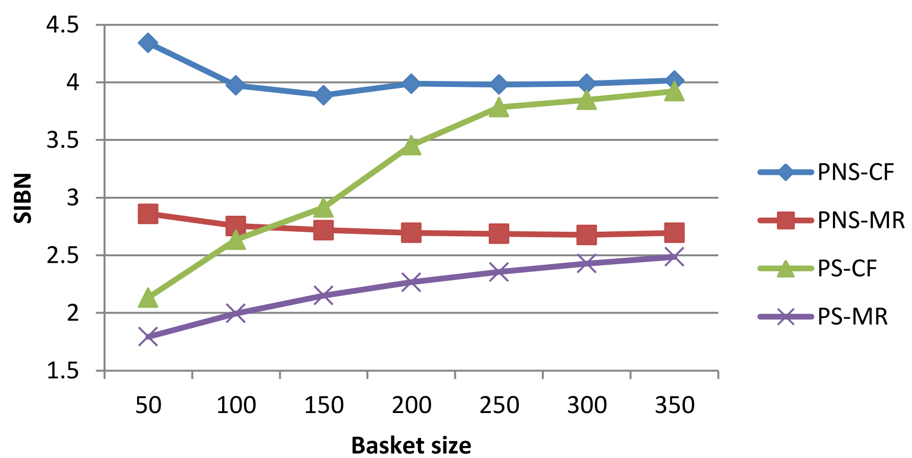

Figure 4: Self-Information Based Novelty (SIBN) of the recommendation algorithms. Other designations are as Figure 3.

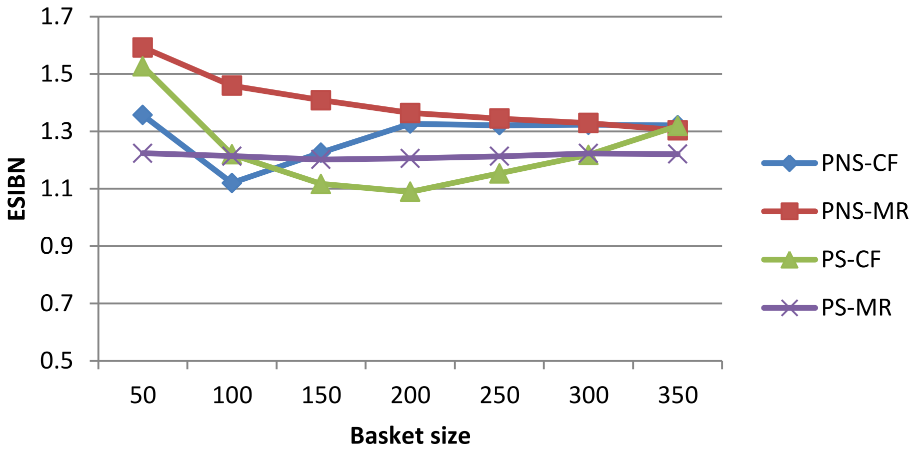

Figure 5: Effective Self-Information Based Novelty (ESIBN) of the recommendation algorithms. Other designations are as Figure 3.

We do not consider basket size lower than 50 items in our experiments. As discussed, the filtering step selects candidate items based on only their popularity time series. Thus, it does not consider the target user in its selections. In other words, the first step of the algorithm is performed in a non-personalized manner. We may adjust the role of the filtering step in our recommendations by modifying the basket size. By decreasing the basket size, the filtering algorithm will be more influential on recommendations. However, if we use small number of candidate items, the recommendation algorithm in the second step will lose its proper functionality. Small number of candidate items forces the recommendation algorithm to recommend almost the same list to different users. In other words, small basket size decreases the personalization in recommendations. The optimal basket size can be determined based on factors such as size of the item space, the number of items in recommendation lists and distribution of items predicted popularity. One may also determine the basket size in a dynamic fashion. For example, the filtering algorithm can do the selection process in a way that candidate items receive at least %80 of users' attention in future time steps. In this work, we do not optimize the basket size and fix it as 50 for both PNS- and PS-based algorithms.

