Contextual Bandits in A Collaborative Environment

Qingyun Wu[1], Huazheng Wang[1], Quanquan Gu[2], Hongning Wang[1]
[1]Department of Computer Science
[2]Department of Systems and Information Engineering
University of Virginia, Charlottesville VA, 22904 USA
{qw2ky,hw7ww,qg5w,hw5x}@virginia.edu

ABSTRACT

Contextual bandit algorithms provide principled online learning solutions to find optimal trade-offs between exploration and exploitation with companion side-information. They have been extensively used in many important practical scenarios, such as display advertising and content recommendation. A common practice estimates the unknown bandit parameters pertaining to each user independently. This unfortunately ignores dependency among users and thus leads to suboptimal solutions, especially for the applications that have strong social components.

In this paper, we develop a collaborative contextual bandit algorithm, in which the adjacency graph among users is leveraged to share context and payoffs among neighboring users while online updating. We rigorously prove an improved upper regret bound of the proposed collaborative bandit algorithm comparing to conventional independent bandit algorithms. Extensive experiments on both synthetic and three large-scale real-world datasets verified the improvement of our proposed algorithm against several state-of-the-art contextual bandit algorithms.

CCS Concepts

•Information systems → Recommender systems; •Theory of computation → Regret bounds; •Computing methodologies → Sequential decision making;

Keywords

Collaborative contextual bandits; online recommendations; reinforcement learning

1. INTRODUCTION

Satisfying users with personalized information plays a crucial role for online service providers to succeed in market. However, the rapid appearance of new information and new users together with the ever-changing nature of content popularity make traditional recommendation approaches, e.g., collaborative filtering [7, 25], incompetent. Modern information service systems now adopt online learning solutions to adaptively find good mappings between available content and users. During online learning, the need to focus on information that raises user interest and, simultaneously, the need to explore new information for globally improving user experience create an explore-exploit dilemma. Significant research attention has been paid on multi-armed bandit algorithms [16, 19, 4, 5], which provide a principled solution of the dilemma. Intuitively, bandit algorithms designate a small amount of traffic to collect user feedback while improving their estimation qualities in realtime.

With the available side information about users or items to be presented, contextual bandits have become a reference solution [3, 10, 21, 14]. Specifically, contextual bandits assume the expected payoff is determined by a conjecture of unknown bandit parameters and given context, which is represented as a set of features extracted from both users and recommendation candidates. Such algorithms are especially advantageous when the space of recommendation is large but the payoffs are interrelated. They have been successfully applied in many important applications, e.g., content recommendation [21, 6] and display advertising [11, 23].

However, a common practice in contextual bandit algorithms estimates the unknown bandit parameters pertaining to each user independently. This unfortunately ignores dependency among users. Due to the existence of social influence [13], e.g., content and opinions sharing among friends in a social network, exploiting the dependency among users raises new challenges and opportunities in personalized information services. For example, in many real-world applications, e.g., content recommendation in Facebook or Twitter, because of the mutual influence among friends and acquaintances, one user's click decision on the recommended items might be greatly influenced by his/her peers. This indicates the knowledge gathered about the interest of a given user can be leveraged to improve the recommendation to his/her friends, i.e., collaborative learning. In other words, the observed payoffs from a user's feedback might be a compound of his/her own preference and social influence he/she receives, e.g., social norms, conformity and compliance. As a result, propagating the knowledge collected about the preference of one user to his/her related peers can not only capitalize on additional information embedded in the dependency among users, which is not available in the context vectors; but also helps conquer data sparsity issue by reducing the sample complexity of preference learning (e.g., known as cold-start in recommender systems [26]). Failing to recognize such information among users will inevitably lead to a suboptimal solution.

In this work, we develop a collaborative contextual bandit algorithm that explicitly models the underlying dependency among users. In our solution, a weighted adjacency graph is constructed, where each node represents a contextual bandit deployed for a single user and the weight on each edge indicates the influence between a pair of users. Based on this dependency structure, the observed payoffs on each user are assumed to be determined by a mixture of neighboring users in the graph. We then estimate the bandit parameters over all the users in a collaborative manner: both

SIGIR '16, July 17-21, 2016, Pisa, Italy

DOI: http://dx.doi.org/10.1145/2911451.2911528

context and received payoffs from one user are prorogated across the whole graph in the process of online updating. The proposed collaborative bandit algorithm establishes a bridge to share information among heterogenous users and thus reduce the sample complexity of preference learning. We rigorously prove that our collaborative bandit algorithm achieves a remarkable reduction of upper regret bound with high probability, comparing to the linear regret with respect to the number of users if one simply runs independent bandits on them. Extensive experiment results on both simulations and large-scale real-world datasets verified the improvement of the proposed algorithm compared with several state-of-the-art contextual bandit algorithms. In particular, our algorithm greatly alleviates the cold-start challenge, in which encouraging performance improvement is achieved on new users and new items.

2. RELATED WORK

Multi-armed bandit algorithms provide principled solutions to the explore/exploit dilemma, which exists in many real-world applications, such as display advertisement selection [11, 23], recommender systems [21, 6], and search engine systems [24, 28]. As opposed to the traditional K-armed bandit problems [5, 4, 16, 19], feature vectors in contextual bandits are created to infer the conditional expected payoff of an action [3, 12, 20, 21]. The setting for contextual bandit with linear payoffs was first introduced in [3], where the expectation of payoff for each action is assumed to be a linear function of its context vector. In the follow-up research [21, 12], LinUCB is introduced to use ridge regression to compute the expected payoff of each action and corresponding confidence interval. Later on, generalized linear models are introduced to parameterize bandit algorithms for non-linear payoffs [14]. Comparing to their context-free counterparts, contextual bandits have achieved superior performance in various application scenarios [21, 14].

The idea of modeling dependency among bandits has been explored in prior research [8, 9, 10, 15, 17]. Studies in [2, 27] explore contextual bandits with assumptions about probabilistic dependencies on the product space of context and actions. Hybrid-LinUCB [21] is such an instance, which uses a hybrid linear model to share observations across users. Social network structures are explored in bandit algorithms for introducing possible dependencies [9, 15]. In [8], parallel context-free K-armed bandits are coupled by the social network structure among the users, where the observed payoffs from neighboring nodes are shared as side-observations to help estimate individual bandits. Besides utilizing existing social networks for modeling relatedness among bandits, there is also work automatically estimates the bandit parameters together with the dependency relation among them, such as clustering the bandits via the learned model parameters during online updating [15]. Some recent work incorporates collaboration among bandits via matrix factorization based collaborative filtering techniques: Kawale et al. preformed online matrix factorization based recommendation via Thompson sampling [18], and Zhao et al. studied interactive collaborative filtering via probabilistic matrix factorization [29].

The most similar work to ours studied in this paper is the GOB.Lin algorithm introduced in [10]. GOB.Lin requires connected users in a network to have similar bandit parameters via a graph Laplacian based model regularization. As a result, GOB.Lin explicitly requires the learned bandit parameters across related users to be close to each other. In our algorithm, we do not have such strong assumption about each individual bandit, but we make explicit assumptions about the reward generation via an additive model: neighboring users' judgements of the recommendations will be shared across, i.e., word-of-mouth, to explain the observed payoffs in different users. This gives us the flexibility in capturing the heterogeneity of preferences among different users in practice, and leads to both theoretically and empirically improved results.

3. METHODOLOGY

We develop a contextual bandit algorithm in a collaborative environment, where the adjacency graph among users is leveraged to share context and payoffs between neighboring bandits during online update. We provide rigorous proof of the resulting upper regret bound, which has a significant regret reduction comparing to running independent bandit algorithms on the same collection of users. In the following discussions, we will first describe the notations and our model assumptions about the collaborative bandit problem, then carefully illustrate our developed bandit algorithm and corresponding regret analysis.

3.1 Bandits in a Collaborative Setting

We consider a contextual bandit problem with a finite, but possibly large, number of arms, which correspond to the candidate item set to be presented (e.g., articles in a content recommendation system). We denote the arm set as $\mathcal{A}$ and the cardinality of $\mathcal{A}$ as K. There are N different users in this collection. At each trial t, a learner observes a given user u_t from user collection and a subset of arms from $\mathcal{A}$, where each arm a is associated with a feature vector $\mathbf{x}_{a_t,u_t} \in \mathbb{R}^d$ summarizing the information of both user u_t and arm a_t at trial t. The learner displays an arm a_t to u_t and receives the corresponding payoff r_{a_t,u_t} from the user. The goal of the learner is to update its arm-selection strategy with respect to the new observations, such that after T trials its *regret* with respect to the oracle arm selection strategy is minimized. In particular, the accumulated T-trial regret for all users is defined formally as,

$$\mathbf{R}(T) = \sum_{t=1}^{T} R_t = \sum_{t=1}^{T} (r_{a_t^*,u_t} - r_{a_t,u_t}) \tag{1}$$

where a_t^* is the best arm to display to user u_t at trial t according to the oracle strategy, $r_{a_t^*,u_t}$ is the corresponding optimal payoff, and R_t is the regret from all users at trial t.

In standard linear contextual bandit problems, the payoffs of each arm with respect to different users are assumed to be governed by a noisy version of an unknown linear function of the context vectors [3, 21]. Specifically, each user i is assumed to associate with an unknown parameter $\boldsymbol{\theta}_i \in \mathbb{R}^d$ (with $\|\boldsymbol{\theta}_i\| \leq 1$), which determines the payoff of a_t by $r_{a_{t,i}} = \mathbf{x}_{a_{t,i}}^{\mathsf{T}} \boldsymbol{\theta}_i + \epsilon_t$, where ϵ_t is drawn from a Guassian distribution $N(0, \sigma^2)$. $\boldsymbol{\theta}$s are independently estimated based on the observations from each individual user. However, due to the existence of mutual influence among users, an isolated bandit can hardly explain all the observed payoffs even for a single user. For example, the context vectors fail to encode such dependency. To capitalize on the additional information embedded in the dependency structure among users (i.e., $\boldsymbol{\theta}$ for different users), we propose to study contextual bandit problems in a collaborative setting.

