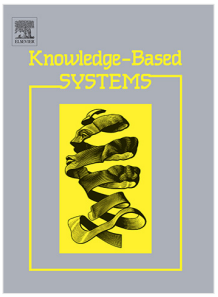
Accepted Manuscript



User preferences modeling using dirichlet process mixture model for a content-based recommender system

Bagher Rahimpour Cami, Hamid Hassanpour, Hoda Mashayekhi

PII: S0950-7051(18)30479-9
DOI: https://doi.org/10.1016/j.knosys.2018.09.028
Reference: KNOSYS 4512

To appear in: *Knowledge-Based Systems*

Received date : 1 March 2018
Revised date : 26 August 2018
Accepted date : 17 September 2018

Please cite this article as: B.R. Cami, et al., User preferences modeling using dirichlet process mixture model for a content-based recommender system, *Knowledge-Based Systems* (2018), https://doi.org/10.1016/j.knosys.2018.09.028

User Preferences Modeling using Dirichlet Process Mixture Model for a Content-Based Recommender System

Bagher Rahimpour Cami[a], Hamid Hassanpour[a,*], Hoda Mashayekhi[a]

[a]*Faculty of Computer Engineering and IT, Shahrood University of Technology, P.O. Box 316, Shahrood, Iran*

Abstract

Recommender systems have been developed to assist users in retrieving relevant resources. Collaborative and content-based filtering are two basic approaches that are used in recommender systems. The former employs the feedback of users with similar interests, while the latter is based on the feature of the selected resources by each user. Recommender systems can consider users' behavior to more accurately estimate their preferences via a list of recommendations. However, the existing approaches rarely consider both interests and preferences of the users. Also, the dynamic nature of user behavior poses an additional challenge for recommender systems. In this paper, we consider the interactions of each individual user, and analyze them to propose a user model and capture user's interests. We construct the user model based on a Bayesian nonparametric framework, called the Dirichlet Process Mixture Model. The proposed model evolves following the dynamic nature of user behavior to adapt both the user interests and preferences. We implemented the proposed model and evaluated it using both the MovieLens dataset, and a real-world dataset that contains news tweets from five news channels (New York Times, BBC, CNN, Reuters and Associated Press). The experimental results and comparisons with several recently developed approaches show the superiority in accuracy of the proposed ap-

*Corresponding author at: Faculty of Computer Engineering and IT, Shahrood University of Technology, P.O. Box 316, Shahrood, Iran. Tel/Fax: +98(23)32300251.

Email addresses: rc_bagher@yahoo.com (Bagher Rahimpour Cami), h.hassanpour@shahroodut.ac.ir (Hamid Hassanpour), hmashayekhi@shahroodut.ac.ir (Hoda Mashayekhi)

proach, and its ability to adapt with user behavior over time.

Keywords: User Preferences Modeling, Temporal Content Based Recommender Systems, User Behavior Modeling.

1. Introduction

Recommender systems have been developed as a useful tool to provide a personalized environment in the World Wide Web (WWW). The main aim of recommender systems is to find relevant resources from the huge number of resources in different contexts such as WWW and social networks. Recommender systems employ users' past transactions within some methodologies and mechanisms to extract their preferences, and help users to select their favorite resources in future. In other words, recommender systems provide a personalized framework. This technology has been widely used in various commercial applications such as Amazon [1], Netflix [2], and online news services (for example, Google news) [3].

There are various methods for developing recommender systems. Existing approaches for constructing a recommender system can be classified in five categories [4, 5]. *Collaborative Filtering* systems constitute the first category. These recommender systems provide a framework for finding groups of similar users to employ their feedbacks about the selected resources for recommendation [6, 7]. *Content-Based* recommender systems constitute the second category. These recommender systems employ the features of resources to constructs the user profile which is later employed in a profile matching algorithm for recommendation [8, 9]. *Context-aware* recommender systems constitute the third category in which employ contextual information such as time, location, and social data to make recommendations [10]. Knowledge-based recommender systems constitute the forth category. These recommender systems extract users requirements and features of resources to produce a *knowledge-base*. Then, the knowledge-base recommender system uses inference and reasoning techniques to discover resources that meet the user's requirements [11]. *Hybrid* recommender systems constitute the fifth category in which different approaches from other categories are combined to improve the recommendation performance [12].

Recommender systems employ the user profile to create a user model for providing proper recommendations. Building an accurate user model plays a main role and is crucial in success of recommender systems [13, 14]. User-

resource interactions typically manifest themselves as explicit and implicit user feedbacks that provide the key indicators for modeling user's preferences in selecting resources, and are essential information for personalizing the recommendation [15]. User modeling approaches can be divided into two general categories, structural (prototype) modeling [14, 16] and behavioral modeling [17], which are stated below.

The structural modeling approaches construct an abstraction of user profile. In these approaches, the feature-based or stereotype modeling methods are used to extract features from profiles of individual users, or a stereotype from profile of several users [18]. Therefore, the constructed user model serves as a structural definition of the user(s).

For example, in collaborative filtering, the abstraction of user profile can be modeled as a feature vector in the form of {*user, resource, rate [, time]*}. Later, this structure can be leveraged by similarity-based algorithms [19] to provide similarity measures based on cumulated users (memory-based/ neighborhood-based collaborative filtering) [20, 21], or to express latent similarity between users or resources (model-based collaborative filtering) [22, 23].

In the content-based recommender systems, a keyword-based vector-space structure is usually used to represent the user profile. Then, a similarity measure is used for matching the user profile with new resources [9, 24].

The behavioral modeling approaches generally use probabilistic frameworks such as Bayesian networks or Hidden Markov Models [17, 25]. These approaches identify the parameters (observable or latent) that affect (or reflect) the user behavior. Then, these parameters are incorporated into a probabilistic graphical model to construct the behavioral model of the user. In these approaches, the user(s') interests are usually denoted by latent parameters (factors). After inferring the latent variables from user(s') profile(s), the model is able to predict the user interests and provide recommendations. Behavioral models can handle the uncertainty of user behavior in modeling process. Also, they can employ the individual user profile, or cumulated user profiles for parameter estimation in order to extract individual or group interests, respectively. For example, in [26] a generative probabilistic model was represented to estimate user ratings. Also, authors in [27] represented an extension of latent Dirichlet allocation to capture user behavior in a forum and learn user interests which can be used for product or news recommendation.

The main contribution of this paper is to represent the temporal user preferences to construct a behavioral user model and employ it in a content-

based recommender system. The existing approaches extract the temporal user interests to predict user behavior [28, 29, 30, 31]. User interests indicate the distinct groups of related resources that the user has selected over time. Recommender systems leverage user interests to find candidate resources that a user may select in future. However, the importance of each resource (resource of each interest) from the user viewpoint is not taken into account. Latent factors are one of the well-known methods which are used to model user interests as the latent parameters. For example, in a news recommender system, user profile contains news articles visited during time. The LDA model [32] is applied on the user profile to extract topics of groups associated with similar news articles visited by the users. We refer to these groups as user interests where all interests have the same priority. A user preference model, other than the interests, captures the preferences of a user in selecting from each interest. User preferences indicate the priority of each interest from the user's viewpoint as a probability distribution [33]. For example, in the news recommender system mentioned above, the preference indicates the priority of each interest (group of news articles that have same topic) that will be represented by a parameter in the model. Therefore, we enhance existing methods which employ user interests and resource similarity for recommendation via capturing and incorporating preferences of users.

In this paper we propose a behavioral user model which employs the individual user profile to construct a temporal content based recommender system. In addition to extracting user interests, a major novelty of the proposed user model is inferring user preferences. It is able to capture the dynamic nature of user interests and preferences over time. We incorporate the user profile in a Bayesian nonparametric framework called Dirichlet Process Mixture Model (DPMM) [34]. We use an extension of DMPP with two main components: distance dependent Chinese Restaurant Process (ddCRP) [35] and Dirichlet Process (DP) [36]. We employ the ddCRP to perform incremental clustering of the user profile for extracting user interests. Also, we calculate the priority of user interests as the user preferences using DP.

We gathered the news tweets from five well-known news agency channels, the New York Times, BBC, CNN, Reuters, and Associated Press in a NewsTweet dataset. Evaluations are performed on this dataset and also the standard MovieLens dataset. We evaluate the accuracy of the proposed model by means of different metrics, and also compare it with other recently developed methods. The empirical results show superiority of the proposed model in predicting user behavior, the effectiveness of preference modeling,

and the model capability in capturing dynamic nature of user behavior.

The rest of the paper is organized as follows: Section 2 provides a brief review of related approaches in recommender systems and the ways to construct the user model. The preliminaries and motivation of the proposed user model is described in Section 3. Section 4 describes the proposed method and related concepts. We give an overview of our dataset, the experimental results and comparisons in Section 5. In Section 6 we present our conclusions and future work.

2. Related Works

Recommender systems were developed to assist users in finding favourite resources. As mentioned earlier, there are two main approaches, *structural modeling* and *behavioral modeling*, to employ users' profiles to construct a recommender system. In *structural modeling* approach, a user profile consists of tuples incorporated in the recommendation procedures. One of the well-known methods in this approach is collaborative filtering that usually represents these tuples as $<$*user_id, resource_id, feedback*$>$. In *neighborhood-based* collaborative filtering techniques, the tuples of users' profile are used as a rating matrix to construct a similarity measure between users (user-based) or resources (item-based) [2, 37]. *Model-based* collaborative filtering techniques apply machine learning methods (for example, Bayes classifiers, support vector machines and K-means clustering) to the rating matrix and extract a model to predict new resources [38, 39]. *Factorization-based* collaborative filtering techniques infer some latent factors from the rating matrix. These latent factors indicate the correlations between users and resources in a latent space. Authors in [2, 30, 40] capture the latent factors using matrix factorization, and employ them to predict the priority of new resources. The latent factor is able to capture user interests via extracting the interaction between users and resources. In contrast, content-based approaches take into account user profiles individually and usually define the user as $<$*user_ia, resource_features*$>$, in which the features of resources are often textual information. For recommendation, features of candidate resources are matched with the resources of user profile and the most similar resources are selected [8, 9].

In *behavioral modeling* approach, probabilistic graphical models such as Bayesian network and Hidden Markov Models [25] are used. A number of parameters exists in the probabilistic graphical model to capture the user

behavior and construct the user model. Then, the users' profiles are used to estimate the parameters. Authors in [27] extended the Latent Dirichlet Allocation (LDA) modeling and created a novel Bayesian network to discover user interests. A graphical model is represented in [17] to capture user behavior during time. This model extracts and combines the time dependent user interests and temporal context (the general interests in a specific time) to predict the rate of new resources.

Temporal information is one of the main feature of users' profiles. This feature is an important factor to construct an adaptive user model and create a temporal recommender system. There are several methods that incorporate temporal information into recommendation procedures. The main characteristic of these approaches is their ability to capture future trends. These methods are usually categorized into *temporal collaborative filtering* and *discrete temporal models* [41, 42]. One of the well known approaches of capturing the dynamic interests of user and its extensions are represented in [30, 43]. In this approach, the matrix factorization is extended for incorporating time information. This model assumes that the user factors change over time, and calculates latent factors time-dependently (temporally). In other words, the matrix factorization models the ratings as a function of time. Most of the other approaches in temporal collaborative filtering have used *decay-function* or *window-intervals* to incorporate time information. These approaches use weighted ratings to calculate the neighborhood or prediction function. Authors in [6, 13] used a decay-function to weight ratings and then the weighted ratings are incorporated to find neighbors. Handling time as a periodic context is represented in [44]. In this paper, time is divided into some intervals in which users perform activities. The similarity of users is calculated using time-dependent activities. Some approaches incorporate time to divide user interests into two general categories: longer-term and short-term. Longer-term interests show the essential interests of users which change very slowly. In contrast, short-term interests show the current user interests and often change with new interactions. Authors in [45] used graph-based methods to explicitly model users' long-term and short-term preferences, and make top-N recommendation [45]. Authors in [31, 46] represented a content-based news recommender system called LOGO that employs a decay function to create long-term and short-term profiles. It uses the long-term to select relevant news groups, and recommends news articles based on a short-term profile. Authors in [17] incorporate time information into a graphical model to represent a temporal context-aware user model. Also, Markov models are

used as sequential information to predict a user's future actions [25].

