Online Interactive Collaborative Filtering Using Multi-Armed Bandit with Dependent Arms

Qing Wang, Chunqiu Zeng, Wubai Zhou, Tao Li, S. S. Iyengar, Larisa Shwartz, *Member, IEEE*, and Genady Ya. Grabarnik

Abstract—Online interactive recommender systems strive to promptly suggest users appropriate items (e.g., movies and news articles) according to the current context including both user and item content information. Such contextual information is often unavailable in practice, where only the users' interaction data on items can be utilized by recommender systems. The lack of interaction records, especially for new users and items, inflames the performance of recommendation further. To address these issues, both collaborative filtering, one of the most popular recommendation techniques relying on the interaction data only, and bandit mechanisms, capable of achieving the balance between exploitation and exploration, are adopted into an online interactive recommendation setting assuming independent items (i.e., arms). This assumption rarely holds in reality, since the real-world items tend to be correlated with each other. In this paper, we study online interactive collaborative filtering problems by considering the dependencies among items. We explicitly formulate item dependencies as the clusters of arms in the bandit setting, where the arms within a single cluster share the similar latent topics. In light of topic modeling techniques, we come up with a novel generative model to generate the items from their underlying topics. Furthermore, an efficient particle-learning based online algorithm is developed for inferring both latent parameters and states of our model by taking advantage of the fully adaptive inference strategy of particle learning techniques. Additionally, our inferred model can be naturally integrated with existing multi-armed selection strategies in an interactive collaborative filtering setting. Empirical studies on two real-world applications, online recommendations on movies and news, demonstrate both the effectiveness and efficiency of our proposed approach.

Index Terms—Recommender systems, interactive collaborative filtering, topic modeling, cold-start problem, particle learning

1 Introduction

The overwhelming amount of data requires an efficient online interactive recommendation system where online users constantly interact with the system, and user feedback is instantly collected to improve recommendation performance. Online interactive recommender systems are challenged to immediately recommend the most proper items (e.g., movies, news articles) to users based on the current user and item content information aiming to continuously maximize users' satisfaction over a long run. To achieve this goal, it becomes a critical task for such recommender systems to constantly track user preferences and recommend interesting items from a large item repository.

In the process of identifying the appropriate match between user preferences and target items, the systems encounter difficulties due to several existing practical challenges. One challenge is the well-known *cold-start* problem since a number of users/items might be completely new to the system, that is, they may have no historical records at all. This problem makes recommender systems ineffective unless additional information including both items and users is collected [1], [2]. The second challenge is that most recommender systems typically assume the entire set of contextual features with respect to both users and items can be accessed to infer users' preference. Due to a number of reasons (e.g., privacy or sampling constraints), it is challenging to obtain all relevant features ahead of time, thus rendering many factors unobservable to recommendation algorithms.

In the first challenge, an *exploration* or *exploitation* dilemma [3] is identified in the aforementioned setting. A tradeoff between two competing goals needs to be considered in recommender systems: maximizing user satisfaction using their consumption history, while gathering new information for improving the goodness of match between user preferences and items [4]. This dilemma is typically formulated as a multi-armed bandit problem where each arm corresponds to an item. The recommendation algorithm determines the strategies for selecting an arm to pull according to the contextual information at each trial. Pulling an arm indicates that the corresponding item is recommended. When an item matches the user preference (e.g., a recommended news article or movie is consumed), a reward is

- *Q. Wang, C. Zeng, W. Zhou, and S. S. Iyengar are with the School of Computing and Information Sciences, Florida International University, Miami, FL 33199. E-mail: {qwang028, czeng001, wzhou005, iyengar}@cs.fiu.edu.*
- *T. Li was with the School of Computing and Information Sciences, Florida International University, Miami, FL 33199.*
- *L. Shwartz is with the Department of Cognitive Service Foundations, IBM T.J. Watson Research Center, Yorktown Heights, NY 10598. E-mail: lshwart@us.ibm.com.*
- *G. Y. Grabarnik is with the Department of Math & Computer Science, St. John's University, Queens, NY 11439. E-mail: grabarng@stjohns.edu.*

Manuscript received 30 Oct. 2017; revised 1 Aug. 2018; accepted 13 Aug. 2018. Date of publication 20 Aug. 2018; date of current version 3 July 2019. (Corresponding author: Tao Li.)
Recommended for acceptance by S. Yan.
For information on obtaining reprints of this article, please send e-mail to: reprints@ieee.org, and reference the Digital Object Identifier below.
Digital Object Identifier no. 10.1109/TKDE.2018.2866041

obtained; otherwise, no reward is provided. The reward information is fed back to optimize the strategies. The optimal strategy is to pull the arm with the maximum expected reward based on the historical interaction on each trial, and then to maximize the total accumulated rewards for the whole series of trials.

Collaborative filtering (CF) is widely applied in recommender systems [5], [6], [7] to address the second challenge. CF has gained its popularity due to its advantage over other recommendation techniques, where CF requires no extra information about items or users for recommendation but only users' historical ratings on items [8], [9]. Further, considering both aforementioned challenges simultaneously aggravates the difficulties when recommending items. Recently, an online interactive collaborative filtering system has been suggested [8], [10] adopting both techniques, multi-armed bandit and collaborative filtering. Typically, one collaborative filtering task is formulated as a matrix factorization problem. Matrix factorization derives latent features for both users and items from the historical interaction records. It assumes that a user's preference (i.e., rating) on a given item can be predicted considering the item and user latent feature vectors. Based on this assumption, multi-armed bandit policies make use of the predicted reward (i.e., user preference) for arm (i.e., item) selection. The feedback occurring between the current user and arm is used to update their latent vectors, without impacting the inference of other arms' latent vectors assuming arms are independent from each other. However, the assumption about the independency among arms rarely holds in real-world applications. For example, in the movie recommendation scenario, each movie corresponds to an arm. The dependent arms (i.e., movies) typically share similar latent topics (e.g., science fiction movies, action movies, etc.), and are likely to receive similar rewards (i.e., ratings or feedback) from users. Intuitively, the dependencies among arms can be utilized for reward prediction improvement and further facilitated the maximization of users' satisfaction in a long run.

In this paper, we introduce an interactive collaborative topic regression model that utilizes bandit algorithms with dependent arms to recommend appropriate items for target users. A sequential online inference method is proposed to learn the latent parameters and infer the latent states. We adopt a generative process based on topic model to explicitly formulate the arm dependencies as the clusters on arms, where dependent arms are assumed to be generated from the same cluster. Every time an arm is pulled, the feedback is not only used for inferring the involved user and item latent vectors, but it is also employed to update the latent parameters with respect to the arm's cluster. The latent cluster parameters further help with reward prediction for other arms in the same cluster. The fully adaptive online inference strategy of particle learning [11] allows our model to effectively capture arm dependencies. In addition, the learnt parameters can be naturally integrated into existing multi-arm selection strategies, such as *UCB* and *Thompson sampling*. We conduct empirical studies on two real-world applications, movie and news recommendations. The experimental results demonstrate the effectiveness of our proposed approach.

The rest of this paper is organized as follows. In Section 2, we provide a brief summary of prior work relevant to collaborative filtering, collaborative topic model, multi-armed bandit and the online inference with particle learning. We formulate the underlying problem in Section 3. The solution to the problem is presented in Section 4. Extensive evaluation results are reported in Section 5. Finally, Section 6 concludes the paper.

2 Related Work

This section highlights the existing works in the literature that are closely relevant to our work.

2.1 Interactive Collaborative Filtering

In recommender systems, collaborative filtering has gained an extensive popularity in recent decades due to its capability of identifying the user preference from the historical interactions between users and items [5], [8], [12], [13], [14], [15], [16]. However, it is still an immense challenge to effectively predict preferences for new users. This challenge typically referred to as the *cold-start* problem [17], [18], [19]. A straightforward solution to address this issue involves two separated stages, where it first explicitly figures out the user profile, then makes further recommendation based on the established user profile [20], [21]. By contrast, some preliminary works, referred to as interactive collaborative filtering (ICF), have recently emerged as an alternative way to deal with the *cold-start* issue [8], [10]. These works do not explicitly fulfill the two stages separately, but formulate the recommendation problem as a multi-armed bandit problem, and then naturally integrate two stages together by striking a balance between exploration and exploitation. Our work is primarily relevant to this research area addressing the ICF problem.

The ICF problem is first introduced in [10], where several multi-armed bandit algorithms (e.g., Thompson sampling [22], UCB [23]) are used for item recommendation in light of the user-item rating prediction with the probabilistic matrix factorization (PMF) framework [13]. However, the proposed method in [10] does not work in a completely online interactive mode since the multi-armed bandit algorithms partially rely on the latent item feature vector distributions, which are learnt with the offline Gibbs sampling in advance. In [8], an efficient Thompson sampling algorithm named particle Thompson sampling (PTS) addresses the ICF problem with Bayesian probabilistic matrix factorization (BPMF) [14] in a completely online mode. To reduce the reward prediction uncertainty, Wang et al. [24] incorporated the contextual features into the learned latent feature vectors for ICF problem. But these methods assume the latent item feature vectors in the ICF setting are independent. Although the work in [25] formulates the arm dependencies as an arm clustering problem, it fails to present an efficient online method to learn arm dependencies. By comparison, we explicitly learn the dependent arms with a generative topic model in the ICF setting and develop an efficient online solution capable of tracking the dependencies between arms as well as addressing the online recommendation.