In this collaborative environment, we place the bandit algorithms on a weighted graph $G = (V, E)$, which encodes the affinity relationship among users. Specifically, each node $v_i \in \{V_1, ..., V_N\}$ in G hosts a bandit parameterized by $\boldsymbol{\theta}_i$ for user i; and the edges in E represent the affinity relation over pairs of users. This graph can be described as an $N \times N$ stochastic matrix $\mathbf{W}$. In this matrix, each element w_{ij} is nonnegative and proportional to the influence that user j has on user i in determining the payoffs of different arms. $w_{ij} = 0$ if and only if user j has no influence on user i. $\mathbf{W}$ is column-wise normalized such that $\sum_{j=1}^{N} w_{ij} = 1$ for $i \in \{1, ..., N\}$. In this work, we assume $\mathbf{W}$ is time-invariant and known to the learner beforehand.

Based on the graph G, collaboration among bandits happens when determining the payoff of a particular arm with respect to a given user. To denote this, we define a $d \times N$ matrix $\boldsymbol{\Theta}$, which consists of parameters from all the bandits in the graph: $\boldsymbol{\Theta} = (\boldsymbol{\theta}_1, \ldots, \boldsymbol{\theta}_N)$. Accordingly, we define a context feature matrix

Algorithm 1 Collaborative LinUCB

Inputs: $\alpha \in \mathbb{R}_+$,$\lambda \in [0,1]$, $\mathbf{W} \in \mathbb{R}^{N \times N}$
Initialize: $\mathbf{A}_1 \leftarrow \lambda \mathbf{I}$, $\mathbf{b}_1 \leftarrow \mathbf{0}$, $\hat{\boldsymbol{\vartheta}}_1 \leftarrow \mathbf{A}_1^{-1}\mathbf{b}_1$, $\mathbf{C}_1 \leftarrow (\mathbf{W}^\top \otimes \mathbf{I})\mathbf{A}_1^{-1}(\mathbf{W} \otimes \mathbf{I})$,
for $t = 1$ to T **do**
  Receive user u_t
  Observe context vectors, $\mathbf{x}_{a_t,u_t} \in \mathbb{R}^d$ for $\forall a \in \mathcal{A}$
  Take action $a_t = \arg\max_{a \in \mathcal{A}} \mathring{\mathcal{X}}_{a_t,u_t}^\top vec(\widehat{\boldsymbol{\Theta}}_t \mathbf{W}) + \alpha\sqrt{\mathring{\mathcal{X}}_{a_t,u_t}^\top \mathbf{C}_t \mathring{\mathcal{X}}_{a_t,u_t}}$
  Observe payoff r_{a_t,u_t}
  $\mathbf{A}_{t+1} \leftarrow \mathbf{A}_t + vec(\mathring{\mathbf{X}}_{a_t,u_t}\mathbf{W}^\top)vec(\mathring{\mathbf{X}}_{a_t,u_t}\mathbf{W}^\top)^\top$
  $\mathbf{b}_{t+1} \leftarrow \mathbf{b}_t + vec(\mathring{\mathbf{X}}_{a_t,u_t}\mathbf{W}^\top)r_{a_t,u_t}$
  $\mathbf{C}_{t+1} \leftarrow (\mathbf{W}^\top \otimes \mathbf{I})\mathbf{A}_{t+1}^{-1}(\mathbf{W} \otimes \mathbf{I})$
  $\hat{\boldsymbol{\vartheta}}_{t+1} \leftarrow \mathbf{A}_{t+1}^{-1}\mathbf{b}_{t+1}$
end for

$\mathbf{X}_t = (\mathbf{x}_{a_{t,1}}, \ldots, \mathbf{x}_{a_{t,N}})$, where the ith column is the context vector $\mathbf{x}_{a_{t,i}}$ for arm a at trial t selected for user i. The collaboration among bandits characterized by the influence matrix $\mathbf{W}$ results in a new bandit parameter matrix $\bar{\Theta} = \boldsymbol{\Theta}\mathbf{W}$, which determines the payoff r_{a_t,u_t} of arm a_t for user u_t at trial t by,

$$r_{a_t,u_t} - diag_{u_t}(\mathbf{X}_t^\top \boldsymbol{\Theta}\mathbf{W}) \sim N(0, \sigma^2) \tag{2}$$

where $diag_{u_t}(\mathbf{X})$ is the operation returning the u_t-th element in the diagonal of matrix $\mathbf{X}$. Eq (2) postulates our *additive* assumption about reward generation in this collaborative environment: the reward r_{a_t,u_t} is not only determined by user u_t's own preference on the arm a_t (i.e., $w_{u_t u_t}\mathbf{x}_{a_t,u_t}^\top \boldsymbol{\theta}_{u_t}$), but also by the judgements from the neighbors who have influence on u_t (i.e., $\sum_{j \neq u_t} w_{u_t j}\mathbf{x}_{a_t,j}^\top \boldsymbol{\theta}_j$). This enables us to distinguish a user's intrinsic preference of the recommended content from his/her neighbors' influence, i.e., personalization. In addition, the linear payoff assumption in our model is to simplify the discussion in this paper; and it can be relaxed via a generalized linear model [14] to deal with nonlinear rewards.

We should note that our model assumption about the collaborative bandits is different from that specified in the GOB.Lin model [10]. In GOB.Lin, connected users in the graph are required to have similar underlying bandit parameters, i.e., via graph Laplacian regularization over the learned bandit parameters. And their assumption about reward generation follows conventional contextual bandit settings, i.e., rewards are independent across users. In our setting, neighboring users do not have to share similar bandit parameters, but they will generate influence on their neighbors' decisions. This assumption is arguably more general, and it leads to an improved upper regret bound and practical performance. Theoretical comparison between these two algorithms will be rigorously discussed in Section 3.3.

3.2 Collaborative Linear Bandit Algorithm

To simplify the notations in our following discussions, we define two long context feature vectors and a long bandit parameter vector based on the vectorize operation $vec(\cdot)$. We define $\mathcal{X}_{a_t} = vec(\mathbf{X}_{a_t}) = (\mathbf{x}_{a_t,1}^\top, \ldots, \mathbf{x}_{a_t,N}^\top)^\top$, which is a concatenation of context feature vectors of the chosen arm a_t at trial t for all the users. And we define $\mathring{\mathcal{X}}_{a_t,u_t} = vec(\mathring{\mathbf{X}}_{a_t,u_t})$, in which $\mathring{\mathbf{X}}_{a_t,u_t}$ is a special case of $\mathbf{X}_{a_t}$: only the column corresponding to the user u_t at time t is set to $\mathbf{x}_{a_t,u_t}^\top$, and all the other columns are set to zero. This corresponds to the situation that at trial t the learner only needs to interact with one user. Correspondingly, we define $\boldsymbol{\vartheta} = vec(\boldsymbol{\Theta}) = (\boldsymbol{\theta}_1^\top, \boldsymbol{\theta}_2^\top, ..., \boldsymbol{\theta}_N^\top)^\top \in \mathbb{R}^{dN}$ as the concatenation of bandit parameter vectors over all the users.

With the collaborative assumption about the expected payoffs defined in Eq (2), we appeal to ridge regression for estimating the unknown bandit parameter $\boldsymbol{\theta}$ for each user. In particular, we simultaneously estimate the global bandit parameter matrix $\boldsymbol{\Theta}$ for all the users as follows,

$$\widehat{\boldsymbol{\Theta}} = \arg\max_{\boldsymbol{\Theta}} \frac{1}{2}\sum_{t=1}^{T}(\mathring{\mathcal{X}}_{a_t,u_t}^\top vec(\boldsymbol{\Theta}_t\mathbf{W}) - r_{a_t,u_t})^2 + \frac{\lambda}{2}tr(\boldsymbol{\Theta}^\top\boldsymbol{\Theta}) \tag{3}$$

where $\lambda \in [0,1]$ is a trade-off parameter of $l2$ regularization in ridge regression.

Since the objective function defined in Eq (3) is quadratic with respect to $\boldsymbol{\Theta}$, we have a closed-form estimation of $\boldsymbol{\Theta}$ as $\hat{\boldsymbol{\vartheta}}_t = \mathbf{A}_t^{-1}\mathbf{b}_t$, in which $\hat{\boldsymbol{\vartheta}} = vec(\widehat{\boldsymbol{\Theta}})$ and $\mathbf{A}_t$ and $\mathbf{b}_t$ are computed as,

$$\mathbf{A}_t = \lambda\mathbf{I} + \sum_{t'=1}^{t} vec(\mathring{\mathbf{X}}_{a_{t'},u_{t'}}\mathbf{W}^\top)vec(\mathring{\mathbf{X}}_{a_{t'},u_{t'}}\mathbf{W}^\top)^\top \tag{4}$$

$$\mathbf{b}_t = \sum_{t'=1}^{t} vec(\mathring{\mathbf{X}}_{a_{t'},u_{t'}}\mathbf{W}^\top)r_{a_{t'},u_{t'}} \tag{5}$$

where $\mathbf{I}$ is a $dN \times dN$ identity matrix.

