The Computer Journal Advance Access published November 5, 2014

doi:10.1093/comjnl/bxu115

A Time-Aware Recommender System Based on Dependency Network of Items

Seyed Mohammadhadi Daneshmand[1], Amin Javari[1],
Seyed Ebrahim Abtahi and Mahdi Jalili*

Department of Computer Engineering, Sharif University of Technology, Tehran, Iran
**Corresponding author: mjalili@sharif.edu*

Recommender systems have been accompanied by many applications in both academia and industry. Among different algorithms used to construct a recommender system, collaborative filtering methods have attracted much attention and been used in many commercial applications. Incorporating the time into the recommendation algorithm can greatly enhance its performance. In this paper, we propose a novel time-aware model-based recommendation system. We show that future ratings of a user can be inferred from his/her rating history. We assume that there is cascade of information between the items such that rating an item can lead to other items being rated. There is indeed a hidden network structure among the items and each user tracks a sequence of items in this network. The dependencies between the items are modeled based on statistical diffusion models and the parameters are obtained through maximum-likelihood estimation. We show that under some mild assumptions, the estimation task becomes a convex optimization problem. A major advantage of the proposed method over classical recommender systems is the ability to include novel items in its recommendation lists besides providing accurate recommendations. The proposed model also results in personalized and diverse recommendations. Experimental evaluations show that the model can be trained based on the ratings of a limited number of users. Furthermore, the proposed model outperforms classical recommendation algorithms in terms of both accuracy and novelty.

Keywords: social networks; recommender systems; information cascade; novelty; diversity; maximum-likelihood estimation

Received 13 July 2013; revised 21 August 2014
Handling editor: Jong Hyuk Park

1. INTRODUCTION

This manuscript is related to two research topics in computer science: recommender system and information cascade. In the following, some background on these topics is discussed.

1.1. Recommender systems

Exploring and retrieving items in applications with large-scale datasets is very time consuming for the users, if not impossible. In order to address this issue, recommender systems aim to help users dealing with this information. Recommender systems generate recommendations to a collection of items which may satisfy users based on their past preferences [1, 2]. They often try to maximize users' satisfaction (and thus, the returning revenue) through recommending appropriate items. Satisfaction of users may have different meanings in different contexts (and different metrics have been proposed to assess satisfaction); however, in the context of recommender systems, the recommendation list is provided such that the target user likely gives high ratings to the recommended items, if purchased.

Methods in recommender systems can be generally categorized into three main classes: collaborative filtering (CF), content-based and hybrid algorithms [3]. In content-based recommenders, first, a profile is produced for each user based on the content of items purchased by that user. Then, the system recommends items with the most similar content to the target user's profile. Knowledge-based recommender systems are a special case of content-based systems, in which users directly determine their favorite contents [4]. In CF algorithms, unlike content-based methods, the system is blind to the contents. In these methods, the recommendation

[1]Both authors contributed equally to this work.

lists are completed based on some similarity scores between items and/or users. The main idea behind CF methods is that users with similar tastes in the past times are likely to have similar tastes in the future. Hybrid methods integrate both content-based and CF methods, aiming to provide better recommendations [5–7].

In content-based recommenders, the more labels provided for the users and items, the better the performance. For example, in a movie recommendation system, information about genre, director or story of a movie can be considered as labels for its content. However, having detailed labeling of the items is not practical in large-scale applications with many items, which restricts functionality of content-based methods. The data sources for CF methods can be easily gathered. CF methods can be generally considered in two main classes: memory-based and model-based [8]. Memory-based CF methods use the entire collection of users' previous ratings to provide the recommendation lists. Although they are simple to implement and have been used in many commercial systems, they have some drawbacks such as sparsity, cold start and scalability problems [1]. Model-based CF methods solve the sparsity problem and represent the datasets in a compressed format resulting in higher accuracy as compared with memory-based methods. Model-based algorithms use statistical and machine learning techniques to learn a model based on the collected ratings from users' previous interactions with items. Several model-based algorithms have been introduced in the literature including dependency networks [9], clustering techniques, Markov-based models [10], latent semantic and matrix factorization methods [11, 12].

It has been shown that considering context can increase performance of model-based methods [13, 14]. Such information refers to the context (e.g. time, location or mood) in which users interact with items [15]. According to the Netflix prize competition, time dimension is one of the most important contextual information resources [16–19]. It has been shown that popularity of items and pattern of users behavior are changed by temporal effects [18]. In general, four categories can be considered for temporal effects: user bias shifting, item bias shifting, time bias and user preference shifting. User preference shifting demonstrates changes in attitude of an individual user while item bias indicates changes in taste of groups of users. Average ratings of users may also change in different times, e.g. a user might be optimistic at certain times and give higher ratings than other times. User bias shifting tracks these changes. Item bias shifting argues about changes in popularity of items in different time steps. Considering these temporal effects in recommender systems may improve their performance, and time-aware models have been extensively studied in the literature [3, 16, 17, 20]. Koren *et al.* proposed a method based on matrix factorization in which vectors of latent factors are generated in each time step [21]. Xiong *et al.* proposed another time-aware recommender system based on a Bayesian probabilistic model [22]. In their model, the time has been incorporated as a distinct feature vector associated with those of users and items. The methods based on Markov chain can also be categorized as time-aware techniques [10]. In this paper, we introduce a novel time-aware recommender system based on information cascade models.

1.2. Information cascade in social networks

Analysis of cascade (or diffusion) on structured relationships has many applications in various fields such as cascade of behavior in social relationships [23], viral marketing in economic networks [24] and analysis of virus propagation in computer networks [25]. The diffusion process starts by infecting some of the nodes that themselves may infect their neighbors. There are a number of models for modeling such processes. Independent cascade model [26] is a well-known statistical diffusion model in which each infected node tries to infect its neighbors independently only once. In this work, we use a progressive independent cascade model [23] in which the infected nodes cannot recover after being infected. Song *et al.* [27] proposed a recommendation method using diffusion of information among users. They assumed that if user u selects some items before user v in a period of time, then user u can be significantly influential in future selections of user v. Indeed, they considered the information flow from user u and its rating history was used to recommend items to user v.

1.3. Recommendation based on information cascade

In this paper, we propose a novel time-aware model-based recommender system. Our main assumption is that rating to item i can be effective in rating to item j that is dependent on transmission rate between items i and j. We propose a model of ratings diffusion as an extended version of an independent cascade model. For recommendation, we estimate the transmission rates using rating history of some users. Estimation of transmission rates is equivalent to inference of hidden structure of the network among items. We assume that there is a hidden structure among the items and each user creates a cascade a sequence of *Like* or *Dislike* ratings on this network. There are a number of methods in the literature for inference of network structure using cascade information, i.e. the infection times of the nodes [28–30], which are properly used in this work.

In the proposed model, dependencies between the items are extracted based on users' ratings and the purchase (or rating) time. To this end, sequence of users' ratings is modeled as cascades among items and a statistical model is employed to extract dependencies between the items based on these cascades. Finally, the past ratings and the extracted dependency network among the items are used to provide the recommendation lists for target users. The main advantage of the proposed model over classical recommendation systems is that it results not only in highly accurate recommendations

but also novel recommendations. Indeed, the proposed model employs information on items' bias shifting to obtain both novel and accurate recommendations. Novelty and diversity of recommender systems are becoming important issues within the community of recommendation systems [31–33]. In diffusion-based models, the links among the items are directed from items that were popular at past time steps to those that will be popular in the future, i.e. the diffusion flows toward future novel items.

