Interactive Online Learning with Incomplete Knowledge

A Dissertation

Presented to

the faculty of the School of Engineering and

Applied Science

University of Virginia

in partial fulfillment
of the requirements for the degree

Doctor of Philosophy

by

Qingyun Wu

August 2020

APPROVAL SHEET

This Dissertation
is submitted in partial fulfillment of the requirements
for the degree of
Doctor of Philosophy

Author Signature: ____________________

This Dissertation has been read and approved by the examining committee:

Advisor:	Hongning Wang
Committee Member:	David Evans
Committee Member:	Quanquan Gu
Committee Member:	Denis Nekipelov
Committee Member:	Lihong Li
Committee Member:	

Accepted for the School of Engineering and Applied Science:

[illegible]

Craig H. Benson, School of Engineering and Applied Science

August 2020

Abstract

The past decades have witnessed a prominent trend of adopting intelligent systems, such as recommendation systems and smart homes, into ordinary people's daily life. One key characteristic of such systems is the need for online sequential decision making: decisions have to be made when the learning agent only has incomplete knowledge about the world/environment. The consequences of such decisions will, in turn, contribute to the data the agent can collect, forming an interactive feedback loop between the agent and the world/environment. This makes conventional offline training based machine learning methods incompetent and urges us to move from the passive learning paradigm to a more interactive and proactive one. It motivates the research of developing interactive online learning solutions, such as contextual bandits, and more generally reinforcement learning, for real-world intelligent systems.

Interactive online learning studies how an agent can interact with an environment to learn a policy that maximizes expected cumulative rewards for a task. In a real-world intelligent system, the learning agent faces environments that consist of human users. This brings at least two significant challenges in developing interactive online learning solutions. First, to capture user heterogeneity, personalized learning solutions are needed. However, the sparsity of each individual user's observation, especially for new users, makes the learning process very slow. Second, many real-world systems are highly dynamic, which is reflected in the fact that users' preferences change over time due to various internal or external factors [1], and item popularity varies due to fast-emerging events and contents. Failing to model such dynamics may lead to sub-optimal decisions.

These fundamental challenges motivate research in this dissertation. We conquer the first challenge by leveraging the existence of dependency among the users. Specifically, we develop a series of collaborative contextual bandit learning solutions in which information can be propagated through explicit or implicit dependency among users. This information propagation helps conquer the data sparsity issue and accelerate the personalized learning process. Rigorous theoretical guarantees are developed, which reveals the benefit of collaboration in the learning process when user dependencies do exist. We conquer the second challenge by moving beyond a commonly used but restrictive stationary environment assumption to a more a realistic non-stationary one. We develop a suite of novel and theoretically sound contextual bandit solutions that automatically detects the potential changes in the environment and adapts its decision making strategy accordingly. Solutions developed this dissertation have been applied to a broad spectrum of recommendation systems, showing their great practical potential.

Acknowledgements

The most fortunate thing for me during my Ph.D. journey is to have Prof. Hongning Wang as my advisor. I want to express my sincere thanks to Prof. Hongning Wang for countless reasons, especially his constant patience, encouragement, and support during my entire Ph.D. journey. Prof. Wang has been a role model for me for his enthusiasm for research, visionary mind, insightful and critical thinking, rigorous attitude to research, integrity, and diligence. In addition to research, I also learned the importance of having a peaceful mind from Prof. Wang, which benefited me a lot in many perspectives.

I would like to thank my dissertation committee members Prof. David Evans, Prof. Quanquan Gu, Prof. Denis Nekipelov, and Dr. Lihong Li for their invaluable help and suggestions on my Ph.D. research and dissertation work. Prof. David Evan has been giving me insightful comments on my research. He kept reminding me to think not only about the technical contributions but also about the broader impact on society. Prof. Quanquan Gu has been acting as my research mentor since my graduate study. He is one of the most influential persons who shaped my initial interest and taste in machine learning research. It is my fortune to be able to work with him and learn from him at the early stage of my Ph.D. study. I want to thank Prof. Denis Nekipelov for providing interdisciplinary suggestions for my dissertation research. Dr. Lihong Li has been a role model in my research. His rigorous attribute to research, insightful mind, and passion in research have been influencing me. I can always learn a lot whenever talking to him. I also want to thank Dr. Li for his suggestions on my dissertation research and very detailed and constructive comments on my dissertation draft.

I want to thank my mentors Dr. Liangjie Hong, Dr. Zheng Wen, Dr. Yasin Abbasi-Yadkori, and Dr. Chi Wang during my internship at Yahoo research, Adobe Research, and Microsoft research. It is my tremendous fortune to be able to closely collaborate with those outstanding scholars. Working with them has greatly broadened my mind. I want to sincerely thank my best collaborator during my Ph.D. and best friend Huazheng Wang, who played another important role in my Ph.D. journey. Among the many things I learned from Huazheng, the important twos are to be open-minded and confident (and I am still learning). I also want to thank everyone in our lab. It is my fortune to have you as lab-mates and friends. I really enjoyed having discussions with you no matter they are about research or daily life. Your enthusiasm for research, great sense of humor, and kindness have inspired me a lot and also made my Ph.D. journey full of laughter and support.

Last but not the least, I want to thank my family for their unconditional love and support.

Contents

List of Tables

List of Figures

Chapter 1

Introduction and Overview

There is no doubt that intelligent systems will become an essential part of most people's daily lives in the near future, if not now. For example, because of the information overload caused by the explosive growth of digital information, we rely more and more on recommendation systems to filter, prioritize, and efficiently deliver relevant information to us; intelligent healthcare systems are transforming many aspects of the traditional medical systems [4], meeting the individual needs of people while improving the efficiency of medical care; intelligent tutoring systems [5], which have consistently been shown to improve the educational outcomes of students when used alone or combined with traditional instruction, are benefiting not only classrooms in developed countries but also are bringing better education to the developing world [6].

Satisfying users with various personalized needs is one of the core missions of many intelligent systems. Machine learning methods are increasingly being used to help with the decision-making process involved in this mission of intelligent systems. For example, methods such as collaborative filtering [7, 8], content-based filtering, and hybrid approaches [9] have long been used in recommender systems to decide what item/content to recommend to satisfy users' personalized information needs. Currently, most of the machine learning methods used in intelligent systems work by first collecting data and then training a fixed model for future predictions. They can provide meaningful predictions by leveraging information demonstrated in the observed data. However, these methods suffer if the information demonstrated in the observed data is incomplete or insufficient to make accurate predictions for the future. For example, in recommendation systems, because of the rapid appearance of new information and new users and the ever-changing nature of content popularity, predictions and decisions made based on the off-line trained models may have become out-of-date by the time the trained models are deployed. In such dynamic environments, an ideal method should be adaptive and make the system self-improving. To this end, feedback from the environment needs to be quickly adopted to improve the predictions and decisions made by the system.

Motivated by this need, modern intelligent systems are now adopting interactive online learning solutions to make predictions or decisions adaptively. Unlike the traditional offline learning methods, in interactive online learning, decisions need to be made on the fly while the system (i.e., the learning agent) only has incomplete knowledge about the users (i.e., the environment). It is also worth noticing that most of the intelligent systems are designed for sophisticated applications, many of which involve interactions with humans, such as the aforementioned recommender systems, health care systems, and tutoring systems. One main challenge in these types of applications is that the learning agent often has limited feedback from the environment. Here limit feedback, on the one hand, means that the learning agent can only get the feedback for the decisions that have been made, and on the other hand, it means that the learning agent cannot ask too many questions as interactions with human users are costly. It is thus critical to learn effectively with such limited feedback. Specifically, during the online decision making, the learning agent, on the one hand, needs to focus on information learned from existing observations such that the decisions are not too bad; simultaneously, it also needs to efficiently explore new information for improving the overall quality of the model, such that better decisions can be made in the future. These two needs create the explore-exploit dilemma in interactive online learning.

1.1 The Contextual Bandit Formulation

Multi-armed bandit algorithms [10–13] provide a principled solution for handling the explore-exploit dilemma. Intuitively, multi-armed bandit algorithms consider different decisions as arms and their main design principle is to designate a small amount of traffic to collect feedback from the environment while improving their estimation qualities on different arms in real time. When side information or context information is available and used in the decision making process, multi-armed bandits become contextual bandits, which have become a reference solution [2, 14–16] in many application scenarios. Such algorithms are especially advantageous in real-world intelligent systems where the space of arms or decisions is large, and the arms are interrelated. They have been applied in in many important applications, e.g., content recommendation [2, 17] and display advertising [18, 19].

In a multi-armed bandit problem, a learner takes turns to interact with the environment with a goal of maximizing its accumulated reward collected from the environment over time T. At round t, the learner makes a choice a_t among a finite, but possibly large, number of arms, i.e., $a_t \in \mathcal{A} = \{a_1, a_2, \dots, a_K\}$, and gets the corresponding reward r_{a_t}. In the contextual bandit setting, each arm a is associated with a feature vector $\mathbf{x}_a \in \mathbb{R}^d$ ($\|\mathbf{x}_a\|_2 \leq 1$ without loss of generality) summarizing the side-information about it at a particular time point. The reward of each arm is usually assumed to be governed by a conjecture of unknown bandit parameter $\boldsymbol{\theta} \in \mathbb{R}^d$ ($\|\boldsymbol{\theta}\|_2 \leq 1$ without loss of generality), which characterizes the environment. This can be specified by a reward mapping function, say $f_{\boldsymbol{\theta}}$: $r_{a_t} = f_{\boldsymbol{\theta}}(\mathbf{x}_{a_t})$. The learner's goal of maximizing the accumulated reward can also be equivalently considered as minimizing the accumulated *regret* with respect to the oracle arm selection strategy. In particular, the accumulated T-trial regret is defined formally as,

$$\mathbf{R}(T) = \sum_{t=1}^{T} R_t = \sum_{t=1}^{T} (\mathbb{E}[r_{a_t^*}] - \mathbb{E}[r_{a_t}]) \tag{1.1}$$

where $a_t^* = \arg\max_a \mathbb{E}[r_{a,t}]$ is the best arm to display at trial t according to the oracle strategy, $r_{a_t^*}$ is the corresponding optimal reward, and $R_t := \mathbb{E}[r_{a_t^*}] - \mathbb{E}[r_{a_t}]$ is the regret at trial t.

Many of the services in intelligent systems can be abstracted as the sequential decision-making process in such a contextual bandit formulation, where the system needs to make a choice among a set of candidate options for different users with the purpose of maximizing its utility. For example, a news recommendation system selects the news articles to present to the users to maximize user clicks (or other types of user engagement), and an intelligent tutoring system recommends the proper exercises to the students to maximize students' learning outcome. In these application scenarios, the intelligent system can be considered as the multi-armed bandit learner; the candidate items to be selected can be considered as the arms. And the different users are different environments the learning agent is interacting with. In the rest of this dissertation, we use the recommendation task as the primary application scenario to illustrate our contribution in such a contextual bandit formulation. However, we emphasize that the proposed research can be applied to a wide spectrum of intelligent systems, as long as the sequential decision-making process involved in their services can be formulated in a similar way.

1.2 Challenges and Insights

Despite their capability to balance exploitation and exploration, there are at least two critical challenges that hinder the application of multi-armed bandit solutions to real-world intelligent systems where human users are involved.

1.2.1 Challenge I: Personalization vs. Learning Efficiency and Privacy

Personalization is becoming not just a desirable feature but an essential need for a modern intelligent system to improve user satisfaction by tailoring the content presentation to suit individual users' needs [20]. With such an emerging need, personalized contextual bandit solutions are being developed [2]. A common practice to realize personalization in contextual bandit algorithms is to estimate the unknown bandit parameters pertaining to each user independently. However, the sparsity of each individual user's observations and the existence of new users and new content makes the personalized online learning process very slow. Meanwhile, it is obvious that users in real-world systems are seldom isolated and thus dependency information can be leveraged to improve the learning efficiency. 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 [21]). Failing to recognize such information among users will inevitably lead to a suboptimal solution.

In addition to the learning efficiency concern, overly personalized recommendations could be a potential source of privacy vulnerability, for adversaries to take advantage of, e.g., infer users' sensitive information, especially in the interactive online learning setting. Real-world privacy breaches have been reported in Amazon's recommendation system [22] and Facebook's advertisement system [23], where an adversary learns considerable amount of information about a user solely based on the systems' recommendation sequences. Compared to the offline learned models, online learning methods directly interact with sensitive user data, e.g., user clicks or purchasing history, and timely update the models to adjust their output, which makes privacy an even more serious concern [24–27]. Due to its importance, private online learning has recently attracted increasing attention in the research community, with a goal to prevent the algorithm's sequential output from revealing a user's private information. While there is existing research on differentially private online convex optimization [24, 28] and contextual bandits [27, 29], private contextual bandits learning that considers the existence of user dependency has not been explored yet.

To address these two challenges brought by personalization, we develop a series of collaborative bandit learning solutions to improve learning efficiency and we equip one of our solutions with privacy protection. This line of research is detailed in Chapter 2. We review some related work in Section 2.1 and then introduce our solutions in the rest of Chapter 2.

In Section 2.2, we introduce our developed collaborative bandit learning framework that is able to explicitly model the underlying dependency among users during online learning. By leveraging user dependency, information sharing will be enabled during online updating. Specifically, a user graph will be constructed according to the available users' dependency information. Each node in the graph represents a contextual bandit agent deployed for a single user, and the weight on each edge indicates the influence between a pair of users. Based on this dependency structure, bandit parameters can be estimated over all the users in a collaborative manner: both context and received payoffs from one user are propagated across the whole graph in the process of online updating. The developed collaborative bandit algorithm establishes a bridge to share information among heterogeneous users and thus reduces the sample complexity of preference learning. Rigorously theoretical analysis and extensive empirical evaluation will be provided to prove the effectiveness of the proposed solution. This work capitalizes on the relatedness among the bandit models deployed across individual users to benefit each of those learning tasks when explicit dependency information is available.

In addition to this explicit modeling of user dependency information, inspired by matrix factorization based collaborative filtering [30–32], we propose another solution, which performs online interactive learning by placing a factorization-based bandit algorithm on each user in the system during interactive online learning. Matrix factorization based collaborative filtering provides a way to capture the underlying dependency among users or items. The basic idea of such solutions is to characterize both recommendation items and users by vectors of latent factors inferred from *historical* user-item preference patterns via low-rank matrix completion [33, 34], with an assumption that only a few factors contribute to an individual's taste [30]. In our solution, by sequentially learning a low-rank structure of the incrementally constructed user-item preference matrix, information sharing can be achieved on both the user side and the item side. This solution is detailed in Section 2.3.

As an important part of this dissertation, we further equip one of our collaborative bandit learning algorithms with privacy guarantees, under the notion of *global differential privacy* [35] and *local differential privacy* [36]. Under global differential privacy, a user is assumed to trust (or say he/she has to trust) the system and provide real engagement data to the system, and the system outputs private recommendations; while under local differential privacy, each user provides perturbed statistics to the system and is no longer required to trust the system or the communication between him/her and system. Details of this work are introduced in Section 2.4.

1.2.2 Challenge II: Non-stationary Environments

In many real-world systems, the only constant is the forever changing user intent and preferences. Users' preferences can be influenced by various internal or external factors [1]. For example, when a sports season ends after a championship, seasonal fans might jump over to following a different sport and not have much interest in the off-season. More importantly, such changes are often not observable to the learners. Multi-armed bandit solutions usually impose stationary assumptions about the environment, which makes them incapable to model or recognize the possible changes of the environment and may lead to constant suboptimal choices.

In this dissertation, moving beyond a restrictive stationary environment assumption, we study a more sophisticated but realistic environment setting where the reward mapping function becomes non-stationary over time. More specifically, we focus on the setting where there are abrupt changes in terms of user preferences (e.g., user interest in a recommender system), and those changes are not observable to the learner beforehand. Between consecutive change points, the reward distribution remains stationary yet unknown, i.e., a piecewise stationary environment. A set of algorithmic solutions, together with their theoretical guarantees, are developed to conquer such a non-stationary environment. We detail our contribution to conquering this challenge in Chapter 3. Based on the properties of the piecewise stationary environment, in any stationary period between two consecutive change points, the reward estimation error of a contextual bandit model trained on the observations collected from that period should be bounded with a high probability [37, 38]. Otherwise, the model's consistent wrong predictions can only come from the change of environment. Based on this insight, we propose to evaluate whether the stationary assumption holds by monitoring a bandit model's reward prediction quality over time. When a bandit model's reward prediction quality is bad enough, a change is considerred to have happened, and new bandit model should be created accordingly. Details of this solution is provided in Section 3.3

In addition, in the contextual bandit setting, the changes in reward distributions caused by a non-stationary environment become *context dependent*. Most existing algorithms that attempt to adapt to this dynamic environment introduce a forgetting mechanism to down weight the historical observations [39, 40], or create a bandit model for each newly detected stationary period [41, 42]. These strategies, however, do not leverage past information optimally. This is because after an abrupt change in the environment, some arms' rewards may be relatively unchanged. This can happen if the underlying change is orthogonal to the context. As a result, strategies that discount observations, or abandon the 'old' model must regain confidence in parameters, incurring a higher regret due to redundant exploration. The possible existence of change-invariant arms in such a non-stationary environment suggests us to reuse the bandit models estimated for those earlier periods, such that more accurate reward estimation on such arms can be achieved sooner to obtain reduced regret in this new period. Based on this insight, we develop a dynamic context-dependent bandit ensemble method to capitalize on this unique context-dependent property in the non-stationary environment. This method makes it possible for those change-invariant arms to reuse old bandit models while not hurting whose arms whose expected reward has changed. Details of this work is described in Section 3.4.

Finally, we recognize that the application of our proposed non-stationary bandit solutions is not limited to the non-stationarity detection in the time horizon, and they can be generalized multi-agent/environment cases, where the non-stationarity can be considered as the heterogeneity across different environments. Based on this insight, we develop a more general solution based on online hypothesis tests, which unifies the online change detection in non-stationary environments and online clustering of learners. Through this unified approach with online hypothesis tests, much more flexible methodologies can be developed to efficiently handle environments where both non-stationarity and collaborative structures exist. Details of this work is described in Section 3.6.

Chapter 2

Bandit Learning in Collaborative Environments

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 [1], 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. 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 [21]). Failing to recognize such information among users will inevitably lead to a suboptimal solution. It is thus necessarily to consider the collaboration effect during the decision-making process of interactive online learning.