4.3 Comparing the proposed algorithms with baseline recommendation algorithms

Figure 6 shows the precision (P@10) of eight recommendation algorithms in five test points. As it is seen, PS-CF and PS-MR has the best precision among these algorithms. Surprisingly, their precision is better than popularity-based algorithm, which is known to provide a recommendation list with high precision among classic recommender systems [Soto 2011]. PS-CF and PS-MR not only have better precision than CF and MR, but also they have less computational complexity as compared to CF and MR. PNS-CF and PNS-MR also show better precision as compared to CF and MR, respectively. It is worth mentioning that classic item-based CF and time decay algorithm – as one of the time aware recommendation algorithms – has almost the same precision.

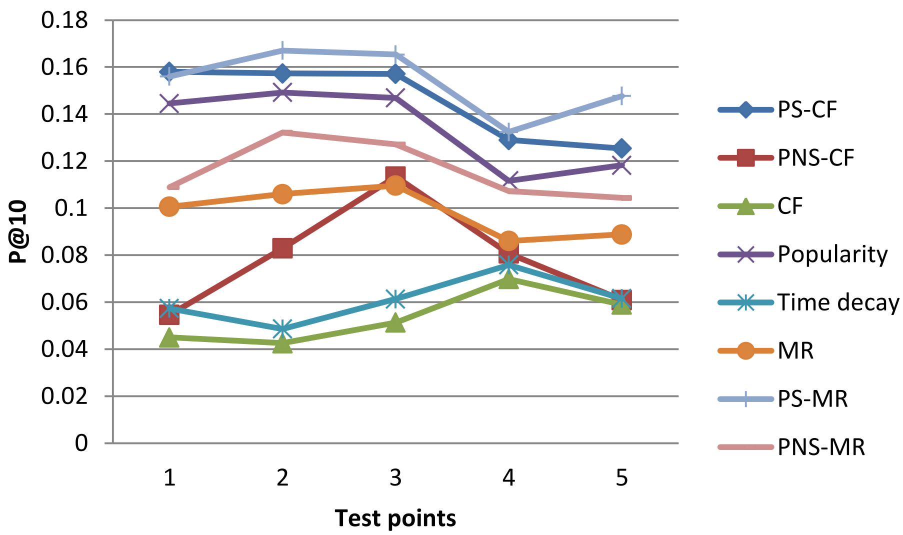

Figure 6: Precision of eight recommendation algorithms (PNS-CF, PNS-MR, PS-CF, PS-MR, Collaborative Filtering (CF), Popularity-based recommender (Popularity), Markov Recommender (MR) and time-decay collaborative filtering (Time Decay)) on five test points of Netflix dataset.

We compare the novelty of the recommendation list, as denoted by SIBN measure and the results are shown in Figure 7. Time decay CF results in the best performance in terms of novelty, followed by CF and PNS-based algorithms. However, as discussed, results obtained through SIBN might be misleading. Figure 8 shows the ESIBN of the algorithms. It is seen that PNS-based recommendation methods have the highest ESIBN in many of the test points followed by PS-based methods. In general, the proposed methods have higher ESIBN as compared to other classic recommender systems, while PNS-based recommenders have better ESIBN than PS-based methods. As we discussed, PS filter aims at maximizing the precision; however, PNS filter also takes into account novelty of recommendations in its filtering algorithm. Thus, we expect higher ESIBN for PNS-based methods compared to PS-based ones. The proposed algorithms outperform time decay algorithm, which results in the highest SIBN. Indeed, as discussed, although item-based CF and time decay algorithms generate recommendation lists with high SIBN, in many cases the novel items they include in their recommendations are unlikely to be purchased by the users. In other words, the novelty of recommended items that are purchased by the users, is higher in the proposed algorithms than CF and time decay methods. Apparently, we are interested in increasing novelty of items which are likely to be purchased by the users. Thus, having better precision and ESIBN, the proposed PS- and PNS-based recommenders outperform classic recommender systems. In order to provide a better view of comparison we integrated the results of the algorithms on different test points. Table 2, presents the average value of P@10, ESIBN and SIBN obtained from 5 test points. As it can be seen from the table, PS- and PNS- based algorithms in general provide better precision and ESIBN compare to the baseline methods.