According to our studies, modeling the temporal user preferences is one of the main shortcoming in the existing recommendation methods. Most of the existing methods employ the user profile to extract user interests [46, 31]. But they do not take into account the priority of each interest. Also, the temporary in the existing method is handled via age of select resources [30]. The proposed method leverages the user preferences to overcome the above mentioned shortcoming. The user preference reflects the user priority in selecting resources within each interest and evolve over time in order to capture the user behavior. In this paper, we extend our previous work [47] via using distance dependent Chinese Restaurant Process (ddCRP) as the prior of latent clusters distribution. Moreover, we use a heuristic method for determining the cluster distribution parameter and allow it to be updated (in the experimental results, we show the improvement of the proposed recommender system).

3. Preliminaries

The motivation for considering user preferences and the prerequisites for describing the proposed content based recommender system are brought in this section. We first illustrate the impact of user preference on his/her behavior in selecting resources from different groups, by means of some real world experiments. The required definitions and notations for describing the proposed system are described in Section 3.2. Finally, the Dirichlet Process Mixture Model that is used as the framework of the proposed model is introduced in Section 3.3.

3.1. The Motivation

We performed some experiments on a real world dataset to further illustrate the concept and influence of user preferences on user interactions. The dataset will be introduced in Section 5.1. We applied the LDA [32] on the dataset to extract groups (topics) from resources. Fig. 1 depicts selection of resources from different groups by all users over time (each colored column represents a distinct group). As observed, some groups are visited more frequently. In other words, these groups have more influence on selection of resources. To analyze users' behavior, we choose a subset of users and plot their resource selection. Fig. 2 illustrates the number of resources that users have selected within each group. Filled circles, depending to their

size, represent the number of selections that are performed by a user from a specific group. This figure indicates that each user prefers to select resource from specified groups. Moreover, Fig. 3 represents the user preferences related to each group in two consecutive time intervals. The user preferences is calculated via the number and timestamp of user activities. Darker blocks indicate larger influence of the corresponding group. Also, the specified areas in Fig 3 clearly expose variability of preferences over time in selecting from different groups.

In this paper we aim to capture these preferences when modeling users behavior, which can improve accuracy of the system in predicting users interactions.

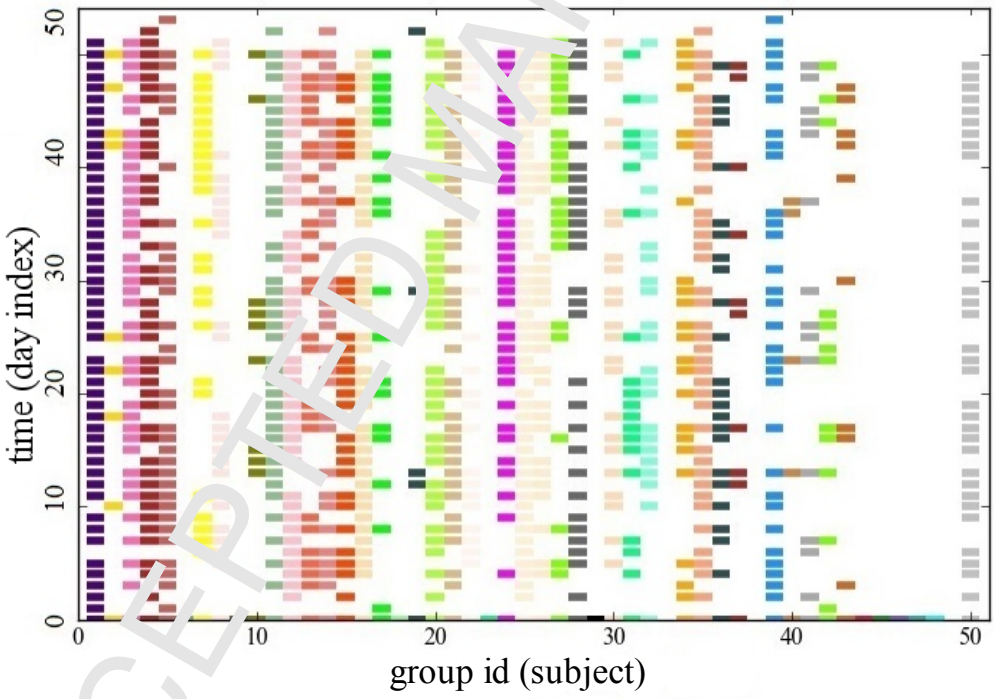


Fig. 1 Selection of resources from groups over time by all users. Each colored column represents related group of resources (subjects).

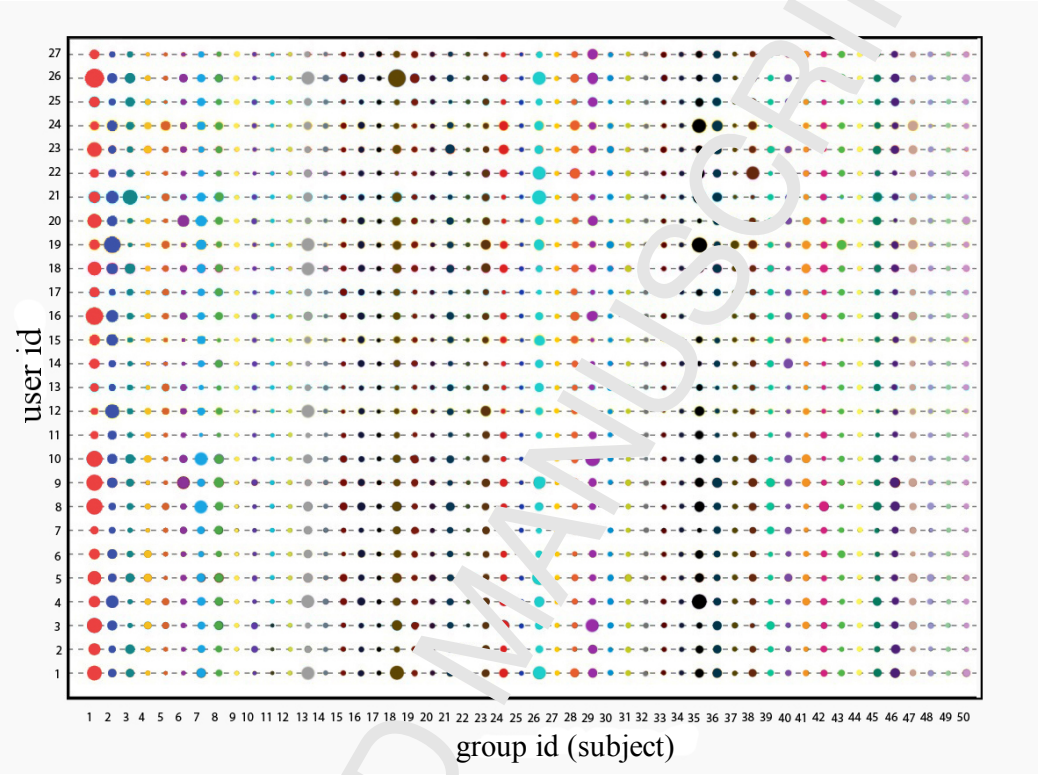
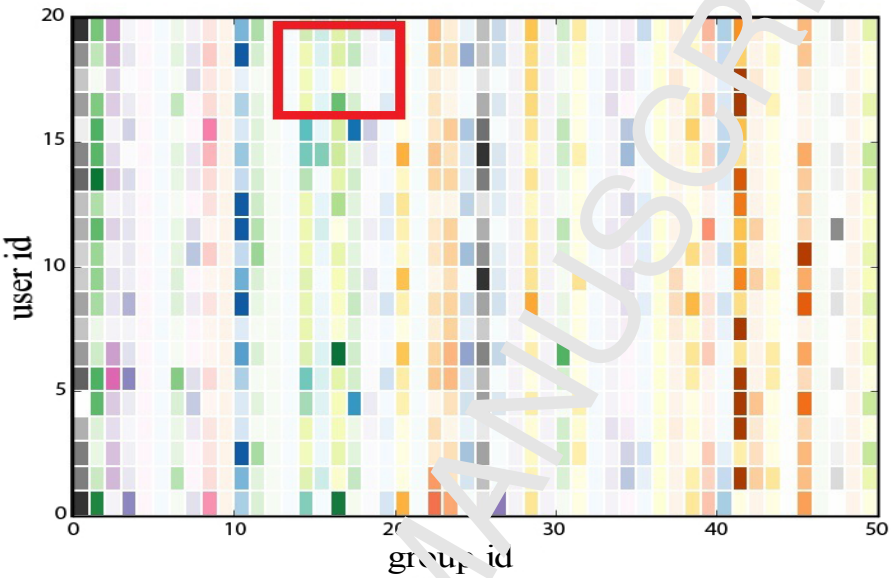
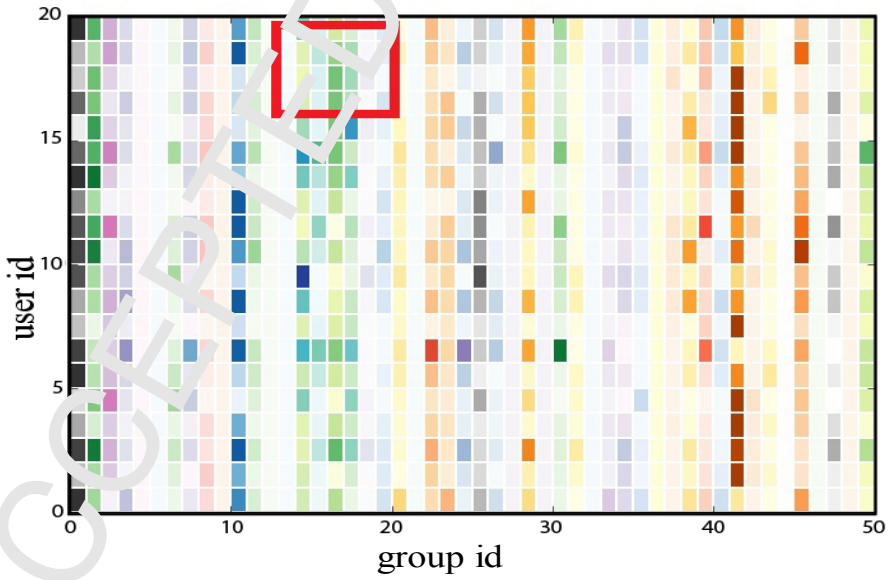


Fig. 2. Users' behavior in selecting from different groups. Filled circles, depending to their size, indicate the number of selections that are performed by a user from a specific group.



(a)



(b)

Fig. 3. (a) and (b) illustrate the influence of user preferences in selecting resources in two consecutive time intervals. Darker blocks indicate larger influence of the corresponding group.