In this chapter, we study how to efficiently perform contextual bandit learning in collaborative environments, where explicit or implicit dependency may exist among the environments, i.e., the users. Due to the existing of user dependencies, such as social influence [1], the observed reward from a user's feedback might be a compound of his/her own preference and social influence he/she receives. 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 [21]). In Section 2.2 and Section 2.3, we develop two types of collaborative learning frameworks to approach this problem, with one leveraging explicit dependency and one leveraging implicit dependency. As an important complement, we develop a general framework to equip one of the collaborative bandit learning frameworks with privacy guarantees in Section 2.4. All the code associated with the empirical study in this chapter is available at `https://github.com/huazhengwang/BanditLib`.

2.1 Related Work

The idea of modeling dependency among bandits has been explored in prior research [15, 43–46]. Studies in [47, 48] explore contextual bandits with assumptions about metric or probabilistic dependencies on the product space of context and actions. Hybrid-LinUCB [2] 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 [44, 45]. In [43], 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 [45]. 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 [49], and Zhao et al. studied interactive collaborative filtering via probabilistic matrix factorization [50]. GOB.Lin [15] 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.

There are some recent developments that focus on online collaborative filtering with multi-armed bandit algorithms, a reference solution for explore-exploit trade-off [2, 13, 14]. [50] studies interactive collaborative filtering via probabilistic matrix factorization. Both context-free and contextual bandit algorithms are introduced to perform online item selection based on the factorization results. [49] performs online low-rank matrix completion, where the explore/exploit balance is achieved via Thompson sampling. [51] introduces a UCB-like strategy to perform interactive collaborative filtering. The algorithm deterministically selects feedback user-item pairs using an index which depends on the covariance matrices of the posterior distributions of both latent user and item vectors. [52] performs co-clustering on users and items for collaborative filtering, where confidence bound on reward estimation is used to decide the clustering structures. However, because of the ad-hoc combinations of collaborative filtering methods and bandit methods in the aforementioned studies, limited theoretical understanding is available in those solutions. In this work, we provide a rigorous regret bound analysis of the developed factorization-based bandit algorithm, and demonstrate the algorithm's convergence property under different conditions. Moreover, our online factorization solution is general enough to incorporate several recent advances in factorization techniques, such as feature-based latent factor models [53, 54] and modeling mutual dependency among users [55, 56], which further improve the proposed algorithm's convergence rate during interactive online learning with users.

Differential privacy [35] provides a formal notion to quantify the amount of information an adversary could obtain by observing the algorithm's output. The common practice is to add Laplace or Gaussian noise to the output; and the scale of noise depends on privacy budget (often denoted as ϵ) and *sensitivity*, which is the change of an algorithm's output caused by the change of input. Prior work has studied the problem of differential privacy for offline collaborative filtering [57–59]. For example, [57] studied differential privacy for item-based collaborative filtering methods. [59] proposed a differentially private matrix factorization method based on Bayesian posterior sampling. And [60] proposed a private matrix factorization method that guarantees user-level *joint differential privacy* by perturbing the low-rank decomposition. Differential privacy was first extended to an online setting for stream data in [61, 62]. Differentially private online learning methods have been studied for online convex optimization [24, 25, 63] and bandit problems [26, 27, 29, 64]. The key technique of these solutions is the *tree-based mechanism*, which was proposed in [61, 62] for privately releasing *sum* statistics in stream data with finite time horizon T. Based on this tree-based mechanism, (globally) differentially private linear bandit was first studied in [29] with guaranteed privacy in collected user reward feedback. However, it is non-trivial to extend the private linear bandits to collaborative bandits setting, where one user's reward feedback directly contributes to other users' model update. In other words, the change of model's input from one user can be measured by the model's output in (potentially) all users.

2.2 Collaborative Bandit Learning with Explicit Dependency

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 context and received payoffs from one user are propagated across the whole graph in the process of online updating. The proposed collaborative bandit algorithm establishes a bridge to share information among heterogeneous 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.

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 [2, 14]. Specifically, each user u_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 the random variable 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 i has on user j in determining the payoffs of different arms. $w_{ij} = 0$ if and only if user i has no influence on user j. $\mathbf{W}$ is normalized such that $\sum_{i=1}^{N} w_{ij} = 1$ for $j \in \{1, ..., N\}$ (the sum of each column is 1). 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 $\mathbf{X}_t = (\mathbf{x}_{a_{t,1}}, \ldots, \mathbf{x}_{a_{t,N}})$, 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,

$$r_{a_t,u_t} - diag_t(\mathbf{X}_t^{\mathsf{T}}\boldsymbol{\Theta}\mathbf{W}) \sim N(0, \sigma^2) \tag{2.1}$$

where $diag_t(\mathbf{X})$ is the operation returning the t-th element in the diagonal of matrix $\mathbf{X}$.

Eq (2.1) 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., $w_{u_t u_t}\mathbf{x}_{a_t,u_t}^{\mathsf{T}}\boldsymbol{\theta}_{u_t}$), but also by the judgements from the neighbors who have influence on u_t (i.e., $\sum_{j \neq u_t} w_{u_t j}\mathbf{x}_{a_{t,j}}^{\mathsf{T}}\boldsymbol{\theta}_j$). 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 work; and it can be relaxed via a generalized linear model [16] 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 [15]. 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 2.3.2.

2.2.1 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 $\mathcal{X}_{a_t} = \text{Vec}(\mathbf{X}_{a_t}) = (\mathbf{x}_{a_{t,1}}^{\mathsf{T}}, \ldots, \mathbf{x}_{a_{t,N}}^{\mathsf{T}})^{\mathsf{T}}$, which is a concatenation of context feature vectors of the chosen arm a_t at trial t for all the users. And we define $\mathring{\mathcal{X}}_{a_t,u_t} = \text{Vec}(\mathring{\mathbf{X}}_{a_t,u_t})$, in which $\mathring{\mathbf{X}}_{a_t,u_t}$ is a special case of $\mathbf{X}_{a_t}$: only the

Algorithm 1 Collaborative LinUCB

```
1: Inputs: $\alpha \in \mathbb{R}_+$,$\lambda \in [0,1]$, $\mathbf{W} \in \mathbb{R}^{N\times N}$
2: Initialize: $\mathbf{A}_1 \leftarrow \lambda\mathbf{I}$, $\mathbf{b}_1 \leftarrow \mathbf{0}$, $\hat{\boldsymbol{\vartheta}}_1 \leftarrow \mathbf{A}_1^{-1}\mathbf{b}_1, \mathbf{C}_1 \leftarrow (\mathbf{W}^\mathsf{T} \otimes \mathbf{I})\mathbf{A}_1^{-1}(\mathbf{W} \otimes \mathbf{I})$,
3: for $t = 1$ to $T$ do
4:     Receive user $u_t$
5:     Observe context vectors, $\mathbf{x}_{a_t,u_t} \in \mathbb{R}^d$ for $\forall a \in \mathcal{A}$
6:     Take action $a_t = \arg\max_{a\in\mathcal{A}} \mathring{\mathcal{X}}_{a_t,u_t}^\mathsf{T}\text{Vec}(\widehat{\boldsymbol{\Theta}}_t\mathbf{W}) + \alpha\sqrt{\mathring{\mathcal{X}}_{a_t,u_t}^\mathsf{T}\mathbf{C}_t\mathring{\mathcal{X}}_{a_t,u_t}}$
7:     Observe payoff $r_{a_t,u_t}$
8:     $\mathbf{A}_{t+1} \leftarrow \mathbf{A}_t + \text{Vec}(\mathring{\mathbf{X}}_{a_t,u_t}\mathbf{W}^\mathsf{T})\text{Vec}(\mathring{\mathbf{X}}_{a_t,u_t}\mathbf{W}^\mathsf{T})^\mathsf{T}$
9:     $\mathbf{b}_{t+1} \leftarrow \mathbf{b}_t + \text{Vec}(\mathring{\mathbf{X}}_{a_t,u_t}\mathbf{W}^\mathsf{T})r_{a_t,u_t}$
10:    $\mathbf{C}_{t+1} \leftarrow (\mathbf{W}^\mathsf{T} \otimes \mathbf{I})\mathbf{A}_{t+1}^{-1}(\mathbf{W} \otimes \mathbf{I})$
11:    $\hat{\boldsymbol{\vartheta}}_{t+1} \leftarrow \mathbf{A}_{t+1}^{-1}\mathbf{b}_{t+1}$
```

column corresponding to the user u_t at time t is set to $\mathbf{x}_{a_t,u_t}^\mathsf{T}$, 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 $\boldsymbol{\vartheta} = \text{Vec}(\boldsymbol{\Theta}) = (\boldsymbol{\theta}_1^\mathsf{T}, \boldsymbol{\theta}_2^\mathsf{T}, ..., \boldsymbol{\theta}_N^\mathsf{T})^\mathsf{T} \in \mathbb{R}^{dN}$ as the concatenation of bandit parameter vectors over all the users.

With the collaborative assumption about the expected payoffs defined in Eq (2.1), 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,

$$\widehat{\boldsymbol{\Theta}} = \arg\min_{\boldsymbol{\Theta}} \frac{1}{2}\sum_{t=1}^{T}(\mathring{\mathcal{X}}_{a_t,u_t}^\mathsf{T}\text{Vec}(\boldsymbol{\Theta}_t\mathbf{W}) - r_{a_t,u_t})^2 + \frac{\lambda}{2}tr(\boldsymbol{\Theta}^\mathsf{T}\boldsymbol{\Theta}) \tag{2.2}$$

where $\lambda \in [0,1]$ is a trade-off parameter of L2 regularization in ridge regression.

Since the objective function defined in Eq (2.2) 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 $\hat{\boldsymbol{\vartheta}} = \text{Vec}(\widehat{\boldsymbol{\Theta}})$ and $\mathbf{A}_t$ and $\mathbf{b}_t$ are computed as,

$$\mathbf{A}_t = \lambda\mathbf{I} + \sum_{t'=1}^{t}\text{Vec}(\mathring{\mathbf{X}}_{a_{t'},u_{t'}}\mathbf{W}^\mathsf{T})\text{Vec}(\mathring{\mathbf{X}}_{a_{t'},u_{t'}}\mathbf{W}^\mathsf{T})^\mathsf{T} \tag{2.3}$$

$$\mathbf{b}_t = \sum_{t'=1}^{t}\text{Vec}(\mathring{\mathbf{X}}_{a_{t'},u_{t'}}\mathbf{W}^\mathsf{T})r_{a_{t'},u_{t'}} \tag{2.4}$$

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 are propagated through the whole graph via the relational matrix $\mathbf{W}$. To understand this, note that $\text{Vec}(\mathring{\mathbf{X}}_{a_t,u_t}\mathbf{W}^\mathsf{T})$ 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 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 $\text{Vec}(\mathring{\mathbf{X}}_{a_t,u_t}\mathbf{W}^\mathsf{T}) = \mathring{\mathcal{X}}_{a_t,u_t}$). 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 2.3.2.

The estimated bandit parameters $\widehat{\mathbf{\Theta}}$ 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 (2.1) 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 (2.1), the following inequality holds with probability at least $1-\delta$,

$$|r_{a_t^*,u_t} - r_{a_t,u_t}| \leq \alpha_t \sqrt{\text{Vec}(\mathring{\mathbf{X}}_{u_t}\mathbf{W}^\mathsf{T})^\mathsf{T}\mathbf{A}_t^{-1}\text{Vec}(\mathring{\mathbf{X}}_{u_t}\mathbf{W}^\mathsf{T})} \tag{2.5}$$

where α_t is a parameter in our algorithm defined in Lemma 1 of Section 2.3.2 and δ is embedded in the computation of α_t. The proof of this inequality can be found in the Appendix.

The inequality in Eq (2.5) gives us a reasonably tight upper confidence bound (UCB) for the expected payoff of a particular arm over all users in the graph G, from which a UCB-style action-selection strategy can be derived. In particular, at trial t, we choose an arm for user u_t by,

$$a_{t,u_t} = \arg\max_{a\in\mathcal{A}}\Big(\mathring{\mathcal{X}}_{a_t,u_t}^\mathsf{T}\text{Vec}(\widehat{\mathbf{\Theta}}_t\mathbf{W}) + \alpha_t\sqrt{\text{Vec}(\mathring{\mathbf{X}}_{u_t}\mathbf{W}^\mathsf{T})^\mathsf{T}\mathbf{A}_t^{-1}\text{Vec}(\mathring{\mathbf{X}}_{u_t}\mathbf{W}^\mathsf{T})}\Big) \tag{2.6}$$

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 $\text{Vec}(\mathring{\mathbf{X}}_{u_t}\mathbf{W}^\mathsf{T}) = (\mathbf{W}\otimes\mathbf{I})\text{Vec}(\mathring{\mathbf{X}}_{u_t})$ $= (\mathbf{W}\otimes\mathbf{I})\mathring{\mathcal{X}}_t$ to simplify Eq (2.6).

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.

2.2.2 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 $\|\hat{\boldsymbol{\vartheta}}_t - \boldsymbol{\vartheta}^*\|_{\mathbf{A}_t} = \sqrt{(\hat{\boldsymbol{\vartheta}}_t - \boldsymbol{\vartheta}^*)^\mathsf{T}\mathbf{A}_t(\hat{\boldsymbol{\vartheta}}_t - \boldsymbol{\vartheta}^*)}$, 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 regret upper bound of the CoLin algorithm.

Theorem 1. *With probability at least $1-\delta$, the accumulated regret of CoLin algorithm satisfies,*

$$\boldsymbol{R}(T) \leq 2\alpha_T \sqrt{2dNT \ln\Big(1 + \frac{\sum_{t=1}^{T}\sum_{j=1}^{N} w_{u_t j}^2}{\lambda dN}\Big)} \tag{2.7}$$

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 [15], 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,

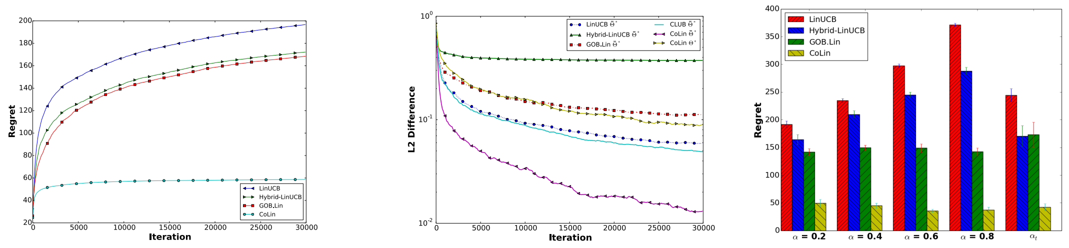
$$0 \leq R_G^2(T) - R_C^2(T) \leq 16TN\ln(1 + \frac{2T}{dN^2}) \sum_{(i,j)\in E} \|\boldsymbol{\theta}_i^* - \boldsymbol{\theta}_j^*\|^2$$

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 experiments on both synthetic and real-world datasets.

2.2.3 Experiments

We performed empirical evaluations of our CoLin algorithm against several state-of-the-art contextual bandit algorithms, including N independent LinUCB [2], hybrid LinUCB with user features [2], GOB.Lin [15], and online cluster of Bandits (CLUB) [45]. 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,



(a) Convergence of accumulated regret (b) Parameter estimation error (c) The effect of α

Figure 2.1: Analysis of regret, bandit parameter estimation and parameter tuning.

including EXP3 [13], UCB1 [12] and ϵ-greedy [12]. 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 dataset via simulations, a large collection of click stream from Yahoo! Today Module dataset [2], and two real-world dataset extracted from the social bookmarking web service Delicious and music streaming service LastFM [15]. 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 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.

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.

Simulation Setting. In simulation, we generate N users, each of which is associated with a d-dimensional parameter vector $\boldsymbol{\theta}^*$, i.e., $\boldsymbol{\Theta}^* = (\boldsymbol{\theta}^*_1, \dots, \boldsymbol{\theta}^*_N)$. 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 $r_{a_{t,i}} = diag_i(\mathbf{X}_t^\mathsf{T}\bar{\Theta}^*) + \epsilon_t$, 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 our CoLin algorithm. In particular, at each trial t, 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 we have the ground-truth bandit parameters available in the simulator, we also compared the quality of learned parameters in each contextual bandit algorithms. This unveils the nature of each bandit algorithm, e.g., how accurately they can recover a user's true preference.

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

Table 2.1: Accumulated 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 2.2: Accumulated 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 2.3: Accumulated 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 2.4: Accumulated 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

in this experiment. We report the accumulated regret as defined in Eq (1.1) and the Euclidean distance between the learnt bandit parameters from different algorithms and the ground-truth in Figure 2.1. 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 2.1 (a) and (b).

As we can find in Figure 2.1 (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 $\mathbf{\Theta}^*$, 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 2.1 (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 2.1 (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 $\mathbf{\Theta}^*$, 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 $\mathbf{\Theta}^*$. 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 $\mathbf{\Theta}^*$ is much worse. The main reason is that CoLin has to decipher $\mathbf{\Theta}$ 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 2.1 (b)), its regret is the 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 (2.1), which still provides validate information for CLUB to estimate $\bar{\Theta}^*$ with reasonable accuracy.

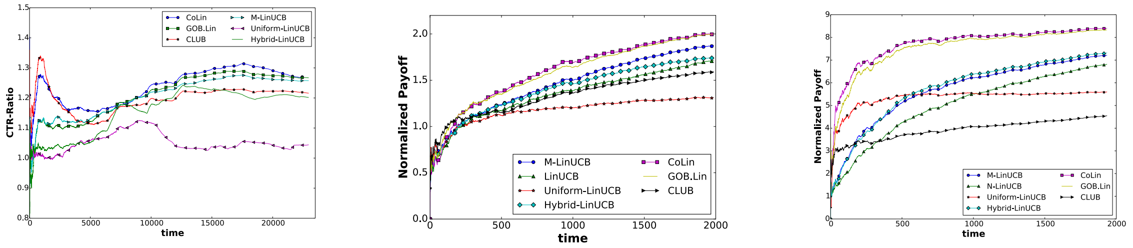
In Figure 2.1 (c), we investigated the effect of exploration parameter α_t's setting in different algorithms. The last column indexed by α_t represented the theoretical values of α computed from the algorithms' corresponding regret analysis. As shown in the results, the empirically tuned α yields comparable performance to the theoretical values, and makes online computation more efficient. As a result, in all our following experiments we will 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 2.1 to 2.4. Because of its poor performance, we did not include CLUB in those tables. Firstly, in Table 2.1, 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 2.3.2, 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 2.2, 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 (2.1).