The other advantage of the proposed model is that it can be learned from ratings of a sampled group of users and the whole dataset is not required to complete the learning task. Indeed, what one needs to learn in the model is to have enough ratings of each of the items to discover the latent structure between them. The effectiveness of the proposed method is assessed on Netflix and Movielens datasets. Our results show that it outperforms classical recommender systems in terms of both diversity and accuracy.

2. PROBLEM FORMULATION

Let us formulate the recommendation problem as follows. There are a set of items $I = \{I_1, I_2, \ldots, I_n\}$ and a set of users $U = \{u_1, u_2, \ldots, u_m\}$, where each user gives ratings to some items at a specific time. The task for recommender system is to recommend a set of items for a target user providing the highest satisfaction for the user.

Users' ratings on items and the times the ratings have been given are represented by matrices $M = [r_{u,i,}]$ and $E = [t_{u,i}]$, where $r_{u,i}$ indicates rating value of user u on item i and $t_{u,i}$ represents the time of this rating. In some applications, users directly rate the items. For example, in Netflix and Movielens datasets, users determine their ratings as five star scores. This type of rating is denoted an explicit rating. The other type is implicit rating, which is defined as user-item co-occurrences. For instance, users' click on items can be considered as positive rate of the user about item. Both types can be mapped to a set of ratings with *like* and *dislike* values. In this paper, we consider only 'like' or 'dislike' states, i.e. the adjacency matrix has values as '+' when a user likes an item or '–' when dislikes it. Let us define rating history of user u as $R_u = \{t_i | i \in I\}$ that means at time t_i user u rates item i. If item i has not been rated by user u, we set $t_i = \infty$. To denote the type of infection (*like* or *dislike*), we use t_i^k where $k \in \{-, +\}$. When the type of the rating is not important, we denote it by t_i. Indeed, there is a single random variable for each item's infection time and k only presents the type of infection. Consider we have the current rating history of user $u(R_u)$ and would like to recommend N items that satisfy the user. Let us suppose T denotes the near future. We should recommend items which are likely to be liked by the user. Consider $P_u(t_i^+ < T|R_u)$ is the probability of item i to be liked by user u in the near future and $P_u(t_i^- < T|R_u)$ is the probability of dislike. We should recommend items with higher values of the term $P_u(t_i^+ < T|R_u) - P_u(t_i^- < T|R_u)$. We propose a method for computing these probabilities to recommend top-N items to each user.

3. EVALUATION OF RECOMMENDATION LISTS

A number of metrics has been introduced in the literature to evaluate performance of recommendation algorithms. Ranking precision is one of the classical metrics to assess top-N recommendation task. Recently, novelty and diversity of the recommendations have been investigated as important metrics. Below, we briefly explain these metrics.

3.1. Precision

Precision is a classic metric to evaluate performance of recommender systems in the top-N recommendation task. Precision of the system, $P_u(N)$, on a recommendation list with N items, is the average precision of the lists recommended to all test users. Precision for the list recommended to user u, $P_u(N)$, is defined as the number of relevant items to the user which are present in the recommendation list. Relevant items to the target user are items for which the user gives them *like* rating in the test set.

3.2. Diversity

It is desirable in recommender systems to recommend items which are personalized for the target user and fit his/her interests. A recommendation list becomes more personalized as the system recommends diverse lists to different users. For example, a recommendation algorithm which recommends almost the same set of items for different users is not desirable in terms of personalization. Inter-list diversity is one of the metrics dealing with personalization in recommendation systems [34]. Inter-list diversity $D(N)$ can be defined as the average distance among the recommended lists to all test users. Obviously, as the average distance becomes higher, the recommendations will be more personalized. Distance between the lists recommended to users i and j can be obtained as:

$$d_{ij} = 1 - \frac{c_{ij}}{N}, \tag{1}$$

where $c_{i,j}$ is the number of common items in the lists recommended to these users and N is the size of the recommended lists.

3.3. Novelty

Although some recommender systems provide accurate lists, their recommendations are in many cases obvious for the users. An item in the recommendation list is obvious when the user knows about the existence of the item before receiving the recommendation list. A recommendation is more desirable as

the list is more informative and non-obvious for the users. Novelty of the recommendation list is a way to consider this issue. Some metrics have also been introduced to evaluate novel recommendations [31]. In this paper, we consider self-information-based novelty to measure novelty of items. According to this measure, users are more likely to know about items which are popular [34]. Self-information-based novelty for a list of recommendations to the user u, $Novelty_u(N)$, can be obtained as:

$$Novelty_u(N) = -\sum_{i=1}^{N} \log_2 \left(\frac{|U|}{Rel_i} \right), \quad (2)$$

where $|U|$ is the number of users in the training set and Rel_i represents the number of users who have given rates to item i. Novelty of an algorithm is the average novelty of the recommendations to all test users.

4. PROPOSED CASCADE-BASED MODEL

In this section, we introduce the proposed recommender system that is based on information cascade models. To compare the performance of the proposed method with some classical recommender systems, we consider memory-based CF and Markov-based model, which have found widespread applications in both academia and industry.

4.1. Memory-based CF

Memory-based CF methods are one of the most popular recommender systems which have been employed in many applications. These methods are in two types: user-based and item-based. In this work, we consider user-based CF that is based on extracting similarities between the users [35]. As the users' similarity scores are extracted based on their previous rating pattern, the algorithm recommends items to the target user based on the rating pattern of its top-K most similar users. Different statistical methods have been introduced to calculate similarities between the users [36, 37]. Two popular methods are Cosine similarity and Pearson correlation. Each of the models has been introduced to deal with different situations (e.g. sparsity problem or cold start problem), and none of them is a golden measure for all cases. We use Pearson correlation in this work. Pearson correlation similarity between users i and j is defined as

$$S_p(i,j) = \frac{\sum_{h=1}^{n} (R_{i,h} - \overline{R_i})(R_{j,h} - \overline{R_j})}{\sqrt{\sum_{h=1}^{n} (R_{i,h} - \overline{R_i})^2} \sqrt{\sum_{h=1}^{n} (R_{j,h} - \overline{R_j})^2}}, \quad (3)$$

where $\overline{R_i}$ is average of the ratings given by user i and $R_{i,h}$ represents the rating given by user i to item h. n is number of items that have received ratings from both users i and j.

4.2. Markov-based recommender

Methods based on Markov chain model the recommendation task as a Markov process [10]. Users purchase items in a sequential manner, and thus, each new item purchased by a user can be considered as transition from one state to another state in a modeled state space. To model the recommendation problem as a Markov process, state space and the transition function between the states should be first defined. In a Markov-based system, state of a user is defined based on k previous items purchased by the user. Here, we define $State_u = \langle I_k, I_{k-1}, \ldots, I_1 \rangle$ as last-k purchased items by user u in a sequential manner. As state space is defined, transition function can be estimated based on training data. Generally, in a Markov model, transition function describes the probability of transition from one state to another one. In the Markov-based recommender system, this value represent the probability that a user who has visited items $I_k, I_{k-1}, \ldots, I_1$, will purchase item I_{k+1} in the next step. Based on the maximum-likelihood estimation and using training data, transition function between the states can be estimated as:

$$PT(< I_k, I_{k-1}, \ldots, I_1 >, < I_{k+1}, I_k, \ldots, I_2 >) = \frac{Count(< I_{k+1}, I_k, \ldots, I_1 >)}{Count(< I_k, I_{k-1}, \ldots, I_1 >)} \quad (4)$$

where $Count(\langle I_k, I_{k-1}, \ldots, I_1 \rangle)$ represents the number of users in the training set that have sequentially purchased the items $I_k, I_{k-1}, \ldots, I_1$. To do the recommendation, first state of the user is defined based on the previous ratings of the target user, and then, items are recommended to a target user considering the users to whom the target items have higher probability of transition.