3.2. Problem Statement

To facilitate the representation, Table 1 gives the description of notations used in this paper. To introduce the proposed approach, we refer to some concepts such as *resource*, *activity*, *user interests*, and *user preferences* which are defined below:

A *resource* is represented by a feature vector consisting of textual information. In our case study, the resources consist of news articles that are prepared after removing stop-words, stemming, and tokenizing. An *activity* is a triplet <*u,r,d*> that denotes a resource r was selected by user u at a specific date d. This selection has the role of an implicit feedback. Each group of related activities performed by a user u is referred to as an *interest*. We denote the i^{th} interest of u, with t_i. The probability that user u selects a resource from a specific interest is referred to as the *user preference*. In other words, user preferences define a probability distribution on user interests, indicating the priority of each interest. We denote the i^{th} preference with θ_i.

Table 1: Notations used to represent the proposed model.

notation	*description*
R	the set of resources
r	a resource
u	a user
d	date (or time)
A^u	activity vector of user u
i, j, k	used for indexing
t_i	the interest (cluster) in i^{th} activity performed by u
T^u	the set of interests of user u
$\mathcal{F}(.)$	the decay function
$\mathcal{D}(.)$	the similarity matrix
$\mathbf{\Theta}^u$	the preference distribution vector of user u
β	Dirichlet Distribution parameter for the priori of preference distribution
α	concentration parameter of ddCRP
G_0	base distribution of Dirichlet Process
Ψ	a set of model parameters

Let $U = \{u_1, u_2, ..., u_{|U|}\}$ be the set of all users, $R = \{r_1, r_2, ..., r_{|R|}\}$ be

the set of all resources, and $A^u = \{a_1, a_2, ..., a_{|A|}\}$ be the set of all activities that are performed by user u. Given A^u, our goal is to infer T^u, the set of user interests and Θ^u, the set of user preferences which together construct the user model. Afterwards, user interests (T^u) are utilized to find out candidate resources, and user preferences (Θ^u) are utilized to determine the priority of each candidate resource. The proposed model should adapt with new activities. In other words, user model should evolve over time by incorporating new activities and updating T^u and Θ^u.

3.3. Dirichlet Process Mixture Model

Clustering is widely used in recommender systems [36]. For example, clustering algorithms are applied in collaborative filtering to find similar users or resources (neighborhood). Determining the number of initial clusters, and updating clusters in a dynamic environment are two major challenges in clustering [48, 34].

Bayesian nonparametric models can be used in this context. Given an observed dataset, Bayesian nonparametric models suppose some latent factors (each representing a cluster) and use posterior inference to determine these factors. Also, the number of clusters (latent factors) is inferred from the data and can evolve as new data are observed [34].

DPMM is a family of nonparametric Bayesian models which can be used for evolutionary clustering. DPMM constructs a single mixture model in which the number of mixture components is infinite [36]. Given $S_1, ..., S_k$ as the partitions of the sample space and G_0 as the base measure, Dirichlet process (DP) constructs the distribution over the probability measure, $G \sim DP(\alpha, G_0)$, as follows:

$$(G(S_1), ..., G(S_k)) \sim Dir(\alpha G_0(S_1), ..., \alpha G_0(S_k)) \ , \tag{1}$$

where α is the concentration parameter and provides control over the variability around G_0. The greater value of α results in increasing the number of clusters and vise versa. Usually its value is estimated based on observations and is updated in the clustering process [49]. DP represents a distribution over the base distribution G_0. The base distribution G_0 can be Gaussian, multinomial or any context-dependent distribution [50]. DP provides a discrete distribution that can be used as prior probabilities for components of a mixture model. To define a DPMM, we assume observations X are generated from k clusters (user interests) T^u, where $k \to \infty$. For each observation x_i,

first we choose its cluster t_i and then generate x_i from the corresponding cluster parameter θ_i as follows:

$$G \sim DP(\alpha, G_0) \tag{2a}$$
$$\theta_i \mid G \sim G \tag{2b}$$
$$x_i \mid \theta_i \sim F(.) \tag{2c}$$

here, $F(.)$ determines a function such as *multinomial* or *categorical* that produces x_i using θ_i. Fig. 4 shows the graphical view of DPMM. The observations X are clustered according to the latent factors Θ. Distribution of clusters is followed by G that represents the DP over clusters; $G \sim DP(\alpha, G_0)$. This is a general representation of DPMM and it can be utilized in different perspectives such as *Stick-breaking construction* and *Chinese Restaurant Process* (CRP) [34, 36].

In this research, a DPMM-based clustering, called *distance dependent CRP* (ddCRP) [35] is used to determine the distribution over clusters. ddCRP uses a data similarity (distance) matrix for clustering similar resources. This similarity matrix is cumulated with the pair-wise similarity measure of resources. In the ddCRP the similar resources are connected to each other and create a cluster that is referred to as a table. The similarity factor can be spatial, temporal, or any other relevant characteristic that may be used to measure the similarity of two resources. ddCRP assigns the first table to the first resource. Afterwards, resource $\mathcal{I}$ is connected to resource $\mathcal{J}$, with the probability distribution of Eq 3.

$$p\left(\mathcal{I} \longrightarrow \mathcal{J} \mid Similarity, \alpha\right) = \begin{cases} Similarity(\mathcal{I}, \mathcal{J}) & \mathcal{I} \neq \mathcal{J} \\ \alpha & \mathcal{I} = \mathcal{J} \end{cases} \tag{3}$$

here, the $Similarity(\mathcal{I}, \mathcal{J})$ shows the similarity between resources $\mathcal{I}$ and $\mathcal{J}$. Interested readers are referred to [35] for more details.

4. The Proposed Recommender System

In this section, we describe the proposed recommender system. We first discuss the proposed user model in which user interests are extracted from the user profile as latent factors using ddCRP. ddCRP perspective of DPMM

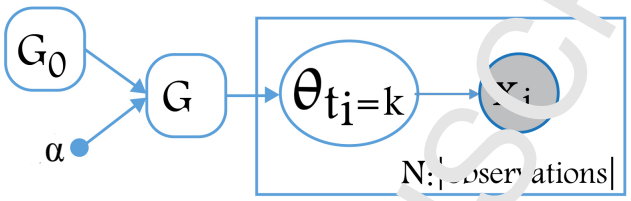


Fig. 4. Graphical view of DPMM. Each observation (x_i) is associated to a destination cluster (t_i). Cluster distribution is followed by a base distribution (G_0) and concentration parameter (α).

lets us construct a dynamic clustering model. Therefore, the user interests may evolve over time. In contrast to available methods that utilize the user interests [31], we infer the user preferences as the main factor that affects the priority of each interest. Calculating the priority of user interests over time is performed using the cluster parameter G_0, which results in dynamic user preferences. To incorporate our approach in a content-based recommender system, the graphical representation of the proposed user model is presented in Section 4.1. Also, the formulation and modeling process are described in Section 4.2. Finally, incorporating the proposed user model to construct a recommendation algorithm is conducted in Section 4.3.

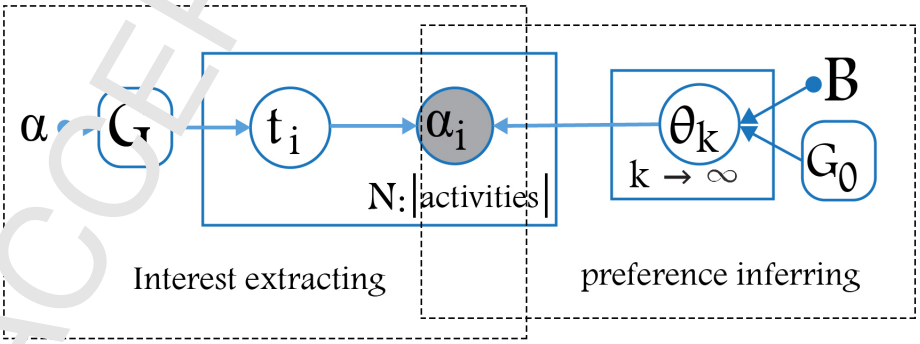


Fig. 5. Graphical view of ddCRP representation of DPMM.

4.1. Representation of User Preferences Modeling

In this section, we illustrate hierarchical Bayesian network (graphical view) of the proposed model and represent the generative process of model parameters. As mentioned earlier, we use the ddCRP representation of DPMM. Fig. 5 illustrates the graphical view of ddCRP representation of DPMM. According to this figure, the proposed user model has two main blocks: *interest extraction* and *preference inference*. The parameters are described below.

- *Hyper-parameter* α: this is the concentration parameter used in ddCRP and controls the distribution of latent clusters.
- *Model parameter* G: this parameter determines the distribution of latent clusters. It is sampled from the ddCRP ($G \sim ddCRP(\alpha)$).
- *Model parameter* t: this parameter indicates the cluster identifier for each observation. From the ddCRP perspective, this latent parameter is referred to the destination table of observations.
- *Parameter* a: this parameter shows the observed data.
- *Model parameter* θ: the features (characteristics) of individual latent clusters are represented with θ_i. For example, if each latent cluster is Gaussian, then $\theta_i = \{\mu_i, \sigma_i\}$.
- *Hyper-parameter* β *and base distribution* G_0: these parameters are used to define the prior distribution of the cluster parameter (θ_i).

We form the posteriori distribution over all latent variables and parameters, and obtain the estimation values of these parameters by approximating the posterior of the ddCRP. To determine the posteriori and prediction inference, we formulate the DPMM as follows:

$$G \sim ddCRP(\alpha) \tag{4a}$$
$$t_i \mid G \sim G(.) \tag{4b}$$
$$\theta \mid \beta, G_0 \sim Dir(\beta, G_0) \tag{4c}$$
$$a_i \mid t_i = z, \theta_z \sim Cat(\theta_z) \tag{4d}$$

also, the description of generative process of user activities is represented as follows:

- For each activity $a_i \in A^u$: draw table assignment t_i: $t_i \sim ddCRP(.)$.
- For each table $k(k \in \{1, 2, \cdots\})$: calculate the distribution of model parameter (θ) using the base measure G_0 and β : $\theta_k \sim Dir(.)$.
- For a new activity $a_x : t_x \sim ddCRP(.)$ and $a_x \sim Ca\ (\theta_{t_x})$.

To specify our user model, we consider the latent clusters (T) and model parameter (Θ) as the user interests and user preferences, respectively. Also, the user profile is used as the observed data (a). By incorporating a user profile into the posterior inference computation, the model parameters are estimated. In the following, we formulate the proposed user model and present the modeling and recommendation algorithms.