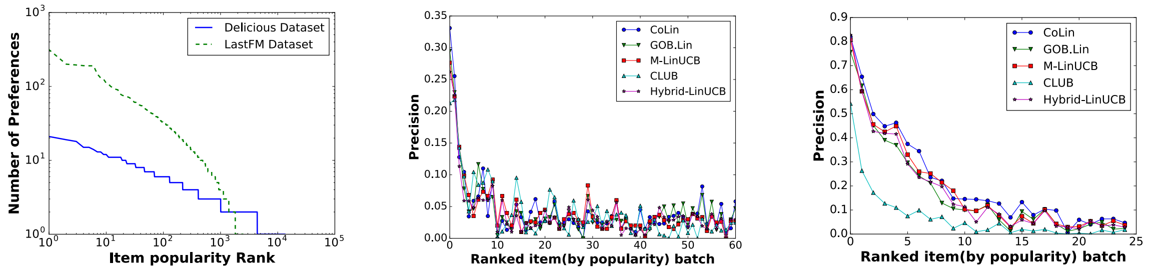
From the accumulated regret shown in Table 2.3, we can find that under moderate noise level, CoLin performed much better than GOB.Lin; but CoLin is more sensitive to noise in $\mathbf{W}$ than GOB.Lin. Because CoLin utilizes a weighted adjacency graph to capture the dependency among users, it becomes 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 to 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 [15]. 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., $\delta = 0$).



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

Figure 2.2: Normalized reward on three real-world datasets



(a) Item popularity (b) Delicious dataset (c) LastFM dataset

Figure 2.3: Item-based analysis on Delicious and LastFM datasets

As shown in Table 2.4, the regret of all bandit algorithms increases as $\mathbf{W}$ becomes sparser, i.e., less information can be shared across 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 $\sum_{t=1}^{T}\sum_{j=1}^{N} w_{u_t j}^2$ and GOB.Lin's regret bound is only related to the graph connectivity.

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 [2] using the unbiased offline evaluation protocol proposed in [3].

The dataset contains 45,811,883 user visits to the Today Module in a ten-day period in May 2009. For each logged event, both the user and each of the 10 candidate articles are associated with a feature vector of six dimensions (including a constant bias feature), which is constructed by a conjoint analysis with a bilinear model [2]. However, this dataset 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 the 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 thus it is denoted as Uniform-LinUCB.

In this experiment, click-through-rate (CTR) was used to evaluate the performance of all bandit algorithms. An algorithm's CTR is defined as the number of clicks its recommendations receive divided by the number of items it recommends, and this is just one way to approximate reward. Average CTR is computed in every 2000 observations (not the accumulated CTR) for each algorithm based on the unbiased offline evaluation protocol proposed in [2, 3]. Following the same evaluation principle used in [2], we normalized the resulting CTR from different bandit algorithms by the corresponding logged random strategy's CTR. We report the normalized CTR results from different contextual bandit algorithms over 160 derived user groups in Figure 2.2 (a).

CoLin outperformed all baselines on this real-world dataset, except CLUB on the first day. Results from other user cluster sizes (40 and 80) showed consistent improvement as demonstrated in Figure 2.2 (a) with 160 user clusters; but due to space limit, we did not include those results. 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 collaborative preference learning, it better controlled the exploration-exploitation trade-off and thus timely recognized the change of items' popularity. However, one potential limitation of CoLin is its computational complexity: because the dimension of global statistic matrix $\mathbf{A}_t$ defined in Eq (2.3) 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 2.4 confirms the feasibility of this direction and we will explore it in our future work.

Experiments on LastFM & Delicious

The LastFM dataset is extracted from the music streaming service Last.fm, and the Delicious dataset is extracted the social bookmark sharing service website Delicious. 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 1,892 users and 17,632 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 1,861 users and 69,226 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 datasets contain the users' social network graph, which makes them a perfect real-world testbed for collaborative bandits.

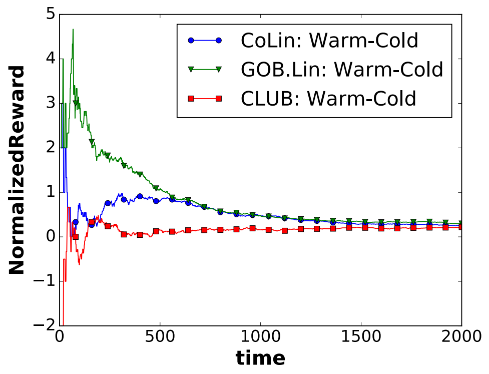
Following the same settings in [15], we pre-processed these two datasets in order to fit them into the contextual bandit setting. We first used all tags associated with a single item to create a TF-IDF feature vector, which uniquely represents the item. Then we used PCA to reduce the dimensionality. In both datasets, we only retained the first 25 principle components to construct the context vectors, i.e., the feature dimension $d = 25$. We generated the candidate arm pool as follows: we fixed the size of candidate arm pool to be $K = 25$; for a particular user u, we picked one item from those nonzero payoff items for user u according to the whole observations in the dataset, and randomly picked the other 24 from those zero-payoff items for user u.

User relation graph is extracted from the social network 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. Users in the same cluster are assumed to share 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 nodes in the original graph; and the edges between different 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 optimized $\{w_{ii}\}_{i=1}^{N}$ which minimizes the term $\sum_i^N \sum_j^N w_{ij}^2$.

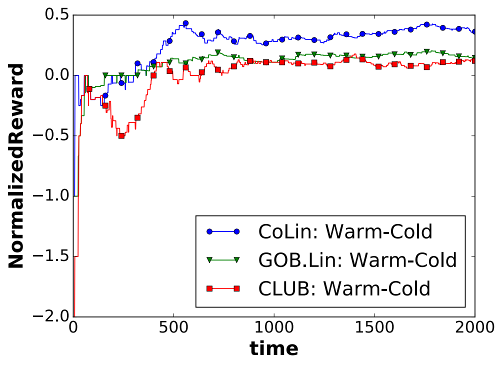
We included three variants of LinUCB, hybrid LinUCB, GOB.Lin and CLUB as baselines. The three variants of LinUCB include: (1) LinUCB that runs independently on each user, denoted as N-LinUCB; (2) LinUCB that is shared in each user cluster, denoted as M-LinUCB (M is the number of clusters); (3) LinUCB that is shared by all the users, denoted as Uniform-LinUCB. Following the setting in [15], GOB.Lin also operates at the user cluster level and it takes the connectivity among clusters as input. We normalized the accumulated reward in each algorithm by a random strategy's accumulated 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 applied the same strategy as we used in simulation to generate the user features.

From the results shown in Figure 2.2 (b) and (c), we can find that CoLin outperforms all the baselines on both Delicious and LastFM datasets. It is worth noting that these two datasets are structurally different, as shown in

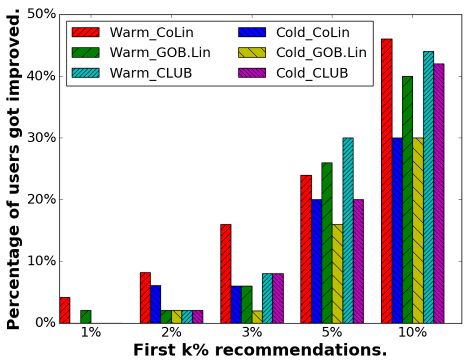
[1]Datasets and their full description is available at http://grouplens.org/datasets/hetrec-2011



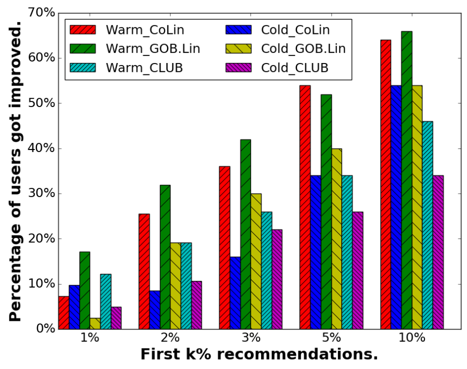
(a) Delicious reward



(b) LastFM reward



(c) Delicious improvement



(d) LastFM improvement

Figure 2.4: Effectiveness of collaboration and User-based analysis

Figure 2.3 (a), the popularity of items on these two datasets 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 grouped the ranked items into different batches and report the average recommendation precision over each batch in Figure 2.3 (b) and Figure 2.3 (c).

From the results about the item-based recommendation precision, we can clearly find that on the LastFM dataset, CoLin achieved improved performance against all the baselines in every category of items, given the popularity of items in this dataset is more balanced. On Delicious dataset, CoLin achieved better performance on the top-ranked items; however, because of the skewness of item popularity, less popular items are still challenging for all the bandit algorithms to correctly recognize on this dataset.

This analysis motivates us to further analyze the user-level recommendation performance of different bandit algorithms, especially to understand the effectiveness of collaboration among users 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, and 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 performed two experiments: one is warm-start, and another 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 users in the second group do not have any observation in the learning bucket, their model parameters can only be updated via the collaboration among bandits, i.e., in CoLin and GOBLin. Then with the model parameters estimated from the learning bucket, we 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 assumes users are independent and there is no information shared among users, LinUCB's performance will not change under warm-start and cold start settings. While in CoLin, GOB.Lin and CLUB, because of the collaboration among users, information is propagated among users. In this case, user preference information learned from the learning bucket can be propagated to the deployment bucket.

We reported 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 2.4 (a) and (b), we reported the gap of accumulated rewards from CoLin GOB.Lin, and CLUB between warm-start and cold-start, normalized by rewards obtained from LinUCB. From Figure 2.4(a) we can notice that on Delicious dataset, although at the very beginning of the warm-start setting both GOBLin 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 models propagated from the first user group to the second. In particular, from Figure 2.4 (a), it is clear that once both algorithms 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 2.4 (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 dataset: users in the second group also prefer the items liked by users in the first group. 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 dataset. 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 is vital to address the cold-start problem in recommendation.

Furthermore, we performed a user-based analysis to examine how many users will benefit from collaborative bandits. 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 reported the percentage of improved users in the first 1%, 2%, 3%, 5%, and 10% observations during online learning. Figure 2.4 (c) and (d) demonstrate that in all collaborative bandit algorithms, the warm-start setting benefits much more users than cold-start setting. This further supports our motivation in developing bandit algorithms in a collaborative environment, which helps alleviate the cold-start challenge.

2.3 Factorization Bandits: Bandit Learning from Implicit Dependency

One way to handle implicit dependency is to use matrix factorization. Matrix factorization based collaborative filtering has become a standard practice in recommender systems [30–32]. The basic idea of such solutions is to characterize both recommendation items and users by vectors of latent factors inferred from *historical* user-item preference patterns via low-rank matrix completion [33, 34], with an assumption that only a few factors contribute to an individual's taste [30]. Despite a few recent advances in specific factorization techniques [53, 54], it is notoriously difficult to perform online interactive recommendation, because the need to focus on items that raise users' interest and, simultaneously, the need to explore new items for improving users' satisfaction in the long run create an explore-exploit dilemma. Periodically repeat model estimation to update latent factors is inept to handle the interactions between a system and its users on the fly, because not only does it overly exploit the learnt model that is biased towards previously frequently recommended items, but it also is prohibitively expensive to afford in terms of computational complexity.

Some preliminary attempts have been made to perform online matrix factorization for collaborative filtering. Basically, multi-armed bandit algorithms [13, 14] are employed to control the exploration of currently less promising recommendations for user feedback, and factorization is applied over the incrementally constructed user-item matrix on the fly. However, these two components are integrated in an *ad-hoc* manner: both contextual and context-free bandits have been explored on top of various factorization methods [49–51], given they only provide an index of candidate items for feedback acquisition. As a result, little is known about whether such combinations would lead to a converging recommendation performance nor would it ensure long-term optimality in theory, i.e., regret bound analysis.

We address the aforementioned challenges by performing online interactive recommendation by placing a factorization-based bandit algorithm on each user in the system. Low-rank matrix completion is performed over an incrementally constructed user-item preference matrix, where an upper confidence bound (UCB) based item selection strategy is developed to balance the exploit/explore trade-off during online feedback acquisition. To better conquer cold-start in recommendation, two special treatments are devised. First, observable contextual features are integrated with the estimated latent factors during matrix factorization. This improves recommendation when the number of candidate items is large, but the payoffs are interrelated, i.e., *context-aware*. Second, the dependence among users (e.g., social influence) is introduced to our bandit algorithm through a collaborative reward generation assumption [65]. It enables information sharing among the neighboring users while online learning, so as to help reduce the overall regret.

More importantly, we rigorously prove that with high probability the developed algorithm achieves a sublinear upper regret bound for interactive recommendation, i.e., the average number of suboptimal recommendations made in our algorithm over time rapidly vanishes with high probability. And considerable regret reduction is achieved on both user and item sides because of our explicit modeling of observable contextual features and dependency among users. Extensive experimentations on both simulations and large-scale real-world datasets confirmed the advantages of the proposed algorithm compared with several state-of-the-art bandit-based factorization methods.

2.3.1 A Factorization Bandit Solution for Interactive Recommendation

Matrix factorization based collaborative filtering solutions map both users $U = \{u_1, u_2, ..., u_N\}$ and recommendation items $\mathcal{A} = \{a_1, a_2, ..., a_M\}$ to a joint latent factor space. The expected reward of an item with respect to a given user is assumed to be an inner product of the latent item factor $\mathbf{v}_a \in \mathbb{R}^l$ and the latent user factor $\boldsymbol{\theta}_u \in \mathbb{R}^l$. Hence, the reward generation process can be formalized as $r_{a,u} = \mathbf{v}_a^\mathsf{T}\boldsymbol{\theta}_u + \eta$, where the random variable η is drawn from a Gaussian distribution $N(0, \sigma^2)$. Regularized quadratic loss over a given set of user-item feedback pairs is usually employed to estimate the latent factors. Formally,

$$\min_{\boldsymbol{\theta}_u, \mathbf{v}_a} \frac{1}{2} \sum_{(a,u)\in\mathcal{K}} (\mathbf{v}_a^\mathsf{T}\boldsymbol{\theta}_u - r_{a,u})^2 + \frac{\lambda_1}{2}\sum_{u\in U}\|\boldsymbol{\theta}_u\|_2 + \frac{\lambda_2}{2}\sum_{a\in\mathcal{A}}\|\mathbf{v}_a\|_2 \tag{2.8}$$

where $\mathcal{K}$ is a set of user-item pairs with known reward (e.g., the offline training set), λ_1 and λ_2 are the trade-off parameters. The key research challenge in interactive matrix factorization is how to select the next feedback user-item pair for model update. Current practice exploits the trained model to collect user feedback, which unfortunately reinforces the bias in a currently inaccurate model. Therefore, properly explore some currently less promising items for model correction becomes necessary for long-term optimality.

Under the context of matrix factorization based collaborative filtering, the uncertainty of reward prediction comes from two sources: 1) the estimation error of latent user factors at trial t, i.e., $\|\hat{\boldsymbol{\theta}}_{u,t} - \boldsymbol{\theta}_u^*\|$, where $\hat{\boldsymbol{\theta}}_{u,t}$ is the current estimate of latent factors for user u, and $\boldsymbol{\theta}_u^*$ is the ground-truth factors; and 2) the estimation error of latent item factors at trial t, i.e., $\|\hat{\mathbf{v}}_{a,t} - \mathbf{v}_a^*\|$. Because of the regularized quadratic loss employed in Eq (2.8), the confidence sets of $\boldsymbol{\theta}_u$ and $\mathbf{v}_a$ estimation can be analytically computed [37], and thus readily be integrated to assemble a UCB-style bandit algorithm for interactive matrix factorization as follows,

$$\big(a_t, \langle\hat{\boldsymbol{\theta}}_{u,t}, \hat{\mathbf{v}}_{a,t}\rangle\big) = \operatorname*{arg\,max}_{\big(a, \langle\boldsymbol{\theta}_u, \mathbf{v}_a\rangle\big)\in\mathcal{D}_t\times\mathcal{C}_{t-1}} \boldsymbol{\theta}_u^\mathsf{T}\mathbf{v}_a \tag{2.9}$$

where $\mathcal{D}_t$ is the set of candidate items for recommendation at trial t, and $\mathcal{C}_{t-1}$ is the confidence set for latent user and item factors $\langle\boldsymbol{\theta}_u, \mathbf{v}_a\rangle$ constructed at last trial.

However, such a straightforward combination of bandit algorithm with matrix factorization cannot effectively solve the cold-start problem, as the estimation uncertainty of the latent factors for new users and new items is at the maximum. This inevitably requires more explorations on the new users and new items and hence leads to a decreased convergence rate of online learning and reduced user satisfaction in practice. We propose to solve these limitations by introducing observed contextual features [53, 66] and user dependence [55, 65] into online factorization. Both of these two techniques have been proved to be effective in offline matrix factorization, but little is known about their utility in an online setting. In particular, we explicitly incorporate these two components into our bandit algorithm's reward generation assumption, to make it a unified framework for interactive matrix factorization.

First, to reduce the reward prediction uncertainty on new items, we introduce observable contextual features into the estimation of latent item factors. Typical item-level contextual features include topic categories for news recommendation [2, 53] and genre for music recommendation [15]. Formally, we denote the observed contextual features for an item a as $\mathbf{x}_a \in \mathbb{R}^d$ and keep using $\mathbf{v}_a \in \mathbb{R}^l$ for its latent part (with $\|(\mathbf{x}_a, \mathbf{v}_a)\|_2 \leq L$). Accordingly, on the user side we redefine $\boldsymbol{\theta}_u = (\boldsymbol{\theta}_u^{\mathbf{x}}, \boldsymbol{\theta}_u^{\mathbf{v}}) \in \mathbb{R}^{d+l}$ (with $\|\boldsymbol{\theta}_u\|_2 \leq S$), in which $\boldsymbol{\theta}_u^{\mathbf{x}} \in \mathbb{R}^d$ corresponds to the context feature $\mathbf{x}_a$ and $\boldsymbol{\theta}_u^{\mathbf{v}} \in \mathbb{R}^l$ corresponds to the latent item factor $\mathbf{v}_a$. These extended user and item factors now determine the rewards in recommendation.