Some recent studies explore the bandit dependencies for a group recommendation delivery by assuming that users in the same group react with similar feedback to the same recommended item [26], [27], [28], [29], [30]. Most existing works utilize the context information for users or predefined social network to build the user dependencies.

Wu et al. [27] exploit social information to find dependency among users for improving the accuracy of reward prediction. Wang et al. [28] propose a context-aware collaborative bandit model, which could incorporate mutual influence among users directly for matrix completion. In [29], an interactive social recommendation model is proposed to predict the target user's preference using a weighted combination of a user's preferences and his/her friends' preferences. A context-dependent clustering of bandits algorithm [31] is investigated, where the clusters over users are based on the current item content. Our work is orthogonal to those studies since we investigate the arm (item) dependencies in a bandit model rather than the dependencies among users. Wang et al. [32] come up with hierarchical multi-armed bandit algorithms leveraging the explicit taxonomy information among items for online recommendation. Our proposed method is capable of instantly learning the item dependencies during the online interactive recommendation process without explicit context information provided.

Multi-armed bandits are widely adopted in diverse applications such as online advertising [1], [33], web content optimization [4], [34], and robotic routing [35]. The core task of bandit problem is to balance the tradeoff between exploration and exploitation. A series of algorithms have been proposed to deal with this problem including ϵ-greedy [36], UCB [23], [37], EXP3 [38], Thompson sampling [39]. In this paper, we model the ICF problem as a multi-armed bandit problem with dependent arms and proposes online recommendation methodologies based on UCB and Thompson sampling.

TABLE 1
Important Notations

Notation	Description
M, N	number of rows (users) and columns (items).
$\mathbf{R} \in \mathcal{R}^{M \times N}$	the rating matrix.
$\mathbb{S}(t)$	the sequence of $(n(t-1), r_{m,n(t-1)})$ observed until time t.
$n(t)$	the recommended item index in the tth iteration.
$r_{m,t}$	the rating (reward) of the mth user by pulling the given item in the tth iteration.
$y_{m,t}$	the predicted rating for the mth user over given item in the tth iteration.
π	the policy to recommend items sequentially.
R_π	the cumulative rating (reward) of the policy π.
K	the number of topics and the number of dimensions for latent vectors.
$\mathbf{p}_m \in \mathcal{R}^K$	the latent feature vector of the mth user.
$\mathbf{q}_n \in \mathcal{R}^K$	the latent feature vector of the nth item.
$\mathbf{\Phi}_k \in \mathcal{R}^N$	the item distribution of the kth topic.
$\mathcal{P}_{m,n(t-1)}$	the set of particles for item $n(t-1)$ given user m at time $t-1$.
$z_{m,t}$	the latent topic of the mth user in the tth iteration.
$x_{m,t}$	the selected item of the mth user in the tth iteration.
λ	Dirichlet priors over topics for topic model.
η	Dirichlet priors over items for topic model.
σ_n^2	the variance of rating prediction.
α, β	the hyper parameters determine the distribution of σ_n^2.
$\boldsymbol{\mu}_\mathbf{q}, \boldsymbol{\Sigma}_\mathbf{q}$	the hyper parameters determine the Gaussian distribution of $\mathbf{q}_n$.
ξ	the observation noise of the rating.

2.2 Sequential Online Inference

Our model leverages topic modeling [40] to formulate arm dependencies and sequential online inference to infer the latent states and learn the unknown parameters. Popular sequential learning methods include sequential monte carlo sampling [41], [42] and particle learning [11], [43].

Sequential Monte Carlo (SMC) sampling consists of a set of Monte Carlo methodologies to solve the filtering problem [44]. These methodologies allow inference of full posterior distributions in general state space models, which may be both nonlinear and non-Gaussian.

Particle learning provides state filtering, sequential parameter learning and smoothing in a general class of state space models [11]. Particle learning is used to approximate the sequence of filtering and smoothing distributions in light of parameter uncertainty for a wide class of state space models. The central idea behind particle learning is to create a particle directly from the approximation to the joint posterior distribution of states and conditional sufficient statistics of fixed parameters in a fully-adapted `resample-propagate` framework. In this paper, we leverage the idea of particle learning for both latent state inference and parameter learning.

3 Problem Formulation

In this section, we provide a mathematical formulation of the interactive collaborative filtering problem into a bandit setting. Then we introduce a new generative model describing the dependency among arms (i.e., items). A glossary of notations mentioned in this paper is summarized in Table 1.

3.1 Basic Concepts and Terminologies

Assume that there are M users and N items in the system. The preferences of the users for the items are recorded by a partially observable matrix $\mathbf{R} = \{r_{m,n}\} \in \mathcal{R}^{M \times N}$, where the rating score $r_{m,n}$ indicates how user m would like item n. The basic collaborative filtering task is to predict the unknown rating score in light of the observed rating scores in $\mathbf{R}$. However, it is very challenging to fulfill the task in practice due to the high dimensionality and sparsity of the rating matrix. Matrix factorization addresses this challenge by mapping each user m and item n to the latent feature vectors $\mathbf{p}_m \in \mathcal{R}^K$ and $\mathbf{q}_n \in \mathcal{R}^K$ in a shared low K-dimension space (typically, $K \ll M, N$). It assumes that the rating $r_{m,n}$ can be predicted by

$$\hat{r}_{m,n} = \mathbf{p}_m^\top \mathbf{q}_n. \tag{1}$$

Therefore, the latent features $\{\mathbf{p}_m\}$ and $\{\mathbf{q}_n\}$ can be learned by minimizing the prediction error for all observed ratings in $\mathbf{R}$, while each unobserved rating value can be estimated using Equation (1) with its corresponding latent features learned by matrix factorization. In practice, since the feedback (i.e., rating scores) from users is received over time, the system is required to address the collaborative filtering problem in an interactive mode, which is referred to as an interactive recommender system.

In an interactive recommender system, user m constantly arrives to interact with the system over time. At each time $t = [1, \ldots, T]$, the system, according to the observed rating history, recommends an item $n(t)$ to the corresponding user m. After consuming item $n(t)$, the feedback (i.e., rating) $r_{m,n(t)}$ from user m is collected by the system and further utilized to update the model for the next item delivery. The interactive recommendation process involves a series of decisions over a finite but possibly unknown time horizon T. Accordingly, such an interactive recommendation process is modeled as a multi-armed bandit problem, where each item corresponds to an arm. Pulling an arm indicates that its corresponding item is being recommended and the rating score is considered as the reward received after pulling the corresponding arm.

Let $\mathbb{S}(t)$ be the available information at time t collected by the system for the target user m,

$$\mathbb{S}(t) = \{(n(1), r_{m,n(1)}), \ldots, (n(t-1), r_{m,n(t-1)})\}. \tag{2}$$

A policy π is defined as a function and used to select an arm based on the current cumulative information $\mathbb{S}(t)$,

$$n(t) = \pi(\mathbb{S}(t)). \tag{3}$$

The total rewards received by the policy π after T iterations is

$$R_\pi = \sum_{t=1}^{T} r_{m,\pi(\mathbb{S}(t))}. \tag{4}$$

The optimal policy π^* is defined as the one with maximum accumulated expected reward after T iterations,

$$\pi^* = \arg\max_\pi \mathbb{E}(R_\pi) = \arg\max_\pi \sum_{t=1}^{T} \mathbb{E}(r_{m,\pi(\mathbb{S}(t))}|t). \tag{5}$$

Therefore, our goal is to identify an optimal policy for maximizing the total rewards. Herein we use reward instead of regret to express the objective function, since maximization of the cumulative rewards is equivalent to minimization of regret during the T iterations [10]. Before selecting one arm at time t, a policy π typically learns a model to predict the reward for every arm according to the historical accumulated information $\mathbb{S}(t)$. The reward prediction helps the policy π make decisions to increase the total rewards.

In the latent factor model [13], [14], the rating is estimated by a product of user and item feature vectors $\mathbf{p}_m$ and $\mathbf{q}_n$ in Equation (1). From the probabilistic perspective, PMF introduces an observation noise ξ, a zero-mean Gaussian noise with variance σ^2 (i.e., $\xi \sim \mathcal{N}(0, \sigma^2)$), to the rating prediction function given in Equation (1). The derived rating prediction is as follows:

$$r_{m,n} = \mathbf{p}_m^\top \mathbf{q}_n + \xi. \tag{6}$$

In this setting, Equation (5) can be re-formulated as:

$$\pi^* = \arg\max_\pi \sum_{t=1}^{T} \mathbb{E}_{\mathbf{p}_m, \mathbf{q}_{\pi(\mathbb{S}(t))}} (\mathbf{p}_m^\top \mathbf{q}_{\pi(\mathbb{S}(t))}|t). \tag{7}$$

Consequently, the goal of an interactive recommender system is reduced to the optimization of the objective function in Equation (7).