4.2. User Modeling Process

In this section, we use the posteriori inference to formulate the user model, then we provide an algorithm to construct the user model. We suppose each resource is represented as a feature vector that consists of textual information. The feature vectors of resources (that are cumulated in the user profile) are processed, then the *data similarity matrix*, $D(.)$ is constructed and utilized in ddCRP.
We form the posteriori distribution and perform calculations to estimate the parameters. Considering t_i^* as the queried table assignment for a selected resource a_i^*, t_{-i} is the destination table of other resources, and $\Psi = \{\alpha, G_0, D(.)\}$ is the set of hyper-parameters. Therefore, the posteriori distribution is formulated as the conditional distribution of Eq. 5.

$$p(t_i^* = k \mid t_{-i}, A^u, \Psi) = \frac{p(t_i^* = k \mid t_{-i}, \Psi) p(A^u \mid (t_i^* = k, t_{-i}), \Psi)}{\sum_{t'} p(A^u \mid t', \Psi) p(t' \mid \Psi)} \tag{5}$$

As illustrated in Eq. 5, the conditional distribution is intractable and we approximate the posteriori as follows:

$$\begin{aligned} &p(t_i^* = k \mid t_{-i}, A^u, \Psi) \propto \\ &\underbrace{p(t_i^* = k \mid t_{-i}, \Psi)}_{term1} \underbrace{p(a_i^* \mid A_{-i}^u, (t_i^* = k, t_{-i}), \Psi)}_{term2} \underbrace{p(A_{-i}^u \mid (t_i^* = k, t_{-i}), \Psi)}_{term3} \end{aligned} \tag{6}$$

the term denoted by $term3$ is the likelihood of observations, and it is constant while performing the table assignment of resource a_i^*. Therefore, the

posteriori is approximated as follows:

$$p(t_i^* = k \mid t_{-i}, A^u, \Psi) \propto \underbrace{p(t_i^* = k \mid t_{-i}, \Psi)}_{term1} \underbrace{p(a_i^* \mid A^u_{-i}, t_i^* = k, t_{-i}), \Psi)}_{term2} \quad (7)$$

In Eq. 7, the term denoted by $term1$ is used to calculate the probable table assignment of resource a_i^*. The table assignment of resource a_i^* is calculated using ddCRP distribution. According to ddCRP, resource a_i^* may be assigned to the existing tables based on similarity matrix $\mathcal{D}(.)$ or to a new table proportional to the concentration parameter α. This distribution is formulated in Eq. 8

$$p\left(t_i^* = k \mid \mathcal{D}(.), \alpha\right) = \begin{cases} \mathcal{D}(a_i, a_{-i}) & k \in t_{-i} \\ \alpha & t_i^* \neq t_{-i} \end{cases} \quad (8)$$

Also, in Eq. 7, the term denoted by $term2$ indicates the likelihood of resource a_i^* under its destination table. Suppose $t_i = l$ is the previous destination of resource a_i^* and $t_i^* = k$ is the new destination of resource a_i^*. This likelihood is calculated under three conditions. If resource a_i^* is assigned to its previous table ($k = l$), or a_i^* creates a new table ($t_i^* \neq t_{-i}$), no change is made in the likelihoods. If resource a_i^* is assigned to another existing table (k is the existing table and $k \neq l$), table assignment of resource a_i^* makes a table split. Hence, the likelihoods are changed. Consequently, we formulate this term as follows:

$$\begin{aligned} &p(a_i^* \mid Z(k)_{-i}, Z(l)_{-i}, \Psi) = \\ &\frac{p(a_i^*, Z(k)_{-i}, Z(l)_{-i} \mid \Psi)}{p(Z(k)_{-i}, Z(l)_{-i} \mid \Psi)} = \frac{p(Z(k,l) \mid \Psi)}{p(Z(k)_{-i} \mid \Psi) p(Z(l)_{-i} \mid \Psi)} \end{aligned} \quad , \quad (9)$$

where $Z(x)$ indicates the resources placed in table x ($Z(x) = \{a_j \mid t_j == x\}$) and $p(Z(.) \mid \Psi)$ shows the likelihood of observations which is calculated as follows:

$$p(Z(x) \mid \Psi) = \sum_{k=1}^{K} \underbrace{p(Z(x)|\theta_k)}_{Cat(Z(x) \mid \Theta)} \times \underbrace{p(\theta_k \mid \Psi)}_{Dir_\Theta(\beta)} \quad (10)$$

$$p(t_i^* = k \mid A^u, \Psi) = \begin{cases} \alpha & t_i^* \neq t_{-i} \\ \mathcal{D}(a_i, a_{-i}) & k = l \\ \mathcal{D}(a_i, a_{-i}) \times \frac{p(Z(k,l)|\Psi)}{p(Z(k)_{-i}|\Psi) p(Z(l)_{-i}|\Psi)} & k \neq l \end{cases} \quad (11)$$

To approximate the posteriori inference, Gibbs sampling [25] which is one of Markov chain Monte Carlo method is used. The Markov chain relies on Gibbs updates, where each variable is updated in turn by sampling from its posterior distribution conditional on all other variables. Algorithm 1 illustrates the Gibbs sampling algorithm that is used to inference parameters. By considering Eq. 8 and Eq. 10, we rewrite Eq. 7 as Eq. 11. We use Eq. 11 to approximate the posteriori inference within Gibbs sampling.

In Gibbs sampling iterations, after determining the table assignment of the i^{th} activity, the latent clusters (tables) can be modified implicitly. Then, we modify the hyper-parameter β and model parameter Θ using Eq. 13 and Eq. 14, respectively. For each latent cluster (table) t, hyper-parameter β is proportional to the activities that are assigned to cluster t. The fitness function $\mathcal{F}(.)$ is used to calculate the influence of user activities as follows:

$$\mathcal{F}(activity) = exp(-Time(activity)/|A^u|). \tag{12}$$

This function is proportional to the timestamp of each user activity. In other words, the newer activities have a higher influence in user preferences and vise versa. We applied the Dirichlet distribution for the priori of model parameter Θ; hence, we use the posteriori of Dirichlet distribution for updating model parameters. The updated β is used to modify the model parameter Θ:

$$\beta_j = \sum_{activity \in table_j} \mathcal{F}(activity) \tag{13}$$

$$(\theta_1^*, ..., \theta_k^*) \sim Dir(\beta^*, G_0) \tag{14}$$

In the above mentioned equations, the base distribution (G_0) and concentration parameter (α) are used for model parameter (Θ) estimation and latent clusters aggregation, respectively. They are described below.

Base distribution. Dirichlet process distribution divides the problem space into measurable partitions. The base distribution G_0 is used as the measure of partitions. In our approach, Dirichlet process is used to model the distribution of user preferences. We formulate the base measure G_0 as follows:

$$G_0(table_i) = \frac{\sum_{activity \in table_i} \mathcal{F}(activity)}{\sum_{j=1}^{|S|}(\sum_{activity \in table_j} \mathcal{F}(activity))}. \tag{15}$$

Base measure G_0 constructs a probability function that is defined for each table $table_i$ and Dirichlet process is defined for the distribution of these probability functions.

Concentration parameter. The distribution of latent clusters are controlled using the concentration parameter α. In this research, the activity clustering is performed based on their content similarity. Therefore, the concentration parameter is proportional to the similarity between the activities. We initiate the concentration parameter heuristically with the average of similarity matrix ($\mathcal{D}(.)$) and update its value in the iterations of Gibbs sampling [41]. We update concentration parameter α and choose the result with the highest likelihood.

4.3. Recommendation Process

By approximating posteriori inference, the parameters, hyper-parameters and the latent variables are estimated. To predict favorite future resources, we perform the two following actions.

(i) ddCRP is used to categorize the candidate resources. A candidate resource r_j is incorporated in Eq. 8. to find out its destination cluster t. In other words, resource r_j is assigned to the user interest (destination cluster) t with a high similarity.

(ii) After determining the related user interest (latent cluster), we use the model parameter (θ) to determine the user preference (priority) conditioned in the interest t. To calculate the user preference for resource r_j, Eq. 16 is used. We use the result of Eq. 16 to rank the candidate resource and recommend the ordered list to the user.

$$p(r_j \mid A^u, \Psi) = \frac{1}{p(A^u \mid \Psi)} \sum_{k=1}^{K} p(t_j^* = k \mid \Psi) \times p(r_j, A^u \mid \theta_k, \Psi) \times p(\theta_k \mid \Psi) \tag{16}$$

We apply the above actions to candidate resources and construct the recommendation list. Algorithm 2 illustrates the recommendation process.

Algorithm 1 Gibbs sampling algorithm for approximating parameters

Input: A^u, $\Psi = \{\alpha, \beta, \mathcal{D}(.), G_0\}$.
Output: model parameter Θ as user preferences distribution
and latent clusters (tables) $\mathbf{T}$ with their members.
Initialization:
N= $size(A^u)$.
Initialize the latent clusters (tables) $\mathbf{T}$.
Randomly place activities on initial tables.
while *not converged* **do**
for $i = 1$ to N **do** ▷ Iterates for each activity
Select i^{th} activity and ignore its table assignment.
$dist\_vector = \{\}$
for $k = 1$ to $\mid \mathbf{T} \mid$ **do** ▷ Iterates for each table t_k
p_{ik} = Compute the probability of assigning the i^{th} activity
to the k^{th} table with Eq. (11).
$dist\_vector = dist\_vector \bigcup p_{ik}$
end for
t_i^* = Choose the destination table of the i^{th} activity
according to $dist\_vector$.
if t_i^* is a new table **then**
$\mathbf{T} = \mathbf{T} \cup \{t_i^*\}$
end if
Update hyper-parameter β using Eq. 13.
Compute model parameter θ using Eq. 14.
Update hyper-parameter α.
end for
end while

Algorithm 2 Prediction process to provide the recommendations.

Input: parameters that specify user model,
such as latent clusters ($\mathbf{T}$), preference distribution (Θ)
and the resource candidate list.
Output: An ordered list for recommendation.
Initialization:
N= $size(candidate\_list)$.
$priority\_list = \{\}$
for $j = 1$ to N **do** ▷ Iterates for each candidate activity
r_j = Select j^{th} resource from $candidate\_list$.
t_j = Choose the destination table of resource r_j using Eq. 8.
p_j = Calculate the likelihood of activity r_j given destination table t_j
using Eq. 16.
$priority\_list = priority\_list \cup \{r_j, p_j\}$
end for
Return Descending sort of $priority\_list$.

5. Experimental Results

We implemented the proposed user model to construct a content-based recommender system. To evaluate the proposed method, we performed a set of experiments using two datasets (introduced in Section 5.3). We calculated the evaluation metrics and compared the results with some of the state-of-the-art recommendation algorithms (presented in Section 5.3). In the following, we describe the dataset and the implementation framework, evaluation metrics, baseline methods, and experiments. Then, we analyze performance of the proposed method on the datasets.

5.1. Dataset and Implementation framework

The proposed method is evaluated using two real-world datasets, MovieLens[1] and NewsTweet[2]. MovieLens is a common reference dataset which is often used in the recommender systems researchs. Each record represents the users' 5-star rating on movies in a specific time, as a (user, resources, rate, timestamp) quaternary. To provide the content information for movies,

[1] https://grouplens.org/datasets/movielens
[2] http://ipdm.shahroodut.ac.ir/downloads/link

we gathered the movie story-line and genre from IMDb[3] for each movie item within the dataset. Then the movie information was processed using text processing tools (such as NLTK [51]) and the textual feature vector was constructed for each movie. NewsTweet was constructed with tweets along with users' access history from popular news channels of Twitter[4] including BBC, CNN, Associated Press, Reuters, and New York Times from May 2017 to August 2017. We used the available APIs of twitter to construct this dataset. Each recored in this dataset consists of {user-id (anonymous), tweet-text, and timestamp}. The statical parameters of these two databases are represented in Table 2.

We provided a multi-process programming framework with Python to construct the proposed method. The experiments were performed using an Intel Core i28 CPU with 30 Gbytes of memory. The computational complexity of user model construction depends on the number of user activities. The model construction is performed per each individual user. Therefore, the running time can be estimated with $O(k \times n)$ in which n being the number of user activities and k is the convergence iteration of Algorithm 1. We can determine an upper bound for convergence iteration. Therefore, the convergence iteration is performed until at least one of the following criteria is reached: excessed the upper bound or a stabled likelihood (equivalent of two consecutive likelihoods). We performed the model construction for users with different number of the activities. Fig 6 shows the average running time for model constructing based on the number of activities associated to each user. We choose the number of activities in range [200..1000] with step 100. The results indicate that the running time is increased linearly with the number of activities. We implemented the recommendation service using three main modules, model constructor, recommender, and evaluator. The model constructor employs the dataset to construct the proposed user model. The recommender uses the user model to provide recommendations. The evaluator calculates the metrics and evaluates the recommendations. The process of constructing and evaluating the recommendation service is depicted in Fig 7.