Second, we incorporate mutual influence among users to reduce the reward prediction uncertainty on new users. Distinct from existing solutions, where the dependency among users (such as social network) is introduced as graph-based regularization over the latent user factors [15, 55], we encode such dependency directly into our reward generation assumptions for matrix factorization. We assume the observed reward from each user is determined by a mixture of neighboring users [65]. Formally, instead of assuming N independent users for factorization, we place them on a weighted graph $G = (V, E)$, which encodes the affinity relation among users, to perform the estimation across them simultaneously. Each node V_u in G is parameterized by the latent user factor $\boldsymbol{\theta}_u$ for user u; and each edge in E represents the influence across users in reward generation. We encode this graph as an $N \times N$ stochastic matrix $\mathbf{W}$, in which each element w_{ij} is nonnegative and proportional to the influence that user j has on user i in determining the reward of different items. $\mathbf{W}$ is column-wise normalized such that $\sum_{j=1}^{N} w_{ij} = 1$ for $i \in \{1, ..., N\}$, and we assume $\mathbf{W}$ is time-invariant and known to the algorithm beforehand.

Based on the introduced contextual features and user relational graph G, we define a $(d+l) \times N$ matrix $\boldsymbol{\Theta} = (\boldsymbol{\theta}_1, \ldots, \boldsymbol{\theta}_N)$, which consists of latent user factors from all N users in graph G, and define $\mathbf{X}_{a_t} = (\mathbf{x}_{a_t,1}, ..., \mathbf{x}_{a_t,N})$ and $\mathbf{V}_{a_t} = (\mathbf{v}_{a_t,1}, ..., \mathbf{v}_{a_t,N})$ for the observable contextual features and latent item factors of the items to be presented to the N users respectively. To simplify the notations for discussion, we decompose $\boldsymbol{\Theta}$ into two sub-matrices, $\boldsymbol{\Theta}^{\mathbf{x}} = (\boldsymbol{\theta}_1^{\mathbf{x}}, \ldots, \boldsymbol{\theta}_N^{\mathbf{x}})$ and $\boldsymbol{\Theta}^{\mathbf{v}} = (\boldsymbol{\theta}_1^{\mathbf{v}}, \ldots, \boldsymbol{\theta}_N^{\mathbf{v}})$, corresponding to the observed context features and latent factors for items. As a result, we enhance our reward generation assumption as follows,

$$r_{a_t,u} = (\mathbf{x}_{a_t}, \mathbf{v}_{a_t})^{\mathsf{T}} \boldsymbol{\Theta} \mathbf{w}_u + \eta_t = \mathbf{x}_{a_t}^{\mathsf{T}} \boldsymbol{\Theta}^{\mathbf{x}} \mathbf{w}_u + \mathbf{v}_{a_t}^{\mathsf{T}} \boldsymbol{\Theta}^{\mathbf{v}} \mathbf{w}_u + \eta_t \tag{2.10}$$

Intuitively, in Eq (2.10) not only the observed contextual features, but also the estimated latent factors will be propagated through the user graph to determine the expected reward of items across users.

We will prove such information sharing greatly reduces sample complexity in learning the latent factors for both users and items. Plugging the enhanced reward generation assumption defined in Eq (2.10) into the regularized quadratic loss function in Eq (2.8), we can easily derive the closed-form solutions for $\boldsymbol{\Theta}$ and $\mathbf{v}_a$ *after* trial t via the alternating least square (ALS) method as $\vec{\boldsymbol{\Theta}}_t) = \mathbf{A}_t^{-1}\mathbf{b}_t$ and $\hat{\mathbf{v}}_{a,t} = \mathbf{C}_{a,t}^{-1}\mathbf{d}_{a,t}$, where the detailed computation of $(\mathbf{A}_t, \mathbf{b}_t, \mathbf{C}_{a,t}, \mathbf{d}_{a,t})$ can be found in Algorithm 2. $\mathbf{I}_1$ and $\mathbf{I}_2$ are identity matrices with dimensions of $(d+l)N \times (d+l)N$ and $l \times l$ respectively. We define $\mathring{\mathbf{X}}_{a_t}$ as a special case of $\mathbf{X}_{a_t}$: only the column corresponding to user u is set to $\mathbf{x}_{a_t,u}$ and all the other columns are zero; and the same notation applies to $\mathring{\hat{\mathbf{V}}}_{a_t}$.

Under our enhanced reward generation assumption defined in Eq (2.10), the confidence set of $\langle \boldsymbol{\theta}_u, \mathbf{v}_a \rangle$ estimation can be analytically computed by the following lemma.

Lemma 2. *With proper initialization of ALS, the Hessian matrix of Eq* (2.8) *is positive definite at the optimizer* $\boldsymbol{\Theta}^*$ *and* $\mathbf{v}_a^*$, *such that for any* $\epsilon_1 > 0$, $\epsilon_2 > 0$, *and* $\delta \in (0, 1)$, *with probability at least* $1 - \delta$, *the estimation*

Algorithm 2 FactorUCB

```
1: **Inputs:** $\lambda_1, \lambda_2 \in (0, +\infty), l \in \mathbb{Z}^+$
2: **Initialize:** $\mathbf{A}_1 \leftarrow \lambda_1 \mathbf{I}_1$, $\mathbf{b}_1 \leftarrow \mathbf{0}^{(d+l)N}$, $\vec{(}\boldsymbol{\Theta}_1) \leftarrow \mathbf{A}_1^{-1}\mathbf{b}_1$
3: **for** $t = 1$ to $T$ **do**
4:     Receive user $u_t$
5:     Observe feature vectors, $\mathbf{x}_a \in \mathbb{R}^d$
6:     **if** item $a$ is new **then**
7:         initialize $\mathbf{C}_{a,t} \leftarrow \lambda_2 \mathbf{I}_2$, $\mathbf{d}_{a,t} \leftarrow \mathbf{0}^l$, $\hat{\mathbf{v}}_{a,t} \leftarrow \mathbf{0}^l$
8:     Select item by $a_t = \arg\max_{a \in \mathcal{A}} \Big( (\mathbf{x}_a, \hat{\mathbf{v}}_{a,t})^\mathsf{T} \hat{\boldsymbol{\Theta}}_t \mathbf{w}_{u_t} + \alpha_t^u \sqrt{vec\big((\mathring{\mathbf{X}}_{a_t}, \mathring{\hat{\mathbf{V}}}_{a_t})\mathbf{W}^\mathsf{T}\big) \mathbf{A}_t^{-1} vec\big((\mathring{\mathbf{X}}_{a_t}, \mathring{\hat{\mathbf{V}}}_{a_t})\mathbf{W}^\mathsf{T}\big)^\mathsf{T}} \Big) +$
   $\alpha_t^a \sqrt{(\hat{\boldsymbol{\Theta}}_t \mathbf{w}_{u_t}) \mathbf{C}_{a,t}^{-1} (\hat{\boldsymbol{\Theta}}_t \mathbf{w}_{u_t})^\mathsf{T}}$
9:     Observe reward $r_{a_t,u_t}$ from user $u_t$
10:    $\mathbf{A}_{t+1} \leftarrow \mathbf{A}_t + \vec{(}\mathring{\mathbf{X}}_{a_t}, \mathring{\hat{\mathbf{V}}}_{a_t})\mathbf{W}^\mathsf{T}) \vec{(}\mathring{\mathbf{X}}_{a_t}, \mathring{\hat{\mathbf{V}}}_{a_t})\mathbf{W}^\mathsf{T})^\mathsf{T}$
11:    $\mathbf{b}_{t+1} \leftarrow \mathbf{b}_t + \vec{(}\mathring{\mathbf{X}}_{a_t}, \mathring{\hat{\mathbf{V}}}_{a_t})\mathbf{W}^\mathsf{T}) r_{a_t,u_t}$
12:    $\vec{(}\boldsymbol{\Theta}_{t+1}) \leftarrow \mathbf{A}_{t+1}^{-1}\mathbf{b}_{t+1}$
13:    $\mathbf{C}_{a_t,t+1} \leftarrow \mathbf{C}_{a_t,t} + (\hat{\boldsymbol{\Theta}}_t^{\mathbf{v}} \mathbf{w}_{u_t})(\hat{\boldsymbol{\Theta}}_t^{\mathbf{v}} \mathbf{w}_{u_t})^\mathsf{T}$
14:    $\mathbf{d}_{a_t,t+1} \leftarrow \mathbf{d}_{a_t,t} + (\hat{\boldsymbol{\Theta}}_t^{\mathbf{v}} \mathbf{w}_{u_t})(r_{a_t,u_t} - \mathbf{x}_{a_t}^\mathsf{T}(\hat{\boldsymbol{\Theta}}_t^{\mathbf{x}} \mathbf{w}_{u_t}))$
15:    $\hat{\mathbf{v}}_{a_t,t+1} \leftarrow \mathbf{C}_{a_t,t+1}^{-1} \mathbf{d}_{a_t,t+1}$
16:    Project $\hat{\boldsymbol{\Theta}}_{t+1}$ and $\hat{\mathbf{v}}_{a_t,t+1}$ with respect to the constraints $\|\boldsymbol{\theta}_u\|_2 \leq S$ and $\|(\mathbf{x}_a, \mathbf{v}_a)\|_2 \leq L$
```

error of latent user and item factors satisfies,

$$\|\vec{(}\boldsymbol{\Theta}_t) - \vec{\boldsymbol{\Theta}}^*)\|_{\mathbf{A}_t} \leq \sqrt{\log\big(\frac{det(\mathbf{A}_t)}{\delta\lambda_1}\big)} + \sqrt{\lambda_1}S + \frac{2}{\sqrt{\lambda_1}} \frac{(q_1+\epsilon_1)(1-(q_1+\epsilon_1)^t)}{1-(q_1+\epsilon_1)} \tag{2.11}$$

$$\|\hat{\mathbf{v}}_{a,t} - \mathbf{v}_a^*\|_{\mathbf{C}_{a,t}} \leq \sqrt{\log\big(\frac{det(\mathbf{C}_{a,t})}{\delta\lambda_2}\big)} + \sqrt{\lambda_2}L + \frac{2}{\sqrt{\lambda_2}} \frac{(q_2+\epsilon_2)(1-(q_2+\epsilon_2)^t)}{1-(q_2+\epsilon_2)} \tag{2.12}$$

in which $q_1 \in (0,1)$ and $q_2 \in (0,1)$.

In Lemma 2, ϵ_1 and ϵ_2 are the precision parameters for ALS, and q_1 and q_2 can be explicitly estimated as described in [67]. The key assumption behind this lemma is the noise distribution in reward generation defined in Eq (2.10) is *stationary*. As a result, this lemma gives us a reasonable construction of the confidence sets for $\boldsymbol{\Theta}$ and $\mathbf{v}_a$ estimation, which can be easily transformed to the estimation uncertainty of payoff $r_{a_t,u}$. The proof sketch of this lemma can be found in the appendix.

Based on Lemma 2, we define α_t^u and α_t^a as the upper bound of $\|\vec{(}\boldsymbol{\Theta}_t) - \vec{\boldsymbol{\Theta}}^*)\|_{\mathbf{A}_t}$ and $\|\hat{\mathbf{v}}_{a,t} - \mathbf{v}_a^*\|_{\mathbf{C}_{a,t}}$ respectively. By applying the UCB principle, the item selection strategy for our bandit algorithm can be derived as step 9 in Algorithm 2. In particular, the first term in our item selection strategy is an online prediction of the expected reward based on the current estimation of latent user factors and item factors. It reflects the tendency for exploiting the current estimates. The second and third terms are related to the estimation uncertainty of $\mathbf{v}_a$ and $\boldsymbol{\Theta}$. They reflect the tendency for exploring currently less promising but highly uncertain items. It is easy to verify that the exploration terms shrink when more observations become available, such that the exploit/explore trade-off is balanced dynamically. Later on we prove that because of the explicit modeling of user dependency (i.e., Eq (2.10)), the exploration term also uniformly shrinks for new users and new items, which lead to considerable regret reduction over all users. We name the resulting bandit algorithm as FactorUCB, and illustrate the detailed procedure of it in Algorithm 2.

2.3.2 Regret Analysis

To quantify the performance of factorUCB, we consider the accumulated (pseudo) regret defined Eq (1.1). Based on Lemma 2 and the developed item selection strategy, we have the following theorem about the upper regret bound of FactorUCB algorithm.

Theorem 2. *Under proper initialization of ALS in Algorithm 2, with probability at least* $1-\delta$*, the accumulated regret of FactorUCB algorithm satisfies,*

$$\begin{aligned}\boldsymbol{R}(T) \leq & 2\alpha_T^u\sqrt{2(d+l)NT\log\Big(1+\frac{L^2\sum_{t=1}^T\sum_j^N w_{u_t,j}^2}{\delta\lambda_1(d+l)N}\Big)} + 2\alpha_T^a\sqrt{2lT\log\Big(1+\frac{S^2\sum_{t=1}^T\sum_j^N w_{u_t,j}^2}{\delta\lambda_2 l}\Big)} \\ & + 2\alpha_T^a\frac{(q_2+\epsilon_2)\big(1-(q_2+\epsilon_2)^T\big)}{1-(q_2+\epsilon_2)}\end{aligned} \tag{2.13}$$

in which q_2 and ϵ_2 are the same as those defined in Lemma 2, α_T^u and α_T^a are the upper bound of $\|\vec{\Theta}_t)-\vec{\Theta}^*)\|_{\mathbf{A}_t}$ and $\|\hat{\mathbf{v}}_{a,t}-\mathbf{v}_a^*\|_{\mathbf{C}_{a,t}}$ over all $t\in\{1,\dots,T\}$ respectively, and δ is also encoded in α_T^u and α_T^a as shown in Eq (2.11) and (2.12). Though required by the theorem that λ_1 and λ_2 have to be sufficiently large, in our empirical evaluations the algorithm's performance is not sensitive to this setting. The specific form of α_T^u and α_T^a and the proof sketch of this theorem are provided in the appendix.

As highlighted in the proof, because the confidence interval is shrinking via exploration, a sublinear regret is achieved after T trials of interactions; otherwise without proper exploration, such as in the conventional offline training and online testing paradigm of matrix factorization, a linear regret is inevitable. Moreover, the resulting regret bound of factorUCB has the following important theoretical properties under different conditions.

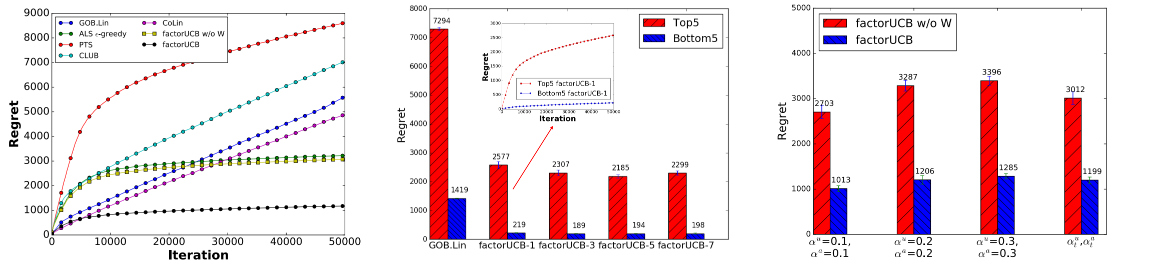
First, the dependency structure among users plays an important role in reducing the regret on both user side and item side. Consider the following two extreme cases. In the first case, when $\mathbf{W}$ is an identity matrix, i.e., no dependency among users, the first term of the upper regret bound in Eq (2.13) degenerates to $O\big(N(d+l)\sqrt{T}\log\frac{T}{N}\big)$, which roots in the reward prediction uncertainty from the estimated latent user factors. And the second term degenerates to $O\big(l\sqrt{T}\log T\big)$, which corresponds to the estimated latent item factors. In the second case, when users are homogenous and have uniform influence among each other, i.e., $\forall i,j, w_{ij}=\frac{1}{N}$, the first term in the regret bound decreases to $O\big(N(d+l)\sqrt{T}\log\frac{T}{N^2}\big)$ and the second decreases to $O\big(l\sqrt{T}\log\frac{T}{N}\big)$. As a result, via modeling user dependency, FactorUCB achieves an $O\big(N(d+l)\sqrt{T}\log N\big)$ regret reduction on the user side and an $O\big(l\sqrt{T}\log N\big)$ regret reduction on the item side.

Second, as denoted in Eq (2.13), the user arrival sequence is recorded in the summation term of $\sum_{t=1}^T\sum_{j=1}^N w_{u_t,j}^2$, which is bounded by T from above, no matter how users arrive to the system (as $\mathbf{w}_u$ is a stochastic vector). Therefore, the upper regret bound of factorUCB stays in $O\big(N(d+l)\sqrt{T}\log\frac{T}{N}\big)$ in the worse case scenario, such as users arrive in an adversarial way – the least connected users come first and most often.

Third, following our enhanced reward generation assumption specified in Eq (2.10), the estimation quality of latent user factors in factorUCB satisfies the following inequality (similar result applies to the estimation quality of latent item factors as well),

$$\|\vec{\Theta}_t)-\vec{\Theta}^*)\|_{\mathbf{A}_t} \leq \sqrt{\log\big(\frac{det(\mathbf{A}_t)}{\delta\lambda_1}\big)} + \sqrt{\lambda_1}S + \frac{2}{\sqrt{\delta\lambda_1}}\sum_{t'=1}^{t}\|\mathbf{v}^*_{a_{t'},u}-\hat{\mathbf{v}}_{a_{t'},u}\|_2 \tag{2.14}$$

If the dimension of latent factors matches the ground-truth, based on the proved convergence property of ALS in [67], the estimation of Θ and $\mathbf{v}_a$ is q-linearly convergent to the optimum $(\Theta^*, \mathbf{v}_a^*)$, which is the conclusion in Lemma 2. But if the dimension is not correctly set and those latent factors are independent from each other, the third term in Eq (2.14) will not converge. It makes α_t^u linearly increase over time as α_t^u is the upper bound of $\|\vec{\Theta}_t)-\vec{\Theta}^*)\|_{\mathbf{A}_t}$. This leads to a linear regret in factorUCB at the worst case. Admittedly, determining the dimension of latent factors is always a bottleneck of factorization-based methods in practice. But by introducing the observable contextual features, especially those strongly correlated with the expected rewards, the reward prediction uncertainty can be reduced as the latent factors only need to fit the residual of reward prediction from the observed features (as shown in the estimation of $\mathbf{v}_a$ in Algorithm 2). This leads to reasonable performance of factorUCB in our empirical evaluations.