The effect of collaboration among bandits is clearly depicted in the above estimation of $\boldsymbol{\Theta}$. Matrix $\mathbf{A}_t$ and vector $\mathbf{b}_t$ store global information shared among all the bandits in the graph. More specifically, the context vector $\mathbf{x}_{a_t,u_t}$ and payoff r_{a_t,u_t} observed in user u_t at trial t can be propagated through the whole graph via the relational matrix $\mathbf{W}$ to other users. To understand this, note that $vec(\mathring{\mathbf{X}}_{a_t,u_t}\mathbf{W}^\top)$ is a dense vector with projected context vectors on every user, while the original $\mathring{\mathcal{X}}_{a_t,u_t}$ is a sparse vector with observations only at active users u_t. Because of this information sharing, at certain trial t, although some users might have not generate any observation yet (i.e., cold-start), they can already start from a non-random initialization of their bandit parameters $\boldsymbol{\theta}_i$. It is easy to verify that when $\mathbf{W}$ is an identity matrix, i.e., users have no influence among each other, the estimation of $\boldsymbol{\Theta}$ degenerates to independently computing N different $\boldsymbol{\theta}$s (since $vec(\mathring{\mathbf{X}}_{a_t,u_t}\mathbf{W}^\top) = \mathring{\mathcal{X}}_{a_t,u_t}$). And the mutual influence will be maximized when $\mathbf{W}$ is a uniform matrix, i.e., all the users have equivalent influence to each other. We have to emphasize that the benefit of this collaborative estimation of $\boldsymbol{\Theta}$ is not to just simply compute the $\boldsymbol{\theta}$s in an integrated manner; but because of the collaboration among users, the estimation uncertainty of all $\boldsymbol{\theta}$s can be quickly reduced comparing to simply running N independent bandit algorithms. This in turn leads to an improved regret bound. We will elaborate the effect of collaboration in online bandit learning with more theoretical justifications in Section 3.3.

The estimated bandit parameters $\widehat{\boldsymbol{\Theta}}$ predict the expected payoff of a particular arm for each user according to the observed context feature matrix $\mathbf{X}_t$. To complete an adaptive bandit algorithm, we need to design the exploration strategy for each user. Our collaborative assumption in Eq (2) implies that r_{a_t,u_t} across users are independent given $\mathbf{X}_t$ and $\mathbf{W}$. As a result, for any σ, i.e., the standard deviation of Gaussian noise in Eq (2), the following inequality holds with probability at least $1 - \delta$,

$$|r_{a_t^*,u_t} - r_{a_t,u_t}| \leq \alpha_t\sqrt{vec(\mathring{\mathbf{X}}_{u_t}\mathbf{W}^\top)^\top\mathbf{A}_t^{-1}vec(\mathring{\mathbf{X}}_{u_t}\mathbf{W}^\top)} \tag{6}$$

where α_t is a parameter in our algorithm defined in Lemma 1 of Section 3.3 and δ is embedded in the computation of α_t. The proof of this inequality can be found in the Appendix.

The inequality specified in Eq (6) gives us a reasonably tight upper confidence bound (UCB) for the expected payoff of a particular arm over all the users in the graph G, from which a UCB-style arm-selection strategy can be derived. In particular, at trial t, we

choose an arm for user u_t according to the following arm-selection strategy,

$$a_{t,u_t} = \arg\max_{a \in \mathcal{A}} \Big(\mathring{\mathcal{X}}_{a_t,u_t}^{\intercal} vec(\widehat{\mathbf{\Theta}}_t \mathbf{W}) + \alpha_t \sqrt{vec(\mathring{\mathbf{X}}_{u_t} \mathbf{W}^{\intercal})^{\intercal} \mathbf{A}_t^{-1} vec(\mathring{\mathbf{X}}_{u_t} \mathbf{W}^{\intercal})} \Big) \quad (7)$$

We name this resulting algorithm as Collaborative Linear Bandit, or CoLin in short. The detailed description of CoLin is illustrated in Algorithm 1, where we use the property that $vec(\mathring{\mathbf{X}}_{u_t} \mathbf{W}^{\intercal}) = (\mathbf{W} \otimes \mathbf{I}) vec(\mathring{\mathbf{X}}_{u_t}) = (\mathbf{W} \otimes \mathbf{I}) \mathring{\mathcal{X}}_t$ to simplify Eq (7).

Another evidence of the benefit from collaboration among users is demonstrated in Algorithm 1. When estimating the confidence interval of the expected payoff for action a_t in user u_t at trial t, CoLin not only considers the prediction confidence from bandit u_t, but also that from its neighboring bandits (as described by the Kronecker product between $\mathbf{W}$ and $\mathbf{I}$). When $\mathbf{W}$ is an identity matrix, such effect disappears. Clearly, this collaborative confidence interval estimation will help the algorithm quickly reduce estimation uncertainty, and thus leads to the optimal solution more rapidly.

One potential issue with CoLin is its computational complexity: matrix inverse has to be performed on $\mathbf{A}_t$ at every trial. First, because of the rank one update of matrix $\mathbf{A}_t$ (8th step in Algorithm 1), quadratic computation complexity is possible via applying the Sherman-Morrison formula. Second, we may compute $\mathbf{A}_t^{-1}$ in a mini-batch manner to further reduce computation with some extra penalty in regret. We will leave this as our future research.

3.3 Regret Analysis

In this section, we provide detailed regret analysis of our proposed CoLin algorithm. We first prove that the estimation error of bandit parameters $\widehat{\mathbf{\Theta}}$ is upper bounded in Lemma 1.

Lemma 1*: For any $\delta > 0$, with probability at least $1 - \delta$, the estimation error of bandit parameters in CoLin is bounded by,*

$$\|\hat{\boldsymbol{\vartheta}}_t - \boldsymbol{\vartheta}^*\|_{\mathbf{A}_t} \leq \sqrt{dN \ln\Big(1 + \frac{\sum_{t'=1}^{t} \sum_{j=1}^{N} w_{u_{t'}j}^2}{\lambda dN}\Big) - 2\ln(\delta)} + \sqrt{\lambda} \|\boldsymbol{\vartheta}^*\|$$

in which $\|\hat{\boldsymbol{\vartheta}}_t - \boldsymbol{\vartheta}^*\|_{\mathbf{A}_t} = \sqrt{(\hat{\boldsymbol{\vartheta}}_t - \boldsymbol{\vartheta}^*)^{\intercal} \mathbf{A}_t (\hat{\boldsymbol{\vartheta}}_t - \boldsymbol{\vartheta}^*)}$, i.e., the matrix norm induced by the positive semidefinite matrix $\mathbf{A}_t$

Based on Lemma 1, we have the following theorem about the upper regret bound of the CoLin algorithm.

Theorem 1: With probability at least $1 - \delta$, the cumulated regret of CoLin algorithm satisfies,

$$\mathbf{R}(T) \leq 2\alpha_T \sqrt{2dNT \ln\Big(1 + \frac{\sum_{t=1}^{T} \sum_{j=1}^{N} w_{u_t j}^2}{\lambda dN}\Big)} \quad (8)$$

in which α_T is the upper bound of $\|\hat{\boldsymbol{\vartheta}}_T - \boldsymbol{\vartheta}^*\|_{\mathbf{A}_T}$ and it can be explicitly calculated based on Lemma 1. The detailed proof of this theorem is provided in the Appendix.

As shown in Theorem 1, the graph structure plays an important role in the upper regret bound of our CoLin algorithm. Consider two extreme cases. First, when $\mathbf{W}$ is an identity matrix, i.e., no influence among users, the upper regret bound degenerates to $O(N\sqrt{T} \ln \frac{T}{N})$. Second, when the graph is fully connected and uniform, i.e., $\forall i, j, w_{ij} = \frac{1}{N}$, such that users have homogeneous influence among each other, and the upper regret bound of CoLin decreases to $O(N\sqrt{T} \ln \frac{T}{N^2})$. That means via collaboration, CoLin achieves an $O(\sqrt{T} \ln N)$ regret reduction for every single user in the graph comparing to the independent case.

Note that the our regret analysis in Theorem 1 is in a very general form, in which we did not make any assumption about the order or frequency that each user will be served. To illustrate the relationship between the proposed collaborative bandit algorithm and conventional independent bandit algorithms in a more intuitive way, we can make a very specific assumption about how a sequential learner interacts with a set of users. Assuming all the users are evenly served by CoLin, i.e., each user interacts with the learner $\bar{T} = \frac{T}{N}$ times. When $\mathbf{W}$ is an identity matrix, the regret bound of CoLin degenerates to the case of running N independent LinUCB, whose upper regret bound is $O(N\sqrt{\bar{T}} \ln \bar{T})$. When $\mathbf{W}$ is uniform, the regret bound reduces to $O(N\sqrt{\bar{T}} \ln \frac{\bar{T}}{N})$, where we achieves an $O(\sqrt{\bar{T}} \ln N)$ regret reduction comparing to running N independent LinUCBs on each single user. The proof of regret bound in this special case is given in the Appendix.

It is necessary to compare the derived upper regret bound of CoLin with that in the GOB.Lin algorithm [10], which also exploits the relatedness among a set of users. In GOB.Lin, the divergence among every pair of bandits (if connected in the graph) is measured by Euclidean distance between the learned bandit parameters. In its upper regret bound, such divergence is accumulated throughout the iterations. In extreme case where users are all connected but associate with totally distinct bandit parameters, GOB.Lin's upper regret bound could be much worse than running N independent bandits, due to this additive pairwise divergence. While in our algorithm, such divergence is controlled by the multiplicative factor $\sum_{t=1}^{T} \sum_j w_{u_t j}^2 \leq T$. We can rigorously prove the following inequalities between the upper regret bound of CoLin ($R_C(T)$) and GOB.Lin ($R_G(T)$) always holds,

$$0 \leq R_G^2(T) - R_C^2(T) \leq 16TN \ln(1 + \frac{2T}{dN^2}) \sum_{(i,j) \in E} \|\boldsymbol{\theta}_i^* - \boldsymbol{\theta}_j^*\|^2$$

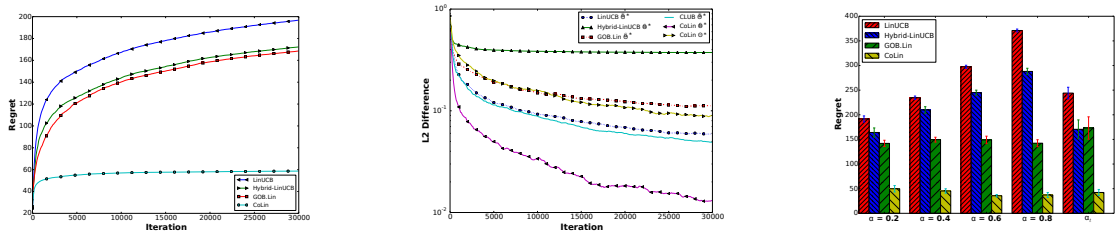
It is clear to notice that if there is no influence between the users in the collection, i.e., no edge in G, these two algorithms' regret bound touches (both degenerate to N independent contextual bandits). Otherwise, GOB.Lin will always lead to a worse and faster growing regret bound than our CoLin algorithm.