Thompson Sampling, one of earliest heuristics for the bandit problem [22], belongs to the probability matching family. Its main idea is to randomly allocate the pulling chance according to the probability that an arm gives the largest expected reward at a particular time t. Based on the objective function in Equation (7), the probability of pulling arm n can be expressed as follows:

$$p(n(t) = n) = \int \mathbb{I}[\mathbb{E}(r_{m,n}|\mathbf{p}_m, \mathbf{q}_n) = \max_i \mathbb{E}(r_{m,i}|\mathbf{p}_m, \mathbf{q}_i)] \, p(\mathbf{p}_m, \mathbf{q}_n|t) d\mathbf{p}_m d\mathbf{q}_n. \tag{8}$$

At each time t, Thompson sampling samples both the user and item feature vectors together from their corresponding distributions, and then selects the item that leads to the largest reward expectation. Therefore, using Thompson sampling, the item selection function can be defined as:

$$n(t) = \arg\max_n (\tilde{\mathbf{p}}_m^\top \tilde{\mathbf{q}}_n|t), \tag{9}$$

where $\tilde{\mathbf{p}}_m$ and $\tilde{\mathbf{q}}_n$ denote the sampled feature vectors for user m and item n, respectively.

To accomplish Thompson sampling, it is critical to model the random variable $\mathbf{p}_m$ and $\mathbf{q}_n$ using distributions, where the latent feature vectors can be easily sampled and the feedback at every time can be reasonably integrated. Most of the previous studies suppose a Gaussian prior for both user and item feature vectors with an assumption that items are independent from each other [8], [10]. However, this assumption rarely holds in real applications. In the following section, we explicitly formulate the dependent arms with a generative model.

3.2 Modeling the Arm Dependency

Based on the fact that similar items (i.e., arms) are likely to receive similar feedback (i.e., rewards) , we assume that a dependency exists among similar items. The dependencies among items can be further leveraged to improve the users' preferences inference on a particular item even if the item has little historical interaction data in the system. The challenge here lies in how to sequentially infer the arms' dependencies as well as the users' preferences simultaneously, providing the feedback over time.

In our work, the arms' dependencies are expressed in the form of the clusters of arms, where the dependent arms fall into the same one. In order to explore the dependencies in the bandit setting, Latent Dirichlet Allocation (LDA) [40], a generative statistic model for topic modeling, is adopted to construct the arms' clusters. We propose the ICTR (Interactive Collaborative Topic Regression) model to infer the clusters of arms as well as the arm selection.

The main idea of our model is to treat an item n as a word, while consider a user m as a document. All the items rated by a user indicate the hidden preferences of the user, analogous to the scenario in topic modeling where the words contained in a document imply its latent topics. Specifically, let K be the number of latent aspects (i.e., topics or clusters) the users concern when consuming items. We assume that $\mathbf{p}_m \in \mathcal{R}^K$ corresponds to the latent vector for user m, where the kth component of $\mathbf{p}_m$ indicates the user's preference over the kth aspect of items. Further, $\mathbf{q}_n \in \mathcal{R}^K$ is

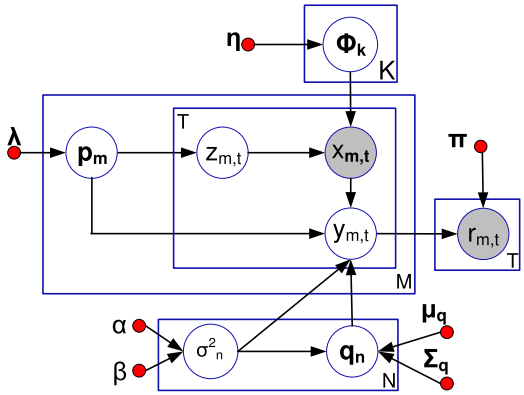


Fig. 1. The graphic model for the ICTR model. Random variable is denoted as a circle. The circle with filled color denotes the observed random variable. Red dot represents a hyper parameter.

supposed to be the latent vector for item n, and the kth component value of $\mathbf{q}_n$ represents that it belongs to the kth cluster. The rating score $r_{m,n}$, given by user m after consuming item n, is assumed to be the inner product of $\mathbf{p}_m$ and $\mathbf{q}_n$. By linking to the topic model, a generative process for user ratings is accordingly introduced and presented in Fig. 1.

Based on the above description, the user latent vector $\mathbf{p}_m$ is assumed to follow a Dirichlet prior distribution with a predefined hyper parameter λ, shown in Equation (10).

$$\mathbf{p}_m|\lambda \sim Dir(\lambda). \tag{10}$$

As presented in Equation (6), we denote $\boldsymbol{\sigma}^2$ as the variance of the noise for reward prediction and assume σ_n^2 is drawn from the Inverse Gamma ($\mathcal{IG}$) distribution shown in the following.

$$p(\sigma_n^2|\alpha,\beta) = \mathcal{IG}(\alpha,\beta), \tag{11}$$

where α and β are predefined hyper parameters for $\mathcal{IG}$ distribution.

Given $\boldsymbol{\sigma}_n^2$, the item latent vector $\mathbf{q}_n$ is generated by a Gaussian prior distribution as follows:

$$\mathbf{q}_n|\mu_{\mathbf{q}},\Sigma_{\mathbf{q}},\sigma_n^2 \sim \mathcal{N}(\mu_{\mathbf{q}},\sigma_n^2\Sigma_{\mathbf{q}}), \tag{12}$$

where $\mu_{\mathbf{q}}$ and $\Sigma_{\mathbf{q}}$ are predefined hyper parameters.

Further, let $\Phi_k \in \mathcal{R}^N$ be the item distribution for topic k. Similar to $\mathbf{p}_m$, Dirichlet distribution is specified as the prior of Φ_k presented in Equation (13).

$$\Phi_k|\eta \sim Dir(\eta), \tag{13}$$

where $\eta \in \mathcal{R}^N$ is the hyper parameter.

When user m arrives to interact with the system at time t, one of K topics, denoted as $z_{m,t}$, is first selected according to the user's latent preference $\mathbf{p}_m$, indicating that user m shows interest in the topic $z_{m,t}$ at this moment. Accordingly, $z_{m,t}$ is supposed to follow a multinomial distribution governed by $\mathbf{p}_m$ as follows,

$$z_{m,t}|\mathbf{p}_m \sim Mult(\mathbf{p}_m). \tag{14}$$

W.L.O.G, we assume $z_{m,t} = k$, and then the item distribution for topic k (i.e., Φ_k) is for generating item $x_{m,t}$ recommended to user m at time t. We assume the random variable $x_{m,t}$ follows the multinominal distribution ruled by Φ_k, i.e.,

$$x_{m,t}|\Phi_k \sim Mult(\Phi_k). \tag{15}$$

W.L.O.G, item n is assumed to be selected by user m at time t (i.e., $x_{m,t} = n$) where the latent vector corresponding to item n is $\mathbf{q}_n$. Let $y_{m,t}$ be the predicted reward (i.e., rating), given by user m at time t. The predicted reward $y_{m,t}$ can be inferred by

$$y_{m,t} \sim \mathcal{N}(\mathbf{p}_m^\top \mathbf{q}_n, \sigma_n^2). \tag{16}$$

By Equation (16), the rewards of different items are predicted. Based on the predicted rewards, the policy π selects an item and recommends it to user m, considering the trade-off between exploitation and exploration. After consuming the recommended item, the system receives the actual reward $r_{m,t}$ from user m. The objective of the model is to maximize the expected accumulative rewards in a long run as described in Equation (5).

In this section, taking the clusters of arms into account, we formally introduced our ICTR model, which integrates matrix factorization with topic modeling in the bandit setting. We develop our solution to infer ICTR model from a Bayesian perspective in the following section.

4 Methodology and Solution

In this section, we present the methodology for online inferences of ICTR model.

The posterior distribution inference involves five random variables, i.e., $\mathbf{p}_m$, $z_{m,t}$, Φ_k, $\mathbf{q}_n$, and σ_n^2. According to the graphical model in Fig. 1, the five random variables belong to two categories: parameter random variable and latent state random variable. Φ_k, $\mathbf{p}_m$, $\mathbf{q}_n$, and σ_n^2 are parameter random variables since they are assumed to be fixed but unknown, and their values do not change with time. Instead, $z_{m,t}$ is referred to as a latent state random variable since it is not observable and its value is time dependent. After pulling arm $n(t)$, where $n(t) = x_{m,t}$ according to Equation (15) at time t, a reward is observed as $r_{m,t}$. Thus, $x_{m,t}$ and $r_{m,t}$ are referred to as observed random variables.

Our goal is to infer both latent parameter variables and latent state random variables to sequentially fit the observed data at time $t-1$, and predict the rewards for arm selection with respect to the incoming user at time t. However, since the inference of our model cannot be conducted by a simple closed-form solution, we adopt the sequential sampling-based inference strategy that is widely used in sequential Monte Carlo sampling [45], particle filtering [46], and particle learning [11] to learn the distribution of both parameter and state random variables. Specifically, particle learning that allows both state filtering and sequential parameter learning simultaneously is a perfect solution to our proposed model inference. In order to develop the solution based on particle learning, we first define a particle as follows.