4.3. Diffusion model for rating

The proposed method is similar to information diffusion in social networks and is indeed a diffusion-based model. The users' ratings diffuse through the hidden structure between the items and the history of the rating values is modeled as a probabilistic model. To estimate the parameters of the model, and thus to provide the recommendation lists, maximum-likelihood framework is used. We first prove that the likelihood function is a convex function under some mild assumption, and then use convex optimization package for estimating parameters of our model.

Let us describe the assumption made in the cascade model using a simple example. Consider that there is a hidden relationship between two items, e.g. two movies have the same directors, actors or story. When a user selects one of these movies, it is likely that this selection leads to selecting the other one as well. The user may dislike the next selection, although he/she has liked the first one. Thus, the hidden structure should be able to model both likes and dislikes. In other words, based on the proposed model, when a user rates item i in type k, it

is expected that this rate can lead the same user to rate item j in type m, and this is dependent on the hidden similarity rate from i to j. The model indeed explores novel and new items from the past history of the user's rating. Furthermore, a user often rates an item at most once, which justifies the choice of progressive independent cascade model.

We consider a network structure among the items on which each user creates two types of opposite cascades (i.e. *Like* and *Dislike*). Let us consider rating time of item i in type k as t_i^k and that of item j in type m as t_j^m. Let us also consider $t_j^m - t_i^k$ as a random variable with a probability density function which is called rating probability density function (RPDF) that is denoted by $f(t_j^m|t_i^k; a_{ij}^{km})$. Since time is a continuous random variable, RPDF represents probability of rating transmission from i to j in the interval $[t_j^m, t_j^m + \mathrm{d}t]$, when $\mathrm{d}t$ tends to zero. To model $t_j^m - t_i^k$, we use statistical methods for life time data [38, 39]. Life time models (or survival models) analyze time of occurrence of some events. Let us consider RPDF depends on parameter a_{ij}^{km} that is denoted as transmission rate [28]. Indeed, transmission rate represents the weight of link from item i to item j in the hidden structure. High value of the transmission rate between i and j makes the rating to item i be more likely to be transmitted to the rating to item j in near future. RPDF can be represented as $f(t_j^m|t_i^k; a_{ij}^{km})$ and the rating cumulative distribution function (RCDF) as $F(t_j^m|t_i^k; a_{ij}^{km})$.

DEFINITION 1. *Pair-wise rating survival probability function.*

Consider item i has been rated in type k in time t_i^k. Following item i being rated, the probability of item j being rated in type m is not high, unless the time t_j^m is a survival probability function [28, 38]. This function (S) can be computed using RCDF as follows:

$$S\left(t_j^m|t_i^k; a_{ij}^{km}\right) = 1 - F\left(t_j^m|t_i^k; a_{ij}^{km}\right). \tag{5}$$

DEFINITION 2. *Pair-wise rating hazard probability function.*

Ratio of the probability of being rated in t_j^m which is caused by that of in t_i^k, to the probability of survival until t_j^m is defined as the hazard function (H) [28, 38]:

$$H\left(t_j^m|t_i^k; a_{ij}^{km}\right) = \frac{f\left(t_j^m|t_i^k; a_{ij}^{km}\right)}{S\left(t_j^m|t_i^k; a_{ij}^{km}\right)}. \tag{6}$$

4.3.1. *Learning pair-wise rating parameters*

Consider $A = \{a_{ij}^{km}|\forall i, j \in U, \forall k, m \in \{-, +\}\}$ as the set of all transmission rates, which should be estimated for recommendation. This is carried out using training set $R = \{R_u|u \in U\}$ where R_u is rating history of user u. The likelihood estimation problem is as follows:

$$A_{\mathrm{ML}} = \arg\max_A\{\ell(R; A)\}, \text{ subject to } a_{ij}^{km} \geq 0 \tag{7}$$

where $\ell(R; A)$ is the likelihood of rating histories with respect to A. This is indeed equivalent to the following minimization problem:

$$A_{\mathrm{ML}} = \arg\min_A\{-\log \ell(R; A)\}, \text{ subject to } a_{ij}^{km} \geq 0 \tag{8}$$

Assuming all elements of R_u are conditionally independent given the set of parameters A, this means that similarities between the users are due only to the hidden network structure. As a simple example, let us consider two users who are friends and not independent; thus, they may have similar rating history. In the cascade mode, we assume that when two users have similar rating history in past times, it is likely that they rate items similarly in future times. In other words, if one knows the past rating histories and also the network structure between the items, the dependency between these two friends can be skipped. One can rewrite the likelihood function as

$$\log \ell(R; A) = \sum_u \log P(R_u; A). \tag{9}$$

4.3.2. *Computing likelihood function*

Items that have been rated before t_i^k can lead item j to receive rating of type m at time $t_j^m \neq \infty$. Since each item can only be rated by a user only once, if item i leads to item j being rated, the items that have been rated before j should not cause j receive rating until t_j^m. Furthermore, item j should not be rated in the opposite type $\sim m$ until t_j^m ($\sim$ is NOT operator, i.e. if m is $-$, then $\sim m$ will be $+$ and vice versa). $P_u(t_j^m|\forall t_i^k < t_j^m; A)$ represents the probability of rating to item j having time of preceding ratings for user u, which can be obtained through the survival probability function as:

$$P_u\left(t_j^m|\forall t_i^k < t_j^m; A\right) = \sum_i \left\{ f\left(t_j^m|t_i^k; a_{ij}^{km}\right) \times \prod_{t_n<t_j, n\neq i} S\left(t_j^m|t_n^k; a_{nj}^{km}\right) \prod_{t_n<t_j} S\left(t_j^{\sim m}|t_n^k; a_{nj}^{k\sim m}\right) \right\}. \tag{10}$$

Using the hazard trick [28] we can rewrite the above probability as:

$$P_u\left(t_j^m|\forall t_i^k < t_j^m; A\right) = \prod_{m\in\{+,-\}} \prod_{t_n<t_j} S\left(t_j^m|t_n^k; a_{nj}^{km}\right) \times \sum_i H\left(t_j^m|t_i^k; a_{ij}^{km}\right). \tag{11}$$

Item j that is not being rated during rating history of user u (i.e. $t_j = \infty$) should survive from the other items for which $t_i^k < \infty$. Let us consider the maximum duration of rating histories as w. In fact, we do not know the exact time of rating

of these items, i.e. we have some censored data [39]. However, we know that these items have not been rated until w. Thus, we should compute conditional probability of surviving until w, as

$$P_u(t_j > w | \forall t_i^k < t_j^m; A) = \prod_{m\in\{+,-\}} \prod_{t_i<t_j} S(w|t_i^k; a_{ij}^{km}). \quad (12)$$

For computing the probability of a rating history, we should compute joint probability of time of infections in the rating history. We can use the chain rule [40] for decomposing of this distribution to conditional distributions, as

$$P(R_u; A) = \prod_{t_j>\infty} P_u(t_j > w | \forall t_i^k < t_j^m; A) \times \prod_{t_j<\infty} P_u(t_j^m | \forall t_i^k < t_j^m; A). \quad (13)$$