(a) Cumulated regret (b) Effect of latent dimensions (c) Effect of (α_t^u, α_t^a)

Figure 2.5: Analysis of regret, hidden feature dimension and parameter tuning.

2.3.3 Experiments

We performed extensive empirical evaluations of our proposed factorUCB algorithm against several state-of-the-art factorization-based and bandit-based collaborative filtering methods, including: 1) Alternating Least Square (ALS) with ϵ-greedy [50], which applies context-free ϵ-greedy algorithm based on both observed features and latent factors, but cannot utilize the user relational graph; 2) Particle Thompson Sampling for matrix factorization (PTS) [49], which combines Thompson sampling with probabilistic matrix factorization based on Rao-Blackwellized particle filter, and it cannot utilize observed features and user relational graph; 3) GOB.Lin [15], which models the dependency among a set of contextual bandits over users via a graph Laplacian based model regularization, but cannot estimate the latent factors; 4) CLUB [45], which clusters users during online learning to enable model sharing; but it only works with contextual features; 5) CoLin [65], which imposes a similar collaborative reward generation assumption over the user relational graph as that in our algorithm, but does not model latent factors; 6) factorUCB w/o W, which is factorUCB with an identity $\mathbf{W}$ matrix, i.e., the dependency among users is not considered; it demonstrates of utility of modeling user dependency in interactive recommendation.

Experiments on synthetic dataset

In simulation, we generated a size-K item pool $\mathcal{A}$, in which each item a is associated with a $(d+l)$-dimension feature vector $(\mathbf{x}_a, \mathbf{v}_a)$. Each dimension is drawn from a set of zero-mean Gaussian distributions with variances sampled from a uniform distribution $U(0,1)$. Principle Component Analysis (PCA) was performed to make all the dimensions *orthogonal* to each other. To simulate the reward generation defined in Eq (2.10), we used all the $(d+l)$-dimension features to compute the true reward for each item, but only revealed the first d dimensions (i.e., $\mathbf{x}_a$) to an algorithm. We simulated N users, each of who is associated with a $(d+l)$-dimension preference vector $\boldsymbol{\theta}_u^*$. Each dimension of $\boldsymbol{\theta}_u^*$ is drawn from a uniform distribution $U(0,1)$. $\boldsymbol{\theta}_u^*$ is treated as the ground-truth latent user factor in reward generation, and is unknown to the algorithms. We then constructed the golden relational stochastic matrix $\mathbf{W}$ for the dependency 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 $\mathbf{W}$ was disclosed to all the algorithms. To increase the learning complexity, at each trial t, our simulator only disclosed a subset of items in $\mathcal{A}$ to the learners for selection, e.g., randomly selected 10 items from $\mathcal{A}$ without replacement. At each trial t, the same set of items were presented to all the algorithms; and the Gaussian noise η_t in Eq (2.10) was sampled once for all those items at each trial. We fixed the dimension d of observable features to 20, the dimension l of latent item factors to 5, user size N to 100, the standard derivation σ of Gaussian noise to 0.1, and the item pool size K to 1000 in our simulation.

Cumulated regret defined in Eq (1.1) was used to evaluate the performance of different algorithms in Figure 2.5 (a), where we set the dimension for latent factors in PTS to 10 (which gave us the best performance) and 5 in ALS ϵ-greedy and factorUCB. We observed that PTS took much longer time to converge, because PTS cannot utilize the observed context features for reward prediction, so that it requires much more observations to improve the accuracy of latent factor estimation. Instead, ALS ϵ-greedy and factorUCB leveraged the context features to quickly reduce the reward prediction uncertainty (i.e., less exploration). Two contextual bandits, i.e., GOB.Lin and CoLin, suffered from linear regret, since they do not model the latent item factors. In addition, factorUCB converged much faster than factorUCB w/o W, which confirmed our theoretical analysis about the regret reduction from user dependency modeling.

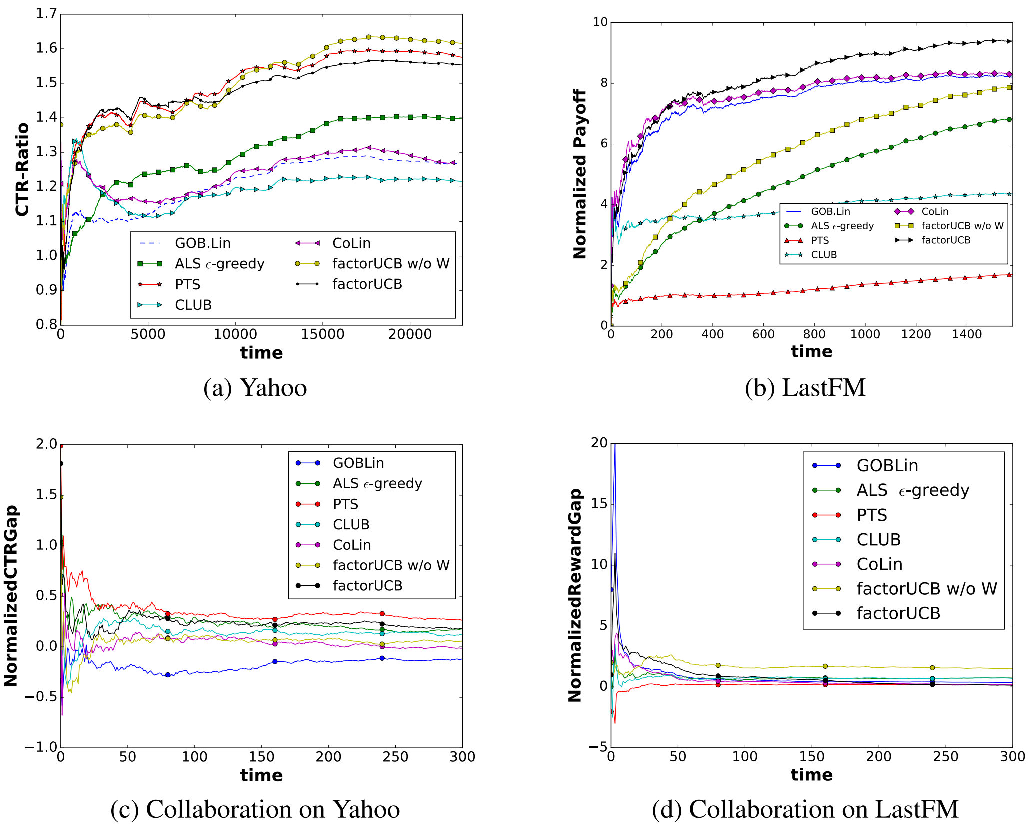


Figure 2.6: Experimental comparisons on real-world datasets.

Because factorUCB requires the dimension of latent factor as input, we test its sensitivity to the setting of latent dimension l. To investigate the importance of correct setup of latent factor dimension in factorUCB, we tested two different ways of latent factor construction in our simulator: 1) we chose the top 5 dimensions with the largest eigenvalue from PCA's result as latent item factors, i.e., we hid the top 5 most informative factors in reward generation from the learners; 2) we hid the bottom 5 most informative factors. And on the algorithm side, we varied the dimension of latent factors used in factorUCB from 1 to 7. From the results shown in Figure 2.5 (b), we can reach three conclusions. First, when the latent factors were the most informative ones, we obtained much worse regret than that in the case of the least informative factors were hidden. Second, the large difference between the regret of a bandit algorithm that does not model the latent factors (such as GOB.Lin) and the one that models latent factors (factorUCB, even with wrong dimensions) emphasizes the necessity of latent factor learning in online recommendation. Third, although our theoretical analysis predicts a linear regret if the latent factor dimension was not accurately set, the actual performance was much more promising. One reason is that our theoretical analysis is for the worst case scenario (upper regret bound), which does not preclude a sub-linear converging regret in practice.

In addition, we also investigated the effect of exploration parameter α_t^u and α_t^a in factorUCB, compared with factorUCB w/o W. In Figure 2.5 (c), each column illustrates a combination of α_t^u and α_t^a used in factorUCB and factorUCB w/o W. The last column indexed by (α_t^u, α_t^a) represents the theoretical values of those two parameters computed from the algorithm's corresponding regret analysis. As shown in the results, the empirically tuned (α^u, α^a) yielded comparable performance to the theoretical values, and made online computation more efficient. As a result, in all our following experiments we will use the manually set α_t^u and α_t^a.

Experiments on real-world datasets

Yahoo dataset: We reported the normalized CTR results from different algorithms over 160 derived user groups in Figure 2.6 (a) (similar relative improvement was obtained with different number of derived user groups). Both variants of FactorUCB outperformed conventional bandit algorithm (i.e., GOB.Lin, CoLin and CLUB) and factorization method (i.e., ALS ϵ-greedy). And clearly via modeling user dependency during online factorization, FactorUCB improves more rapidly than PTS when more observations become available.

LastFM dataset: We normalized the accumulated reward from different algorithms by that from a random algorithm, and reported the results in Figure 2.6 (b). We can clearly notice that PTS performed the worst, while two contextual bandits (i.e., GOB.Lin and CoLin) achieved much better performance than it. This indicates the observed context features in this dataset were sufficiently informative for the algorithms to make accurate recommendations. A purely factorization-based method got penalized by not utilizing such information. On the other hand, we also noticed that factorUCB converged much faster than factorUCB w/o W, which again demonstrates the utility of user dependency modeling for addressing cold-start in recommendation.

To further investigate the effect of modeling context features and user dependency in alleviating cold-start in recommendation, we designed a set of controlled experiments. We first split users into two groups using a max-cut algorithm on the constructed user relational graph to maximize the connectivity between these two groups. Observations in the first user group are called "learning group" and those in the second group are called "testing group." To simulate cold-start, we only executed algorithms on the testing group. Correspondingly, we simulated *warm-start* by first running algorithms on the learning group to pre-estimate the models, and then continuing them on the testing group. Since users in the testing group were isolated from the learning group, their model parameters could only be initialized by the propagated information via the user relational graph, if an algorithm explicitly modeled that.

We measured the differences in average CTR on Yahoo and accumulated rewards on LastFM between *warm-start* and *cold-start* in Figure 2.6 (c) and (d). On the Yahoo dataset, factorization-based algorithms (i.e., factorUCB, PTS and ALS ϵ-greedy) benefit the most from the collaboration in latent factor estimation: latent item factors estimated in the learning group helped them better estimate user preferences in testing group. On the LastFM dataset, considerable improvement was achieved in algorithms explicitly modeling user dependency, i.e., factorUCB, GOB.Lin and CoLin.

2.4 Privacy Protection in Collaborative Bandit Learning

The challenges regarding the risk of privacy breach in a collaborative bandit based recommender system are unique. In such a system, the algorithm recommends an item to a user, and the user provides feedback (e.g., click) based on his/her true preference. The feedback (reward) is then used to update not only the model's reward estimation on this user, but also other on users via the imposed dependency among users. As a result, any change in one user's feedback promptly leads to changes in the algorithm's output, e.g., different sequences of recommended items, potentially for *all* users. This is originally designed to improve subsequent recommendations collectively across all users. But a user's private information could thus be inferred and revealed simply by releasing the recommendation sequence, e.g., extraction attack, even if this user's feedback is kept private in the system.

In this work, we propose the first study to equip collaborative bandit algorithms with privacy guarantees, under the notion of *global differential privacy* [35] and *local differential privacy* [36]. Under global differential privacy, a user is assumed to trust (or say he/she has to trust) the system and provide real engagement data to the system, and the system outputs private recommendations; while under local differential privacy, each user provides perturbed statistics to the system and is no longer required to trust the system or the communication between him/her and system. As the very first study on private collaborative bandits, we focus on algorithms that leverage *known dependency* (e.g., social connections) among users, such as [15, 65]. Specifically, these algorithms propagate the reward collected from one user to update his/her peers' bandit models, according to a given and fixed user dependency structure.

One common practice to achieve privacy guarantee is to inject noise to perturb certain statistics derived from private information in the learning process, either on the server side to achieve global differential privacy or on the client side to achieve local differential privacy [35, 36, 68]. However, how to efficiently inject noise in the collaborative bandit learning setting is non-trivial, because of the inherent information sharing mechanism. Specifically, to preserve privacy in collaborative bandits, we apply the tree-based mechanism [61, 62] to add Laplace noise to the models' statistics to guarantee privacy on each user's *reward* feedback (e.g., user clicks).

We conduct sensitivity analysis, to which the key is to calibrate the noise scale with respect to the *structure of collaboration* defined by the user dependency graph. Our insight is that a careful sensitivity analysis over the collaboration structure offers the opportunity to inject minimum amount of noise and better balance the *privacy and utility trade-off*. In this work, we employ the collaborative bandit algorithm developed in this dissertation, i.e., Collaborative LinUCB (CoLin) [65], as the baseline algorithm, which represent a classic types of social network based collaboration structure. We develop its private versions to illustrate a general solution framework for private collaborative bandit. We prove the private algorithms reduce the added regret caused by privacy-preserving mechanism compared to its linear bandits counterparts, i.e., collaboration actually helps to achieve stronger privacy with the same amount of injected noise. We also empirically evaluate the algorithms on both synthetic and real-world public datasets to validate its effectiveness and show the improved trade-off between utility and privacy from our proposed solution framework.

2.4.1 Differential Privacy

For a contextual bandit algorithm that interacts with users over time horizon T, denote $S = \{r_t\}_{t=1}^T$ as the reward sequence, where r_t is the reward feedback from user u_t at time t. S' is considered as an adjacent neighboring sequence of S, if it only differs from S at one point of reward r_i. The output of a bandit algorithm $\mathcal{O}$ (which is observed by the adversary) is the sequence of its selected arms, i.e., $\{a_t\}_{t=1}^T$.

Definition 1 (Global Differential Privacy (DP) [35]). *A randomized mechanism $\mathcal{M}$ is ϵ-differentially private if for any adjacent neighboring sequences $\{S, S'\}$ and output, $\mathbb{P}(\mathcal{M}(S) \in \mathcal{O}) \le e^{\epsilon}\mathbb{P}(\mathcal{M}(S') \in \mathcal{O})$.*

Global differential privacy ensures the adversary observes almost the same output from a private algorithm, in a probabilistic sense, if only one input data point is changed. The difference between the corresponding output is characterized by ϵ. Laplace or Gaussian noise is commonly introduced to disguise the output, where the noise scale is related to the privacy budget ϵ and the *sensitivity* of $\mathcal{M}$. We formally define sensitivity below.

Definition 2 (Sensitivity [35]). *For any adjacent neighboring sequences $\{S, S'\}$, global sensitivity of a function $f(\cdot)$ is defined as $\Delta_f = \max_{S,S'}|f(S) - f(S')|$.*

Global differential privacy protects sensitive user data from an adversary who has access to the algorithm's output. But it requires the user to send his/her authentic data to the server. Thus, the server and the communication between user and server have to be trusted. To lift the trust needed from the user, local differential privacy (LDP) is proposed [36]. The key idea is that the privacy mechanism needs to perturb the sensitive statistics on the client side before sending it to the server for further computation. Local differential privacy has been adopted in many real-world applications, such as the RAPPOR system developed by Google to collect web browsing behaviour [69], and Apple provides this privacy protection when collecting users' usage and typing history [70]. Note that the input and output of a local differential privacy mechanism could be different from the global differential privacy mechanism, even for the same problem, as they impose different privacy requirements. Let S_i be the reward sequence of user u_i such that $\bigcup_i S_i = S$. The formal definition of local differential privacy is provided below, where a user perturbs his/her private statistics S_i using mechanism L locally, and then send the noisy statistics to the server.

Definition 3 (Local Differential Privacy (LDP) [36]). *A randomized mechanism $\mathcal{M}$ is ϵ-locally differentially private if for any input $\{S_i, S_i'\}$ and output $\mathcal{O}$, $\mathbb{P}(\mathcal{M}(S_i) \in \mathcal{O}) \le e^{\epsilon}\mathbb{P}(\mathcal{M}(S_i') \in \mathcal{O})$*

The key difference between LDP and DP is that a DP mechanism takes all users' data S as input and requires the output to be indifferentiable, while LDP mechanism takes only one user's data S_i as input and generates randomized responses per user (locally) for downstream tasks.