Definition 1 (Particle). *A particle for predicting the reward $y_{m,t}$ is a container that maintains the current status information for both user m and item $x_{m,t}$. The status information comprises of random variables such as $\mathbf{p}_m$, σ_n^2, Φ_k, $\mathbf{q}_n$, and $z_{m,t}$, as well as the hyper parameters of their corresponding distributions, such as λ, α, β, η, $\mu_{\mathbf{q}}$ and $\Sigma_{\mathbf{q}}$.*

In particle learning, each particle corresponds to a sample for modeling inference status information. At each time stamp, particles are re-sampled according to their fitness to the current observable data. Then, the re-sampled particles are propagated to new particles and obtain the status information for the next time stamp. In the following sections, we develop our solution based on particle learning.

4.1 Re-sample Particles with Weights

At time $t-1$, a fixed-size set of particles is maintained for the reward prediction for each arm $n(t-1)$ given user m. We denote the particle set at time $t-1$ as $\mathcal{P}_{m,n(t-1)}$ and assume the number of particles in $\mathcal{P}_{m,n(t-1)}$ is B. Let $\mathcal{P}^{(i)}_{m,n(t-1)}$ be the ith particles given both user m and item $n(t-1)$ at time $t-1$, where $1 \leq i \leq B$. Each particle $\mathcal{P}^{(i)}_{m,n(t-1)}$ has a weight, denoted as $\rho^{(i)}$, indicating its fitness for the new observed data at time t. Note that $\sum_{i=1}^{B} \rho^{(i)} = 1$. The fitness of each particle $\mathcal{P}^{(i)}_{m,n(t-1)}$ is defined as the likelihood of the observed data $x_{m,t}$ and $r_{m,t}$. Therefore,

$$\rho^{(i)} \propto p(x_{m,t}, r_{m,t} | \mathcal{P}^{(i)}_{m,n(t-1)}). \tag{17}$$

Further, $y_{m,t}$ is the predicted value of $r_{m,t}$. The distribution of $y_{m,t}$, determined by $\mathbf{p}_m$, $\mathbf{q}_n$, $z_{m,t}$, Φ_k , and σ_n^2, has been described in Section 3.2.

Therefore, we can compute $\rho^{(i)}$ as proportional to the density value given $y_{m,t} = r_{m,t}$ and $x_{m,t} = n$. Thus, we obtain

$$\begin{aligned}\rho^{(i)} \propto & \sum_{z_{m,t}=1}^{K} \{\mathcal{N}(r_{m,t} | (\mathbf{p}_m^\top \mathbf{q}_n, \sigma_n^2) \\ & \bullet p(z_{m,t} = k, x_{m,t} = n | \mathcal{P}^{(i)}_{m,n(t-1)})\},\end{aligned}$$

where

$$\begin{aligned}& p(z_{m,t} = k, x_{m,t} = n | \mathcal{P}^{(i)}_{m,n(t-1)}) \\ &= \iint_{\mathbf{p}_m, \Phi_k} p(z_{m,t} = k, x_{m,t} = n, \mathbf{p}_m, \Phi_k | \lambda, \eta) d\mathbf{p}_m d\Phi_k \\ &= \int_{\mathbf{p}_m} Mult(z_{m,t} = k | \mathbf{p}_m) Dir(\mathbf{p}_m | \lambda) d\mathbf{p}_m \\ &\quad \bullet \int_{\Phi_k} Mult(x_{m,t} = n | \Phi_k) Dir(\Phi_k | \eta) d\Phi_k \\ &= E(\mathbf{p}_{m,k} | \lambda) \bullet E(\Phi_{k,n} | \eta).\end{aligned} \tag{18}$$

Thus, we have

$$\rho^{(i)} \propto \sum_{z_{m,t}=1}^{K} \{\mathcal{N}(\mathbf{r}_{m,t} | (\mathbf{p}_m^\top \mathbf{q}_n, \sigma_n^2) \bullet E(\mathbf{p}_{m,k} | \lambda) \bullet E(\Phi_{k,n} | \eta)\}, \tag{19}$$

where $E(\mathbf{p}_{m,k}|\lambda)$ and $E(\Phi_{k,n}|\eta)$ represent the conditional expectations of $\mathbf{p}_{m,k}$ and $\Phi_{k,n}$ given the observed reward λ and η of $\mathcal{P}^{(i)}_{m,n(t-1)}$. The expectations can be inferred by $E(\mathbf{p}_{m,k}|\lambda) = \frac{\lambda_k}{\sum_{k=1}^{K} \lambda_k}$ and $E(\Phi_{k,n}|\eta) = \frac{\eta_{k,n}}{\sum_{n=1}^{N} \eta_{k,n}}$.

Before updating any parameters, a re-sampling process is conducted. We replace the particle set $\mathcal{P}_{m,n(t-1)}$ with a new set $\mathcal{P}_{m,n(t)}$, where $\mathcal{P}_{m,n(t)}$ is generated from $\mathcal{P}_{m,n(t-1)}$ using sampling with replacement based on the weights of particles. Then sequential parameter updating is based on $\mathcal{P}_{m,n(t)}$.

4.2 Latent State Inference

Provided with the new observation $x_{m,t}$ and $r_{m,t}$ at time t, the random state $z_{m,t}$ can be one of K topics and the posterior distribution of $z_{m,t}$ is shown as follows:

$$z_{m,t} | x_{m,t}, r_{m,t}, \mathcal{P}^{(i)}_{m,n(t-1)} \sim Mult(\boldsymbol{\theta}), \tag{20}$$

where $\boldsymbol{\theta} \in \mathcal{R}^K$ is the parameter specifying the multinominal distribution. According to Equation (18), since

$$p(z_{m,t} | x_{m,t}, r_{m,t}, \lambda, \eta) \propto p(z_{m,t}, x_{m,t} | r_{m,t}, \lambda, \eta),$$

θ can be computed by $\theta_k \propto E(\mathbf{p}_{m,k}|r_{m,t}, \lambda) \bullet E(\Phi_{k,n}|r_{m,t}, \eta)$. Further, $E(\mathbf{p}_{m,k}|r_{m,t}, \lambda)$ and $E(\Phi_{k,n}|r_{m,t}, \eta)$ can be obtained as follows,

$$\begin{aligned}E(\mathbf{p}_{m,k} | r_{m,t}, \lambda) &= \frac{\mathcal{I}(z_{m,t} = k) r_{m,t} + \lambda_k}{\sum_{k=1}^{K} [\mathcal{I}(z_{m,t} = k) r_{m,t} + \lambda_k]}, \\ E(\Phi_{k,n} | r_{m,t}, \eta) &= \frac{\mathcal{I}(x_{m,t} = n) r_{m,t} + \eta_{k,n}}{\sum_{n=1}^{N} [\mathcal{I}(x_{m,t} = n) r_{m,t} + \eta_{k,n}]}.\end{aligned} \tag{21}$$

Where $\mathcal{I}(\bullet)$, an indicator function, returns 1 when the input boolean expression is true, otherwise returns 0. Specifically, if $r_{m,t} \in \{0, 1\}$, the value of $r_{m,t}$ indicates whether $x_{m,t}$ should be included in the preferred item list of user m. If $r_{m,t} \in [0, +\infty)$, the value of $r_{m,t}$ implies how user m likes item $x_{m,t}$. Therefore, our solution can effectively handle the non-negative rating score at different scales.

4.3 Parameter Statistics Inference

At time $t-1$, the sufficient statistics for the parameter random variables ($\mathbf{q}_n$, σ_n^2, $\mathbf{p}_m$, Φ_k) are ($\mu_{\mathbf{q}}$, $\Sigma_{\mathbf{q}}$, α, β, λ, η). Assume $\mu'_{\mathbf{q}}$, $\Sigma'_{\mathbf{q}}$, α', β', λ', and η' are the sufficient statistics at time t, which are updated on the basis of both the sufficient statistics at time $t-1$ and the new observation data (i.e, $x_{m,t}$ and $r_{m,t}$). The sufficient statistics for parameters are updated as follows:

$$\begin{aligned}&\Sigma'_{\mathbf{q}_n} = (\Sigma^{-1}_{\mathbf{q}_n} + \mathbf{p}_m \mathbf{p}_m^\top)^{-1}, \quad \mu'_{\mathbf{q}_n} = \Sigma'_{\mathbf{q}_n} (\Sigma^{-1}_{\mathbf{q}_n} \mu_{\mathbf{q}_n} + \mathbf{p}_m r_{m,t}) \\ &\alpha' = \alpha + \frac{1}{2} \\ &\beta' = \beta + \frac{1}{2} (\mu_{\mathbf{q}_n}^\top \Sigma^{-1}_{\mathbf{q}_n} \mu_{\mathbf{q}_n} + r_{m,t}^\top r_{m,t} - \mu'^\top_{\mathbf{q}_n} \Sigma'^{-1}_{\mathbf{q}_n} \mu'_{\mathbf{q}_n}) \\ &\lambda'_k = \mathcal{I}(z_{m,t} = k) r_{m,t} + \lambda_k, \quad \eta'_{k,n} = \mathcal{I}(x_{m,t} = n) r_{m,t} + \eta_{k,n}.\end{aligned} \tag{22}$$

At time t, the sampling process for the parameter random variables σ_n^2, $\mathbf{q}_n$, $\mathbf{p}_m$ and Φ_k is summarized as below

$$\begin{aligned}&\sigma_n^2 \sim \mathcal{IG}(\alpha', \beta'), \quad \mathbf{q}_n | \sigma_n^2 \sim \mathcal{N}(\mu'_{\mathbf{q}_n}, \sigma_n^2 \Sigma'_{\mathbf{q}_n}), \\ &\mathbf{p}_m \sim Dir(\lambda'), \quad \Phi_k \sim Dir(\eta').\end{aligned} \tag{23}$$

4.4 Integration with Policies

In our ICTR model, when user m arrives at time t, reward $r_{m,t}$ is unknown since it is not observed until one of arms $x_{m,t}$ is pulled. Without observed $x_{m,t}$ and $r_{m,t}$, the particle re-sampling, latent state inference, and parameter statistics inference for time t cannot be conducted. Therefore, we utilize the latent vectors $\mathbf{p}_m$ and $\mathbf{q}_n$, sampled from their corresponding posterior distributions by Equation (23) at time

$t-1$, to predict the reward for each arm. In this section, two policies based on Thompson sampling and UCB for ICF are integrated with our model.