Replacing (11) and (12) in (13), one obtains

$$P(R_u; A) = \prod_{t_j>\infty} \prod_{m\in\{+,-\}} \prod_{t_i<t_j} S(w|t_i^k; a_{ij}^{km}) \times \prod_{t_j<\infty} \prod_{m\in\{+,-\}} \prod_{t_i<t_j} \times S(t_j^m|t_i^k; a_{ij}^{km}) \times \prod_{t_j<\infty} \left\{ \sum_{t_i<t_j} H(t_j^m|t_i^k; a_{ij}^{km}) \right\}. \quad (14)$$

Now, using log function properties and equations (14) and (9), we can rewrite the likelihood function using three functions that each is a combination of hazard and survival functions. More precisely,

$$\begin{aligned} \log \ell(R; A) &= \sum_u \log P(R_u; A) \\ &= \sum_u \varphi_1(R_u; A) + \varphi_2(R_u; A) + \varphi_3(R_u; A) \\ \varphi_1(R_u; A) &= \sum_{t_j<\infty} \log \left(\sum_{t_i<t_j} H(t_j^m|t_i^k; a_{ij}^{km}) \right) \\ \varphi_2(R_u; A) &= \sum_{t_j<\infty} \sum_{t_i<t_j} \sum_{m\in\{+,-\}} \log S(t_j^m|t_i^k; a_{ij}^{km}) \\ \varphi_3(R_u; A) &= \sum_{t_j>\infty} \sum_{t_i<t_j} \sum_{m\in\{+,-\}} \log S(w|t_i^k; a_{ij}^{km}) \end{aligned} \quad (15)$$

In this section, we prove that if RPDF meets some conditions, the likelihood function will be a concave function and the likelihood estimation will be a convex optimization problem. To prove the concavity of the log of the likelihood function, we use the following lemmas.

LEMMA 1. *If f_1 and f_2 are concave functions, then $f_1 + f_2$ will be a concave function* [41].

LEMMA 2. *If h is concave and nondecreasing and g is also concave, then $h(g(x))$ will be concave* [41].

Now, we use these two lemmas for proving the following lemma.

LEMMA 3. *If the survival probability function is log concave and the hazard function is concave, then the log of the likelihood function will be concave function.*

Proof.

- $\sum H(t_i^m|t_j^k; a_{ij}^{km})$ is a concave function, since the hazard functions are concave and based on Lemma 1, sum of concave function is concave.
- The sum of the hazard functions is concave and the log function is a concave non-decreasing function. Thus, based on Lemma 2, $\varphi_1(R_u; A)$ is a concave function.
- The sum of the survival functions is concave based on Lemma 2. Thus, $\varphi_2(R_u; A)$ and $\varphi_3(R_u; A)$ are concave functions.
- The sum of concave function φ_1, φ_2 and φ_3 will be a concave function. Thus, the likelihood function will be a log concave function. □

4.3.3. *Estimation method*

To estimate the transmission rates, we use the RPDF whose survival function is log concave and hazard function is concave. Gomez Rodriguez *et al.* [28] introduced some probability distributions with these properties. We use exponential probability distribution, since some experiments in real-world social networks showed that pair-wise probability distribution function is exponential [30]. Using exponential RPDF, the maximum-likelihood estimation will be a convex optimization problem (based on Lemma 3). For solving this problem, we use MATLAB CVX package that is a powerful tool for solving convex optimization problems [42, 43]. Also, we speed up the optimization process by a distributed optimization method and setting the infeasible rates to zeros [28].

4.3.4. *Recommending based on users' rating history*

As mentioned, we recommend top items with respect to $P_u(t_i^+ < T|R_u) - P_u(t_i^- < T|R_u)$ that is equal to $F_u(t_i^+ = T|R_u) - F_u(t_i^- = T|R_u)$. We can compute this probability using the following equation:

$$F_u(t_i^m = T|R_u; A) = 1 - S_u(t_i^m = T|R_u; A) = 1 - \prod_{t_j<t_i} S_u(t_i^m = T|t_j^k; A). \quad (16)$$

In fact, we sort the items according to difference of conditional probability in high rate and conditional probability in low rate in the future time T.

4.3.5. *Structural analysis of the cascade model*

The proposed model is based on extraction of cascade network among items that is based on the time difference between the ratings. Intuitively, we can state that in the proposed model, the cascade dependency from items i to j is intensified as a larger number of users in the training dataset rate item j after item i within a short-time interval. As the cascade dependency network among items is extracted, in the recommendation phase, the algorithm recommends items receiving higher cascades from those previously rated by the target user. This might result in a bias to recommend items showing increasing trend in their popularity, leading to obtain high precision. Items with growing popularity trend are likely to be purchased by the users in near future. The proposed method intelligently filters out the items that were popular in the past times and are losing their popularity. This helps to improve the novelty of the recommendations. Let us consider two items i and j with $a_{ij}{}^{+}$ and $a_{ji}{}^{+}$ as transmission rates between them. If item i was more popular than item j in the past time steps, then $a_{ij}{}^{+} > a_{ji}{}^{+}$. The proposed diffusion model intrinsically selects the items which are likely to be popular in the future, resulting in high novelty of the recommendation lists.

Another issue about the model is its ability to extract the dependency network from ratings of a limited number of users. Once the model has learned the network structure among the items using histories collected from a small subset of users, it can be applied for recommendation to a large set of users. Indeed, the model only requires having enough ratings for each item. Therefore, it is a scalable algorithm for applications with a huge number of users.

As mentioned, we used exponential RPDF in our analysis in which recent users' interests have more intense influence on current users' desires. Nonetheless, temporal effects can be adjusted by choosing different RPDF, as long as the concavity conditions hold, i.e. log-concavity of survival probability function and concave hazard function. Functions such as Rayleigh and Power-law distribution, which satisfy the concavity conditions, can be used for such a purpose [28]. The kernelized cascade model [44] provides a method to estimate RPDF in lieu of the parametric model used in our formulation.

4.3.6. *Hybrid cascade method*

In this section, we discuss how to combine contextual information with collaborating filtering in the proposed model. Consider for each item i we have a set of features (or labels) $f_i = \{f_i^1, f_i^2, \ldots, f_i^k\}$, which are characteristics of the item. The goal is using these features in improving the recommendation. Since our method recommends based on a similarity network inferred by cascade of ratings, the contextual information should be included in the process of inferring the dependency network structure. To this end, feature-enhanced probabilistic model [45] can be utilized. According to this model, conditional probability of infection is

$$f(t_j^m | t_i^k; a_{ij}^{km}, f_i, f_j) = \gamma a_{ij}^{km} \exp(-d(f_i, f_j)) \times \exp(-a_{ij}^{km}(t_j^m - t_i^k)) I(t_j^m > t_i^k). \quad (17)$$

This means that the probability of transferring user from i to j depends on the similarity of two feature vectors f_i and f_j ($d(f_i, f_j)$ is a similarity measure, e.g. cosine similarity) in addition to the transmission rate and time. Based on the above formulation, the inference method considers similarity between features of items in addition to the time of ratings. The main assumption of the hybrid method is that a user follows items which have more similar features as well as larger value of transmission rate. It can be easily shown that the new probability distribution function also leads to a convex optimization problem [45], and thus, the proposed cascade-based model can be simply used in hybrid recommendations.