Table 2: Average Precision, Effective Self Information based Novelty (ESIBN) and Self Information based Novelty (SIBN) of eight recommendation algorithms on five test points

	CF	TD	PS-CF	PNS-CF	MR	PS-MR	PNS-MR	Popularity
Precision	0.0535	0.0609	0.1453	0.0785	0.0982	0.1536	0.1159	0.1340
ESIBN	1.4883	1.6999	1.9287	1.8895	1.4482	1.5354	1.9149	0.7488
SIBN	5.1753	5.6325	2.6133	4.6094	3.1385	2.4425	3.1536	1.2589

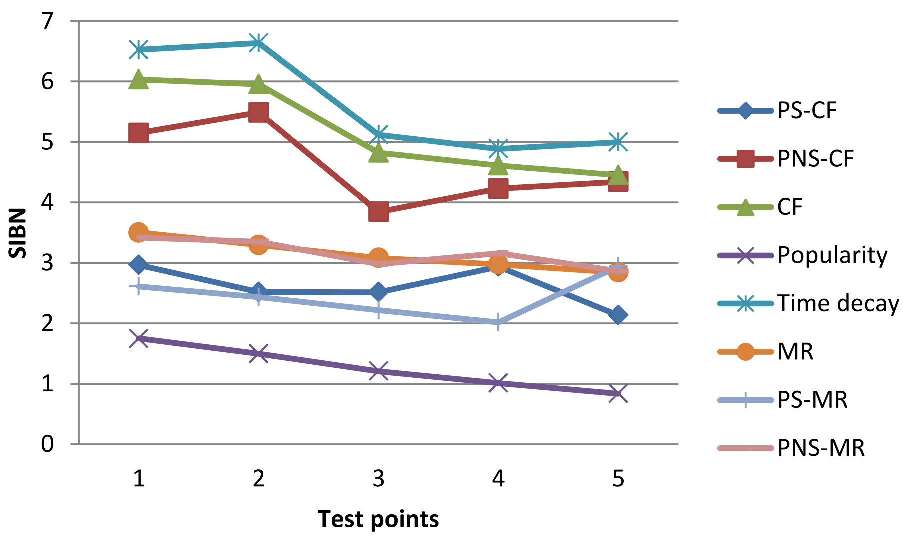

Figure 7: Self-Information Based Novelty (SIBN) of eight recommendation algorithms. Other designations are as Figure 6.

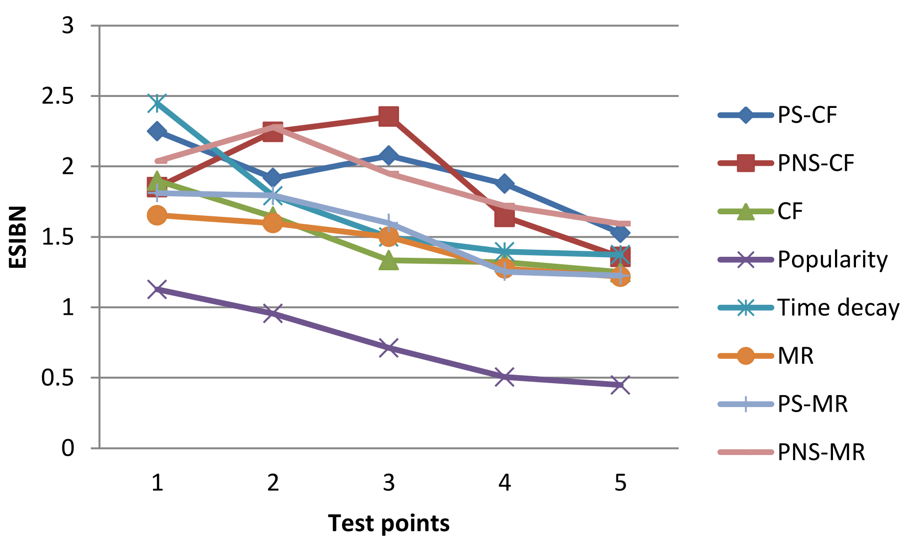

Figure 8: Effective Self-Information Based Novelty (ESIBN) of eight recommendation algorithms. Other designations are as Figure 6.