In the model, every item has B independent particles given user m. Each particle i contains its latent variables and parameters, and produces an independent reward prediction $r_{m,t}^{(i)}$. Specifically, according to Thompson sampling discussed in Section 3.1, we predict the reward of pulling arm n with the average value of rewards from B particles. The policy based on Thompson sampling selects an arm $n(t)$ based on the following equation,

$$n(t) = \arg\max_{n} (\bar{r}_{m,n}), \tag{24}$$

where $\bar{r}_{m,n}$ denotes the average reward, i.e.,

$$\bar{r}_{m,n} = \frac{1}{B}\sum_{i=1}^{B} \mathbf{p}_m^{(i)\top}\mathbf{q}_n^{(i)}.$$

Moreover, UCB policy selects an arm based on the upper bound of the predicted reward. Assuming that $r_{m,t}^{(i)} \sim \mathcal{N}(\mathbf{p}_m^{(i)\top}\mathbf{q}_n^{(i)}, \sigma^{(i)2})$, the UCB-based policy is developed by the mean and variance of predicted reward, i.e.,

$$n(t) = \arg\max_{n} (\bar{r}_{m,n} + \gamma\sqrt{v}), \tag{25}$$

where $\gamma \geq 0$ is a predefined threshold and the variance is expressed as $v = \frac{1}{B}\sum_{i}^{B} \sigma^{(i)2}$.

4.5 Algorithm

Putting all the aforementioned inference together, an algorithm for ICTR model is provided below.

Online inference for ICF problem starts with MAIN procedure presented in Algorithm 1. As user m arrives at time t, EVAL procedure computes a score for each arm, where we define the score as the average reward. The arm with the highest score is selected to be pulled. After receiving a reward by pulling an arm, the new feedback is used to update ICTR model by UPDATE procedure. Especially in UPDATE procedure, we use the *resample-propagate* strategy in particle learning [11] rather than the *propagate-resample* strategy in particle filtering [46]. With the *resample-propagate* strategy, particles are re-sampled by taking $\rho^{(i)}$ as the ith particle's weight, where the $\rho^{(i)}$ indicates the fitness of the observation at time t given the particle at time $t-1$. The *resample-propagate* strategy is considered as an optimal and fully adapted strategy avoiding an importance sampling step.

In addition, existing algorithms [8], [10] consider all the arms independently, while our model takes the clusters of arms into account by learning the topic-related random variables (e.g., Φ_k), which are shared among all the arms.

5 Empirical Study

To demonstrate the efficiency of our proposed algorithm, we conduct our experimental study over two popular real-world dataset: Yahoo! Today News and MovieLens (10M). First, we outline the general implementation of the baselines. Second, we start with a brief description of the datasets and evaluation method. Finally, we show and discuss the comparative experimental results of both the proposed algorithms and the baselines, and a case study on movie topic distribution analysis of MovieLens (10M). Methods in [26], [27], [28], [47] are excluded from the baselines since our work is orthogonal to those methods.

Algorithm 1. The Algorithm for ICTR Model

procedure MAIN(B) ▷ Main entry.
  Initialize B particles, i.e., $\mathcal{P}_{m,n(0)}^{(1)} \ldots \mathcal{P}_{m,n(0)}^{(B)}$.
  for $t \leftarrow 1, T$ **do**
    User m arrives for item recommendation.
    $n(t) = \arg\max_{n=1,N} \text{EVAL}(m, n)$ by Equation (24) or Equation (25).
    Receive $r_{m,t}$ by rating item $n(t)$.
    UPDATE(m, $n(t)$, $r_{m,t}$).
  end for
end procedure
procedure EVAL(m, n) ▷ Get a rating score for item n, given user m.
  for $i \leftarrow 1, B$ **do** ▷ Iterate on each particle.
    Get the user latent vector $\mathbf{p}_m^{(i)}$.
    Get the item latent vector $\mathbf{q}_n^{(i)}$.
    Predict ith reward $r_{m,t}^{(i)}$.
  end for
  Compute the average reward as the final reward $r_{m,t}$.
  return the score.
end procedure
procedure UPDATE(m, $n(t)$, $r_{m,t}$) ▷ Update the inference.
  for $i \leftarrow 1, B$ **do** ▷ Compute weights for each particle.
    Compute weight $\rho^{(i)}$ of particle $\mathcal{P}_{m,n(t)}^{(i)}$ by Equation (17).
  end for
  Re-sample $\mathcal{P}'_{m,n(t)}$ from $\mathcal{P}_{m,n(t)}$ according to the weights $\rho^{(i)}$s.
  for $i \leftarrow 1, B$ **do** ▷ Update statistics for each particle.
    Update the sufficient statistics for $z_{m,t}$ by Equation (21).
    Sample $z_{m,t}$ according to Equation (20).
    Update the statistics for $\mathbf{q}_n$, σ_n^2, $\mathbf{p}_m$, Φ_k by Equation (22).
    Sample $\mathbf{q}_n$, σ_n^2, $\mathbf{p}_m$, Φ_k by Equation (23).
  end for
end procedure

5.1 Baseline Algorithms

In the experiment, we demonstrate the performance of our methods by comparing them with the following baseline algorithms:

(1) Random: it randomly selects an item recommending to the target user.
(2) ϵ-greedy(ϵ): it randomly selects an item with probability ϵ and selects the item of the largest predicted reward with probability $1-\epsilon$, where ϵ is a predefined parameter.
(3) UCB(λ): it picks item $j(t)$ with the highest rewards at time t as follows:

$$j(t) = \arg\max_{j=1,\ldots,N} (\hat{\mu}_i + \lambda\sqrt{\frac{2ln(t)}{n_{i(t)}}}),$$

where $n_{i(t)}$ is the number of times that item n_i has been recommended until time t.

TABLE 2
Description of Datasets

Dataset	Yahoo News	MovieLens (10M)
#users	226,710	71,567
#items	652	10,681
#ratings	280,410,150	10,000,054

(4) TS($S_i(t), F_i(t)$): Thompson sampling described in Section 3.1, randomly draws the expected reward from the Beta posterior distribution, and selects the item with the largest predicted reward. $S_i(t)/F_i(t)$ is the number of positive/negative feedback on item i until time t.
(5) PTS(d, p): particle Thompson sampling for matrix factorization approximates the posterior of latent feature vectors by updating a set of particles. Here d is the dimension of latent feature vector and p is the number of particles.

Our methods proposed in this paper include:

(1) ICTRTS(d, p): it denotes our proposed interactive collaborative topic regression model with TS. Here d is the dimension of latent feature vector and p is the number of particles.
(2) ICTRUCB(d, p, γ): it indicates our proposed model with UCB. Similar to UCB, γ is given. Here d is the dimension of latent feature vector and p represents the number of particles.

All algorithms are implemented using Java 1.8. All empirical experiments are running on Linux 2.6.32. The server is equipped with Intel(R) Xeon(R) CPU with 24 cores runing at speed of 2.50 GHZ. The total volume of memory is 158 GB.

5.2 Datasets Description

We use two real-world datasets shown in Table 2 to evaluate our proposed algorithms.

Yahoo! Today News. The core task of personalized news recommendation is to display appropriate news articles on the web page for users. The system often takes the user's instant feedback into account to improve the prediction of his/her preferences, where the user feedback is about whether he/she clicks the recommended article or not. Here, we formulate the personalized news recommendation problem as an instance of bandit problem, where each arm corresponds to a news article. The experimental dataset is a collection based on a sample of anonymized user interaction on the news feeds published by Yahoo! Research Lab.[1] The dataset contains 15 days' visit events of user-news item interaction data by randomly selecting news articles for recommendation. Besides, user's information (e.g., demographic information) is provided for each visit event and represented as the user identification, where users with the same information are identified as one user. In our experiments, the visit events of the first day are utilized for selecting proper parameters of ICTR model, while two million of the remaining are for the evaluation. Each interactive record in the historical logs consists of user ID, news article ID, rating feedback and a timestamp.