5. RESULTS

5.1. Datasets

The performance of the proposed recommendation method is evaluated on two real datasets: Netflix and Movielens. These two datasets have been frequently used as benchmarks in recommendation systems. Both datasets contain explicit ratings of the users in a range 1–5 on a set of movies. They also contain time stamps of the ratings which make them appropriate to be used as benchmark in time-aware recommender systems. As we discussed, the proposed method, in its current form, has restrictions in the number of items that it can manage, which is mainly due to the limitations of the CVX toolbox used to solve the convex optimization problem. Therefore, we randomly sample the original Netflix and Movielens datasets. Basic statistics of the sampled datasets have been summarized in Table 1.

5.2. Test methodology

In general, there are two methods for evaluating recommender systems: online evaluation and offline evaluation. In online evaluation, recommendation algorithms are implemented on a real-world application and according to feedback generated by the users, the quality of recommendations is assessed.

TABLE 1. Statistics of datasets.

	Movielens	Netflix
Number of users	6009	17 000
Number of items	1000	1000
Number of edges	1 210 802	1 183 014

However, this type of evaluation is often costly and cannot be implemented in many cases. In this work, we performed the evaluation based on an offline method in a top-N recommendation task. Although there are some drawbacks for this method, it could be used as an alternative for online evaluation and has found widespread application in evaluating recommender systems [46–48]. Indeed, the offline method tries to simulate the online evaluation. In the offline evaluation, the original dataset is split to the test and the training sets. To this end, we first sorted the ratings regarding to their time stamps, and then determined a particular day as the test day. Ratings with time stamps before the determined day were considered as training set and those after that day as test set. For both datasets, we determine the reference day in a way to have 80% of ratings in the training set and 20% in the test set. The training set is used to learn the parameters of the model, which is then tested on test set.

Once the model is trained, we provide a recommendation list of five items for users in the test set. To be in the test set, the users should have at least one rating in the training set and five ratings in the test set (i.e. due to top-5 recommendation). To evaluate the recommendations for each user in the test set, the recommended items are compared against the user's real ratings. Since our model is based on two-level rating (i.e. like or dislike), we use a threshold value to decide whether or not the user likes the item. In this work, we use the threshold of 2 meaning that if a user gives a rating of >2 to an item, it likes the item; otherwise, it dislikes. This form of offline evaluation has been frequently used in previous works [16]. Evidently, the performance changes by varying the threshold, the higher the threshold, the lower the precision of the recommendations. However, this is the case for all algorithms.

5.3. Experimental results and discussion

In this section, we first investigate generalizability of the learned model as a function of the number of users in the training set. In the second experiment, we evaluate the performance of three recommendation algorithms, based on three metrics introduced in the previous section.

5.3.1. Convergence of the model as a function of the number of users

The proposed recommender system is a model-based method. In such methods, first a model is trained based on users' ratings in the training set, and then, it is used to provide the recommendations for each user. One of the main characteristics of the cascade-based model is that the trained model can be used for users who do not have any rating in training data. This is an important feature since one of the main disadvantages of many model-based methods is their need to have all users in the training set, which makes the computations be expensive. The proposed cascade-based model solves the problem by learning the model from a sampled group of users. This can significantly reduce the computational complexity of the method as compared with other model-based techniques.

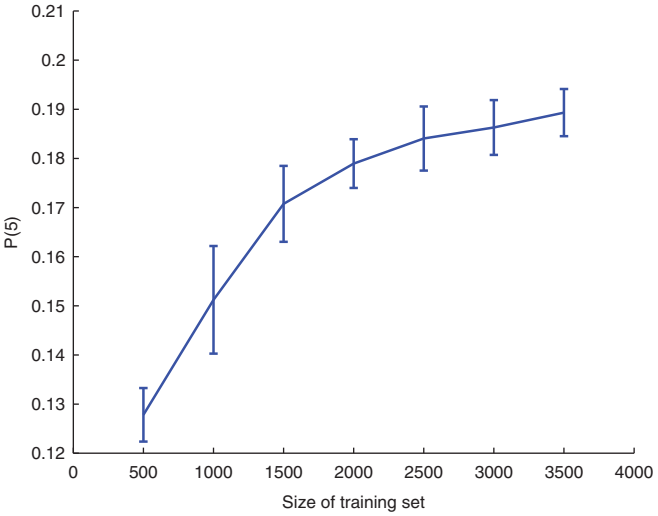

FIGURE 1. Precision of recommendations as a function of the number of users in the training set ($P(5)$) in Movielens dataset.

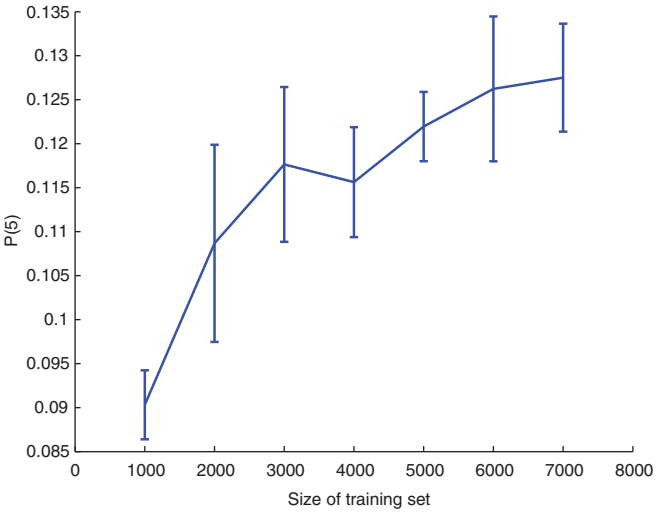

FIGURE 2. Precision of recommendations as a function of the number of users in the training set ($P(5)$) in Netflix dataset.

In the first experiment, we generate the training and the test sets in which users in the training data are completely different from those in the test set. We select ratings of a set of 10 000 and 2500 users, respectively, for Netflix and Movielens datasets as the test sets. Also, we train the model using the training data with different number of users for both datasets. We increase the number of users from 1000 to 7000 in Netflix and from 500 to 3500 in Movilens datasets, and evaluate the precision of the recommendations. Figures 1 and 2 show the precision as a function of the number of users in the training set in Movilens and Netflix datasets, respectively. The results show that in the Netflix dataset as the number of users

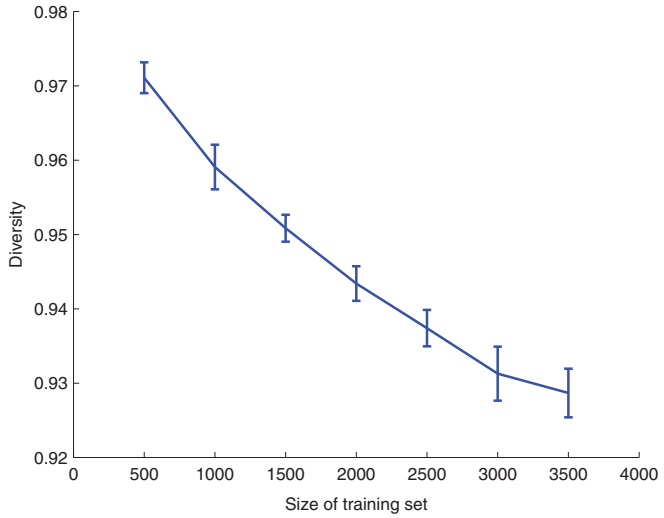

FIGURE 3. Diversity of recommendations as a function of the number of users in the training set (D) in Movielens dataset.