[3] http://www.imdb.com
[4] https://www.twitter.com

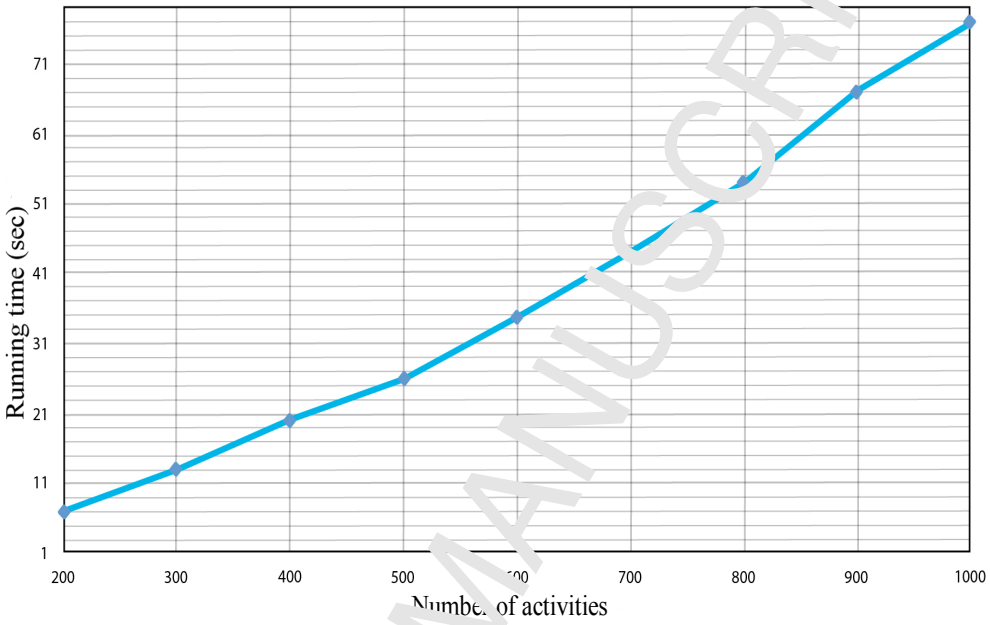


Fig. 6. The running time to construct user model for different number of activities.

5.2. Evaluation metrics

The proposed method recommends a list of resources to the user. In other words, it provides a list of top@N resources to the user. We use the accuracy metrics including *precision*, *recall*, and $F1-measure$ to evaluate the proposed method [52]. Also the $DCG@N$ metric [7, 47] is used to measure the usefulness, or gain of a resource based on its position in the recommended list. We calculate these metrics for different lengths of recommendation list including $top@N = \{5,\ 10,\ 15,\ 20\}$.

5.3. Baseline Methods and Toolkits

We compare our proposed method with other state-of-the-art methods that are incorporated for constructing a recommender system. These methods include the item-item based collaborative filtering [53], matrixfactorization (SVD) [21], Funk-SVD [54], timeSVD [55], non-negative matrix-factorization (NMF) [28], content-based filtering [9], and LOGO [46, 31]. The item-item based collaborative considers the group interests but does

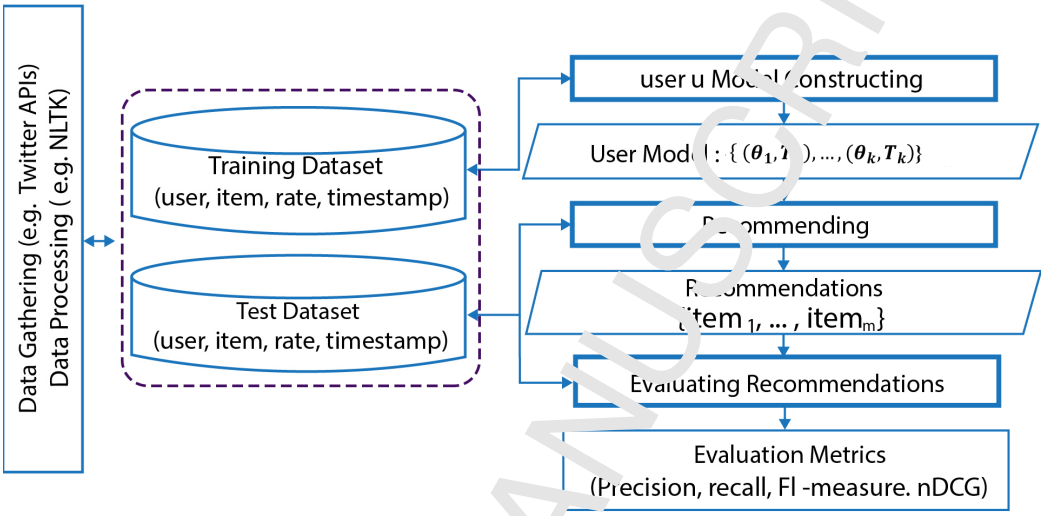


Fig. 7. The process of constructing and evaluating the proposed recommender system.

Table 2: properties of the datasets.

Properties	Datasets	
	MovieLens	*NewsTweet*
number of users	4743	2018
number of resources	8787	46936
number of feedbacks	1M	376085

not include preferences. SVD, timeSVD, funk-SVD and NMF use the latent factors to capture the user interests. LOGO and content-based filtering use the content similarity of resources (new resources and user profile) and perform recommendation based on the individual user interests. In contrast to all of the above mentioned methods, we employ the individual user profile and model the user preferences as well as user interests. Among these methods, timeSVD and LOGO are temporal recommender system. These methods are implemented in popular toolkits in the recommender systems context. The item-item based filtering, funk-SVD and content-based filtering are implemented in LensKit[5] [56]. The SVD and NMF are implemented in

[5]http://lenskit.grouplens.org/

Surprise[6] [57]. Also, the timeSVD is implemented in LibRec[7] [58]. We used these toolkits (LensKit, Surprise, LibRec) to incorporate the above methods and provide comparison results. There is not a public implementation of LOGO, and we implemented it with Python.

5.4. Experiments and Results

We performed several experiments for evaluating different aspects of the proposed recommender system (RS, namely), including *recommendation accuracy*, *preference impact*, *adaptivity* (*temporality*), and *concentration impact*.

Recommendations accuracy. We evaluate accuracy of the RS and perform comparison using the two datasets. We executed the RS several times (up to fifty times) and calculated the evaluation metrics per run. The results of metrics calculation using New Tweet dataset for different executions are illustrated in Fig. 8. Values of F1-measure and DCG are depicted in Fig. 8a and Fig. 8b, respectively. The averages of F1-measure and DCG are highlighted in Fig. 8. Also, we implemented the LOGO algorithm and compared its results with the results of the RS. The numerical results are depicted in Table 3.

We employed the MovieLens dataset to evaluate RS. Also, the evaluation metrics of the selected baseline methods (described in Section 5.3) are presented. The results of evaluation metrics are depicted in Table 4. According to the accuracy metrics such as F1-measure (Table 3 and Table 4), the RS provides better recommendations compared to other state-of-the-art methods.

The results indicate that the proposed recommender system has a desirable performance compared to the existing methods. The results confirm that using preferences to capture the priority of user interests improves the performance of recommender system. Moreover, the results show that using DPMM to construct user model could identify the user preferences effectively.

Preference impact. To illustrate the impact of the user preference on the improvement of recommendation accuracy, we execute the RS in two cases: *interest based* and *preference based*. For *interest based*, we only used the interests extraction block of the user model that is depicted in Fig. 5.

[6] http://surprise.readthedocs.io/en/stable
[7] https://www.librec.net/

Table 3: Performance comparison of the proposed recommender system (RS) and LOGO methods using various Evaluation Metrics.

top@N	Method	precision	recall	F1 ± *Variance*	nDCG
5	LOGO	0.35	0.10	0.14	0.37
	RS	0.87	0.14	**0.24** ± 0.0003	**0.87**
10	LOGO*	0.21	0.25	0.22	
	LOGO	0.32	0.16	0.19	0.44
	RS	0.88	0.29	**0.43** ± 0.0009	**0.87**
15	LOGO	0.31	0.22	0.25	0.47
	RS	0.80	0.40	**0.53** ± 0.0015	**0.82**
20	LOGO*	0.27	0.36	0.31	
	LOGO	0.31	0.25	0.27	0.49
	RS	0.75	0.50	**0.59** ± 0.0025	**0.78**

*LOGO**: The original results of [46].

For *preference base*, we used the whole user model. In this experiment, we used MovieLens dataset, and calculated the F1-measure and nDCG metrics for these executions and depicted them in Fig 9. The results highlight the importance of preference modeling. Merely considering interests without their priority from the user viewpoint produces less accuracy results in predicting future resources. Therefore, incorporating both user interests and preferences for user modeling can improve the recommendation performance.

Adaptivity. To examine the influence of temporal context on the user preferences, we divided the training data into two continuity disjoint partitions by means of the *timestamps* (*date*) feature. We used the first partition of dataset to create user model and evaluated the metrics. Then, we applied the second partition of dataset to update the user model, and again evaluated the metrics. The process of analyzing the adaptivity behavior of the proposed user model is illustrated in Fig. 10. In this experiment we used the MovieLens dataset in which the last partition of the dataset for each user is the test data. The results of F1-measure and DCG metric are depicted in Fig. 11. As illustrated in Fig. 11 the performance of the RS is persistent over time. In other words, the RS can evolve over time and capture the changes of user preferences to specify user model accurately. Also, we depicted the changes of user interests and user preferences for two randomly users in Fig. 12b. This figure shows the number of users interests and their associated preferences that are created in each partition. As observed, the

Table 4: Comparison of the *precision* and *NDCG* metrics for the proposed RS and other methods.

top@N	Method	precision	nDCG
5	Lenskit: item-item	0.364	0.318
	Lenskit: Funk-SVD	0.193	0.305
	Lenskit: Content-based	0.234	0.251
	Surprise: SVD	0.177	0.230
	Surprise: NMF	0.168	0.210
	LibRec: TimeSVD	0.234	0.251
	proposed RS	0.753	0.461
10	Lenskit: item-item	0.462	0.492
	Lenskit: Funk-SVD	0.454	0.481
	Lenskit: Content-based	0.399	0.423
	Surprise: SVD	0.347	0.409
	Surprise: NMF	0.328	0.383
	LibRec: TimeSVD	0.40	0.422
	proposed RS	0.771	0.595
15	Lenskit: item-item	0.572	0.610
	Lenskit: Funk-SVD	0.565	0.601
	Lenskit: Content-based	0.510	0.550
	Surprise: SVD	0.473	0.535
	Surprise: NMF	0.455	0.509
	LibRec: TimeSVD	0.510	0.550
	proposed RS	0.689	0.656

level of interest may change when new user activities are introduced to the model. Accordingly, the preference of each interest may also change. As depicted for one sample user, the number of interests is increased, while for the second user, this number is decreased when new data is observed.

Concentration impact. We performed a scanning process to check sensitivity of our recommender system to the concentration parameter α. As described in Section 4.1, we determine the average value of the similarity matrix $P(.)$. Then, a range of values around this average is selected for α in the experiments. The RS is executed for four different subset of users. The initial α and final α of the trained model for each user are shown in Fig. 13. Also, the *Cosine similarity* metric was calculated for these vectors and the results are shown in this figure. The result indicates the high correlation between

initial α and final α of the trained model. The results show that initializing α with the average similarity matrix provides a higher performance for the RS compared to our previous work [47] that used experiments (k-folding) to assign a fixed value to α.