In addition, limited by the use of graph Laplacian, GOB.Lin can only capture the binary connectivity relation among users. CoLin differentiates the strength of connections with a stochastic relational graph. This makes CoLin more general when modeling the relatedness among bandits and provides a tighter upper regret bound. This effect is also empirically verified by our extensive experiments on both synthetic and real-world data sets.

4. EXPERIMENTS

We performed empirical evaluations of our CoLin algorithm against several state-of-the-art contextual bandit algorithms, including N independent LinUCB [21], hybrid LinUCB with user features [21], GOB.Lin [10], and online cluster of Bandits (CLUB) [15]. Among these algorithms, hybrid LinUCB exploits user dependency via a set of hybrid linear models over user features, GOB.Lin encodes the user dependency via graph-based regularization over the learned bandit parameters, and CLUB clusters users during online learning to enable model sharing. In addition, we also compared with several popularly used context-free bandit algorithms, including EXP3 [5], UCB1 [4] and ϵ-greedy [4]. But their performance is much worse than the contextual bandits, and thus we do not include their performance in the following discussions.

We tested all the algorithms on a synthetic data set via simulations, a large collection of click stream from Yahoo! Today Module dataset [21], and two real-world dataset extracted from the social bookmarking web service Delicious and music streaming service LastFM [10]. Extensive experiment comparisons confirmed the advantage of our proposed CoLin algorithm against all the baselines. More importantly, comparing to the baselines that also exploit user dependencies, CoLin performs significantly better in identifying



(a) Convergence of cumulated regret (b) Accuracy of bandit parameter estimation (c) The effect of α

Figure 1: Analysis of regret, bandit parameter estimation and parameter tuning.

users' preference on less popular items (items that are only observed among a small group of users). This validates that with the proposed collaborative learning among users, CoLin better alleviates the cold-start challenge comparing to the baselines.

4.1 Experiments on synthetic dataset

In this experiment, we compare the bandit algorithms based on simulations and use the accumulated regret and bandit parameter estimation accuracy as the performance metrics.

4.1.1 Simulation Setting

In simulation, we generate N users, each of which is associated with a d-dimensional parameter vector $\boldsymbol{\theta}^*$, i.e., $\boldsymbol{\Theta}^* = (\boldsymbol{\theta}_1^*, \ldots, \boldsymbol{\theta}_N^*)$. Each dimension of $\boldsymbol{\theta}_i^*$ is drawn from a uniform distribution $U(0,1)$ and normalized to $\|\boldsymbol{\theta}_i^*\| = 1$. $\boldsymbol{\Theta}^*$ is treated as the ground-truth bandit parameters for reward generation, and they are unknown to bandit algorithms. We then construct the golden relational stochastic matrix $\mathbf{W}$ for the graph of users by defining $w_{ij} \propto \langle \boldsymbol{\theta}_i^*, \boldsymbol{\theta}_j^* \rangle$, and normalize each column of $\mathbf{W}$ by its L1 norm. The resulting $\mathbf{W}$ is disclosed to the bandit algorithms. In the end, we generate a size-K action pool $\mathcal{A}$. Each action a in $\mathcal{A}$ is associated with a d-dimensional feature vector $\mathbf{x}_a$, each dimension of which is drawn from $U(0,1)$. We also normalize $\mathbf{x}_a$ by its L1 norm. To construct user features for hybrid LinUCB algorithm, we perform Principle Component Analysis (PCA) on the relational matrix $\mathbf{W}$, and use the first 5 principle components to construct the user features.

To simulate the collaborative reward generation process among users, we first compute $\bar{\Theta}^* = \boldsymbol{\Theta}^* \mathbf{W}$ and then compute the payoff of action a for user i at trial t as $r_{a_{t,i}} = diag_i(\mathbf{X}_t^\top \bar{\Theta}^*) + \epsilon_t$, where $\epsilon_t \sim N(0, \sigma^2)$. To increase the learning complexity, at each trial t, our simulator only discloses a subset of actions in $\mathcal{A}$ to the learning algorithms for selection, e.g., randomly select 10 actions from $\mathcal{A}$ without replacement. In simulation, based on the known bandit parameters $\bar{\Theta}^*$, the optimal action $a_{t,i}^*$ and the corresponding payoff $r_{a_{t,i}^*}$ for each bandit i at trial t can be explicitly computed.

Under this simulation setting, we compared hybrid LinUCB, N independent LinUCB, GOB.Lin, CLUB and CoLin. At each trial, the same set of actions are presented to all the algorithms; and the Gaussian noise ϵ_t is fixed for all those actions at trial t. In our experiments, we fixed the feature dimension d to 5, article pool size K to 1000, and set the trade-off parameter λ for L2 regularization to 0.2 in all the algorithms. We compared the regret of different bandit algorithms during adaptive learning. Furthermore, since the ground-truth bandit parameters are available in the simulator, we also compared the quality of learned parameters in each contextual bandit algorithm. This unveils the nature of each bandit algorithm, e.g., how accurately they can recover a user's true preference.

4.1.2 Results & Analysis

We first set the user size N to 100 and fix the standard deviation σ to 0.1 in the Gaussian noise for reward generation. All the contextual bandit algorithms are executed up to 300 iterations per user in this experiment. We report the cumulated regret as defined in Eq (1) and the Euclidean distance between the learnt bandit parameters from different algorithms and the ground-truth in Figure 1. To reduce randomness in simulation-based evaluation, we reported the mean and standard deviation of final regret from different algorithms after 30,000 iterations over 5 independent runs for results in all following experiments. To increase visibility, we did not plot error bars in Figure 1 (a) and (b).

As we can find in Figure 1 (a), simply running N independent LinUCB algorithm gives us the worst regret, which is expected. Hybrid LinUCB, which exploits user dependency via a set of hybrid linear models over user features performed better than LinUCB, but still much worse than CoLin. Although GOB.Lin also exploits the graph structure when estimating the bandit parameters, its assumption about the dependency among bandits is too restrictive to well capture the information embedded in the interaction with users. We should note that in our simulation, by multiplying the relational matrix $\mathbf{W}$ with the ground-truth bandit parameter matrix $\boldsymbol{\Theta}^*$, the resulting bandit parameters $\bar{\Theta}^*$ align with GOB.Lin's assumption, i.e., neighboring bandits are similar. And $\bar{\Theta}^*$ is used in reward generation. Therefore, our simulation does not produce any bias against GOB.Lin. In Figure 1 (a) we did not include CLUB, whose regret grew linearly. After looking into the selected arms from CLUB, we found because of the aggregated decisions from users in the automatically constructed user clusters, CLUB always chose suboptimal arms, which led to a linearly increasing regret.

In Figure 1 (b), we compared accuracy of the learnt bandit parameters from different algorithms. Because of their distinct modeling assumptions, LinUCB, hybrid LinUCB, CLUB and GOB.Lin cannot directly estimate $\boldsymbol{\Theta}^*$, i.e., the true bandit parameters for each user. Instead, they can only estimate $\bar{\Theta}^*$, which directly governs the generation of observed payoffs. Only CoLin can estimate both $\bar{\Theta}^*$ and $\boldsymbol{\Theta}^*$. As we can find in the results, CoLin gave the most accurate estimation of $\bar{\Theta}^*$, which partially explains its superior performance in regret. We also find that LinUCB actually achieved a more accurate estimation of $\bar{\Theta}^*$ than GOB.Lin, but its regret is much worse. To understand this, we looked into the actual execution of LinUCB and GOB.Lin, and found that because of the graph Laplacian regularization in GOB.Lin, it better controlled exploration in arm selection and therefore picked the optimal arm more often than LinUCB. Hyrbid LinUCB's estimation of $\bar{\Theta}^*$ is the worst, but it is expected: hybird LinUCB uses a shared model and a personalized model to fit the observations. Comparing to CoLin's estimation quality of $\bar{\Theta}^*$, its estimation of $\boldsymbol{\Theta}^*$ is much worse. The main reason is that CoLin has to decipher $\boldsymbol{\Theta}$ from the estimated $\bar{\Theta}$ based on $\mathbf{W}$, where noise is accumulated to prevent accurate estimation. Nevertheless, this result demonstrates the possibility of discovering each individual user's true preference from their compound feedback. This is meaningful for many practical applications, such as user modeling and social influence analysis.

We also notice that although CLUB's estimated $\bar{\Theta}^*$ is almost as good as LinUCB's (as shown in Figure 1 (b)), its regret is the

Table 1: Cumulated regret with different bandit size (σ=0.1).

Bandit Size (N)	40	80	100	200
LinUCB	75.31±5.11	168.42±9.90	191.53±6.18	355.56±7.23
HybridLinUCB	59.12±2.11	150.09±5.29	164.11±9.19	311.43±11.59
GOB.Lin	58.49±5.04	143.42±5.28	141.96±6.36	275.32±10.51
CoLin	21.78±12.84	47.73±4.31	49.83±6.55	77.38±20.59

Table 2: Cumulated regret with different noise level (N=100).