1. http://webscope.sandbox.yahoo.com/catalog.php

MovieLens (10M). Online movie recommender service aims to maximize the customer's satisfaction by recommending proper movies to target users according to their preferences. Specifically, several movies are selected out of a movie set and displayed to users, and then users' feedback on displayed movies are collected for improving the user satisfaction. Thereby, the problem of movie recommendation can be formulated as a bandit problem where an arm is a movie, a pull is regarded as a movie selection, and the reward is indicated by the user's rating on the recommended movie. In our experiments, each rating associates user ID, movie ID, and a timestamp. In order to use the *replayer* evaluation method, we assume the rating data is produced by the users when the movies are randomly recommended. The rating score in the dataset ranges from 1 to 5. Additionally, we choose the top-N (N = 100) popular movies to form a movie set, from which one movie is recommended to a user by algorithms in every trial.

5.3 Evaluation Method and Metrics

The evaluation methods for traditional non-interactive recommender systems assume the independence among the items at different time stamps once the offline model is built. In an online interactive recommender system, the recommended items at previous time stamps are used to update the recommendation model, and then further effect the recommendation items at current time stamp.

We apply the *replayer* method to evaluate our proposed algorithms on the aforementioned two datasets. The *replayer* method, first introduced in [48], provides an unbiased offline evaluation for multi-armed bandit algorithms via historical logs, where the logs are assumed to be generated by random recommendation. The main idea of *replayer* is to replay each user visit in the historical logs to the algorithm under evaluation. If the recommended item by the testing algorithm is identical to the one in the historical log, this visit is considered as a match between the historical recommendation and the testing recommendation algorithm. The *replayer* method only counts those matched visits in for the accumulated reward computation. Since the recommendation algorithms may result in different numbers of matched visits, the average reward (i.e., the accumulated rewards divided by the number of matched visits) is adopted for evaluation.

Particularly, in the scenario of news article recommendation, a matched visit corresponds to an impression, and a reward of one is obtained by a click, so the average reward also represents the average Click Through Rate (CTR). In the scenario of movie recommendation, we set the reward of one if the rating score of the recommended movie is no less than four, indicating that the user likes the recommended movie. If the rating is less than four, a reward value of zero is obtained. Thus, the average reward in this scenario indicates the success rate of movie recommendation. To sum it up, in our setting, the reward is one if the recommended article (movie) is clicked (liked), otherwise it is zero.

Consider a matched visit shown in Table 3. If the item in the logs is clicked or liked by a user, the recommended item is referred to as a true positive (TP), otherwise it is referred to as a false positive (FP). The average reward is computed as $\frac{TP*1+FP*0}{TP+FP} = \frac{TP}{TP+FP}$, corresponding to the formula for the

TABLE 3
Evaluation Metric Computation for *Replayer*

		Item in Random Recommendation Logs	
		Clicked/Liked	Not Clicked/Not Liked
Recommended Item	Matched	TP	FP
	Not Matched	N/A	N/A

precision for matched visits. However, for the unmatched visits, we can not determine whether an item is false negative or true negative since no ground truth is provided. Therefore, the computation of the average reward in this case as the recall, relying on the false negative, is not feasible.

5.4 Recommendation Evaluation

In this section we first conduct the *replayer* evaluation method for each algorithm with different parameter settings. The aforementioned average reward is used as the performance metric in the experiments.

All baseline algorithms are configured with different parameter settings provided in Table 4. The settings of all algorithms with the highest average reward are highlighted in bold. Our algorithm `ICTRUCB(2,10,1.0)` achieves the best performance among all algorithms on Yahoo! Today News, and the performance comparisons among different algorithms along different time buckets are illustrated in Fig. 2. For MovieLens (10M), `ICTRTS(3,10)` outperforms all others and the corresponding performance comparisons are shown in Fig. 3.

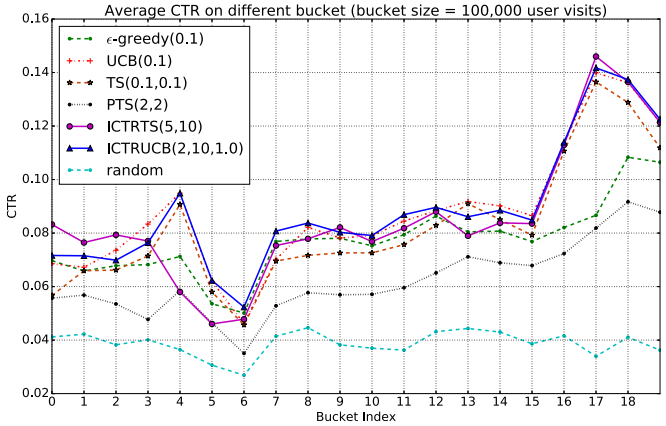


Fig. 2. The average CTR of Yahoo! Today News data is given along each time bucket. All algorithms shown here are configured with their best parameter settings.

Our proposed algorithms outperform the baseline algorithms using independent arms because ICTR model can leverage the dependencies among items by clustering items (arms) using items' latent aspects. The feedback received after recommending an item is not only used to update the

TABLE 4
Average CTR/Rating on Two Real-World Datasets

Algorithm	Yahoo! Today News				MovieLens (10M)			
	mean	std	min	max	mean	std	min	max
`ϵ-greedy(0.01)`	0.06916	0.00312	0.06476	0.07166	0.70205	0.06340	0.60752	0.78934
`ϵ-greedy(0.1)`	**0.07566**	0.00079	0.07509	0.07678	**0.82038**	0.01437	0.79435	0.83551
`ϵ-greedy(0.3)`	0.07006	0.00261	0.06776	0.07372	0.80447	0.01516	0.77982	0.82458
`ϵ-greedy(1.0)`	0.03913	0.00051	0.03842	0.03961	0.60337	0.00380	0.59854	0.60823
`UCB(0.01)`	0.05240	0.00942	0.04146	0.06975	0.62133	0.10001	0.45296	0.73369
`UCB(0.1)`	**0.08515**	0.00021	0.08478	0.08544	**0.73537**	0.07110	0.66198	0.85632
`UCB(0.5)`	0.05815	0.00059	0.05710	0.05893	0.71478	0.00294	0.63623	0.64298
`UCB(1.0)`	0.04895	0.00036	0.04831	0.04932	0.63909	0.00278	0.60324	0.61296
`TS(0.01,0.01)`	0.07853	0.00058	0.07759	0.07921	**0.83585**	0.00397	0.82927	0.84177
`TS(0.1,0.1)`	**0.07941**	0.00040	0.07869	0.07988	0.83267	0.00625	0.82242	0.84001
`TS(0.5,0.5)`	0.07914	0.00106	0.07747	0.08041	0.82988	0.00833	0.81887	0.84114
`TS(1.0,1.0)`	0.07937	0.00079	0.07788	0.08044	0.83493	0.00798	0.82383	0.84477
`PTS(2,2)`	**0.06069**	0.00575	0.05075	0.06470	**0.70484**	0.03062	0.64792	0.74610
`PTS(2,10)`	0.05699	0.00410	0.05130	0.06208	0.65046	0.01124	0.63586	0.66977
`PTS(5,10)`	0.05778	0.00275	0.05589	0.06251	0.63777	0.00811	0.62971	0.65181
`PTS(5,20)`	0.05726	0.00438	0.05096	0.06321	0.62289	0.00714	0.61250	0.63567
`PTS(10,20)`	0.05490	0.00271	0.05179	0.05839	0.61819	0.01044	0.60662	0.63818
`ICTRTS(2,5)`	0.06888	0.00483	0.06369	0.07671	0.70386	0.15772	0.48652	0.85596
`ICTRTS(2,10)`	0.06712	0.01873	0.03731	0.08487	0.56643	0.10242	0.42974	0.67630
`ICTRTS(3,10)`	0.06953	0.00783	0.05857	0.07804	**0.88512**	0.00052	0.88438	0.88553
`ICTRTS(5,10)`	**0.08321**	0.08236	0.08492	0.06292	0.55748	0.14168	0.38715	0.73404
`ICTRTS(7,10)`	0.05066	0.00885	0.04229	0.06423	0.517826	0.07120	0.42297	0.59454
`ICTRTS(7,20)`	0.04925	0.00223	0.04672	0.05285	0.61414	0.12186	0.44685	0.73365
`ICTRUCB(2,10,0.01)`	0.06673	0.01233	0.04588	0.08112	0.44650	0.06689	0.38678	0.53991
`ICTRUCB(2,10,1.0)`	**0.08597**	0.00056	0.08521	0.08675	**0.86411**	0.01528	0.85059	0.88547
`ICTRUCB(3,10,0.05)`	0.07250	0.00426	0.06799	0.07694	0.54757	0.13265	0.43665	0.73407
`ICTRUCB(3,10,1.0)`	0.08196	0.00296	0.07766	0.08530	0.57805	0.08716	0.46453	0.67641
`ICTRUCB(5,10,0.01)`	0.07009	0.00722	0.06411	0.08244	0.62282	0.02572	0.59322	0.65594
`ICTRUCB(5,10,1.0)`	0.08329	0.00140	0.08098	0.08481	0.80038	0.24095	0.29625	0.88554

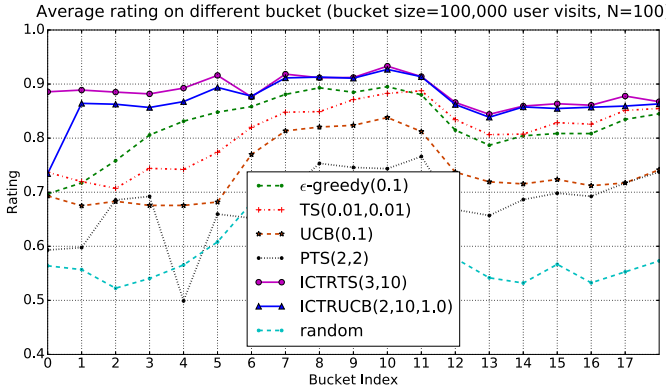


Fig. 3. The average rating of MovieLens (10M) data is given along each time bucket. All algorithms shown here are configured with their best parameter settings.

model parameters related to this item, but also utilized to refine the parameters for the item's cluster. As a result, the updated cluster parameters further influence the model's parameter inference for other items within the same cluster. The effect of the clustering is illustrated in more details in the next section.