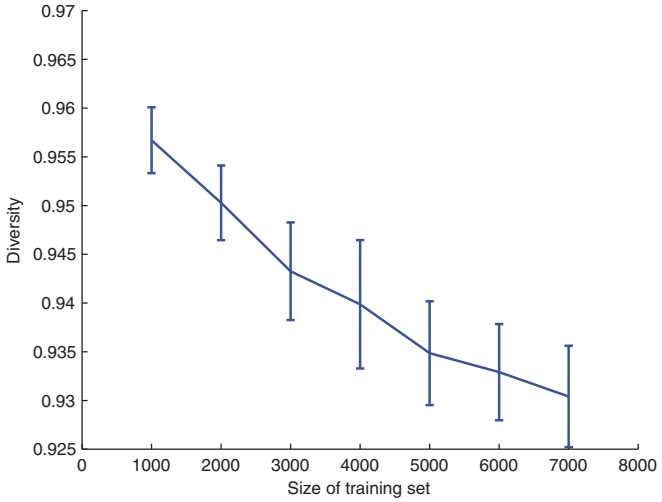

FIGURE 4. Diversity of recommendations as a function of the number of users in the training set (D) in Netflix dataset.

increases from 5000, no significant improvement is obtained in the precision of the model. In Movilens dataset, as the number of users increases from 2500, no significant improvement is obtained. Therefore, the proposed model can learn the required parameters from a limited number of users in the training set and has satisfactory generalizability. This is an important feature, since many real-world recommendation systems handle millions of users and a model-based method using all (or many) of these users in not practical. Figures 3 and 4 represent novelty and Figs 5 and 6 show diversity as a function of the number of users in the training set. According to these results, as size of the training set increases, the model places the items with higher popularity in its recommendations, and thus, the similarity of the lists recommended to different user increases. When the model is not trained, all items have the same chance to appear in recommendation lists; however, it is not desirable to give the same chance to all items to appear in the list. Some items are attractive for a large group of users, and thus, although giving higher chance to these items lowers novelty and precision, it improves accuracy of recommendations.

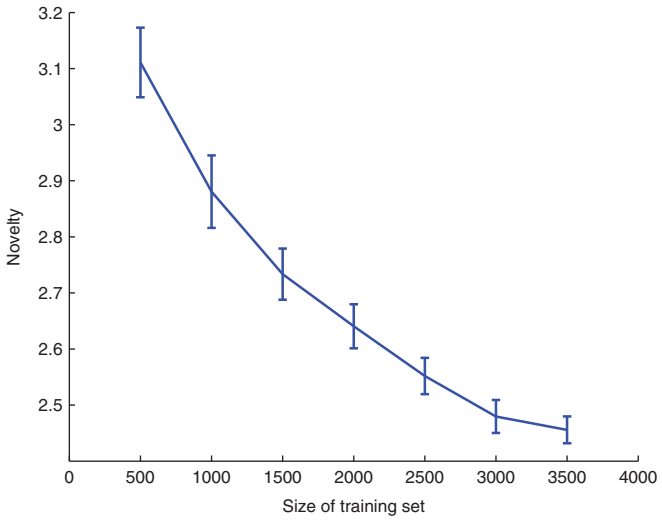

FIGURE 5. Novelty of recommendations as a function of the number of users in the training set in Movielens dataset.

5.3.2. *Comparing cascade-based model with classical recommender systems*

In this section, we compare performance of the proposed cascade-based model with three classic recommendation algorithms: user-based CF, popularity-based recommender systems and Markov-based model. As in the previous section, to compare these three algorithms, we use a set of 10 000 and 2500 users from the original Netflix and Movielens datasets, respectively.

In the user-based CF, To find the top-5 items for each user, 50 of its nearest neighbor users (based on the similarity measure obtained for the users) are considered. In the Markov-based model, we considered the size of the chain as 2. The popularity-based method recommends to a user the most popular items that have not yet been rated by the user. Popularity of items can be determined based on the number of ratings on each item. For these recommendation methods, we compare the precision (P(N)), the self-information-based novelty (Novelty) and the inter-list diversity (Diversity).

Figure 7 shows the precision of these recommender systems in the two datasets. The cascade-based method results in 0.14 and 0.19 precision in Netflix and Movielens datasets, respectively. This is higher than the precision of both the user-based CF and the Markov-based model. The popularity-based method is known to give a highly accurate recommendation list; and the proposed method could give us a comparative

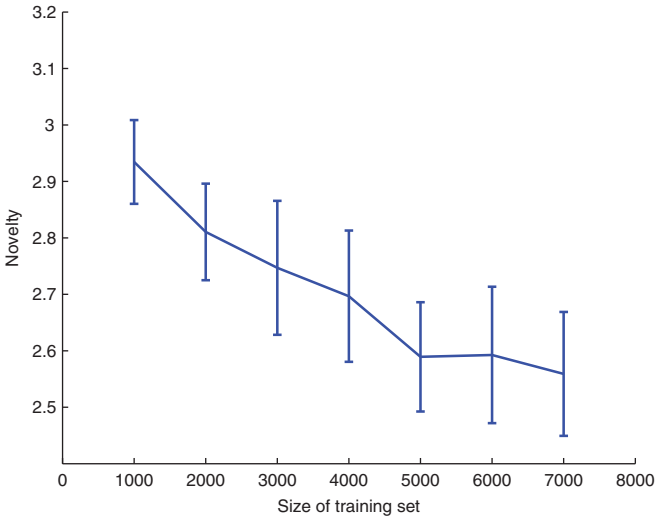

FIGURE 6. Novelty of recommendations as a function of the number of users in the training set in Netflix dataset.

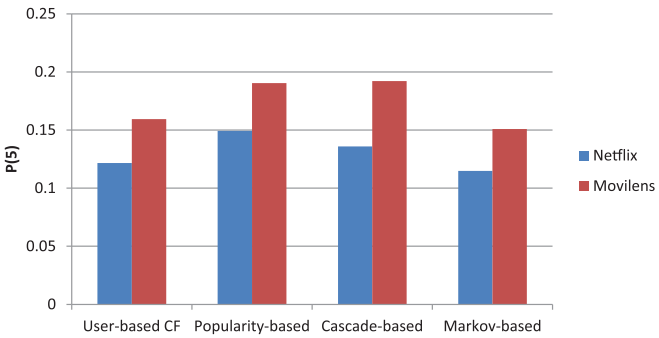

FIGURE 7. Precision ($P(5)$) of four different recommendation algorithms including user-based **CF**, popularity-based, cascade-based and Markov-based methods on Netflix and Movielens datasets.

precision with the popularity-based method (its precision is even a bit better in the Movielens dataset).

We compare the novelty of the recommendation lists as their self-information-based novelty and the results are shown in Fig. 8. The Markov-based model results in the best performance in terms of novelty which is followed by cascade-based, user-based CF and popularity-based model. We know that the more the novelty of the recommendation lists, the less expected the target user is to be aware of existence of the recommended items. Although, it is desired to recommend lists with high novelty, but besides precision of the recommendations should be considered. As can be seen in Figs. 7 and 8, Markov-based model has the lowest precision among other algorithms. Indeed having high novelty Markov model suffers from low precision. One may say that random recommendation may have very high novelty but obviously output of the random recommender is not desirable to the users.