Noise (σ)	0.01	0.05	0.1	0.3
LinUCB	92.72±2.56	116.53±3.07	191.53 ±6.18	830.47±69.48
HybridLinUCB	69.47±2.12	91.48±1.94	164.11±9.19	759.93±39.15
GOB.Lin	58.85±6.25	82.33±1.53	141.96±6.36	708.13±43.73
CoLin	41.69±6.95	40.95±4.43	49.83±6.55	83.98±8.57

Table 3: Cumulated regret with different noise level on matrix $\mathbf{W}$ (N=100).

matrix noise (δ)	0	0.01	0.03	0.05
HybridLinUCB	164.11±9.19	171.74±7.67	171.51±13.31	163.93±9.19
GOB.Lin	141.96±6.36	163.28±5.63	164.36±6.66	169.87±11.89
CoLin	49.83±6.55	54.42±2.45	101.39±6.01	239.88±13.86

Table 4: Cumulated regret with different matrix sparsity level.

Sparsity (M/N)	20/100	40/100	60/100	80/100
HybridLinUCB	135.98±5.11	141.15±4.82	150.49±4.58	160.12±7.55
GOB.Lin	133.30±3.98	126.13±5.59	143.29±6.49	143.42±5.82
CoLin	39.74±8.80	30.76±3.66	37.29±3.55	49.56±8.88

worst. As we described earlier, CLUB's aggregated decision at user cluster level constantly forced the algorithm to choose sub-optimal arms; but the reward generation for each arm in our simulator follows that defined in Eq (2), which still provides validate information for CLUB to estimate $\bar{\Theta}^*$ with reasonable accuracy.

In Figure 1 (c), we investigated the effect of exploration parameter α_t's setting in different algorithms. The last column indexed by α_t represents the theoretical values of α computed from the algorithms' corresponding regret analysis. As shown from results, the empirically tuned α yields comparable performance to the theoretical values, and makes online computation more efficient. As a result, in all our following experiments we use a fixed α instead of a computed α_t.

To further investigate the convergence property of different bandit algorithms, we examined the following four scenarios: 1) various user sizes N, 2) different noise level σ, 3) a corrupted affinity matrix $\mathbf{W}$, and 4) a sparse affinity matrix $\mathbf{W}$, in reward generation. We report the results in Table 1 to 4. Because of its poor performance, we did not include CLUB in those tables.

Firstly, in Table 1, we fixed the noise level σ to 0.1 and varied the user size N from 40 to 200. We should note in this experiment the total number of iterations varies as every user will interact with the bandit learner 300 times. The regret in LinUCB goes linearly with respect to the number of users, since no information is shared across them. Via model sharing, hybrid LinUCB achieved some regret reduction compared with LinUCB; but its regret still increases linearly with the number of users. Compared with the independent bandits, we can clearly observe the regret reduction in CoLin with increasing number of users. As we discussed in Section 3.3, although GOB.Lin exploits the dependency among users, its regret might be even worse than running N independent LinUCBs, especially when the divergence between users is large.

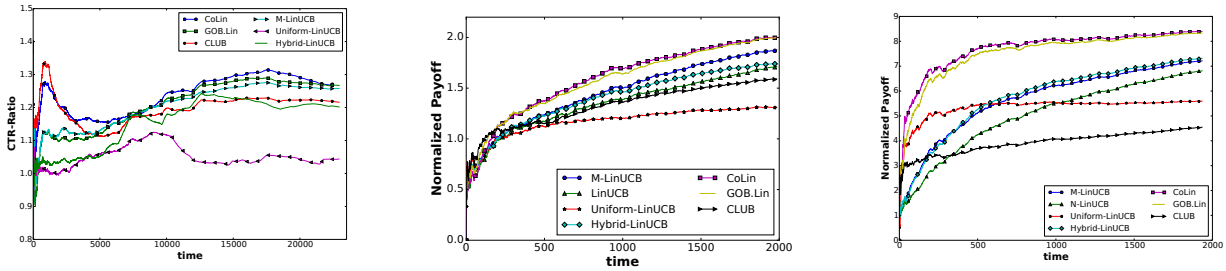
Secondly, in Table 2, we fixed N to 100 and varied the noise level σ from 0.01 to 0.3. We can notice that CoLin is more robust to noise in the feedback: its regret grows much slower than all baselines. Our current regret analysis does not consider the effect of noise in reward, as long as it has a zero mean and finite variance. It would be interesting to incorporate this factor in regret analysis to provide more insight of collaborative bandit algorithms.

Thirdly, in CoLin, we have assumed the adjacency matrix $\mathbf{W}$ is known to the algorithm beforehand. However, in reality one might not precisely recover this matrix from noisy observations, e.g., via social network analysis. It is thus important to investigate the robustness of collaborative bandit algorithms to a noisy $\mathbf{W}$. We fixed the user size N to 100 and corrupted the ground-truth adjacency matrix $\mathbf{W}$: add Gaussian noise $N(0, \delta)$ to w_{ij} and normalize the resulting matrix. We refer to this noisy adjacency matrix as $\mathbf{W}_0$. The simulator still uses the true adjacency matrix $\mathbf{W}$ to compute the reward of each action for a given user; while the noisy matrix $\mathbf{W}_0$ will be provided to the bandit algorithms, i.e., CoLin and GOB.Lin. This newly introduced Gaussian noise is different from the noise in generating the rewards as described in Eq (2).

From the cumulated regret shown in Table 3, we can find that under moderate noise level, CoLin performed much better than GOB.Lin; but CoLin is more sensitive to noise in $\mathbf{W}$ than GOB.Lin. Because CoLin utilizes a weighted adjacency graph to capture the dependency among users, it becomes more sensitive to the estimation error in $\mathbf{W}$. While in GOB.Lin, because only the graph connectivity is used and the random noise is very unlikely to change the graph connectivity, its performance is more stable. Further theoretical analysis of how an inaccurate estimation of $\mathbf{W}$ would affect the resulting regret will be an interesting future work yet to explore.

Finally, the regret analysis of CoLin shows that its upper regret bound is related with the graph structure through the term $\sum_{t=1}^{T}\sum_{j=1}^{N} w_{u_t j}^2$ and GOB.Lin's regret bound is related to the graph connectivity [10]. We designed another set of simulation experiments to verify the effect of graph structure on CoLin and GOB.Lin. In this experiment, we set the user size N to 100 and controlled the graph sparsity as follows: for each user in graph G, we only included the edges from his/her top M most influential neighbors (measured by the edge weight in $\mathbf{W}$) in the adjacency matrix, and normalized the resulting adjacency matrix to a stochastic matrix. No noise is added to $\mathbf{W}$ in this experiment (i.e., $\delta = 0$).

As shown in Table 4, the regret of all bandit algorithms increases as $\mathbf{W}$ becomes sparser, i.e., less information can be shared across



(a) Yahoo dataset (b) Delicious dataset (c) LastFM dataset

Figure 2: Normalized reward on three real-world datasets

users. We can observe that the regret of CoLin increases faster than that in GOB.Lin, since more information becomes unavailable to CoLin. The results empirically verified that CoLin's regret bound is directly related to the graph structure described by the term $\sum_{t=1}^{T}\sum_{j=1}^{N} w_{utj}^2$ and GOB.Lin's regret bound is only related to the graph connectivity.

4.2 Experiments on Yahoo! Today Module

In this experiment, we compared our CoLin algorithm with LinUCB, hybrid LinUCB, GOB.Lin and CLUB on a large collection of ten days' real traffic data from Yahoo! Today Module [21] using the unbiased offline evaluation protocol proposed in [22].

The dataset contains 45,811,883 user visits to the Today Module in a ten-day period in May 2009. For each visit, both the user and each of the 10 candidate articles are associated with a feature vector of six dimensions (including a constant bias feature), constructed by a conjoint analysis with a bilinear model [21]. However, this data set does not contain any user identity. This forbids us to associate the observations with individual users. To address this limitation, we first clustered all users into user groups by applying K-means algorithm on the given user features. Each observation is assigned to its closest user group. The weight in the adjacency matrix $\mathbf{W}$ is estimated by the dot product between the centroids from K-means' output, i.e., $w_{ij} \propto \langle u_i, u_j \rangle$. The CoLin and GOB.Lin algorithms are then executed over those identified user groups. For LinUCB baseline, we tested two variants: one is individual LinUCBs running over the identified user groups and it is denoted as M-LinUCB; another one is a uniform LinUCB shared by all the users, i.e., it does not distinguish individual users, and it is denoted as Uniform-LinUCB.

In this experiment, click-through-rate (CTR) was used to evaluate the performance of all bandit algorithms. Average CTR is computed in every 2000 observations (not the cumulated CTR) for each algorithm as the performance metric based on the unbiased offline evaluation protocol proposed in [22, 21]. Following the same evaluation principle used in [21], we normalized the resulting CTR from different bandit algorithms by the logged random strategy's CTR. We report the normalized CTR from different contextual bandit algorithms over the 160 derived user groups in Figure 2 (a).

CoLin outperformed all baselines on this real-world data set, except CLUB on the first day. Results from other user cluster sizes (40 and 80) showed consistent improvement as demonstrated in Figure 2 (a) with 160 user clusters; but due to space limit, we did not include those results on cluster size of 40 and 80. As we can find CLUB achieved the best CTR on the first day; but as some popular news articles became out-of-date, CLUB cannot correctly recognize their decreased popularity, and thus provided degenerated recommendations. But in CoLin, because of the collaborative preference learning, it better controlled exploration-exploitation tradeoff and thus timely recognized the change of items' popularity. However, one potential limitation of CoLin algorithm is its computational complexity: because the dimension of global statistic matrix $\mathbf{A}_t$ defined in Eq (4) is $dN \times dN$, the running time of CoLin scales quadratically with the number of users. It makes CoLin less attractive in practical applications where the size of users is large. One potential solution is to enforce sparsity in the estimated $\mathbf{W}$ matrix such that distributed model update is possible, i.e., only share information within the connected users. The simulation study in Table 4 confirms the feasibility of this direction and we will explore it in our future work.