As the experiments imply, there are three main achievements in this research. First, the accuracy results show that the proposed method for user modeling is able to identify user's preferences and improves the performance of the recommender system. In other words, the proposed method can capture user's behavior using the DPMM framework. Second, the adaptivity results indicate that the performance of the RS is persistence with emerging new user activities. It can evolve over time without constructing the model from scratch. Finally, the analysis of concentration parameter shows that the RS can determine this parameter according to the dataset adaptively and dose not use a fixed predefined value for the concentration parameter.

6. Conclusions

In this paper, we presented an adaptive method for content-based recommender system. In the proposed model, both user interests and preferences are considered. Our studies showed that user preference is an important factor in selecting resources by the users. A DPMM framework was used to model user preferences and to capture user behavior in selecting resource during time. We used the ddCRP representation of DPMM to cluster user activities, to calculate the distribution of user preferences, and to update them over time. Experimental results show that the proposed method improves the accuracy of the content-based recommender system compared to existing methods. According to these results, the proposed method improves the accuracy of recommendations by incorporating the user preferences. Also, the proposed method has the minimum model hyper-parameters (which are calculated semi-automatically) related to the other methods.

For future works, some additional influential parameters can be used in the base distribution to improve the proposed method. For example, we can consider the user's rate feedback to capture the preference of a user related to each latent factors (interests). Also, the collaborative approach can be considered to improve the diversity measure of recommendation via handling the group preferences. Furthermore, the concentration parameter may be updated in an evolutionary process.

References

[1] Greg Linden, Brent Smith, and Jeremy York. Amazon.com recommendations: Item-to-item collaborative filtering. *IEEE Internet computing*, 7(1):76–80, 2003.

[2] Gediminas Adomavicius and YoungOk Kwon. Improving aggregate recommendation diversity using ranking-based techniques. *IEEE Transactions on Knowledge and Data Engineering*, 24(5):896–911, 2012.

[3] Charu C Aggarwal. An introduction to recommender systems. In *Recommender Systems*, pages 1–28. Springer, 2016.

[4] Charu C Aggarwal. *Recommender systems*. Springer, 2016.

[5] Jesús Bobadilla, Fernando Ortega, Antonio Hernando, and Abraham Gutiérrez. Recommender systems survey. *Knowledge-based systems*, 46:109–132, 2013.

[6] Nan Zheng and Qiudan Li. A recommender system based on tag and time information for social tagging systems. *Expert Systems with Applications*, 38(4):4575–4587, 2011.

[7] Hao Wu, Yijian Pei, Bo Li, Zongzhan Kang, Xiaoxin Liu, and Hao Li. Item recommendation in collaborative tagging systems via heuristic data fusion. *Knowledge-Based Systems*, 75:124–140, 2015.

[8] Michael J Pazzani and Daniel Billsus. Content-based recommendation systems. In *The adaptive web*, pages 325–341. Springer, 2007.

[9] Pasquale Lops, Marco De Gemmis, and Giovanni Semeraro. Content-based recommender systems: State of the art and trends. In *Recommender systems handbook*, pages 73–105. Springer, 2011.

[10] Cosimo Palmisano, Alexander Tuzhilin, and Michele Gorgoglione. Using context to improve predictive modeling of customers in personalization applications. *IEEE transactions on knowledge and data engineering*, 20(11):1535–1549, 2008.

[11] Walter Carrer-Neto, María Luisa Hernández-Alcaraz, Rafael Valencia-García, and Francisco García-Sánchez. Social knowledge-based recommender system. application to the movies domain. *Expert Systems with applications*, 39(12):10990–11000, 2012.

[12] Robin Burke. Hybrid recommender systems: Survey and experiments. *User modeling and user-adapted interaction*, 12(4):331–370, 2002.

[13] Heung-Nam Kim, Abdulmajeed Alkhaldi, Abdulmotaleb El Saddik, and Geun-Sik Jo. Collaborative user modeling with user-generated tags for social recommender systems. *Expert Systems with Applications*, 38(7):8488–8496, 2011.

[14] Panagiotis Symeonidis, Alexandros Nanopoulos, and Yannis Manolopoulos. Feature-weighted user model for recommender systems. In *International Conference on User Modeling*, pages 97–106. Springer, 2007.

[15] Susan Gauch, Mirco Speretta, Aravind Chandramouli, and Alessandro Micarelli. User profiles for personalized information access. *The Adaptive Web*, pages 54–89, 2007.

[16] Gawesh Jawaheer, Peter Weller, and Patty Kostkova. Modeling user preferences in recommender systems: A classification framework for explicit and implicit user feedback. *ACM Transactions on Interactive Intelligent Systems (TiiS)*, 4(2):8, 2014.

[17] Hongzhi Yin, Bin Cui, Ling Chen, Zhiting Hu, and Xiaofang Zhou. Dynamic user modeling in social media systems. *ACM Transactions on Information Systems (TOIS)*, 33(3):10, 2015.

[18] Peter Brusilovsky and Eva Millán. User models for adaptive hypermedia and adaptive educational systems. In *The adaptive web*, pages 3–53. Springer, 2007.

[19] Haifeng Liu, Zheng Hu, Ahmad Mian, Hui Tian, and Xuzhen Zhu. A new user similarity model to improve the accuracy of collaborative filtering. *Knowledge-Based Systems*, 56:156–166, 2014.

[20] J Ben Schafer, Dan Frankowski, Jon Herlocker, and Shilad Sen. Collaborative filtering recommender systems. In *The Adaptive Web*, pages 291–324. Springer, 2007.

[21] Yehuda Koren, Robert Bell, and Chris Volinsky. Matrix factorization techniques for recommender systems. *Computer*, 42(8), 2009.

[22] Abhinandan S Das, Mayur Datar, Ashutosh Garg, and Shyam Rajaram. Google news personalization: scalable online collaborative filtering. In *Proceedings of the 16th international conference on World Wide Web*, pages 271–280. ACM, 2007.

[23] Yi Cai, Ho-fung Leung, Qing Li, Huaqing Min, Jie Tang, and Juanzi Li. Typicality-based collaborative filtering recommendation. *IEEE Transactions on Knowledge and Data Engineering*, 26(3):766–779, 2014.

[24] Charu C Aggarwal. Content-based recommender systems. In *Recommender Systems*, pages 139–166. Springer, 2016.

[25] Nachiketa Sahoo, Param Vir Singh, and Tridas Mukhopadhyay. A hidden markov model for collaborative filtering. *MIS Quarterly*, 36(4):1329–1356, 2012.

[26] Kai Yu, Anton Schwaighofer, Volker Tresp, Xiaowei Xu, and H-P Kriegel. Probabilistic memory-based collaborative filtering. *IEEE Transactions on Knowledge and Data Engineering*, 16(1):56–69, 2004.

[27] Chaotao Chen and Jiangtao Ren. Forum latent dirichlet allocation for user interest discovery. *Knowledge-Based Systems*, 126:1–7, 2017.

[28] Xin Luo, Mengchu Zhou, Yunni Xia, and Qingsheng Zhu. An efficient non-negative matrix-factorization-based approach to collaborative filtering for recommender systems. *IEEE Transactions on Industrial Informatics*, 10(2):1273–1284, 2014.

[29] Carlos A Gomez-Uribe and Neil Hunt. The netflix recommender system: Algorithms, business value, and innovation. *ACM Transactions on Management Information Systems (TMIS)*, 6(4):13, 2016.

[30] Yehuda Koren. Collaborative filtering with temporal dynamics. *Communications of the ACM*, 53(4):89–97, 2010.

[31] Lei Li, Li Zheng, Fan Yang, and Tao Li. Modeling and broadening temporal user interest in personalized news recommendation. *Expert Systems with Applications*, 41(7):3168–3177, 2014.

[32] David M Blei, Andrew Y Ng, and Michael I Jordan. Latent dirichlet allocation. *Journal of machine Learning research*, 3(Jan):993–1022, 2003.

[33] Katherine L Spangler. Reading interests vs. reading preferences: using the research. *The Reading Teacher*, 36(9):876–878, 1983.

[34] Samuel J Gershman and David M Blei. A tutorial on bayesian nonparametric models. *Journal of Mathematical Psychology*, 56(1):1–12, 2012.

[35] David M Blei and Peter I Frazier. Distance dependent chinese restaurant processes. *Journal of Machine Learning Research*, 12(Aug):2461–2488, 2011.

[36] Yee Whye Teh, Michael I. Jordan, Matthew J. Beal, and David M. Blei. Hierarchical dirichlet processes. *User modeling and user-adapted interaction*, 101(476):1566–1581, 2006.

[37] Nour El Islem Karabadji, Sabrina Belaqjoudi, Hassina Seridi, Sabeur Aridhi, and Wajdi Dhifli. Improving memory-based user collaborative filtering with evolutionary multi-objective optimization. *Expert Systems with Applications*, 2018.

[38] Charu C Aggarwal. Neighborhood-based collaborative filtering. In *Recommender Systems*, pages 29–70. Springer, 2016.

[39] Gediminas Adomavicius and Alexander Tuzhilin. Toward the next generation of recommender systems: A survey of the state-of-the-art and possible extensions. *IEEE transactions on knowledge and data engineering*, 17(6):734–749, 2005.

[40] Baiyu Chen, Zi Yang, and Zhouwang Yang. An algorithm for low-rank matrix factorization and its applications. *Neurocomputing*, 275:1012–1020, 2018.

[41] Charu C Aggarwal. Time-and location-sensitive recommender systems. In *Recommender Systems*, pages 283–308. Springer, 2016.

[42] Pedro G Campos, Fernando Díez, and Iván Cantador. Time-aware recommender systems: a comprehensive survey and analysis of existing evaluation protocols. *User Modeling and User-Adapted Interaction*, 24(1-2):67–119, 2014.

[43] Ling Cai, Jun Xu, Ju Liu, and Tao Pei. Integrating spatial and temporal contexts into a factorization model for poi recommendation. *International Journal of Geographical Information Science*, 32(3):524–546, 2018.

[44] Yali Si, Fuzhi Zhang, and Wenyuan Liu. Ctf-ara: An adaptive method for poi recommendation based on check-in and temporal features. *Knowledge-Based Systems*, 128:59–70, 2017.

[45] Liang Xiang, Quan Yuan, Shiwan Zhao, Li Chen, Xiatian Zhang, Qing Yang, and Jimeng Sun. Temporal recommendation on graphs via long- and short-term preference fusion. In *Proceedings of the 16th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 723–732. ACM, 2010.

[46] Lei Li, Li Zheng, and Tao Li. Logo: a long-short user interest integration in personalized news recommendation. In *Proceedings of the fifth ACM conference on Recommender systems*, pages 317–320. ACM, 2011.

[47] Bagher Rahimpour Cami, Hamid Hassanpour, and Hoda Mashayekhi. User trends modeling for a content-based recommender system. *Expert Systems with Applications*, 87(30):209–219, 2017.

[48] Chhavi Rana and Sanjay Kumar Jain. An evolutionary clustering algorithm based on temporal features for dynamic recommender systems. *Swarm and Evolutionary Computation*, 14:21–30, 2014.

[49] Thomas Minka. Estimating a dirichlet distribution. *Technical report M.I.T.*, 2000.