5. CONCLUSION

Designing efficient recommender systems is an emerging field for individual users and businesses to provide personalized recommendations. Recommender systems use past information of users' preferences in order to predict probable future interests, and thus, providing a list of recommendation items to the users. The most important issue addressed in previous works on recommender systems is precision of the recommendation list, i.e., providing a list with the best accuracy on the test dataset. To this end, methods based on similarity between items and users are often employed. However, this usually makes the recommended items to be too popular, while the users prefer to be recommended diverse and novel items. Diversity and novelty of recommendation list is a new challenging issue in this field that has recently attracted attention.

In this work, considering Netflix dataset, we proposed two filtering algorithms based on the popularity of the items, which could be used as adds-on to classic recommender algorithms. Slow dynamics of the popularity time series were forecasted using wavelet decomposition method. The proposed filtering algorithms were applied on item-based collaborative filtering and Markov-based recommender systems. First, a list of items that are predicted to be popular in future time steps were created, and then, standard collaborative filtering or Markov recommender were used to personalize the items among this list. This decreases the computational load of the recommendation as compared to collaborative filtering or Markov algorithms. We also proposed another filtering algorithm by not recommending the items with high popularity values in the past time that show decreased popularity. This helped in enhancing the novelty of the recommendation lists. The results showed significant outperformance of the proposed algorithms as compared to four classic recommendation systems. The proposed methods outperformed classic recommenders not only in terms of precision, but also in terms of effective self-information novelty. It is worth mentioning that the proposed filtering stage can be merged with any recommendation systems without significantly increasing the computational load.

In the future, one might consider more structural data of the networks to improve both precision and novelty. For example, considering community structure in one-mode projected networks within users or items may help in designing algorithms with better precision. Also, there is a need for developing a combined metric measuring precision and diversity in the recommendation list.

REFERENCES

ADOMAVICIUS, G. AND KWON, Y. 2011. Maximizing Aggregate Recommendation Diversity: A Graph-Theoretic Approach. In *Proceedings of Workshop on Novelty and Diversity in Recommender Systems*, 3-10.

ADOMAVICIUS, G. AND TUZHILIN, A. 2005. Toward the next generation of recommender systems: a survey of the state-of-the-art and possible extensions. *IEEE Transactions on Audio Electroacoustics Knowledge and Data Engineering 17*, 734-749.

ADOMAVICIUS, G. AND TUZHILIN, A. 2011. Context-aware recommender systems. In *Recommender systems handbook* Springer, 217-253.

AGRAWAL, R., GOLLAPUDI, S., HALVERSON, A. AND IEONG, S. 2009. Diversifying Search Results. In *Web Search and Data Mining* ACM, Barcelona, Spain, 5-14.

BALABANOVIĆ, M. 1997. Fab: content-based, collaborative recommendation. *Communications of the ACM 40*, 66-72.

BALTRUNAS, L. AND RICCI, F. 2013. Experimental evaluation of context-dependent collaborative filtering using item splitting. *User Modeling and User-Adapted Interaction*, 1-28.

BASU, C., HIRSH, H. AND COHEN, W. 1998. Recommendation as classification: Using social and content-based information in recommendation. In *Proceedings of the National Conference on Artificial Intelligence*1998.

BURKE, R. 2002. Hybrid recommender systems: survey and experiments. *User Modeling and User-Adaptive Interaction 12*, 331-370.

CACHEDA, F., CARNEIRO, V., FERNÁNDEZ, D. AND FORMOSO, V. 2011. Comparison of collaborative filtering algorithms: Limitations of current techniques and proposals for scalable, high-performance recommender systems. *ACM Transactions on the Web (TWEB) 5*, 2.