Figure 9 shows personalization of the recommendation algorithms in terms of inter-list diversity. Cascade-based model

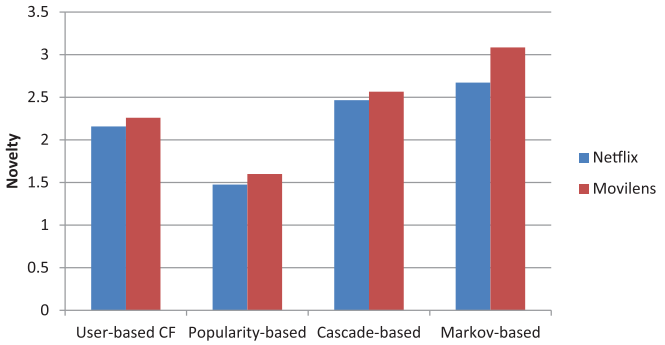

FIGURE 8. Self-information-based novelty (*N*ovelty) of four different recommendation algorithms including user-based **CF**, popularity-based, cascade-based and Markov-based methods on Netflix and Movielens datasets.

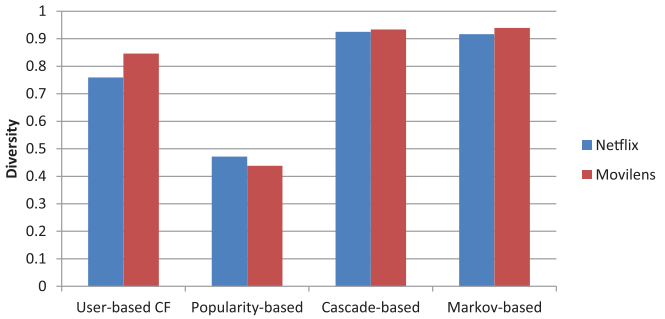

FIGURE 9. Inter-list diversity (*D*iversity) of four different recommendation algorithms including user-based **CF**, popularity-based, cascade-based and Markov-based methods on Netflix and Movielens datasets.

and Markov model have almost similar results in terms of the diversity. Also as it can be seen that the cascade model outperforms the user based CF and popularity models in terms of the diversity.

In the Movielens dataset, there is ∼0.93 inter-list diversity followed by the user-based CF with 0.84 and the popularity-based with 0.44. The cascade-based model outperforms the user-based CF and the popularity-based model by 0.15 and 0.40 in Netflix dataset.

In sum, considering the results of the Markov-based model, we can state that this model includes items that are less related to the target users. Indeed, providing satisfaction for the users depends on novelty and personalization of lists as well as precision of the recommendations. For example, although popularity-based recommender provides an accurate list, it is not a good strategy, since it does not consider personalization and novelty. The cascade-based model considers both these factors in its recommendations. Thus, compared with the user-based CF and the popularity-based methods, the proposed cascade-based model not only demonstrates accurate recommendations, but also provides novel and personalized lists for each user. Both the popularity-based and the user based CF

are biased toward recommending items with high popularity resulting to rather low novelty values. They consider only a part of item space with high popularity, and thus, suffer from low personalization in their recommendations. Our proposed model tries to overcome these problems. A reason behind higher personalization and novelty of the cascade-based model is that it performs the recommendation based on cascades among items. Let us consider an item with low popularity and a user who has previously rated items with high dependency to this item. In such a case, the cascade-based model is likely to recommend the item to the user, since it is surrounded by those which have been previously rated by the user. Thus, the model gives a chance to items with low popularity to be included in recommendation lists. It has been shown that there is a dilemma between accuracy and personalization in recommendation systems [47, 49] and they do not go in the same direction. Often, when precision improves, the novelty worsens and vice versa. Having a method resulting in better precision and novelty than the user-based CF would be valuable and our proposed model-based algorithm could be such a method.

6. CONCLUSION

Recommender systems have application in many systems such as online social networks, movie store, music distributions and book stores. In previous works, precision of recommendation lists was the most important issue addressed in designing the algorithms. However, it has been recently shown that personalization and novelty are also important metrics influencing satisfaction of users. Diversity and novelty of recommendations has recently attracted much attention and become a challenging issue in this field. In this work, we propose a novel time-aware model-based recommendation algorithm. The model is built based on extraction of dependencies among items. Considering the sequential manner of users' rating on items, we model these sequences as cascades among the items. We assume that there is a hidden network structure among items and proper cascade models are used to infer this structure. We prove that under some mild conditions, the maximum-likelihood estimation problem is a convex optimization problem, and thus, can be easily solved. Our main contribution in this work is to link two domains of social networks analysis and mining: diffusion on networks and recommender systems. The cascade-based structure of the model allows including novel and personalized items in the recommendation lists besides having high accuracy. Our results show that, as compared with the user-based CF (a classic recommender system used in many application), the proposed cascade-based model provides recommendations with not only higher accuracy, but also higher novelty and diversity. Also, novelty and diversity of the proposed method are much higher than the popularity-based method, while their accuracy was comparable.

FUNDING

This research was supported by Australian Research Council through project No DE140100620.

REFERENCES

[1] 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 Trans. Audio Electroacoust. Knowl. Data Eng.*, **17**, 734–749.

[2] Resnick, P. and Varian, H.R. (1997) Recommender systems. *Commun. ACM*, **40**, 56–58.

[3] Ricci, F. and Shapira, B. (2011) *Recommender systems handbook*. Springer.

[4] Pazzani, M. and Billsus, D. (2007) *Content-based recommendation systems.* The adaptive web, 325–341.

[5] J. Ben Schafer, Jon Herlocker, D.F. and Shilad Sen (2007) *Collaborative filtering recommender systems.* LNCS: Lecture Notes In Computer Science, **4321**, 291–324.

[6] Cobos, C., Rodriguez, O., Rivera, J., Betancourt, J., Mendoza, M., León, E. and Herrera-Viedma, E. (2013) A hybrid system of pedagogical pattern recommendations based on singular value decomposition and variable data attributes. *Inf. Process. Manage.*, **49**, 607–625.

[7] Debattista, J., Scerri, S., Rivera, I. and Handschuh, S. (2012) Ontology-Based Rules for Recommender Systems. *International Workshop on Recommender Systems meet Big Data & Semantic Technologies* (*SeRSy*), pp. 49–60.

[8] Bobadilla, J., Ortega, F., Hernando, A. and Gutiérrez, A. (2013) *Recommender systems survey. Knowl.-Based Syst.*, **46**, 109–132.

[9] Heckerman, D., Chickering, D.M., Meek, C., Rounthwaite, R. and Kadie, C. (2001) Dependency networks for inference, collaborative filtering, and data visualization. *J. Mach. Learn. Res.*, **1**, 49–75.

[10] Shani, G., Brafman, R.I. and Heckerman, D. (2002) An MDP-Based Recommender System. *Proc. 18th Conf. on Uncertainty in Artificial Intelligence*, pp. 453–460. San Francisco: Morgan Kaufmann.

[11] Hofmann, T. (2004) Latent semantic models for collaborative filtering. *ACM Trans. Inf. Syst.* (TOIS), **22**, 89–115.

[12] Koren, Y., Bell, R. and Volinsky, C. (2009) *Matrix factorization techniques for recommender systems.* Computer, **42**, 30–37.

[13] Baltrunas, L. and Ricci, F. (2013) Experimental evaluation of context-dependent collaborative filtering using item splitting. *User Model. User-Adapt. Interact.*, **23**, 1–28.

[14] Liu, X. and Aberer, K. (2013) SoCo: A Social Network Aided Context-Aware Recommender System. *Proc. of the 22nd Int. Conf. on World Wide Web*, pp. 781–802. International World Wide Web Conferences Steering Committee.