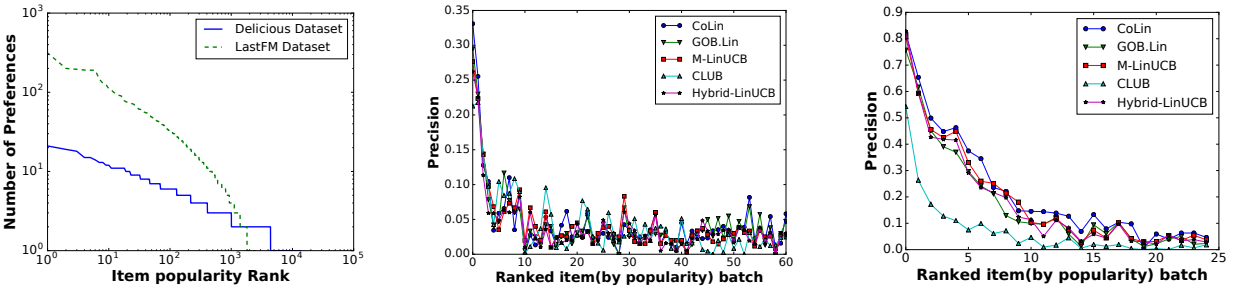
4.3 Experiments on LastFM & Delicious

The LastFM dataset is extracted from the music streaming service Last.fm (http://www.last.fm), and the Delicious data set is extracted the social bookmark sharing service website Delicious (https://delicious.com). These two datasets were generated by the Information Retrieval group at Universidad Autonomade Madrid for the HetRec 2011 workshop with the goal of investigating the usage of heterogeneous information in recommendation system[1]. The LastFM dataset contains 1892 users and 17632 items (artists). We used the information of "listened artists" of each user to create payoffs for bandit algorithms: if a user listened to an artist at least once, the payoff is 1, otherwise 0. The Delicious dataset contains 1861 users and 69226 items (URLs). We generated the payoffs using the information about the bookmarked URLs for each user: the payoff is 1 is the user bookmarked a particular URL, otherwise 0. Both of these two data sets contain the users' social network graph, which makes them a perfect testbed for collaborative bandits.

Following the same settings in [10], we pre-processed these two datasets in order to fit them into the contextual bandit setting. Firstly, we used all tags associated with a single item to create a TF-IDF feature vector, which uniquely represents the item. Then we used PCA to reduce the dimensionality of the feature vectors. In both datasets, we only retained the first 25 principle components to construct the context vectors, i.e., the feature dimension $d = 25$. We then generate the candidate arm pool as follows: we fix the size of candidate arm pool to be $K = 25$; for a particular user u, we pick one item from those nonzero payoff items for user u according to the whole observations in the dataset, and randomly pick the other 24 from those zero-payoff items for user u.

User relation graph is naturally extracted from the social network information provided by the datasets. In order to make the graph denser and make the algorithms computationally feasible, we performed graph-cut to cluster users into M clusters (following the same setting as in [10]). Users in the same clusters are assumed to have the same bandit parameters. In our experiments, M was set to be 50, 100, and 200. Our reported results are from the setting of $M = 200$, and similar results were achieved in other settings of M. After user clustering, a weighted graph can be generated: the nodes are the clusters of the original graph; and the edges between differ-

[1]Datasets and their full description is available at http://grouplens.org/datasets/hetrec-2011



(a) Item popularity of the two datasets (b) Item-based Precision on Delicious (c) Item-based Precision on LastFM

Figure 3: Item-based analysis on Delicious and LastFM datasets

ent clusters are weighted by the number of inter-cluster edges in the original graph. In CoLin, we also need the diagonal elements in $\mathbf{W}$, which is undefined in a graph-cut based clustering algorithm. We computed the diagonal elements based on the derived regret bound of CoLin. Specifically, we first set $w_{ij} \propto c(i,j)$, where $c(i,j)$ is the number of edges between cluster i and j; then we optimize $\{w_{ii}\}_{i=1}^{N}$ which minimizes the term $\sum_i^N \sum_j^N w_{ij}^2$.

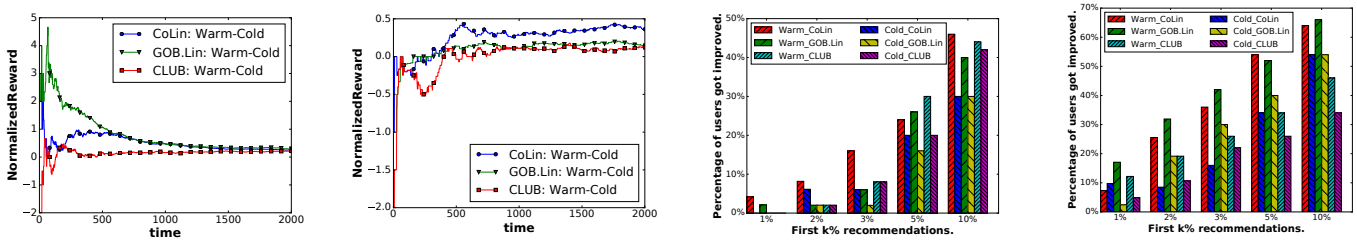
We included three variants of LinUCB, hybrid LinUCB, GOB.Lin and CLUB as our baselines. The three variants of LinUCB include: 1) LinUCB that runs independently on each user, denoted as N-LinUCB; 2) LinUCB that is shared in each user cluster, denoted as M-LinUCB (M is the number of clusters); 3) LinUCB that is shared by all the users, denoted as Uniform-LinUCB. Following the setting it [10], GOB.Lin also operates at the user cluster level and it takes the connectivity among clusters as input. We normalized the cumulated reward in each algorithm by a random strategy's cumulated reward, and compute the average accumulated normalized reward in every 50 iterations. Note that user features required by hybrid LinUCB are not given in these two datasets. We construct the user features by applying PCA on CoLin's adjacency matrix $\mathbf{W}$ and retained the first 25 principle components.

From the results shown in Figure 2 (b) and (c), we can find that CoLin outperforms all the baselines on both Delicious and LastFM datasets. It's worth noting that these two datasets are structurally different, as shown in Figure 3 (a), the popularity of items on these two data sets differs significantly: on LastFM dataset, there are a lot more popular artists whom everybody listens to than the popular websites which everyone bookmarks on Delicious dataset. Thus the highly skewed distribution of item popularity makes recommendation on Delicious dataset much more challenging. Because most of items are only bookmarked by a handful of users, exploiting the relatedness among users to propagate feedback become vital. While on LastFM since there are much more popular items that most users would like, most algorithms can easily recognize the quality of items. In order to better understand this difference, we performed detailed item-level analysis to examine the effectiveness of different algorithms on items with varied popularity. Specifically, we first ranked all the items in these two datasets in a descending order of item popularity and then examined the item-based recommendation precision from all the bandit algorithms, e.g., percentage of item recommendations that are accepted by the users. In order to better visualize the results, we further grouped ranked items into different batches and report the average recommendation precision over each batch in Figure 3 (b) and Figure 3 (c).

From the item-based recommendation precision results, we can clearly find that on the LastFM dataset, CoLin achieved superior performance against all the baselines in every category of items, given the popularity of items in this data set is more balanced. On Delicious dataset, CoLin achieved superior performance on top-ranked items, however, because of the skewness of item popularity distribution, less popular items are still challenging for all the compared algorithms.

This analysis motivates us to further analyze the user-level recommendation performance of different bandit algorithms, especially to understand the effectiveness of collaboration among bandits in alleviating the cold-start problem. To quantitatively evaluate this, we first ranked the user clusters in a descending order with respect to the number of observations in it. We then selected top 50 clusters as group 1. From the bottom 100 user clusters, we select 50 of them who are mostly connected to the users in group 1, as refer to them as group 2. The first group of users is called "learning bucket" and the second group is called "testing bucket". Based on this separation of user clusters, we designed two different experiment settings: one is warm-start, another one is cold-start. In the warm-start setting, we first run all the algorithms on the learning bucket to estimate parameters for both group of users, such as $\mathbf{A}_t$ and $\mathbf{b}_t$ in CoLin. However, because user group 2 does not have any observation in the learning bucket, their model parameters can only be updated via collaboration among bandits, i.e., in CoLin and GOBLin. Then with the model parameters estimated from the learning bucket, we run and evaluate different algorithms on the deployment bucket. Correspondingly, in the cold-start setting, we directly run and evaluate the bandit algorithms on the deployment bucket. It is obvious that since LinUCB assume users are independent and there is no sharing parameters among users, LinUCB's performance will not change under warm-start and cold start setting. While in CoLin, GOB.Lin and CLUB, because of the collaboration among users, model parameters are shared among users. In this case, user preference information learned from the learning bucket can be propagated to the deployment bucket.

We report the performance on the first 10% observations in the deployment bucket instead of the whole observations in order to better simulate the cold-start situation (i.e., all the algorithms do not have sufficient observations to confidently estimate model parameters). In Figure 4 (a) and (b) we report the gap of cumulated rewards from CoLin GOB.Lin, and CLUB between warm-start setting and cold-start setting, normalized by rewards obtained from LinUCB. From Figure 4(a) we can notice that on Delicious dataset, although at the very beginning of the warm-start setting both GOB-Lin and CoLin performed worse than the cold-start setting, both algorithms in warm-start quickly improved and outperformed the cold-start setting. One possible explanation is that the algorithms might take the first several iterations to adapt the model propagated from user group 1 to user group 2. In particular, from Figure 4 (a), it is clear that once both algorithm are adapted, the improvement between warm-start and cold-start on CoLin is larger than that on GOB.Lin. This verified CoLin's effectiveness in address the cold-start challenge. From Figure 4 (b), we can find that warm-start helps both algorithms immediately at the first several iterations on LastFM dataset. This might be caused by the flat distribution of item popularity in this data set: users in group 2 also prefer the



(a) Delicious (b) LastFM (c) Delicious (d) LastFM

Figure 4: Effectiveness of collaboration and User-based analysis

items liked by user group 1. We should note that the larger gap from GOB.Lin than that from CoLin between warm-start and cold-start settings does not mean CoLin is worse than GOB.Lin; but it indicates the cold-start CoLin learned faster than the cold-start GOB.Lin on this data set. And the final performance of both cold-start and warm-start CoLin was better than GOB.Lin in the corresponding settings. We can also notice that cold-start CLUB performed very similarly as warm-start CLUB. It means the user clusters automatically constructed in CLUB does not help collaborative learning. This experiment confirms that appropriate observation sharing among users indeed helps address the cold-start problem in recommendation.