2.4.2 Global Differential Privacy for CoLin

In Collaborative LinUCB (CoLin [65]), contextual bandit models are placed on a weighted graph $G = (\mathcal{V}, \mathcal{E})$, which encodes the affinity relationship among users. Specifically, each node $v_i \in \mathcal{V}$ in G hosts a bandit model parameterized by $\boldsymbol{\theta}_i$ for user i; and the edges in $\mathcal{E}$ represent the affinity relation over pairs of users. This graph is encoded as an $N \times N$ stochastic matrix $\mathbf{W}$, in which each element w_{ij} is nonnegative and proportional

Algorithm 3 Differentially Private CoLin (DP-CoLin)

```
1: Inputs: $\delta \in \mathbb{R}_+$,$\lambda \in [0,1]$, $\mathbf{W} \in \mathbb{R}^{N\times N}$, $\Delta$
2: Initialize: $\mathbf{A}_1 \leftarrow \lambda \mathbf{I}_{dN\times dN}$, $\mathbf{b}_1 \leftarrow \mathbf{0}$, $\hat{\boldsymbol{\vartheta}}_1^p \leftarrow \mathbf{A}_1^{-1}\mathbf{b}_1$,
3: for $t = 1$ to $T$ do
4:     Receive user $u_t$, observe context vectors, $\mathbf{x}_{a_t,u_t} \in \mathbb{R}^d$ and construct $\tilde{\mathbf{x}}_{a_t,u_t} = \text{Vec}(\mathring{\mathbf{X}}_{a_t,u_t}\mathbf{W}^\top)$ for $\forall a \in \mathcal{A}$
5:     Take action $a_t = \arg\max_{a\in\mathcal{A}} \tilde{\mathbf{x}}_{a_t,u_t}^\top \hat{\boldsymbol{\vartheta}}_t^p + \alpha_t\sqrt{\tilde{\mathbf{x}}_{a_t,u_t}^\top \mathbf{A}_t^{-1}\tilde{\mathbf{x}}_{a_t,u_t}}$, where $\alpha_t$ is given by Lemma 4.
6:     Observe payoff $r_{a_t,u_t}$
7:     $\mathbf{A}_{t+1} \leftarrow \mathbf{A}_t + \tilde{\mathbf{x}}_{a_t,u_t}\tilde{\mathbf{x}}_{a_t,u_t}^\top$, $\mathbf{b}_{t+1} \leftarrow \mathbf{b}_t + \tilde{\mathbf{x}}_{a_t,u_t} r_{a_t,u_t}$
8:     Sample noise $\eta_t \sim$ TreeMechanism$(\Delta,\epsilon)$, in which $\Delta = \max_i L\|\mathbf{W}_i\|_2$
9:     $\mathbf{b}_{t+1}^p \leftarrow \mathbf{b}_{t+1} + \eta_t$, $\hat{\boldsymbol{\vartheta}}_{t+1}^p \leftarrow \mathbf{A}_{t+1}^{-1}\mathbf{b}_{t+1}^p$
```

to the influence that user i has on user j. $\mathbf{W}$ is normalized such that $\sum_{i=1}^N w_{ij} = 1$ for $j \in \{1, ..., N\}$, and it is assumed to be time-invariant and known to the learner beforehand. Accordingly, CoLin postulates an *additive* reward generation assumption: the expected reward $\mathbb{E}[r_{a_t,u_t}]$ is not only determined by user u_t's own preference on the arm a_t, but also by that from the neighbors who have influence on u_t as $\mathbb{E}[r_{a_t,u_t}] = \sum_{j=1}^N w_{u_t j}\mathbf{x}_{a_{t,j}}^\top \boldsymbol{\theta}_j$; or equivalently this can be described as,

$$r_{a_t,u_t} \sim N\big(\text{Vec}(\mathring{\mathbf{X}}_{a_t,u_t}\mathbf{W}^\top)^\top \text{Vec}(\boldsymbol{\Theta}), \sigma^2\big) \tag{2.15}$$

where Vec$(\cdot)$ is the matrix vectorization operation, $\boldsymbol{\Theta}$ is a $d \times N$ matrix consisting of parameters from all the bandits in the graph: $\boldsymbol{\Theta} = (\boldsymbol{\theta}_1, \dots, \boldsymbol{\theta}_N)$, and $\mathring{\mathbf{X}}_{a_t,u_t}$ is a $d \times N$ matrix with only the column corresponding to user u_t at time t set to $\mathbf{x}_{a_t,u_t}^\top$ and all the other columns set to zero. By defining $\tilde{\mathbf{x}}_{a_t,u_t} = \text{Vec}(\mathring{\mathbf{X}}_{a_t,u_t}\mathbf{W}^\top)$ and $\boldsymbol{\vartheta} = \text{Vec}(\boldsymbol{\Theta})$, Eq (2.15) can be re-written as $r_{a_t,u_t} \sim N(\tilde{\mathbf{x}}_{a_t,u_t}^\top\boldsymbol{\vartheta}, \sigma^2)$.

With such a collaborative reward generation assumption, CoLin appeals to ridge regression for estimating the global bandit parameter matrix $\boldsymbol{\vartheta}_t$ over all the users at time t. It has a closed-form solution $\hat{\boldsymbol{\vartheta}}_t = \mathbf{A}_t^{-1}\mathbf{b}_t$, in which $\mathbf{A}_t = \lambda\mathbf{I}_{dN} + \sum_{t'=1}^{t-1}\tilde{\mathbf{x}}_{a_{t'},u_{t'}}\tilde{\mathbf{x}}_{a_{t'},u_{t'}}^\top$ and $\mathbf{b}_t = \sum_{t'=1}^{t-1}\tilde{\mathbf{x}}_{a_{t'},u_{t'}} r_{a_{t'},u_{t'}}$. $\mathbf{I}_{dN}$ is an identity matrix and λ is the trade-off parameter for the L2 regularization in ridge regression.

The required information sharing in CoLin brings unique challenges in protecting users' reward feedback, i.e., the change in one user's reward feedback can be effectively inferred from all users' observed recommendation sequences. The recommendation sequences for all users thus have to be perturbed to obtain differential privacy. But instead of directly adding noise to the model's output, i.e., its choice of arms, we choose to add noise η_t to the sufficient statistics $\mathbf{b}_t = \sum_{t'}^{t-1}\tilde{\mathbf{x}}_{a_{t'}} r_{t'}$ in CoLin, where we sample η_t from a tree-based mechanism [61,62]. Because differential privacy is immune to post-processing [71], this ensures differential privacy on the algorithm's output. We name this private derivation of CoLin as (Globally) Differentially Private CoLin (DP-CoLin), and provide its details in Algorithm 3.

The key in DP-CoLin is to derive the sensitivity of CoLin. Analyzing sensitivity in a linear bandit is straightforward [29], as the sensitivity on $\mathbf{b}_t$ can be directly bounded by $\|\mathbf{x}_a\|_2|r_a - r_a'| \leq L$, where the reward difference is bounded by 1 and the norm of context vector is bounded by L. However, for collaborative bandits, the context vectors encode user dependency and have a higher dimension $\tilde{\mathbf{x}}_{a,u} \in \mathbb{R}^{dN}$. A trivial bound is $\|\tilde{\mathbf{x}}_{a,u}\|_2|r_a - r_a'| \leq NL$; but we argue this is not tight enough and unnecessarily introduces large noise. Below we analyze the privacy guarantee of DP-CoLin with a tighter sensitivity bound, which calibrates the noise with respect to the structure of collaboration embedded in $\mathbf{W}$.

Privacy Analysis of DP-CoLin.

Lemma 3 provides the sensitivity of model statistics $\mathbf{b}_t$ in CoLin, based on which we develop the privacy guarantee of DP-CoLin.

Lemma 3 (Sensitivity of $\mathbf{b}_t$ in CoLin). *Sensitivity of* $\mathbf{b}_t$ *is* $\Delta = \max_i L\|\mathbf{W}_i\|_2$, *where* $\mathbf{W}_i$ *is the i-th row of user dependecy matrix* $\mathbf{W}$ *and L is the norm of context vector* $\mathbf{x}$.

The proof of this lemma is provided in Appendix. Note that the sensitivity Δ of CoLin is related to the structure of $\mathbf{W}$; and we discuss two extreme cases of $\mathbf{W}$ to illustrate its effect on privacy protection. Consider when $\mathbf{W}$ is an identity matrix, the resulting sensitivity by our Lemma 3 is L, which is the same as in linear bandits, since there is no influence among users. When $\mathbf{W}$ is a uniform matrix, i.e., users have homogeneous influence among each other and $w_{ij} = \frac{1}{N}$, Lemma 3 shows the sensitivity is $\frac{L}{\sqrt{N}}$. This result is significant: stronger user dependency in CoLin not only leads to lower regret [65], but also smaller sensitivity of $\mathbf{b}_t$, which directly reduces the level of required noise to guarantee privacy. This result is also intuitive: when every user has uniform influence on each other, it becomes harder to tell whose action causes the observed change in the algorithm's output. Less perturbation is thus needed to protect a single user's privacy. This improvement can hardly be obtained by directly applying existing conclusions on linear bandits.

Based on the above sensitivity analysis, we prove privacy guarantee of DP-CoLin in the following.

Theorem 3 (Privacy of DP-CoLin). *Algorithm 3 with global sensitivity Δ defined in Lemma 3 is ϵ-differentially private.*

Proof. By applying tree-based mechanism [61,62] with privacy budget ϵ and sensitivity Δ as shown in line 9-11 of Algorithm 3, the perturbed statistics $\mathbf{b}_t^p$ is ϵ-differentially private. Since differential privacy is immune to post-processing [71], this consequently makes the model parameter $\hat{\boldsymbol{\vartheta}}_t^p$ and the sequence of recommendation $\{a_t : t \in [1..T]\}$ produced by $\hat{\boldsymbol{\vartheta}}_t^p$ also ϵ-differentially private. $\square$

Regret Analysis of DP-CoLin.

We first prove the corresponding confidence bound of parameter estimation in DP-CoLin, i.e., α_t in line 5 of Algorithm 3, which governs its upper confidence bound based arm selection for online learning. In the following discussion, we use $\|\mathbf{B}\|_{\mathbf{A}} = \sqrt{\mathbf{B}^\mathsf{T}\mathbf{A}\mathbf{B}}$ to denote the matrix norm of vector $\mathbf{B}$.

Lemma 4 (Confidence Bound of DP-CoLin). *For any $\delta > 0$, with probability at least $1-\delta$, the estimation error of bandit parameters in DP-CoLin is bounded by,*

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

The proof is provided in Appendix. The right-hand side of the inequality in Lemma 4 gives us α_t that is used in line 5 of Algorithm 3 for arm selection. We notice that in order to maintain a private bandit model $\hat{\boldsymbol{\vartheta}}_t^p$, the parameter estimation error of DP-CoLin suffers from an additional term $\frac{\Delta}{\epsilon}\log T\sqrt{\log t}\log\frac{1}{\delta}$ comparing to that in CoLin due to the added noise η_t. Based on Lemma 4, we have the following theorem about the upper regret bound of the DP-CoLin algorithm, which shows the trade-off between privacy budget ϵ and regret.

Theorem 4 (Regret of DP-CoLin). *With probability at least $1-\delta$, the accumulated regret of DP-CoLin algorithm satisfies,*

$$\begin{aligned}\boldsymbol{R}(T) \leq 2\sqrt{2dNT\log\Big(1+\frac{\sum_{t=1}^{T}\sum_{j=1}^{N} w_{u_tj}^2}{\lambda dN}\Big)}\Bigg(\sqrt{\lambda}\|\boldsymbol{\vartheta}^*\| + \sqrt{dN\log\Big(1+\frac{\sum_{t'=1}^{t}\sum_{j=1}^{N} w_{u_{t'}j}^2}{\lambda dN}\Big) - 2\log(\delta)} \\ + \frac{\max_i L\|\mathbf{W}_i\|_2}{\epsilon}\log^{1.5} T\log\frac{1}{\delta}\Bigg)\end{aligned} \tag{2.16}$$

Specifically, the added regret of DP-CoLin comparing to the CoLin is the last term, i.e.,

$$\frac{2\max_i L\|\mathbf{W}_i\|_2}{\epsilon}\log^{1.5} T\log\frac{1}{\delta}\sqrt{2dNT\log\Big(1+\frac{\sum_{t=1}^{T}\sum_{j=1}^{N} w_{u_tj}^2}{\lambda dN}\Big)}$$

We illustrate the proof details in Appendix. From Theorem 4, we can find that the dependency structure plays an important role in the added regret, and again we discuss those two extreme cases of $\mathbf{W}$ to explain its effect. If $\mathbf{W}$ is an identity matrix, the added regret is in the order of $O\big(\frac{\sqrt{N}}{\epsilon}\log^{1.5}T\sqrt{\log\frac{T}{N}}\sqrt{T}\log\frac{1}{\delta}\big)$. And if $\mathbf{W}$ is a uniform matrix, the added regret is in the order of $O\big(\frac{1}{\epsilon}\log^{1.5}T\sqrt{\log\frac{T}{N}}\sqrt{T}\log\frac{1}{\delta}\big)$. It is important to note that the collaboration structure also helps reduce the added regret, by a factor of $\frac{1}{\sqrt{N}}$, required to achieve privacy. In the meanwhile, the regret reduction from collaboration in the original CoLin is still preserved in the first part of Eq (2.16) in DP-CoLin.

2.4.3 Local Differential Privacy for CoLin

Global differential privacy for CoLin requires each user to send true reward (e.g., clicks) to the server, which then aggregates the data, injects noise, and publishes a privacy preserving output. Local differential privacy lifts the trust on the server by asking each user to perturb his/her data locally, before any disclosure to non-trustful server or the communication. Intuitively, this stronger privacy guarantee is at the cost of worse utility.

We present the Locally Differentially Private CoLin algorithm (LDP-CoLin) in Algorithm 4 in Appendix due to the space limit. LDP-CoLin requires a different communication mechanism: instead of directly sending reward r_{a_t,u_t} to the server, each user u maintains $\mathbf{b}_{u,t}=\sum_{t'=1}^{t_u-1}\tilde{\mathbf{x}}_{a_{t'},u}r_{a_{t'},u}$ locally as shown in line 8 of Algorithm 4. Each user perturbs their own $\mathbf{b}_{u,t}$ by a tree-based mechanism, where noise scales with per-user sensitivity Δ_u (line 8-9), and then sends it to the server. The server aggregates the received statistics to get $\mathbf{b}_t^p$ as shown in line 12, and uses it for model estimation and subsequent recommendations. Again in LDP-CoLin the key is to analyze the sensitivity, which controls the minimum amount of noise needed for privacy protection.

Algorithm 4 Locally Differentially Private CoLin (LDP-CoLin)

```
1: Inputs: $\delta \in \mathbb{R}_+$, $\lambda \in [0,1]$, $\mathbf{W} \in \mathbb{R}^{N\times N}$, $\Delta_{1:N}$
2: Initialize: $\mathbf{A}_1 \leftarrow \lambda\mathbf{I}_{dN\times dN}$, $\mathbf{b}_{u,1} \leftarrow \mathbf{0}$ for $\forall u$, $\hat{\boldsymbol{\vartheta}}_1 \leftarrow \mathbf{A}_1^{-1}\mathbf{b}_1$,
3: for $t = 1$ to $T$ do
   // Sever side:
4:     Receive user $u_t$, observe context vectors $\mathbf{x}_{a_t,u_t} \in \mathbb{R}^d$, and construct $\tilde{\mathbf{x}}_{a_t,u_t} = \text{Vec}(\mathring{\mathbf{X}}_{a_t,u_t}\mathbf{W}^\top)$ for $\forall a \in \mathcal{A}$
5:     Take action $a_t = \arg\max_{a\in\mathcal{A}} \tilde{\mathbf{x}}_{a_t,u_t}^\top\hat{\boldsymbol{\vartheta}}_t + \alpha_t\sqrt{\tilde{\mathbf{x}}_{a_t,u_t}^\top\mathbf{A}_t\tilde{\mathbf{x}}_{a_t,u_t}}$, where $\alpha_t$ is given by Lemma 6.
   // User side:
6:     Observe $r_{a_t,u_t}$
7:     Update locally $\mathbf{b}_{u_t,t_{u_t}+1} \leftarrow \mathbf{b}_{u_t,t_{u_t}} + \tilde{\mathbf{x}}_{a_t,u_t}r_{a_t,u_t}$
8:     Sample noise $\eta_{u_t,t_{u_t}} \sim \text{TreeMechanism}_{u_t}(\Delta_{u_t},\epsilon)$, in which $\Delta_{u_t} = L\|\mathbf{W}_{u_t}\|_2$
9:     Send perturbed statistics $\mathbf{b}^p_{u_t,t_{u_t}+1} \leftarrow \mathbf{b}_{u_t,t_{u_t}+1} + \eta_{u_t,t_{u_t}}$ to server
10:    $t_{u_t} \leftarrow t_{u_t} + 1$
   // Server side:
11:    $\mathbf{A}_{t+1} \leftarrow \mathbf{A}_t + \tilde{\mathbf{x}}_{a_t,u_t}\tilde{\mathbf{x}}_{a_t,u_t}^\top$, $\mathbf{b}^p_{t+1} \leftarrow \sum_u \mathbf{b}^p_{u,t_{u_t}}$, $\hat{\boldsymbol{\vartheta}}^p_{t+1} \leftarrow \mathbf{A}_{t+1}^{-1}\mathbf{b}^p_{t+1}$
```

Privacy Analysis of LDP-CoLin

We first analyze the sensitivity Δ_u of $\mathbf{b}_{u,t}$ for each user u, and then show that Algorithm 4 is locally differentially private using this per-user sensitivity.

Lemma 5 (Sensitivity of $\mathbf{b}_{u,t}$ in CoLin). *Sensitivity of* $\mathbf{b}_{u,t}$ *for user* u *is* $\Delta_u = L\|\mathbf{W}_u\|_2$.

The proof is similar to Lemma 3 and the details are provided in Appendix. The main difference is that sensitivity Δ_u is for a specific user u, which only relies on his/her dependent neighbors, i.e., $\mathbf{W}_u$.

Theorem 5 (Privacy of DP-CoLin). *Randomized response* $\mathbf{b}^p_{u,t}$ *in Algorithm 4 with sensitivity* Δ_u *defined in Lemma 5 is* ϵ*-locally differentially private.*

The proof is similar to DP-CoLin but works in the local setting: as shown in line 8-9 of Algorithm 4, each user u maintains his/her own tree-based mechanism with privacy level ϵ and sensitivity Δ_u *locally*. The local statistics $\mathbf{b}_{u,t}$ are perturbed by the tree-based mechanism thus is ϵ-locally differentially private, and thus are $\hat{\boldsymbol{\vartheta}}_t^p$ and the resulting recommendation sequence.

Regret Analysis of LDP-CoLin

Due to local noise injection, the server's arm selection strategy has to be revised accordingly, which can be guided by the following lemma.

Lemma 6 (Confidence Bound of LDP-CoLin)**.** *Let t_i be the number of times where user i interacts with the system up to time t, i.e., $\sum_i t_i = t$. For any $\delta > 0$, with probability at least $1-\delta$, the estimation error of bandit parameters in LDP-CoLin is bounded by,*

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

The proof detail is shown in Appendix. Similarly, the right-hand side of the inequality gives us α_t which is used in line 5 of Algorithm 4. Based on it, we have the following theorem about the upper regret bound of LDP-CoLin.

Theorem 6 (Regret of LDP-CoLin)**.** *With probability at least $1-\delta$, the accumulated regret of LDP-CoLin algorithm (Algorithm 4) satisfies,*

$$\boldsymbol{R}(T) \leq 2\sqrt{2dNT\log\Big(1+\frac{\sum_{t=1}^{T}\sum_{j=1}^{N} w_{u_t j}^2}{\lambda dN}\Big)}\Bigg(\sqrt{\lambda}\|\boldsymbol{\vartheta}^*\| + \sqrt{dN\log\Big(1+\frac{\sum_{t'=1}^{t}\sum_{j=1}^{N} w_{u_{t'}j}^2}{\lambda dN}\Big) - 2\log(\delta)} \tag{2.17}$$

$$+ \frac{1}{\epsilon}\log\frac{1}{\delta}\sqrt{\sum_{i=1}^{N}\|\mathbf{W}_i\|^2\log^3 T_i}\Bigg)$$

Specifically, the added regret of LDP-CoLin comparing to the non-private CoLin is the last term.