[50] Dilan Görür and Carl Edward Rasmussen. Dirichlet process gaussian mixture models: Choice of the base distribution. *Journal of Computer Science and Technology*, 25(4):653–664, 2010.

[51] Steven Bird. Nltk: the natural language toolkit. In *Proceedings of the COLING/ACL on Interactive presentation sessions*, pages 69–72. Association for Computational Linguistics, 2006.

[52] Iman Avazpour, Teerat Pitakrat, Lars Grunske, and John Grundy. Dimensions and metrics for evaluating recommendation systems. In *Recommendation systems in software engineering*, pages 245–273. Springer, 2014.

[53] Mukund Deshpande and George Karypis. Item-based top-n recommendation algorithms. *ACM Transactions on Information Systems (TOIS)*, 22(1):143–177, 2004.

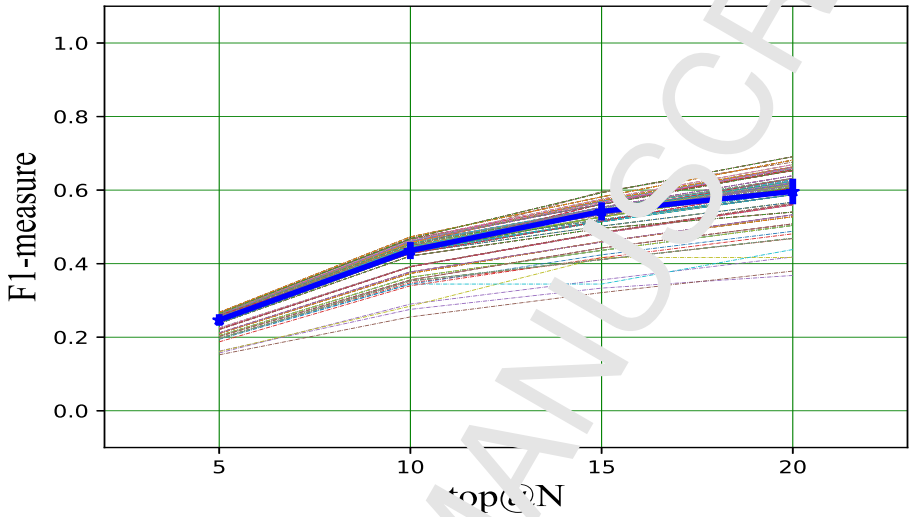
[54] Ludovik Çoba and Markus Zanker. Replication and reproduction in recommender systems research-evidence from a case-study with the rrecsys library. In *International Conference on Industrial, Engineering and Other Applications of Applied Intelligent Systems*, pages 305–314. Springer, 2017.

[55] Yehuda Koren and Robert Bell. Advances in collaborative filtering. In *Recommender systems handbook*, pages 77–118. Springer, 2015.

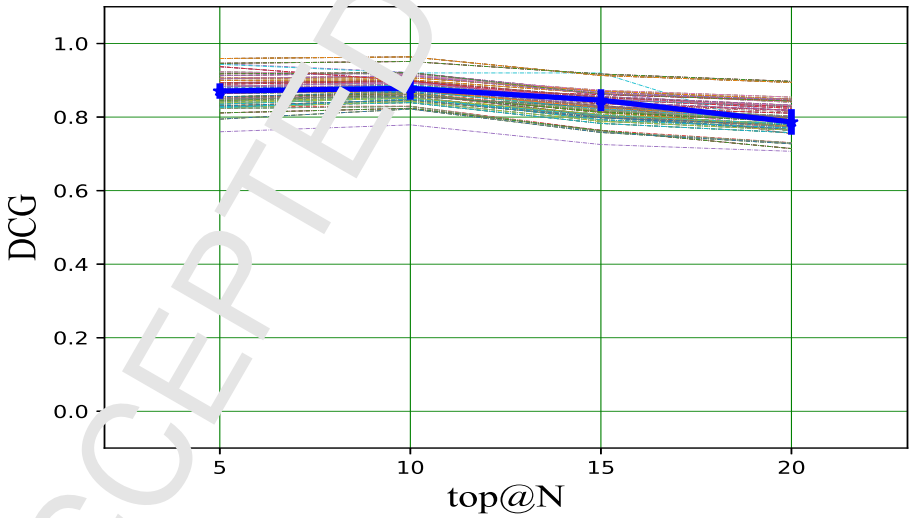
[56] Michael D Ekstrand, Michael Ludwig, Joseph A Konstan, and John T Riedl. Rethinking the recommender research ecosystem: reproducibility, openness, and lenskit. In *Proceedings of the fifth ACM conference on Recommender systems*, pages 133–140. ACM, 2011.

[57] Nicolas Hug. Surprise, a python library for recommender systems., 2017.

[58] Guibing Guo, Jie Zhang, Zhu Sun, and Neil Yorke-Smith. Librec: A java library for recommender systems. In *UMAP Workshops*, 2015.

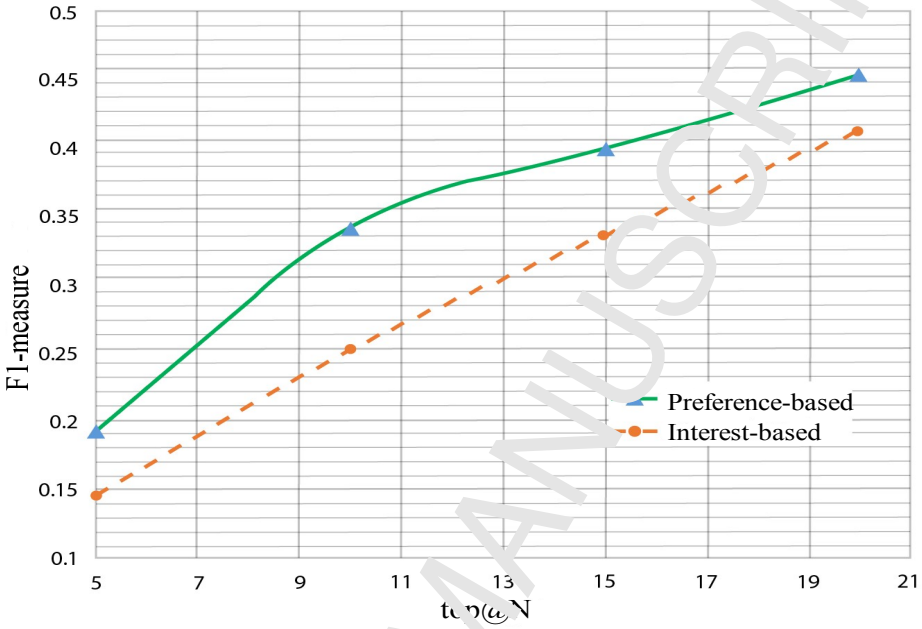


(a)

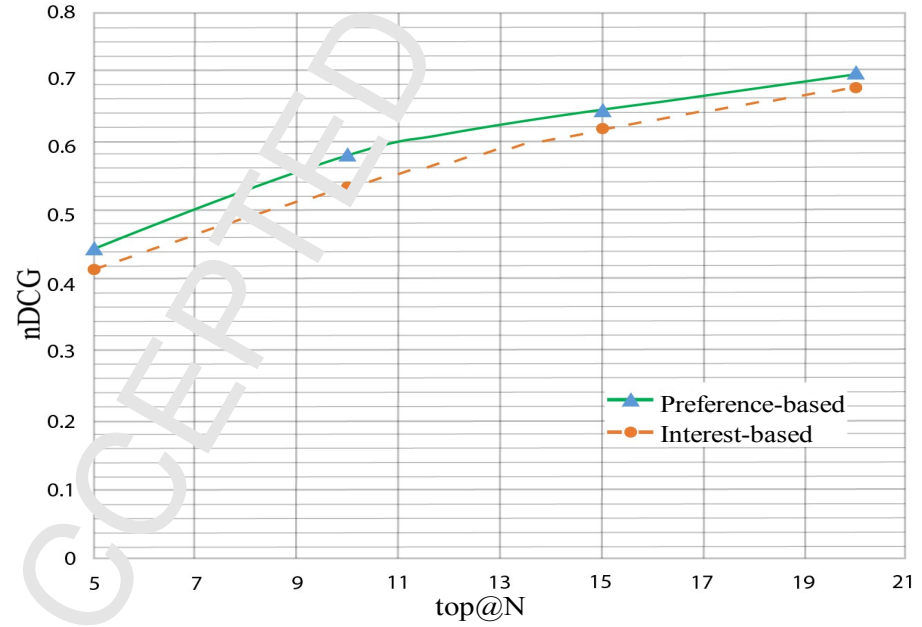


(b)

Fig. 8. Evaluation metrics for different executions using NewsTweet dataset. Bold-line represents the average of evaluation metrics. The results of each individual execution are depicted with a dashed-line.



(a) F1-measure



(b) DCG metric.

Fig. 8. Preference impact: the evaluation metrics for two different executions, *interest based* and *preference base* using MovieLens dataset. Dashed-line and solid-line represent the *interest based* and *preference based* results, respectively.