Furthermore, we performed a user-based analysis to examine how many users will benefit from the collaborative bandits setting. We define an improved user as the user who is served with improved recommendations from a collaborative bandit algorithm (i.e. CoLin, GOB.Lin and CLUB) than those from isolated LinUCBs. We report the percentage of improved users in the first 1%, 2%, 3%, 5%, and 10% observations during online learning process. Figure 4 (c) and (d) demonstrate that on all collaborative bandit algorithms, the warm-start setting benefits much more users than in the cold-start setting. This result further supports our motivation of this research that collaboration among bandits can help alleviate the cold-start challenge.

5. CONCLUSIONS

In this paper, we developed CoLin algorithm to study contextual bandit problem in a collaborative environment. In CoLin, context and payoffs are shared among the neighboring bandits during online update, and it helps reduce the preference learning complexity (i.e., requires less observations to achieve satisfactory prediction performance) and leads to reduced overall regret. We religiously proved a reduced upper regret bound comparing to independent bandit algorithms. Experimental results based on extensive simulations and three real-world datasets consistently confirmed the improvement of the proposed collaborative bandit algorithm against several state-of-the-art contextual bandit algorithms.

In our current setting, we have assumed the graph structure among users is known to the algorithm beforehand. It is meaningful to study how to dynamically estimate the graph structure during online update with provable regret bound. The computation complexity in CoLin is now expensive. Utilizing the sparse graph structure to decentralize the computation is an important and interesting research direction. Besides, CoLin should not be only limited to linear payoffs; various types of link functions and regret definitions can be explored as our future work.

6. ACKNOWLEDGMENTS

We thank the anonymous reviewers for their insightful comments. This paper is based upon work supported by the National Science Foundation under grant IIS-1553568.

7. REFERENCES

[1] Y. Abbasi-yadkori, D. Pál, and C. Szepesvári. Improved algorithms for linear stochastic bandits. In *NIPS*, pages 2312–2320. 2011.

[2] K. Amin, M. Kearns, and U. Syed. Graphical models for bandit problems. *Proceedings of UAI 2011*, 2011.

[3] P. Auer. Using confidence bounds for exploitation-exploration trade-offs. *Journal of Machine Learning Research*, 3:397–422, 2002.

[4] P. Auer, N. Cesa-Bianchi, and P. Fischer. Finite-time analysis of the multiarmed bandit problem. *Mach. Learn.*, 47(2-3):235–256, May 2002.

[5] P. Auer, N. Cesa-Bianchi, Y. Freund, and R. E. Schapire. Gambling in a rigged casino: The adversarial multi-armed bandit problem. In *Foundations of Computer Science, 1995. Proceedings., 36th Annual Symposium on*, pages 322–331, 1995.

[6] D. Bouneffouf, A. Bouzeghoub, and A. L. Gançarski. A contextual-bandit algorithm for mobile context-aware recommender system. In *Neural Information Processing*, pages 324–331. 2012.

[7] J. S. Breese, D. Heckerman, and C. Kadie. Empirical analysis of predictive algorithms for collaborative filtering. Technical Report MSR-TR-98-12, Microsoft Research, May 1998.

[8] S. Buccapatnam, A. Eryilmaz, and N. B. Shroff. Multi-armed bandits in the presence of side observations in social networks. In *Decision and Control (CDC), 2013 IEEE 52nd Annual Conference on*, pages 7309–7314. IEEE, 2013.

[9] S. Buccapatnam, A. Eryilmaz, and N. B. Shroff. Stochastic bandits with side observations on networks. *SIGMETRICS Perform. Eval. Rev.*, 42(1):289–300, June 2014.

[10] N. Cesa-Bianchi, C. Gentile, and G. Zappella. A gang of bandits. *In Pro. NIPS*, 2013.

[11] O. Chapelle and L. Li. An empirical evaluation of thompson sampling. In *Advances in Neural Information Processing Systems*, pages 2249–2257, 2011.

[12] W. Chu, L. Li, L. Reyzin, and R. E. Schapire. Contextual bandits with linear payoff functions. In *International Conference on Artificial Intelligence and Statistics*, pages 208–214, 2011.

[13] R. B. Cialdini and M. R. Trost. Social influence: Social norms, conformity and compliance. 1998.

[14] S. Filippi, O. Cappe, A. Garivier, and C. Szepesvári. Parametric bandits: The generalized linear case. In *NIPS*, pages 586–594, 2010.

[15] C. Gentile, S. Li, and G. Zappella. Online clustering of bandits. In *Proceedings of the 31st International Conference on Machine Learning (ICML-14)*, pages 757–765, 2014.

[16] J. C. Gittins. Bandit processes and dynamic allocation indices. *Journal of the Royal Statistical Society. Series B (Methodological)*, pages 148–177, 1979.

[17] S. Kar, H. Poor, and S. Cui. Bandit problems in networks: Asymptotically efficient distributed allocation rules. In *Decision and Control and European Control Conference (CDC-ECC), 2011 50th IEEE Conference on*, pages 1771–1778, Dec 2011.

[18] J. Kawale, H. H. Bui, B. Kveton, L. Tran-Thanh, and S. Chawla. Efficient thompson sampling for online matrix-factorization recommendation. In *Advances in Neural Information Processing Systems*, pages 1297–1305, 2015.

[19] T. L. Lai and H. Robbins. Asymptotically efficient adaptive allocation rules. *Advances in applied mathematics*, 6(1):4–22, 1985.

[20] J. Langford and T. Zhang. The epoch-greedy algorithm for multi-armed bandits with side information. In *NIPS*, pages 817–824. 2008.

[21] L. Li, W. Chu, J. Langford, and R. E. Schapire. A contextual-bandit approach to personalized news article recommendation. In *Proceedings of 19th WWW*, pages 661–670. ACM, 2010.
[22] L. Li, W. Chu, J. Langford, and X. Wang. Unbiased offline evaluation of contextual-bandit-based news article recommendation algorithms. In *Proceedings of 4th WSDM*, pages 297–306. ACM, 2011.
[23] W. Li, X. Wang, R. Zhang, Y. Cui, J. Mao, and R. Jin. Exploitation and exploration in a performance based contextual advertising system. In *Proceedings of 16th SIGKDD*, pages 27–36. ACM, 2010.
[24] F. Radlinski, R. Kleinberg, and T. Joachims. Learning diverse rankings with multi-armed bandits. In *Proceedings of 25th ICML*, pages 784–791. ACM, 2008.
[25] B. Sarwar, G. Karypis, J. Konstan, and J. Riedl. Item-based collaborative filtering recommendation algorithms. In *Proceedings of 10th WWW*, pages 285–295. ACM, 2001.
[26] A. I. Schein, A. Popescul, L. H. Ungar, and D. M. Pennock. Methods and metrics for cold-start recommendations. In *Proceedings of 25th SIGIR*, pages 253–260. ACM, 2002.
[27] A. Slivkins. Contextual bandits with similarity information. *The Journal of Machine Learning Research*, 15(1):2533–2568, 2014.
[28] Y. Yue and T. Joachims. Interactively optimizing information retrieval systems as a dueling bandits problem. In *Proceedings of 26th ICML*, pages 1201–1208. ACM, 2009.
[29] X. Zhao, W. Zhang, and J. Wang. Interactive collaborative filtering. In *Proceedings of the 22nd CIKM*, pages 1411–1420. ACM, 2013.

APPENDIX

Proof of Lemma 1:

Consider the objective function of ridge regression defined in Eq (3). By taking the gradient of $L(\mathbf{\Theta})$ with respect to $\mathbf{\Theta}$ and applying our model assumption specified in Eq (2), we have,

$$\mathbf{A}_t(\hat{\boldsymbol{\vartheta}}_t - \boldsymbol{\vartheta}^*) = \sum_{t'=1}^{t} vec(\mathring{\mathbf{X}}_{a_{t'},u'_t}\mathbf{W}^\mathsf{T})\epsilon_{t'} - \lambda\boldsymbol{\vartheta}^*$$

in which $\epsilon_{t'}$ is the Gaussian noise at time t' in reward generation.