5.5 A Case Study: Topic Distribution Analysis on MovieLens (10M)

We conduct an experiment to demonstrate that our model can effectively capture the dependency between items, i.e., finding the latent topics among movies and clustering similar movies together. In this experiment, top-N (N = 8) popular movies are selected and topic number (K = 2) is set for our model. After millions of training iterations, the learned latent movie feature vectors will represent each movie's topic distribution over the two latent topics, in which the ith dimension of the feature vector encodes the probability that the movie belongs to the ith movie topic cluster. We separately choose four movies with the highest value of the first element and the second element of these latent feature vectors, and list their IDs, names, and movie types in Table 5, which clearly proves our assumption that the model is able to capture the dependency between items and cluster similar movies together.

5.6 Time Cost

The cumulative time cost of each algorithm on both datasets is presented in Figs. 4a and 4b, where all algorithms are configured with their best parameter settings. Our proposed algorithms have higher running time since they needs to learn the latent features for arms. However, the computational complexity of both `ICTRUCB(1,1,1.0)` and `ICTRTS(1,1)` is comparable to the baselines'. We also evaluate the time costs of `ICTRTS` and `ICTRUCB` with different number of particles and latent feature vector dimensions on the two datasets (see Figs. 4c and 4d). It shows that the time cost grows linearly with the number of particles and dimensions of latent feature vector.

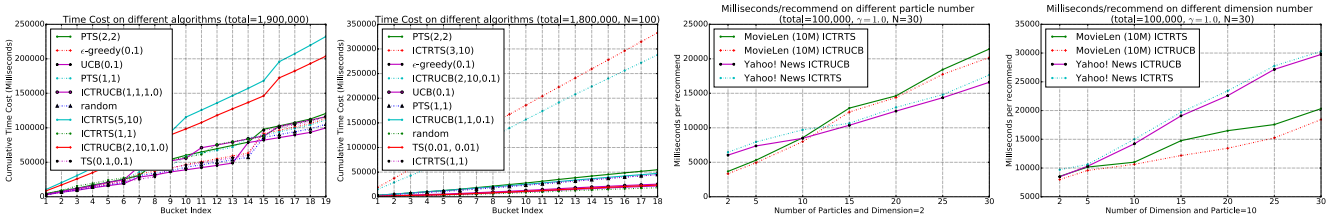
The observations can be summarized as follows: (1) MovieLens (10M) requires much more time than Yahoo! Today New due to a larger amount of items and users. (2) In general, `UCB`-based algorithms (e.g., `ICTRUCB`, `UCB`) are faster than `TS`-based ones (e.g., `ICTRTS`, `PTS`) since the `TS`-based algorithms highly depend on the sampling process.

6 Conclusion and Future Work

In this paper, we propose an interactive collaborative topic regression model that adopts a generative process based on topic model to explicitly formulate the arm dependencies as the clusters on arms, where dependent arms are assumed to be generated from the same cluster. Every time an arm is pulled, the feedback is not only used for inferring the involved user and item latent vectors, but also employed to update the latent parameters with respect to the arm's cluster. The latent cluster parameters further help with the reward prediction for other arms in the same cluster. We

TABLE 5
Movie Topic Distribution of MovieLen (10M)

Topic Cluster I			Topic Cluster II		
MovieId	MovieName	MovieType	MovieId	MovieName	MovieType
32	12 Monkeys	Sci-Fi,Thriller	344	Pet Detective	Comedy
50	Usual Suspects	Crime,Mystery,Thriller	588	Aladdin	Children,Animation,Comedy
590	Dances with wolves	Adventure,Drama,Western	595	Beauty and the Beast	Animation,Children,Musical
592	Batman	Action,Crime,Sci-Fi,Thriller	2857	Yellow Submarine	Adventure,Animation,Comedy,Musical



(a) Cumulative time cost of Yahoo! Today News is given along each time bucket.

(b) Cumulative time cost of MovieLens (10M) is given along each time bucket.

(c) Time cost is given with different number of particles.

(d) Time cost is given with different number of latent feature vector dimensions.

Fig. 4. Time cost comparison on both two datasets.

conduct empirical studies on two real-world applications, including movie and news recommendation, and the experimental results demonstrate the effectiveness of our proposed approach.

Individual preferences on news and movies usually evolve over time. One possible research direction is to extend our model considering the time-varying property in user preferences for better online personal recommendation [1]. In addition, we would like to provide a comprehensive regret analysis [31] of our model in the future work.

Acknowledgments

The work was supported in part by the National Science Foundation under Grant Nos. CNS-1461926 and FIU Dissertation Year Fellowship.

References

[1] C. Zeng, Q. Wang, S. Mokhtari, and T. Li, "Online context-aware recommendation with time varying multi-armed bandit," in *Proc. 22nd ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, 2016, pp. 2025–2034.

[2] S. Chang, J. Zhou, P. Chubak, J. Hu, and T. S. Huang, "A space alignment method for cold-start TV show recommendations," in *Proc. 24th Int. Conf. Artif. Intell.*, 2015, pp. 3373–3379.

[3] A. Barraza-Urbina, "The exploration-exploitation trade-off in interactive recommender systems," in *Proc. 11th ACM Conf. Recommender Syst.*, 2017, pp. 431–435.

[4] L. Li, W. Chu, J. Langford, and R. E. Schapire, "A contextual-bandit approach to personalized news article recommendation," in *Proc. 19th Int. Conf. World Wide Web*, 2010, pp. 661–670.

[5] J. B. Schafer, D. Frankowski, J. Herlocker, and S. Sen, "Collaborative filtering recommender systems," in *The Adaptive Web*. Berlin, Germany: Springer, 2007, pp. 291–324.

[6] J. Bennett, S. Lanning et al., "The netflix prize," in *Proc. KDD Cup Workshop*, 2007, Art. no. 35.

[7] Y. Koren, R. Bell, and C. Volinsky, "Matrix factorization techniques for recommender systems," *Comput.*, vol. 42, no. 8, pp. 30–37, 2009.

[8] J. Kawale, H. H. Bui, B. Kveton, L. Tran-Thanh, and S. Chawla, "Efficient thompson sampling for online matrix-factorization recommendation," in *Proc. Int. Conf. Neural Inf. Process. Syst.*, 2015, pp. 1297–1305.

[9] J. L. Herlocker, J. A. Konstan, A. Borchers, and J. Riedl, "An algorithmic framework for performing collaborative filtering," in *Proc. 22nd Annu. Int. ACM SIGIR Conf. Res. Develop. Inf. Retrieval*, 1999, pp. 230–237.

[10] X. Zhao, W. Zhang, and J. Wang, "Interactive collaborative filtering," in *Proc. 22nd ACM Int. Conf. Inf. Knowl. Manage.*, 2013, pp. 1411–1420.

[11] C. Carvalho, M. S. Johannes, H. F. Lopes, and N. Polson, "Particle learning and smoothing," *Statistical Sci.*, vol. 25, no. 1, pp. 88–106, 2010.

[12] B. Sarwar, G. Karypis, J. Konstan, and J. Riedl, "Item-based collaborative filtering recommendation algorithms," in *Proc. 10th Int. Conf. World Wide Web*, 2001, pp. 285–295.

[13] R. Salakhutdinov and A. Mnih, "Probabilistic matrix factorization," in *Proc. 20th Int. Conf. Neural Inf. Process. Syst.*, 2007, pp. 1257–1264.

[14] R. Salakhutdinov and A. Mnih, "Bayesian probabilistic matrix factorization using Markov chain Monte Carlo," in *Proc. 25th Int. Conf. Mach. Learn.*, 2008, pp. 880–887.

[15] L. Peska, K. Buza, and J. Koller, "Drug-target interaction prediction: A Bayesian ranking approach," *Comput. Methods Programs Biomed.*, vol. 152, pp. 15–21, 2017.

[16] L. Wu, C.-J. Hsieh, and J. Sharpnack, "Large-scale collaborative ranking in near-linear time," in *Proc. 23rd ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, 2017, pp. 515–524.

[17] K. Buza and L. Peska, "ALADIN: A new approach for drug–target interaction prediction," in *Proc. Joint Eur. Conf. Mach. Learn. Knowl. Discovery Databases*, 2017, pp. 322–337.

[18] H. J. Ahn, "A new similarity measure for collaborative filtering to alleviate the new user cold-starting problem," *Inf. Sci.*, vol. 178, no. 1, pp. 37–51, 2008.