CAMPOS, P.G., DÍEZ, F. AND CANTADOR, I. 2013. Time-aware recommender systems: a comprehensive survey and analysis of existing evaluation protocols. *User Modeling and User-Adapted Interaction*, 1-53.

CATTUTO, C., LORETO, V. AND PIETRONERO, L. 2007. Semiotic dynamics and collaborative tagging. *Proceedings of the National Academy of Science of the United States of America 104*, 1461-1464.

CELMA, Ò. AND HERRERA, P. 2008. A new approach to evaluating novel recommendations. In *Recommendation Systems* ACM, Lausanne, Switzerland, 179-186.

CLARKE, C.L.A., KOLLA, M., CORMACK, G.V., VECHTOMOVA, O., ASHKAN, A., BÜTTCHER, S. AND MACKINNON, I. 2008. Novelty and diversity in information retrieval evaluation. In *SIG Information Retrieval* ACM, Singapore, 659-666.

DESHPANDE, M. AND KARYPIS, G. 2004. Item-based top-N recommendation algorithms. *ACM Transactions on Information Systems 22*.

DING, Y. AND LI, X. 2005. Time weight collaborative filtering. In *International Conference on Information and Knowledge Management.* ACM, Bremen, Germany, 485-492.

GE, M., DELGADO-BATTENFELD, C. AND JANNACH, D. 2010. Beyond accuracy: evaluating recommender systems by coverage and serendipity. In *Proceedings of the fourth ACM conference on Recommender systems* ACM, 257-260.

HAMILTON, J.D. 1994. *Time series analysis*. Cambridge Univ Press.

HSIEH, T.-J., HSIAO, H.-F. AND YEH, W.-C. 2011. Forecasting stock markets using wavelet transforms and recurrent neural networks: an integrated system based on artificial bee colony algorithm. *Applied soft computing 11*, 2510-2525.

J. BEN SCHAFER, D.F., JON HERLOCKER AND SHILAD SEN 2007. Collaborative filtenng recommender systems. *LNCS: Lecture Notes In Computer Science 4321*, 291-324.

JAMES BENNETT, C.E., BING LIU, PADHRAIC SMYTH AND DOMONKOS TIKK 2007. KDD Cup and workshop 2007. *ACM SIGKDD Explorations Newsletter 9*.

JAVARI, A. AND JALILI, M. 2014. A probabilistic model to resolve diversity-accuracy challenge of recommendation systems. *Data Mining and Knowledge Discovery,to appear*.

KONSTAN, J., MILLER, B., MALTZ, D., HERLOCKER, J., GORDON, L. AND RIEDL, J. 1997. GroupLens: applying collaborative filtering in usenet news. *Communications of the ACM 40*, 77-87.

KOREN, Y. 2010. Collaborative filtering with temporal dynamics. *Communications of the ACM 53*, 89-97.

LIBEN-NOWELLY, D. AND KLEINBERG, J. 2003. The link prediction problem for social networks. In *Twelfth Annual ACM International Conference on Information and Knowledge Management*, 556-559.

LÜ, L. AND ZHOU, T. 2011. Link prediction in complex networks: A survey. *Physica A 390*, 1150-1170.

MANOS PAPAGELIS, D.P. 2005. Qualitative analysis of user-based and item-based prediction algorithms for recommendation agents. *Engineering Applications of Artificial Intelligence 18*, 781–789.

MASLOV, S. AND ZHANG, Y.-C. 2001. Extracting hidden information from knowledge networks. *Physical Review Letters 87*, 248701.

MCGINTY, L. AND SMYTH, B. 2003. On the role of diversity in conversational recommender systems. In *Lecture Notes in Artificial Intelligence*, K.D. ASHLEY AND D.G. BRIDGE Eds. Springer Verlag, 276-290.

MCNEE, S.M., RIEDL, J. AND KONSTAN, J.A. 2006. Being accurate is not enough: how accuracy metrics have hurt recommender systems. In *CHI'06 extended abstracts on Human factors in computing systems* ACM, 1097-1101.