Define $\mathbf{S}_t = \sum_{t'=1}^{t} vec(\mathring{\mathbf{X}}_{a_{t'},u'_t}\mathbf{W}^\mathsf{T})\epsilon_{t'}$, we have,

$$\hat{\boldsymbol{\vartheta}}_t - \boldsymbol{\vartheta}^* = \mathbf{A}_t^{-1}(\mathbf{S}_t - \lambda\boldsymbol{\vartheta}^*)$$

Because $\mathbf{S}_t$ is a martingale, according to Theorem 1 and 2 in [1],

$$\|\hat{\boldsymbol{\vartheta}}_t - \boldsymbol{\vartheta}^*\|_{\mathbf{A}_t} \leq \sqrt{2\ln(\frac{det(\mathbf{A}_t)^{1/2}det(\lambda\mathbf{I})^{-1}}{\delta})} + \sqrt{\lambda}\|\boldsymbol{\vartheta}^*\| \quad (9)$$

Since $\|\mathbf{x}_{a_{t,i}}\| \leq 1$, $trace(\mathbf{A}_t) \leq \lambda dN + \sum_{t'=1}^{t}\sum_{j=1}^{N} w^2_{u'_t j}$, we have $det(\mathbf{A}_t) \leq (\frac{trace(\mathbf{A}_t)}{dN})^{dN} \leq (\lambda + \frac{\sum_{t'=1}^{t}\sum_{j=1}^{N} w^2_{u'_t j}}{dN})^{dN}$. Similarly, we have $det(\lambda\mathbf{I}_{dN}) \leq \lambda^{dN}$. Putting all these into Eq (9), we have,

$$\|\hat{\boldsymbol{\vartheta}}_t - \boldsymbol{\vartheta}^*\|_{\mathbf{A}_t} \leq \sqrt{dN\ln(1 + \frac{\sum_{t'=1}^{t}\sum_{j=1}^{N} w^2_{u'_t j}}{\lambda dN}) - 2\ln(\delta)} + \sqrt{\lambda}\|\boldsymbol{\vartheta}^*\|$$

□

Proof of Theorem 1:

According to the definition of regret in Eq (1), the regret of CoLin at time t can be written as,

$$\begin{aligned}
R_t &= r_{a^*_t,u_t} - r_{a_t,u_t} \\
&= vec(\mathring{\mathbf{X}}^*_{u_t}\mathbf{W}^\mathsf{T})^\mathsf{T}\boldsymbol{\vartheta}^* - vec(\mathring{\mathbf{X}}_{u_t}\mathbf{W}^\mathsf{T})^\mathsf{T}\boldsymbol{\vartheta}^* \\
&\leq vec(\mathring{\mathbf{X}}_{u_t}\mathbf{W}^\mathsf{T})^\mathsf{T}\hat{\boldsymbol{\vartheta}}_{t-1} + \alpha_t\|vec(\mathring{\mathbf{X}}_{u_t}\mathbf{W}^\mathsf{T})\|_{\mathbf{A}^{-1}_{t-1}} - vec(\mathring{\mathbf{X}}_{u_t}\mathbf{W}^\mathsf{T})^\mathsf{T}\boldsymbol{\vartheta}^* \\
&\leq \|vec(\mathring{\mathbf{X}}_{u_t}\mathbf{W}^\mathsf{T})\|_{\mathbf{A}^{-1}_{t-1}}\|\hat{\boldsymbol{\vartheta}}_{t-1} - \boldsymbol{\vartheta}^*\|_{\mathbf{A}_{t-1}} + \alpha_t\|vec(\mathring{\mathbf{X}}_{u_t}\mathbf{W}^\mathsf{T})\|_{\mathbf{A}^{-1}_{t-1}} \\
&\leq 2\alpha_t\|vec(\mathring{\mathbf{X}}_{u_t}\mathbf{W}^\mathsf{T})\|_{\mathbf{A}^{-1}_{t-1}}
\end{aligned}$$

where the first inequality is based on the following two inequalities,

(1) Based on CoLin's arm selection strategy, if arm $\mathbf{X}_t$ is chosen at time t, it must satisfy,

$$\begin{aligned}
&vec(\mathring{\mathbf{X}}_{u_t}\mathbf{W}^\mathsf{T})^\mathsf{T}\hat{\boldsymbol{\vartheta}}_{t-1} + \alpha_t\|vec(\mathring{\mathbf{X}}_{u_t}\mathbf{W}^\mathsf{T})\|_{\mathbf{A}^{-1}_{t-1}} \\
&\geq vec(\mathring{\mathbf{X}}^*_{u_t}\mathbf{W}^\mathsf{T})^\mathsf{T}\hat{\boldsymbol{\vartheta}}_{t-1} + \alpha_t\|vec(\mathring{\mathbf{X}}^*_{u_t}\mathbf{W}^\mathsf{T})\|_{\mathbf{A}^{-1}_{t-1}}
\end{aligned}$$

(2) Based on Cauchy-Schwarz inequality, we have,

$$\begin{aligned}
&vec(\mathring{\mathbf{X}}^*_{u_t}\mathbf{W}^\mathsf{T})^\mathsf{T}\hat{\boldsymbol{\vartheta}}_{t-1} + \alpha_t\|vec(\mathring{\mathbf{X}}^*_{u_t}\mathbf{W}^\mathsf{T})\|_{\mathbf{A}^{-1}_{t-1}} - vec(\mathring{\mathbf{X}}^*_{u_t}\mathbf{W}^\mathsf{T})^\mathsf{T}\boldsymbol{\vartheta}^* \\
&\geq -\|vec(\mathring{\mathbf{X}}^*_{u_t}\mathbf{W}^\mathsf{T})^\mathsf{T}\|_{\mathbf{A}^{-1}_{t-1}}\|\hat{\boldsymbol{\vartheta}}_{t-1} - \boldsymbol{\vartheta}^*\|_{\mathbf{A}_{t-1}} + \alpha_t\|vec(\mathring{\mathbf{X}}^*_{u_t}\mathbf{W}^\mathsf{T})\|_{\mathbf{A}^{-1}_{t-1}} \\
&\geq -\alpha_{t-1}\|vec(\mathring{\mathbf{X}}^*_{u_t}\mathbf{W}^\mathsf{T})\|_{\mathbf{A}^{-1}_{t-1}} + \alpha_t\|vec(\mathring{\mathbf{X}}^*_{u_t}\mathbf{W}^\mathsf{T})\|_{\mathbf{A}^{-1}_{t-1}} \\
&\geq 0
\end{aligned}$$

and therefore, we have

$$vec(\mathring{\mathbf{X}}^*_{u_t}\mathbf{W}^\mathsf{T})^\mathsf{T}\hat{\boldsymbol{\vartheta}}_{t-1} + \alpha_t\|vec(\mathring{\mathbf{X}}^*_{u_t}\mathbf{W}^\mathsf{T})\|_{\mathbf{A}^{-1}_{t-1}} \geq vec(\mathring{\mathbf{X}}^*_{u_t}\mathbf{W}^\mathsf{T})^\mathsf{T}\boldsymbol{\vartheta}^*$$

And according to Lemma 11 in [1], we have,

$$\ln(\frac{det(\mathbf{A}_T)}{det(\lambda\mathbf{I})}) \leq \sum_{t=1}^{T}\|vec(\mathring{\mathbf{X}}_{u_t}\mathbf{W}^\mathsf{T})\|^2_{\mathbf{A}^{-1}_{t-1}} \leq 2\ln(\frac{det(\mathbf{A}_T)}{det(\lambda\mathbf{I})})$$

Thus the accumulated regret at time T in CoLin can be bounded by,

$$\begin{aligned}
\mathbf{R}(T) &\leq \sqrt{T\sum_{t=1}^{T} R_t^2} \leq \sqrt{T4\alpha_T^2\sum_{t=1}^{T}\|vec(\mathring{\mathbf{X}}_{u_t}\mathbf{W}^\mathsf{T})\|^2_{\mathbf{A}^{-1}_{t-1}}} \\
&= \sqrt{T8\alpha_T^2\ln\Big(\frac{det(\mathbf{A}_T)}{det(\lambda\mathbf{I})}\Big)} \\
&\leq 2\alpha_T\sqrt{2dNT\ln\Big(\frac{\sum_{t=1}^{T}\sum_{j=1}^{N} w^2_{u_t j}}{\lambda dN} + 1\Big)}
\end{aligned}$$

□

Analysis for the special case of evenly served users :

If we assume users are evenly served, say each user is served $\bar{T} = T/N$ times. With this assumption, the regret of CoLin can be further written as,

$$\begin{aligned}
\mathbf{R}(T) &\leq \sum_{i=1}^{N}\sqrt{\bar{T}\sum_{t=\bar{T}(i-1)+1}^{t'=\bar{T}i} R_t^2} \\
&\leq \sum_{i=1}^{N}\sqrt{\bar{T}\Big(\ln^2(\frac{det(\mathbf{A}_{t'})}{det(\mathbf{A}_t)}) + \lambda\|\boldsymbol{\vartheta}^*\|\ln(\frac{det(\mathbf{A}_{t'})}{det(\mathbf{A}_t)})\Big)}
\end{aligned} \quad (10)$$

(1) In the extreme case that W is identity matrix, the $\mathbf{A}$ matrix in CoLin becomes a block diagonal matrix, which consists of $\mathbf{A}^{u_i}$, $i \in 1,..,N$. Using the block diagonal matrix's property that $det(\mathbf{A}) = \prod_i^N det(\mathbf{A}^{u_i})$, the regret can be further simplified as,

$$\begin{aligned}
\mathbf{R}(T) &\leq \sum_{i=1}^{N}\sqrt{\bar{T}\Big(\ln^2\big(det(\mathbf{A}^{u_i}) + \lambda\|\boldsymbol{\vartheta}^*\|\ln(det(\mathbf{A}^{u_i}))\big)\Big)} \\
&\leq N\sqrt{\bar{T}\Big(d^2\ln^2(1 + \frac{\bar{T}}{d}) + \lambda\|\boldsymbol{\vartheta}^*\|d\ln(1 + \frac{\bar{T}}{d})\Big)}
\end{aligned} \quad (11)$$

which is $O\big(N\ln(\bar{T})\sqrt{\bar{T}}\big)$ and is identical with N independent LinUCB.

(2) In the extreme case where W is uniform, we have,

$$\mathbf{A}'_t = \mathbf{A}_t + \bar{T}vec(\mathring{\mathbf{X}}\mathbf{W}^\mathsf{T})vec(\mathring{\mathbf{X}}\mathbf{W}^\mathsf{T})^\mathsf{T}$$

such that,

$$det(\mathbf{A}'_t) = det(\mathbf{A}_t)(1 + \bar{T}\|vec(\mathring{\mathbf{X}}\mathbf{W}^\mathsf{T})\|^2_{\mathbf{A}^{-1}_t}) \quad (12)$$

By applying Eq(12) and according to Eq (10) the regret bound of CoLin can be written as,

$$\mathbf{R}(T) \leq N\sqrt{\bar{T}\Big(\ln^2(1 + \frac{\bar{T}}{\lambda N}) + \lambda\|\boldsymbol{\vartheta}^*\|\ln(1 + \frac{\bar{T}}{\lambda N})\Big)}$$

which is $O\big(N\ln(\frac{\bar{T}}{N})\sqrt{\bar{T}}\big)$.