[19] A. I. Schein, A. Popescul, L. H. Ungar, and D. M. Pennock, "Methods and metrics for cold-start recommendations," in *Proc. 25th Annu. Int. ACM SIGIR Conf. Res. Develop. Inf. Retrieval*, 2002, pp. 253–260.

[20] A. M. Rashid, I. Albert, D. Cosley, S. K. Lam, S. M. McNee, J. A. Konstan, and J. Riedl, "Getting to know you: Learning new user preferences in recommender systems," in *Proc. 7th Int. Conf. Intell. User Interfaces*, 2002, pp. 127–134.

[21] A. M. Rashid, G. Karypis, and J. Riedl, "Learning preferences of new users in recommender systems: An information theoretic approach," *ACM SIGKDD Explorations Newsletter*, vol. 10, no. 2, pp. 90–100, 2008.

[22] O. Chapelle and L. Li, "An empirical evaluation of thompson sampling," in *Proc. Int. Conf. Neural Inf. Process. Syst.*, 2011, pp. 2249–2257.

[23] D. Bouneffouf, A. Bouzeghoub, and A. L. Gançarski, "A contextual-bandit algorithm for mobile context-aware recommender system," in *Proc. Int. Conf. Neural Inf. Process.*, 2012, pp. 324–331.

[24] H. Wang, Q. Wu, and H. Wang, "Learning hidden features for contextual bandits," in *Proc. 25th ACM Int. Conf. Inf. Knowl. Manage.*, 2016, pp. 1633–1642.

[25] S. Pandey, D. Chakrabarti, and D. Agarwal, "Multi-armed bandit problems with dependent arms," in *Proc. 24th Int. Conf. Mach. Learn.*, 2007, pp. 721–728.

[26] C. Gentile, S. Li, and G. Zappella, "Online clustering of bandits," in *Proc. 31st Int. Conf. Mach. Learn.*, 2014, pp. 757–765.

[27] Q. Wu, H. Wang, Q. Gu, and H. Wang, "Contextual bandits in a collaborative environment," in *Proc. 39th Int. ACM SIGIR Conf. Res. Develop. Inf. Retrieval*, 2016, pp. 529–538.

[28] H. Wang, Q. Wu, and H. Wang, "Factorization bandits for interactive recommendation," in *Proc. 31st AAAI Conf. Artif. Intell.*, 2017, pp. 2695–2702.

[29] X. Wang, S. C. Hoi, C. Liu, and M. Ester, "Interactive social recommendation," in *Proc. ACM Conf. Inf. Knowl. Manage.*, 2017, pp. 357–366.

[30] L. Song, C. Tekin, and M. van der Schaar, "Online learning in large-scale contextual recommender systems," *IEEE Trans. Serv. Comput.*, vol. 9, no. 3, pp. 433–445, May/Jun. 2016.

[31] C. Gentile, S. Li, P. Kar, A. Karatzoglou, G. Zappella, and E. Etrue, "On context-dependent clustering of bandits," in *Proc. Int. Conf. Mach. Learn.*, 2017, pp. 1253–1262.

[32] Q. Wang, T. Li, S. Iyengar, L. Shwartz, and G. Y. Grabarnik, "Online it ticket automation recommendation using hierarchical multi-armed bandit algorithms," in *Proc. SIAM Int. Conf. Data Mining*, 2018, pp. 657–665.

[33] S. Pandey and C. Olston, "Handling advertisements of unknown quality in search advertising," in *Proc. Int. Conf. Neural Inf. Process. Syst.*, 2007, pp. 1065–1072.

[34] D. Agarwal, B.-C. Chen, and P. Elango, "Explore/exploit schemes for web content optimization," in *Proc. 9th IEEE Int. Conf. Data Mining*, 2009, pp. 1–10.

[35] B. Awerbuch and R. Kleinberg, "Online linear optimization and adaptive routing," *J. Comput. Syst. Sci.*, vol. 74, no. 1, pp. 97–114, 2008.

[36] M. Tokic, "Adaptive ε-greedy exploration in reinforcement learning based on value differences," in *Proc. 33rd Annu. German Conf. Advances Artif. Intell.*, 2010, pp. 203–210.

[37] D. K. Mahajan, R. Rastogi, C. Tiwari, and A. Mitra, "LogUCB: An explore-exploit algorithm for comments recommendation," in *Proc. 21st ACM Int. Conf. Inf. Knowl. Manage.*, 2012, pp. 6–15.

[38] P. Auer, N. Cesa-Bianchi, Y. Freund, and R. E. Schapire, "The nonstochastic multiarmed bandit problem," *SIAM J. Comput.*, vol. 32, no. 1, pp. 48–77, 2002.

[39] S. Agrawal and N. Goyal, "Thompson sampling for contextual bandits with linear payoffs," in *Proc. Int. Conf. Mach. Learn.*, 2013, pp. 127–135.

[40] D. M. Blei, A. Y. Ng, and M. I. Jordan, "Latent dirichlet allocation," *J. Mach. Learn. Res.*, vol. 3, no. Jan, pp. 993–1022, 2003.

[41] J. H. Halton, "Sequential Monte Carlo," *Math. Proc. Cambridge Philosoph. Soc.*, vol. 58, no. 01, pp. 57–78, 1962.

[42] A. Doucet, N. De Freitas, and N. Gordon, "An introduction to sequential Monte Carlo methods," in *Sequential Monte Carlo Methods in Practice*. Berlin, Germany: Springer, 2001, pp. 3–14.

[43] C. Zeng, Q. Wang, W. Wang, T. Li, and L. Shwartz, "Online inference for time-varying temporal dependency discovery from time series," in *Proc. IEEE Int. Conf. Big Data*, 2016, pp. 1281–1290.

[44] A. Doucet, S. Godsill, and C. Andrieu, "On sequential Monte Carlo sampling methods for Bayesian filtering," *Statist. Comput.*, vol. 10, no. 3, pp. 197–208, 2000.

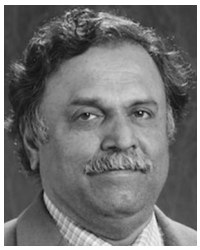
[45] A. Smith, A. Doucet, N. de Freitas, and N. Gordon, *Sequential Monte Carlo Methods in Practice*. Berlin, Germany: Springer Science & Business Media, 2013.
[46] P. M. Djurić, J. H. Kotecha, J. Zhang, Y. Huang, T. Ghirmai, M. F. Bugallo, and J. Miguez, "Particle filtering," *IEEE Signal Process. Mag.*, vol. 20, no. 5, pp. 19–38, Sep. 2003.
[47] L. Zhou and E. Brunskill, "Latent contextual bandits and their application to personalized recommendations for new users," arXiv:1604.06743, 2016.
[48] L. Li, W. Chu, J. Langford, T. Moon, and X. Wang, "An unbiased offline evaluation of contextual bandit algorithms with generalized linear models," *J. Mach. Learn. Res.*, vol. 26, pp. 19–36, 2012.

Qing Wang received the BS and MS degrees in computer science from Zhengzhou University and Xidian University, in 2009 and 2013, respectively. She is currently working toward the PhD degree in the School of Computer Science, Florida International University. Her research interests include interactive recommender system, multi-armed bandit, and large scale data mining.

Chunqiu Zeng received the BS and MS degrees in computer science from Sichuan University, in 2006 and 2009, respectively. He is currently working toward the PhD degree in the School of Computer Science, Florida International University. His research interests include event mining, system management, and large scale data mining.

Wubai Zhou received the BS degree in computer science and technology from Wuhan University, in 2012. He is currently working toward the PhD degree in the School of Computer Science, Florida International University. His research interests include system oriented data mining, system management, and machine learning.

Tao Li received the PhD degree in computer science from the University of Rochester, in 2004. He was an eminent scholar professor with the School of Computer Science, Florida International University when he died in Dec. 2017. His research interests was in data mining and machine learning studying both on the algorithmic and application issues, computing system management, and information retrieval. He was a recipient of the USA NSF CAREER Award and multiple IBM Faculty Research Awards.



S. S. Iyengar is the director and Ryder professor with the School of Computer Science, Florida International University. He is a member of the European Academy of Sciences, a fellow of the Institute of Electrical and Electronics Engineers (IEEE), a fellow of the Association of Computing Machinery (ACM), a fellow of the American Association for the Advancement of Science (AAAS), and a fellow of the Society for Design and Process Science (SDPS). He is a leading researcher with the fields of distributed sensor networks, computational robotics, and oceanographic applications, and is perhaps best known for introducing novel data structures and algorithmic techniques for large-scale computations in sensor technologies and image processing applications.

Larisa Shwartz (M'05) received the PhD degree in mathematics from UNISA University. She is currently a researcher at the IBM T.J. Watson Research Center, Yorktown Heights, NY. She has research experience in mathematics and computer science, but is now focusing on IT service management technologies for service delivery. She has more than 55 publications and 52 patents. She is a member of the IEEE.

Genady Ya. Grabarnik is currently a professor in the Math and CS Department at St John's University. He is a trained mathematician and has authored more than 80 papers. He spent 10 years at the IBM T.J. Watson Research Center where his work was celebrated with a number of awards including Outstanding Technical Achievement Award and Research Achievement Awards. He is a prolific inventor with more than 65 US patents. His interests include research in functional analysis, inventions, and research in computer science and artificial intelligence.