[15] Adomavicius, G. and Tuzhilin, A. (2011) Context-Aware Recommender Systems. In F. Ricci and Shapira, B. (eds), *Recommender Systems Handbook.* Springer.

[16] Campos, P.G., Díez, F. and Cantador, I. (2013) Time-aware recommender systems: a comprehensive survey and analysis of

existing evaluation protocols. *User Model. User-Adapt. Interact*, **24**, 1–53.

[17] Wei, S., Ye, N. and Zhang, Q. (2012) Time-Aware Collaborative Filtering for Recommender Systems. In *Pattern Recognition*, pp. 663–670. Springer.

[18] Xiang, L. and Yang, Q. (2009) Time-Dependent Models in Collaborative Filtering Based Recommender System. *Int. Joint Conf. on Web Intelligence and Intelligent Agent Technology*. Washington, DC, USA, pp. 450–457. IEEE.

[19] Javari, A. and Jalili, M. (2014) Accurate and novel recommendations: an algorithm based on popularity forecasting. *Trans. Intell. Syst. Technol.*, Special issue on Diversity and Discovery in Recommender Systems and Exploratory Search, **5**, 1–22.

[20] Baltrunas, L. and Amatriain, X. (2009) Towards Time-Dependant Recommendation Based on Implicit Feedback. *Workshop on Context-Aware Recommender Systems* (*CARS'09*), New York, NY, USA, October 25.

[21] Koren, Y. (2010) Collaborative filtering with temporal dynamics. *Commun. ACM*, **53**, 89–97.

[22] Xiong, L., Chen, X., Huang, T.-K., Schneider, J. and Carbonell, J.G. (2010) Temporal Collaborative Filtering with Bayesian Probabilistic Tensor Factorization. *Proc. SIAM Data Mining*.

[23] Kleinberg, J. (2007) Cascading Behavior in Networks: Algorithmic and Economic Issues. In N. Nisan, Roughgarden, T., Tardos, E. and Vazirani, V.V. (eds), *Algorithmic Game Theory*, pp. 613–632. Cambridge University Press.

[24] Leskovec, J., Adamic, L.A. and Huberman, B.A. (2007) The dynamics of viral marketing. *ACM Trans. Web (TWEB)*, **1**, 5:1–5:39.

[25] Kephart, J.O. and White, S.R. (1991) Directed-Graph Epidemiological Models of Computer Viruses. *Proc. of 1991 IEEE Computer Society Symposium on Research in Security and Privacy, 1991*, 20–22 May 1991, pp. 343–359.

[26] Kempe, D., Kleinberg, J. and Tardos, E. (2003) Maximizing the Spread of Influence Through a Social Network. *Proc. of the 9th ACM SIGKDD Int. Conf. on Knowledge Discovery and Data Mining*, Washington, DC, pp. 137–146. ACM, New York.

[27] Song, X., Tseng, B.L., Lin, C.-Y. and Sun, M.-T. (2006) Personalized Recommendation Driven by Information Flow. *Proc. of the 29th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, Seattle, Washington, USA, pp. 509–516. ACM, New York.

[28] Gomez Rodriguez, M., Balduzzi, D. and Schölkopf, B. (2011) Uncovering the Temporal Dynamics of Diffusion Networks. *Int. Conf. on Machine Learning*, Bellevue, Washington, USA.

[29] Myers, S.A. and Leskovec, J. (2011) *On the Convexity of Latent Social Network Inference*. *Advances in Neural Infromation Processing Systems (NIPS)*.

[30] Goyal, A., Bonchi, F. and Lakshmanan, L.V.S. (2010) Learning Influence Probabilities in Social Networks. *Proc. of the 3rd ACM Int. Conf. on Web Search and Data Mining*, New York, USA, pp. 241–250. ACM, New York.

[31] Shani, G. and Gunawardana, A. (2011) Evaluating Recommendation Systems. In F. Ricci, Rokach, L., Shapira, B. and Kantor, P.B. (eds), *Recommender Systems Handbook*, pp. 257–297. Springer.

[32] Hurley, N. and Zhang, M. (2011) Novelty and diversity in top-N recommendation—analysis and evaluation. *ACM Trans. Internet Technol. (TOIT)*, **10**, 14:1–14:30.

[33] Castells, P., Vargas, S. and Wang, J. (2011) Novelty and Diversity Metrics for Recommender Systems: Choice, Discovery and Relevance. *Proc. of Int. Workshop on Diversity in Document Retrieval (DDR)*, Dublin, Ireland, April 18, pp. 29–37.

[34] Vargas, S. and Castells, P. (2011) *Rank and Relevance in Novelty and Diversity Metrics for Recommender Systems*. *Recomemnation Systems*, Chicago, IL, USA, pp. 109–116. ACM.

[35] Su, X. and Khoshgoftaar, T.M. (2009) A survey of collaborative filtering techniques. *Adv. Artif. Intell.*, **2009**, 14:1–14:19.

[36] Fouss, F., Pirotte, A., Renders, J.-M. and Saerens, M. (2007) Random-walk computation of similarities between nodes of a graph with application to collaborative recommendation. *IEEE Trans. Knowl. Data Eng.*, **19**, 355–369.

[37] Ahn, H.J. (2008) A new similarity measure for collaborative filtering to alleviate the new user cold-starting problem. *Inf. Sci.*, **178**, 37–51.

[38] Lawless, J.F. (1982) *Statistical Models and Methods for Lifetime Data*. Wiley, New York.

[39] Borgan, O., Gjessing, H.K. and Gjessing, S. (2008) *Survival and Event History Analysis: A Process Point of View*. Springer.

[40] Papoulis, A. and Pillai, S.U. (2002) *Probability, Random Variables and Stochastic Processes*. Tata McGraw-Hill Education.

[41] Boyd, S. and Vandenberghe, L. (2004) *Convex Optimization*. Cambridge University Press.

[42] Grant, M. and Boyd, S. (2014) *CVX: Matlab Software for Disciplined Convex Programming*, *version 2.1*. http://cvxr.com/cvx.

[43] Grant, M. and Boyd, S. (2008) Graph Implementations for Nonsmooth Convex Programs. In V. Blondel, Boyd, S. and Kimura, H. (eds), *Recent Advances in Learning and Control.* Springer

[44] Du, N., Song, L., Smola, A. J. and Yuan, M. (2012) Learning Networks of Heterogeneous Influence. *Advances in Neural Infromation Processing Systems (NIPS)*.

[45] Wang, L., Ermon, S. and Hopcroft, J. (2012) Feature-Enhanced Probabilistic Models for Diffusion Network Inference. In P. Flach, Bie, T. and Cristianini, N. (eds), *Machine Learning and Knowledge Discovery in Databases*, pp. 499–514. Springer, Berlin, Heidelberg.

[46] Jahrer, M., Töscher, A. and Legenstein, R. (2010) Combining Predictions for Accurate Recommender Systems. *Proc. of the 16th ACM SIGKDD Int. Conf. on Knowledge Discovery and Data Mining*, pp. 693–702. ACM.

[47] 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. *Proc. Natl. Acad. Sci. U.S.A.*, **107**, 4511–4515.

[48] Cremonesi, P., Koren, Y. and Turrin, R. (2010) Performance of Recommender Algorithms on Top-*n* Recommendation Tasks. *Proc. the 4th ACM Conf. on Recommender Systems*, pp. 39–46. ACM.

[49] Javari, A. and Jalili, M. (2014) A probabilistic model to resolve novelty-accuracy challenge of recommendation systems. *Knowl. Inf. Syst.*, in press.