NEWMAN, M.E.J. AND PARK, J. 2003. Why social networks are different from other types of networks. *PHYSICAL REVIEW E 68*, 036122.

PAZZANI, M.J. AND BILLSUS, D. 2007. Content-based recommendation systems. In *Lecture Notes in Computer Science*, P. BRUSILOVSKY, A. KOBSA AND W. NEJDL Eds. Springer Verlag, 325-341.

RESNICK, P. AND VARIAN, H.R. 1997. Recommender systems. *Communications of the ACM 40*, 56-58.

RICCI, F. AND SHAPIRA, B. 2011. *Recommender systems handbook*. Springer.

SARWAR, B., KARYPIS, G., KONSTAN, J. AND RIEDL, J. 2001. Item-based collaborative filtering recommendation algorithms. In *Proceedings of the 10th international conference on World Wide Web*2001 ACM.

SHANG, M.-S., LU, L., ZHANG, Y.-C. AND ZHOU, T. 2010. Empirical analysis of web-based user-object bipartite networks. *Europhysics Letters 92*, 48006.

SHANI, G., BRAFMAN, R.I. AND HECKERMAN, D. 2002. An MDP-based recommender system. In *Proceedings of the Eighteenth conference on Uncertainty in artificial intelligence* Morgan Kaufmann Publishers Inc., 453-460.

SHANI, G. AND GUNAWARDANA, A. 2011. Evaluating recommendation systems. In *Recommender systems handbook* Springer, 257-297.

SHANI, G., HECKERMAN, D. AND BRAFMAN, R.I. 2006. An MDP-based recommender system. *Journal of Machine Learning Research 6*, 1265.

SHIN, D., LEE, J.-W., YEON, J. AND LEE, S.-G. 2009. Context-aware recommendation by aggregating user context. In *Commerce and Enterprise Computing, 2009. CEC'09. IEEE Conference on* IEEE, 423-430.

SOLTANI, S. 2002. On the use of the wavelet decomposition for time series prediction. *Neurocomputing 48*, 267-277.

SOTO, P.G.C. 2011. Temporal models in recommender systems: an exploratory study on different evaluation dimensions Universidad Autónoma de Madrid Escuela Politécnica Superior.

STECK, H. 2011. Item popularity and recommendation accuracy. In *Recomemndation Systems* ACM, Chicago, Illinois, USA, 125-132.

VARGAS, S. AND CASTELLS, P. 2011. Rank and relevance in novelty and diversity metrics for recommender systems. In *Recomemnation Systems* ACM, Chicago, Illinois, USA, 109-116.

XIANG, L. AND YANG, Q. 2009. Time-dependent models in collaborative filtering based recommender system. In *International Joint Conference on Web Intelligence and Intelligent Agent Technology* IEEE, Washington DC, USA, 450-457.

XIONG, L., CHEN, X., HUANG, T.-K., SCHNEIDER, J. AND CARBONELL, J.G. 2010. Temporal collaborative filtering with bayesian probabilistic tensor factorization. In *Proceedings of SIAM Data Mining*.

YOUSEFI, S., WEINREICH, I. AND REINARZ, D. 2005. Wavelet-based prediction of oil prices. *Chaos, Solitons and Fractals 25*, 265-275.

ZHOU, T., KUSCSIK, Z., LIU, J.-G., MEDO, M., WAKELING, J.R. AND ZHANG, Y.-C. 2010. Solving the apparent diversity-accuracy dilemma of recommender systems. *Proceedings of the National Academy of Science of the United States of America 107*, 4511-4515.

ZHOU, T., REN, J., MEDO, M. AND ZHANG, Y.-C. 2007. Bipartite network projection and personal recommendation. *PHYSICAL REVIEW E 76*, 046115.

ZIEGLE, C.-N., MCNEE, S.M., KONSTAN, J.A. AND LAUSEN, G. 2005. Improving recommendation lists through topic diversification. In *World Wide Web* ACM, Chiba, Japan, 22-32.