Due to space limit, we omit the details of this proof. Note that Theorem 6 is in a general form in which we do not make any assumption about the users' arriving frequency or order. To better illustrate the added regret, we discuss a special case where the frequency of each user interacting with the system is the same, i.e., $T_i = \frac{T}{N}$. The added regret can thus be simplified as,

$$\frac{2}{\epsilon}\log^{1.5}\frac{T}{N}\log\frac{1}{\delta}\sqrt{\sum_{i=1}^{N}\|\mathbf{W}_i\|^2}\sqrt{2dNT\log\Big(1+\frac{\sum_{t=1}^{T}\sum_{j=1}^{N} w_{u_t j}^2}{\lambda dN}\Big)}.$$

Consider the best case scenario where $\mathbf{W}$ is a uniform matrix, e.g., maximum collaboration, the added regret in LDP-CoLin is in the order of $O\Big(\frac{\sqrt{N}}{\epsilon}\log^2\frac{T}{N^2}\sqrt{T}\log\frac{1}{\delta}\Big)$, while DP-CoLin only has the added regret of $O\Big(\frac{1}{\epsilon}\log^{1.5}T\sqrt{\log\frac{T}{N^2}}\sqrt{T}\log\frac{1}{\delta}\Big)$. In fact, in both cases of the illustrative dependency structure, e.g., no collaboration and uniform collaboration, the added regret of LDP-CoLin is roughly $\sqrt{N}$-times larger compared with DP-CoLin's, and increases when the number of users grows. This is the inevitable cost to protect privacy in the local (user) level. We verified this relationship between the number of users and regret in our empirical evaluations later as well.

2.4.4 Experiments

We performed empirical evaluations of our developed private collaborative bandit algorithms against several baseline algorithms including the non-private collaborative bandit algorithms CoLin [65] and GOBLin [15], non-private LinUCB [2] and private LinUCB [29]. The datasets include a synthetic dataset from simulation, and two real-world datasets for music recommendation and bookmark recommendation. We compare models' accumulated regret on the synthetic dataset and accumulated reward on real-world datasets.

Evaluation Datasets

• **Synthetic dataset.** To build a synthetic dataset, we follow the settings in [15, 65] to simulate a collaborative online recommendation environment. Specifically, we generate N users, each of which is associated with a d-dimensional parameter vector $\boldsymbol{\theta}^*$, i.e., $\boldsymbol{\Theta}^* = (\boldsymbol{\theta}_1^*, \dots, \boldsymbol{\theta}_N^*)$. Each dimension of $\boldsymbol{\theta}_i^*$ is drawn from a uniform distribution $U(0, 1)$ and normalized to $\|\boldsymbol{\theta}_i^*\|_2 = 1$. $\boldsymbol{\Theta}^*$ is treated as the ground-truth bandit parameters for reward generation, and they are withheld from bandit algorithms. We construct the golden relational stochastic matrix $\mathbf{W}$ for the graph of users by defining $w_{ij} \propto \langle \boldsymbol{\theta}_i^*, \boldsymbol{\theta}_j^* \rangle$. We delete the edges where w_{ij} is smaller than a predefined threshold, and get the final user graph G by normalizing each column of $\mathbf{W}$ by its L1 norm. Note that since w_{ij} is generated proportionally to the similarity between $\boldsymbol{\theta}_i^*$ and $\boldsymbol{\theta}_j^*$, the resulting graph naturally satisfies the collaborative assumption in GOBLin [15], i.e., connected users share similar $\boldsymbol{\theta}^*$. The resulting user graph G represented by the relational matrix $\mathbf{W}$ are disclosed to the bandit algorithms. In the end, we generate a size-K arm pool $\mathcal{A}$. Each arm a in $\mathcal{A}$ is associated with a d-dimensional feature vector $\mathbf{x}_a$, each dimension of which is also drawn from $U(0, 1)$. We normalize $\mathbf{x}_a$ by its L2 norm.

To simulate the collaborative reward generation process among users, we compute the reward of arm a for user i at time t as $r_{a_t,i} = \text{Vec}(\mathring{\mathbf{X}}_{a_t,i}\mathbf{W}^\mathsf{T})^\mathsf{T}\text{Vec}(\boldsymbol{\Theta}^*) + \gamma_t$ following Eq (2.15), where $\gamma_t \sim N(0, \sigma^2)$. To increase the learning complexity, at each time t, our simulator only discloses a subset of arms in $\mathcal{A}$ to the learning algorithms, e.g., randomly select 10 arms from $\mathcal{A}$ without replacement. In simulation, based on the known bandit parameters $\boldsymbol{\Theta}^*$, the optimal arm $a_{t,i}^*$ and the corresponding reward $r_{a_{t,i}^*}$ for each user i at time t can be explicitly computed. In our experiment, we set the number of users $N = 10$ and size of arm pool $K = 1,000$. We run $T = 30,000$ iterations and interact with users evenly, which means we serve each user i in total $T_i = 3,000$ iterations.

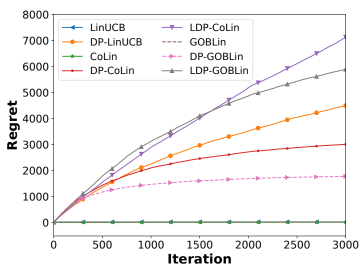
• **LastFM and Delicious datasets.** These two datasets and the pre-processing of them are the same as that in CoLin.

Experiment Results

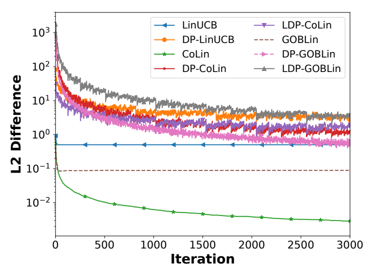
• **Regret comparison.** On the synthetic dataset, accumulated regret is used to evaluate the performance of the compared algorithms. In the real-world datasets, since we do not have an oracle policy, we instead use each learning algorithm's accumulated reward for evaluation. The accumulated regret (the lower the better) on the synthetic dataset and accumulated reward (the higher the better) on real-world datasets are reported in Figure 2.7 (a) and Figure 2.8 respectively. We set the privacy budget $\epsilon = 2$ for all private algorithms in our experiments by default.

In both synthetic and real-world datasets, the non-private collaborative bandits performed better than their globally and locally private counterparts, which is surely expected. We also observe that compared with the globally differentially private collaborative bandit algorithms, i.e., DP-CoLin and DP-GOBLin, the locally differentially private algorithms have significantly worse regret (smaller accumulated reward). This is also expected as local differential privacy is a stronger privacy definition on the user side, and more model perturbation has to be introduced to achieve so. Specifically, as our analysis in Section 2.4.3 suggested, the added regret of LDP collaborative bandit algorithms are roughly $\sqrt{N}$-times larger than their DP counterparts.

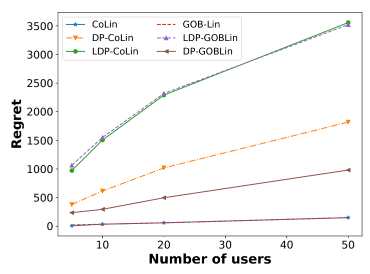
We also notice that DP-CoLin and DP-GOBLin performed better than DP-LinUCB in both synthetic and real-world datasets. The improvement comes from two sources: 1) collaborative learning, which improves the convergence rates of model parameter estimation as discussed in [15, 65]; and 2) privacy mechanism under the collaborative environment, which adds less noise than DP-LinUCB when users are not all independent or disconnected. Accordingly to Figure 2.7 (a), it is obvious that comparing to the regret difference between LinUCB and GOBLin or CoLin, the regret difference between DP-LinUCB and DP-GOBLin or DP-CoLin is much larger. This confirms that the main reason of regret reduction is the calibrated privacy mechanisms developed in this work.



(a) Cumulative regret

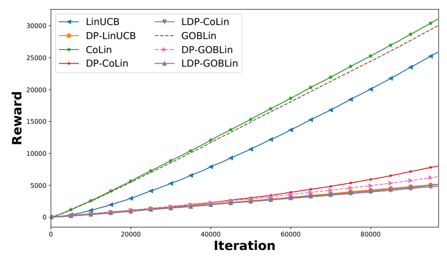


(b) Parameter estimation error

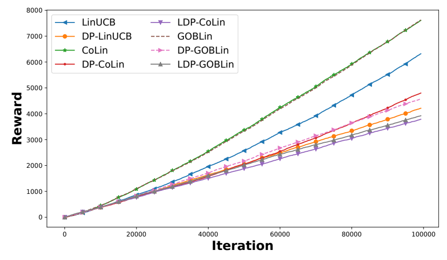


(c) Regret under different number of users

Figure 2.7: Experimental results on synthetic dataset.



(a) Cumulative reward on LastFM dataset.



(b) Cumulative reward on Delicious dataset.

Figure 2.8: Experimental results on real-world datasets.

• **Parameter estimation quality.** To better illustrate the performance of different bandit algorithms, we also studied their parameter estimation quality, which directly measures the algorithms' online learning convergence. Specifically, we reported the L2 difference between the estimated bandit parameter $\hat{\vartheta}_t$ and the ground-truth parameter ϑ^* in Figure 2.7 (b). We observe that private collaborative bandit algorithms have a slower model convergence than their non-private counterparts. Moreover, local differential privacy clearly imposes a much larger estimation error comparing to their counterparts with global differential privacy (note that the y-axis is on a log-scale), which further confirms the required cost to guarantee privacy in the local setting.

Detailed Algorithm-level Analysis

To better understand the trade-off between privacy and utility in collaborative bandit learning, we varied the privacy parameter ϵ and number of users in our evaluation.

• **Effect of privacy budget ϵ.** In Table 2.5, we reported the accumulated regret of the collaborative bandit algorithms with global and local differential privacy under different privacy parameter ϵ. We vary ϵ from 0.5 to 10. We run each experiment for $T = 10,000$ iterations and report the average regret of 5 repeated runs. From the results, we notice a clear trade-off between the required privacy level ϵ and the resulting regret. Stronger privacy requirement (i.e., a smaller ϵ) requires the privacy mechanism to introduce more noise, which directly inflates regret. This result also supports our theoretical analysis that the added regret of the private collaborative bandit algorithms is in the order of $O(\frac{1}{\epsilon})$.

• **Effect of number of users N.** In Figure 2.7 (c), we show the accumulated regret of the collaborative bandit algorithms with global and local differential privacy under different number of users N. We run $T = 10,000$ iterations and all users are evenly served for $\frac{T}{N}$ times. We vary N from 5 to 50. From the result we observe that the regret increases with the number of users. By looking at the difference between the regret of non-private algorithms and their private versions, we can notice that the added regret increases with number of users N. This also validates our theoretical analysis that the added regret for LDP collaborative bandit algorithms is roughly $\sqrt{N}$ times larger than their DP versions, which is the inevitable cost to protect privacy at the local level.

Table 2.5: Cumulative regret across different bandit algorithm under different privacy level ϵ.

ϵ	0.5	1	2	5	10
DP-LinUCB	3082.90±82.69	2683.69±89.74	1504.14±30.40	910.97±20.83	496.11±14.72
DP-CoLin	2619.56±29.44	2450.70±50.51	1327.70±23.79	884.19±23.12	297.18±6.80
DP-GOBLin	2672.56±29.13	2550.22±19.67	964.63±13.61	685.65±6.47	246.92±9.70
LDP-CoLin	3310.53±51.85	3095.10±48.97	2389.05±61.24	1795.40±31.21	938.76±26.16
LDP-GOBLin	3268.70±65.61	3004.80±75.08	2334.62±63.78	1743.57±36.40	1060.53±28.99

2.5 Conclusion

In this chapter, we study contextual bandit learning in collaborative environments. We first developed CoLin [65], in which context and payoffs are shared among the neighboring bandits during the online update, and it helps reduce the preference learning complexity (i.e., requires fewer 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. Then we develop a factorization-based bandit algorithm [72] in which observable contextual features and dependency among users are leveraged to improve the algorithm's convergence rate and help conquer cold-start in recommendation. A high probability sublinear upper regret bound is proved, where considerable regret reduction is achieved on both user and item sides. At last, we studied the problem of protecting global and local differential privacy for collaborative bandits. Our solution framework allows the privacy mechanism to calibrate the noise scale with respect to the user dependency graph. Our theoretical analysis proves the desired privacy guarantee under both settings in CoLin. We also rigorously proved the corresponding upper regret bound of the derived private algorithms. Most importantly, we showed the added regret caused by differential privacy mechanism is still sublinear and benefits from the collaboration structure. Experimental results based on extensive simulations and three real-world datasets consistently confirmed the improvement of the proposed collaborative bandit algorithms against several state-of-the-art contextual bandit algorithms in recommendation tasks.

We showed that collaborative learning can help reduce learning complexity. However, as a downside, collaborative online learning solutions may, at the same time, increase computation complexity. For example, the computation complexity at each interaction increases from $O(d^3)$ to $O(d^3N^3)$, where d is the number of feature dimensions and N is the number of users, in the two collaborative bandit solutions developed in this dissertation (it is also true for one existing collaborative bandit learning method [15]). This computation complexity is simply unaffordable for a system that has a large number of users. In order to make these solutions more practical and scalable, it is especially important to reduce the computation complexity of collaborative bandit learning solutions in future work.

In this chapter, we have been focusing on how to improve learning efficiency by leveraging existing dependency information among the environments, i.e., users. A stationary environment assumption has been made when developing such solutions. In fact, such a stationary environment assumption is extensively used by default in most of the existing contextual bandit and multi-armed bandit solutions. However, this assumption is hardly true in most of the real-world application scenarios, where there can be dramatic changes in users' preference or item popularity. In the next chapter, we introduce our efforts in developing practical contextual bandit solutions that are able to handle a realistic non-stationary environment.

Chapter 3

Bandit Learning in Non-Stationary Environments

Most existing stochastic bandit learning solutions, including the contextual bandit solutions, assume a stationary environment with unknown yet fixed reward mapping function [2, 16, 65, 73, 74]. In practice, this translates to the assumption that users' preferences remain static over time. However, this assumption rarely holds in reality as users' preferences can be influenced by various internal or external factors [1]. If a learning algorithm fails to model or recognize the possible changes in the environment, it would constantly make suboptimal choices, e.g., keep making out-of-date recommendations to users.

In this chapter, we study the problem of contextual bandit learning in a non-stationary environment. More specifically, we focus on the abruptly changing environment, or piecewise stationary environment, in which the environment undergoes abrupt changes at unknown time points but remains stationary between two consecutive change points. Under such a non-stationary environment, we first develop two non-stationary contextual bandit learning solutions. In our solutions, instead of maintaining only one bandit learning agent, we use insights from online statistical hypothesis tests to adaptively add a learner or remove a learner. By doing so, our solutions are able to maintain a dynamic set of active base bandit learners and form dynamic ensembles of them to interact with the current environment. With these solutions, a near-optimal regret bound is achieved when learning in environments with abrupt changes; we prove that linear regret is inevitable otherwise. Then we develop a more general solution based on online hypothesis tests to unify the online change detection in non-stationary environments and online clustering of learners. This unified approach with online hypothesis tests can efficiently handle environments where both non-stationarity and collaborative structures exist.

This chapter is organized as follows: in Section 3.1 we introduce existing literature about interactive online learning in non-stationary environments; in Section 3.2, we introduce the setting of the non-stationary environment, i.e., a piecewise stationary environment, studied in this dissertation; in Section 3.3 and Section 3.4 we introduce the two solutions developed for the piecewise stationary environment and the corresponding theoretical guarantees; in Section 3.5 we present our empirical study of the proposed solutions on a synthetic dataset and three real-world datasets; in Section 3.6, we present a unified approach for change detection and cluster detection in the piecewise stationary environment. All the code associated with the empirical study in this chapter is available at `https://github.com/qw2ky/NonstationaryBanditLib`.

3.1 Related Work

For the piecewise stationary environment studied in this dissertation, Hartland et al. [40] proposed the $\gamma-$Restart algorithm, in which a discount factor γ is introduced to exponentially decay the effect of past observations. Garivier and Moulines [39] proposed a discounted-UCB algorithm, which is similar to the $\gamma-$Restart algorithm in discounting the historical observations. They also proposed a sliding window UCB algorithm, where only observations inside a sliding window are used to update the bandit model. Yu and

Mannor [42] proposed a windowed mean-shift detection algorithm to detect the potential abrupt changes in the environment. An upper regret bound of $O\big(\Gamma_T \log(T)\big)$ is proved for the proposed algorithm, in which Γ_T is the number of ground-truth changes up to time T. However, they assume that at each iteration, the agent can query a subset of arms for additional observations. Slivkins and Upfal [75] considered a continuously changing environment, in which the expected reward of each arm follows Brownian motion. They proposed a UCB-like algorithm, which considers the volatility of each arm in such an environment. The algorithm restarts in a predefined schedule to account for the change of reward distribution.

Most existing solutions for non-stationary bandit problems focus on context-free scenarios, which cannot utilize the available contextual information for reward modeling. Ghosh et al. proposed an algorithm in [76] to deal with environment misspecification in contextual bandit problems. Their algorithm comprises a hypothesis test for linearity followed by a decision to use either the learned linear contextual bandit model or a context-free bandit model. But this algorithm still assumes a stationary environment, i.e., neither the ground-truth linear model nor unknown models are changing over time. Liu et al. [77] proposed to use cumulative sum and Page-Hinkley Test to detect sudden changes in the environment. An upper regret bound of $O(\sqrt{\Gamma_T T}\log T)$ is proved for one of their proposed algorithms. However, this work is limited to a simplified Bernoulli bandit environment. Recently, Luo et al [78] studied the non-stationary bandit problem and proposed several bandit algorithms with statistical tests to adapt to changes in the environment. They analyzed various notions of regret, including interval regret, switching regret, and dynamic regret. Hariri et al. [41] proposed a contextual Thompson sampling algorithm with a change detection module, which involves iteratively applying a combination of cumulative sum charts and bootstrapping to capture potential changes of user preference in interactive recommendation tasks. But no theoretical analysis is provided about this proposed algorithm. To the best of our knowledge, among the existing non-stationary bandit solutions, no work utilizes the context-dependent property of reward changes in a piecewise stationary environment. Hence, none of aforementioned work is able to exploit the existence of *context dependent changes*.