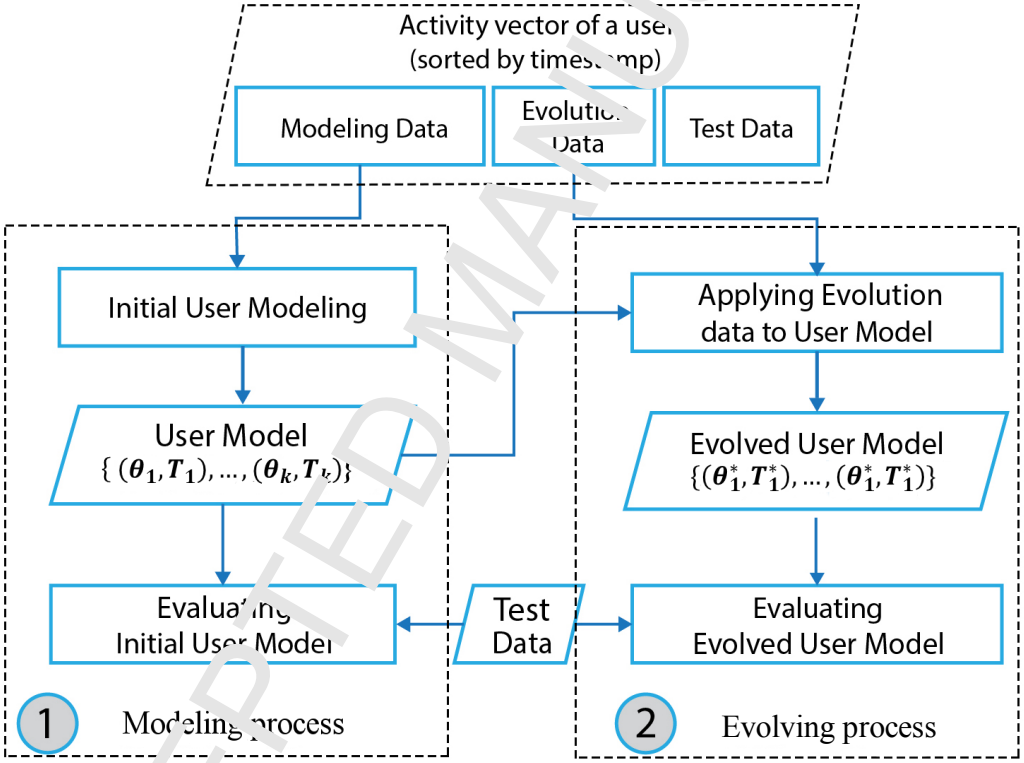
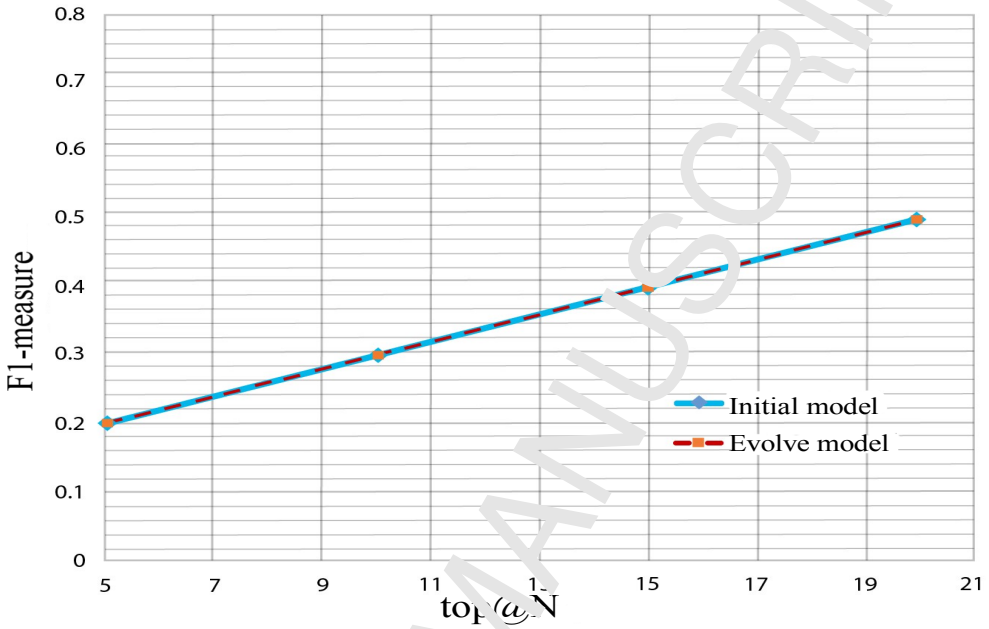
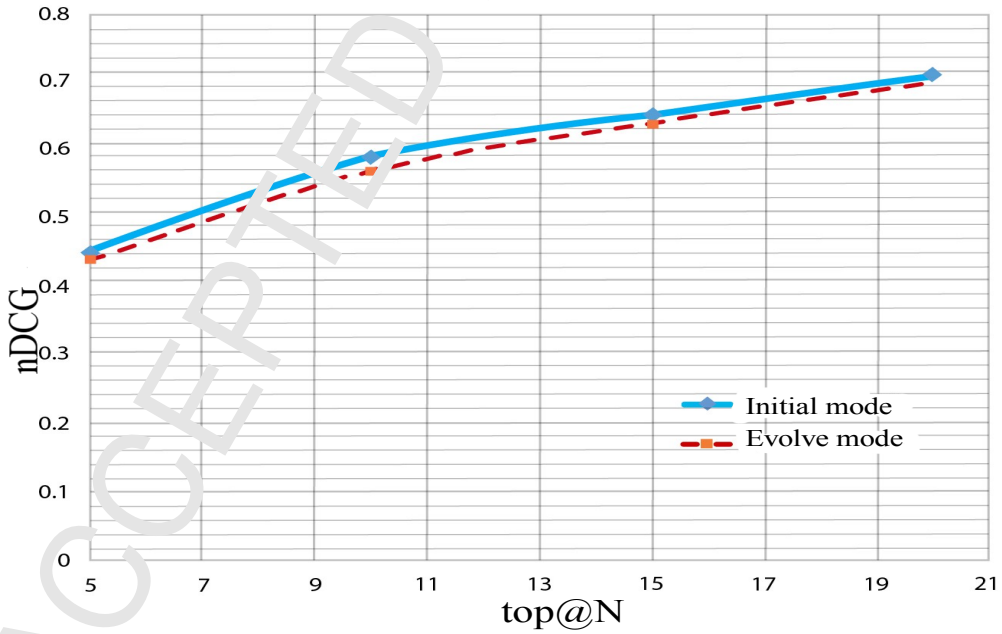


Fig. 10. The process of adaptivity evaluation.

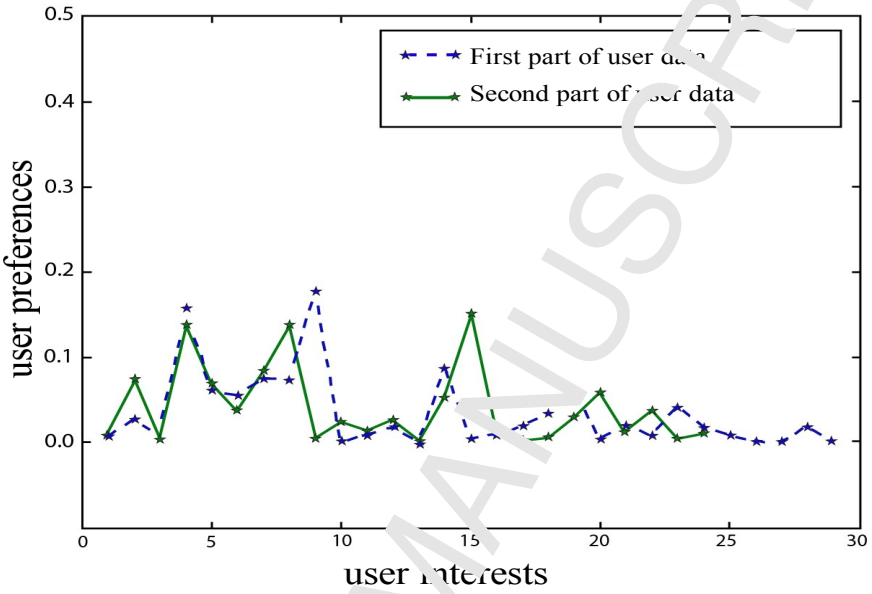


(a) F1-measure

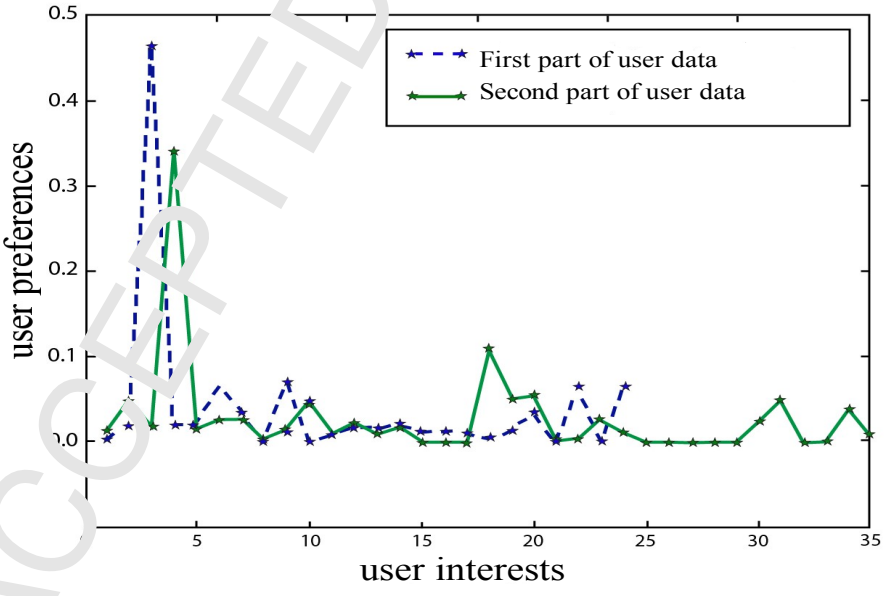


(b) DCG metric.

Fig. 11. Adaptivity evaluation: the evaluation metrics for two different consecutive partitions of MovieLens dataset. Dashed and solid lines represent the evolving and modeling results, respectively.

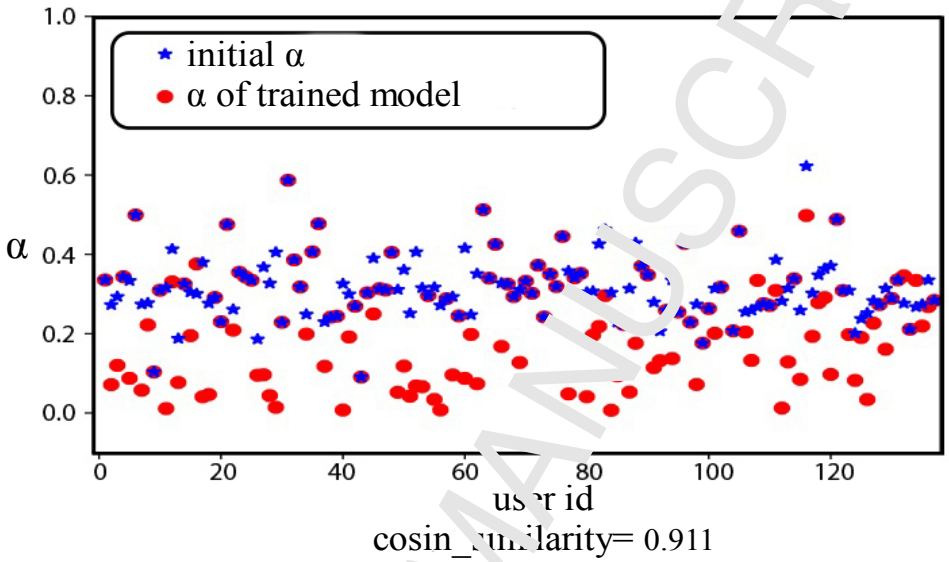


(a) First randomly selected user.

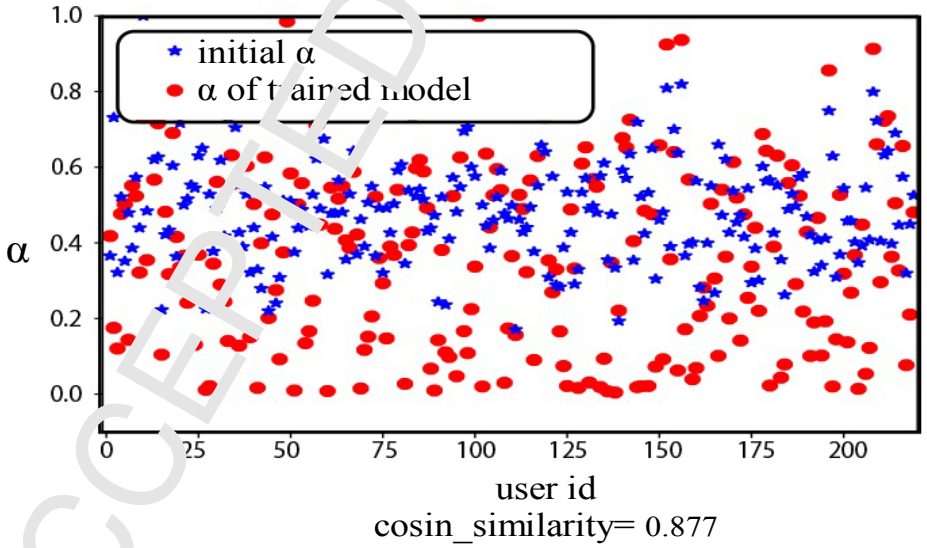


(b) Second randomly selected user.

Fig. 12. The adaptivity behavior of the proposed recommender system according to the activities of two randomly selected users.



(a)



(b)

Fig 13. The correlation of initiation α and final α of trained model for two different subset of users.