3.2 A Piecewise Stationary Environment

Let us first recall the contextual bandit formulation in a stationary environment. In a multi-armed bandit problem, a learner takes turns to interact with the environment, such as a user or a group of users in a recommender system, with a goal of maximizing its accumulated reward collected from the environment over time T. At round t, the learner makes a choice a_t among a finite, but possibly large, number of arms, i.e., $a_t \in \mathcal{A} = \{a_1, a_2, \ldots, a_K\}$, and gets the corresponding reward r_{a_t}, such as a user clicks on a recommended item. In a contextual bandit setting, each arm a is associated with a feature vector $\mathbf{x}_a \in \mathbb{R}^d$ ($\|\mathbf{x}_a\|_2 \leq 1$ without loss of generality) summarizing the side-information about it at a particular time point. The reward of each arm is assumed to be governed by a conjecture of unknown bandit parameter $\boldsymbol{\theta} \in \mathbb{R}^d$ ($\|\boldsymbol{\theta}\|_2 \leq 1$ without loss of generality), which characterizes the environment. This can be specified by a reward mapping function $f_{\boldsymbol{\theta}}$: $r_{a_t} = f_{\boldsymbol{\theta}}(\mathbf{x}_{a_t})$. In a stationary environment, $\boldsymbol{\theta}$ is constant over time.

In a non-stationary environment, the reward distribution over arms varies over time because of the changes in the environment's bandit parameter $\boldsymbol{\theta}$. In this dissertation, we consider a environment with abrupt changes, a.k.a. a piecewise stationary environment [39–41], in which the ground-truth parameter $\boldsymbol{\theta}$ changes arbitrarily at arbitrary time, but remains constant between any two consecutive change points described as follows:

$$\underbrace{r_0, r_1, \cdots, r_{t_{c_1}-1}}_{\text{governed by } f_{\boldsymbol{\theta}^*_{c_0}}}, \underbrace{r_{t_{c_1}}, r_{t_{c_1}+1}, \cdots, r_{t_{c_2}-1}}_{\text{governed by } f_{\boldsymbol{\theta}^*_{c_1}}}, \cdots, \underbrace{r_{t_{c_\Gamma}}, r_{t_{c_\Gamma}+1}, \cdots, r_T}_{\text{governed by } f_{\boldsymbol{\theta}^*_{c_{\Gamma-1}}}} \tag{3.1}$$

In Eq (3.1) the change points $\{t_{c_j}\}_{j=1}^{\Gamma_T-1}$ of the underlying reward distribution and the corresponding ground-truth bandit parameters $\{\boldsymbol{\theta}^*_{c_j}\}_{j=0}^{\Gamma_T-1}$ are unknown to the learner. We only assume there are at most $\Gamma_T - 1$ change points in the environment up to time T, with $\Gamma_T \ll T$. To simplify the discussion, linear structure in $f_{\boldsymbol{\theta}_u}(\mathbf{x}_{a_t})$ is postulated, but it can be readily extended to more complicated dependency structures, such as generalized linear models [16], without changing the design of our algorithm. Specifically, we have,

$$r_t = f_{\boldsymbol{\theta}^*_t}(\mathbf{x}_{a_t}) = \mathbf{x}_{a_t}^{\mathsf{T}} \boldsymbol{\theta}^*_t + \eta_t \tag{3.2}$$

in which η_t is Gaussian noise drawn from $N(0, \sigma^2)$.

3.3 Dynamic Bandit Learning

Under such a non-stationary environment described in Section 3.2, in this section we propose a two-level hierarchical bandit algorithm, which automatically detects and adapts to changes in the environment by maintaining a suite of contextual bandit models during identified stationary periods based on its interactions with the environment. We make the following assumptions about the non-stationary environment in addition to the piecewise stationary property.

Assumption 1. *For any two consecutive change points t_{c_j} and $t_{c_{j+1}}$ in the environment, there exists $\Delta_j > 0$, such that when $t \geq t_{c_{j+1}}$ at least ρ $(0 < \rho \leq 1)$ portion of all the arms satisfy,*

$$|\mathbf{x}_{a_t}^{\mathsf{T}}\boldsymbol{\theta}^*_{t_{c_{j+1}}} - \mathbf{x}_{a_t}^{\mathsf{T}}\boldsymbol{\theta}^*_{t_{c_j}}| > \Delta_j \tag{3.3}$$

Remark 1. *The above assumption is general and mild to satisfy in many practical scenarios, since it only requires a portion of the arms to have recognizable change in their expected rewards. For example, a user may change his/her preference in sports news but not in political news. The arms that do not satisfy Eq* (3.3) *can be considered as having small deviations in the generic reward assumption made in Eq* (3.2). *We will later prove our bandit solution remains its regret scaling in the presence of such small deviation.*

3.3.1 Dynamic Linear UCB Algorithm

Based on the above assumption about the non-stationary environment, in any stationary period between two consecutive change points, the reward estimation error of a contextual bandit model trained on the observations collected from that period should be bounded with a high probability [37,38]. Otherwise, the model's consistent wrong predictions can only come from the change of environment. Based on this insight, we can evaluate whether the stationary assumption holds by monitoring a bandit model's reward prediction quality over time. Based on the reward prediction quality evaluation, we can maintain one or a set of bandit models whose reward prediction quality are good enough, by creating new ones and abandoning the bad ones on the fly.

Specifically, we propose a hierarchical bandit algorithm, in which a master multi-armed bandit model operates over a set of slave contextual bandit models to interact with the changing environment. The master model monitors the slave models' reward estimation error over time, which is referred to as 'badness' in this work, to evaluate whether a slave model is admissible for the current environment. Based on the estimated 'badness' of each slave model, the master model dynamically discards *out-of-date* slave models or creates new ones. At each round t, the master model selects a slave model with the smallest lower confidence bound (LCB) of 'badness' to interact with the environment, i.e., the most promising slave model. The obtained observation $(\mathbf{x}_{a_t}, r_{a_t})$ is shared across all admissible slave models to update their model parameters. The process is illustrated in Figure 3.1.

Any contextual bandit algorithm [2, 16, 65, 73] can serve as our slave model. Due to the simplified linear reward assumption made in Eq (3.2), we choose LinUCB [2] for the purpose in this paper; but our proposed algorithm can be readily adapted to any other choices of the slave model. This claim is also supported by our later regret analysis. As a result, we name our algorithm as Dynamic Linear Bandit with Upper Confidence Bound, or dLinUCB in short. In the following discussion, we first briefly describe our chosen slave model LinUCB. Then we formally define the concept of 'badness', based on which we design the strategy for creating and discarding slave bandit models. Lastly, we explain how dLinUCB selects the most promising slave model from the admissible model set. The detailed description of dLinUCB is provided in Algorithm 5.

Slave bandit model: LinUCB. Each slave LinUCB model maintains all historical observations that the master model has assigned to it. Based on the assigned observations, a slave model m gets an estimate of user preference $\hat{\boldsymbol{\theta}}_t(m) = \mathbf{A}_t^{-1}(m)\mathbf{b}_t(m)$ [2], in which $\mathbf{A}_t(m) = \lambda\mathbf{I} + \sum_{i\in\mathcal{I}_{m,t}}\mathbf{x}_{a_i}\mathbf{x}_{a_i}^{\mathsf{T}}$, $\mathbf{I}$ is a $d\times d$ identity matrix, λ is the coefficient for $L2$ regularization; $\mathbf{b}_t(m) = \sum_{i\in\mathcal{I}_{m,t}}\mathbf{x}_{a_i}r_{a_i}$, and $\mathcal{I}_{m,t}$ is an index set recording when the observations are assigned to the slave model m up to time t. According to [37], with a high probability $1-\delta_1$ the expected reward estimation error of model m is upper bounded: $|\hat{r}_{a_t}(m) - \mathbb{E}[r_{a_t}]| \leq B_t(m,a)$, in which $B_t(m,a) = \big(\sigma^2\sqrt{d\ln(1+\frac{|\mathcal{I}_{m,t}|}{\lambda\delta_1})} + \sqrt{\lambda}\big)\|\mathbf{x}_a\|_{\mathbf{A}_t^{-1}(m)}$. Based on the upper confidence bound

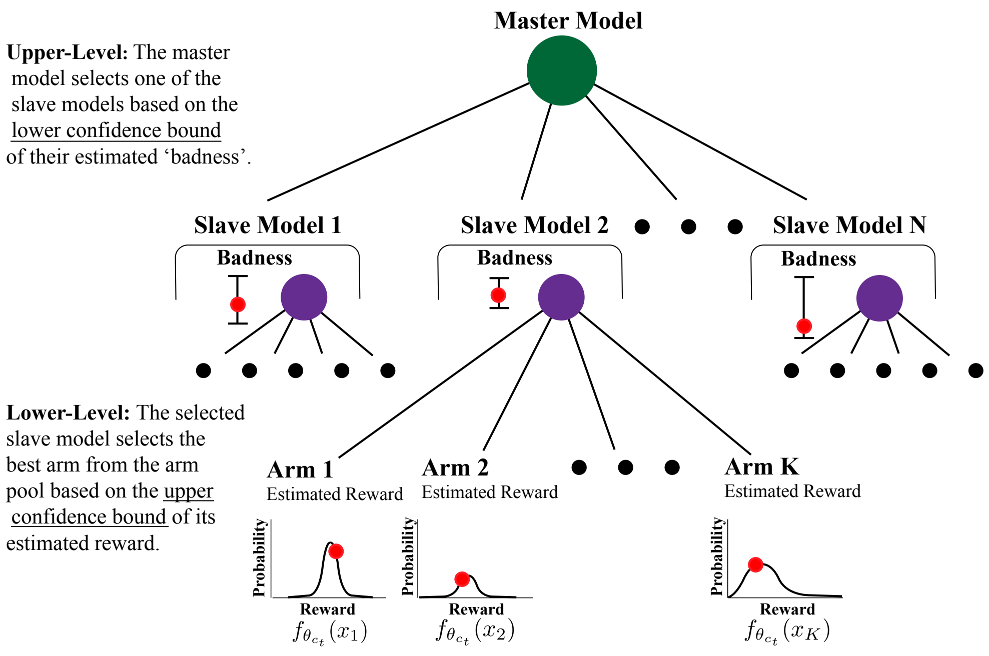

Master Model
Upper-Level: The master model selects one of the slave models based on the lower confidence bound of their estimated 'badness'.
Slave Model 1
Slave Model 2
Slave Model N
Badness
Badness
Badness
Lower-Level: The selected slave model selects the best arm from the arm pool based on the upper confidence bound of its estimated reward.
Arm 1
Estimated Reward
Arm 2
Estimated Reward
Arm K
Estimated Reward
Probability
Reward
$f_{\theta_{c_t}}(x_1)$
$f_{\theta_{c_t}}(x_2)$
$f_{\theta_{c_t}}(x_K)$

Figure 3.1: Illustration of dLinUCB. The master bandit model maintains the 'badness' estimation of slave models over time to detect changes in the environment. At each round, the most promising slave model is chosen to interact with the environment; and the acquired feedback is shared across all admissible slave models for model update.

principle [12], a slave model m takes an action using the following arm selection strategy (i.e., line 6 in Algorithm 5):

$$a_t(m) = \arg\max_{a \in \mathcal{A}} \left(\mathbf{x}_a^\top \hat{\boldsymbol{\theta}}_t(m) + B_t(m, a)\right) \tag{3.4}$$

Slave model creation and abandonment. For each slave bandit model m, we define a binary random variable $e_i(m)$ to indicate whether the slave model m's prediction error at time i exceeds its confidence bound,

$$e_i(m) := \mathbb{1}\left\{|\hat{r}_i(m) - r_i(m)| > B_i(m, a_i) + \epsilon\right\} \tag{3.5}$$

where $\epsilon = \sqrt{2}\sigma \text{erf}^{-1}(\delta_1 - 1)$ and $\text{erf}^{-1}(\cdot)$ is the inverse of Gauss error function. ϵ represents the high probability bound of Gaussian noise in the received feedback.

According to Eq (3.7) in Theorem 7, if the environment stays stationary since the slave model m has been created, we have $\mathbb{P}(e_i(m) = 1) \leq \delta_1$, where $\delta_1 \in (0, 1)$ is a hyper-parameter in $B_i(m, a)$. Therefore, if we observe a sequence of consistent prediction errors from the slave model m, it strongly suggests a change of environment, so that this slave model should be abandoned from the admissible set. Moreover, we introduce a size-τ sliding window to only accumulate the most recent observations when estimating the expected error in slave model m. The benefit of sliding window design will be discussed with more details later in Section 3.4.3.

We define $\hat{e}_t(m) := \frac{\sum_{i=t-\tilde{\tau}(m)}^{t} e_i(m)}{\tilde{\tau}(m)}$, which estimates the 'badness' of slave model m within the most recent period $\tilde{\tau}$ to time t, i.e., $\tilde{\tau}(m) = \min\{t - t_m, \tau\}$, in which t_m is when model m was created. Combining the concentration inequality in Theorem 14 (provided in the appendix), we have the assertion that if in the period $[t - \tilde{\tau}(m), t]$ the stationary hypothesis is true, for any given $\delta_1 \in (0, 1)$ and $\delta_2 \in (0, 1)$, with a probability at least $1 - \delta_2$, the expected 'badness' of slave model m satisfies,

$$\hat{e}_t(m) \leq \mathbb{E}[e_t(m)] + \sqrt{\frac{\ln(1/\delta_2)}{2\tilde{\tau}(m)}} \leq \delta_1 + \sqrt{\frac{\ln(1/\delta_2)}{2\tilde{\tau}(m)}} \tag{3.6}$$

Algorithm 5 Dynamic Linear UCB (dLinUCB)

```
1: Inputs: $\lambda > 0, \tau > 0, \delta_1, \delta_2 \in (0,1), \tilde{\delta_1} \in [0, \delta_1]$
2: Initialize: Maintain a set of slave models $\mathcal{M}_t$ with $\mathcal{M}_1 = \{m_1\}$, initialize $m_1$: $\mathbf{A}_1(m_1) = \lambda\mathbf{I}$, $\mathbf{b}_1(m_1) = \mathbf{0}$, $\hat{\boldsymbol{\theta}}_1(m_1) = \mathbf{0}$; and initialize the 'badness' statistics of it: $\hat{e}_1(m_1) = 0$, $d_1(m_1) = 0$
3: for $t = 1$ to $T$ do
4:     Choose a slave model from the active slave model set $\tilde{m}_t = \arg\min_{m \in \mathcal{M}_t} \big(\hat{e}_t(m) - \sqrt{\ln \tau} \times d_t(m)\big)$
5:     Observe candidate arm pool $\mathcal{A}_t$, with $\mathbf{x}_a \in \mathbb{R}^d$ for $\forall a \in \mathcal{A}_t$
6:     Take action $a_t = \arg\max_{a \in \mathcal{A}_t} \big(\mathbf{x}_a^\mathsf{T}\hat{\boldsymbol{\theta}}_t(\tilde{m}_t) + B_t(\tilde{m}_t, a)\big)$, in which $B_t(\tilde{m}_t, a)$ is defined in Eq (3.4)
7:     Observe payoff $r_{a_t}$
8:     Set CreatNewFlag = True
9:     for $m \in \mathcal{M}_t$ do
10:        $e_t(m) = \mathbb{1}\{|\hat{r}_t(m) - r_t| > B_t(m, a) + \epsilon\}$, where $\hat{r}_t(m) = \mathbf{x}_{a_t}^\mathsf{T}\hat{\boldsymbol{\theta}}_t(m)$ and $\epsilon = \sqrt{2}\sigma\mathrm{erf}^{-1}(\delta_1 - 1)$
11:        if $e_t(m) = 0$ then
12:            Update slave model: $\mathbf{A}_{t+1}(m) = \mathbf{A}_t(m) + \mathbf{x}_{a_t}\mathbf{x}_{a_t}^\mathsf{T}$, $\mathbf{b}_{t+1}(m) = \mathbf{b}_t(m) + \mathbf{x}_{a_t} r_t$, $\hat{\boldsymbol{\theta}}_{t+1} = \mathbf{A}_{t+1}^{-1}(m)\mathbf{b}_{t+1}(m)$
13:            $\tilde{\tau}(m) = \min\{t - t_m, \tau\}$, where $t_m$ is when $m$ was created
14:            Update 'badness' $\hat{e}_t(m) = \frac{\sum_{i=t-\tilde{\tau}}^{t} e_i(m)}{\tilde{\tau}(m)}$, $d_t(m) = \sqrt{\frac{\ln 1/\delta_2}{2\tilde{\tau}(m)}}$
15:        if $\hat{e}_t(m) < \tilde{\delta_1} + d_t(m)$ then
16:            Set CreatNewFlag = False
17:        else if $\hat{e}_t(m) \geq \delta_1 + d_t(m)$ then
18:            Discard slave model $m$: $M_{t+1} = M_t - m$
19:    if CreateNewFlag or $\mathcal{M}_t = \emptyset$ then
20:        Create a new slave model $m_t$: $\mathcal{M}_{t+1} = \mathcal{M}_t + m_t$
21:        Initialize $m_t$: $\mathbf{A}_t(m_t) = \lambda\mathbf{I}$, $\mathbf{b}_t(m_t) = \mathbf{0}$, $\hat{\boldsymbol{\theta}}_t(m_t) = \mathbf{0}$
22:        Initialize 'badness' statistics of $m_t$: $\hat{e}_t(m_t) = 0$, $d_t(m_t) = 0$
```

Eq (3.6) provides a tight bound to detect changes in the environment. If the environment is unchanged, within a sliding window the estimation error made by an up-to-date slave model should not exceed the right-hand side of Eq (3.6) with a high probability. Otherwise, the stationary hypothesis has to be rejected and thus the slave model m should be discarded. Accordingly, if none of the slave models in the admissible bandit set satisfy this condition, a new slave bandit model should be created for this new environment. Specifically, the master bandit model controls the slave model creation and abandonment in the following way.

• *Model abandonment*: when the slave model m's estimated 'badness' exceeds its upper confidence bound defined in Eq (3.6), i.e., $\hat{e}_t(m) > \delta_1 + \sqrt{\frac{\ln(1/\delta_2)}{2\tilde{\tau}(m)}}$, it will be discarded and removed from the admissible slave model set. This corresponds to line 18-20 in Algorithm 5.

• *Model creation*: When no slave model's estimated 'badness' is within its expected confidence bound, i.e., no slave model satisfies $\hat{e}_t(m) \leq \tilde{\delta_1} + \sqrt{\frac{\ln(1/\delta_2)}{2\tilde{\tau}(m)}}$, a new slave model will be created. $\tilde{\delta_1} \in [0, \delta_1]$ is a parameter to control the sensitivity of dLinUCB, which affects the number of maintained slave models. When $\tilde{\delta_1} = \delta_1$, the threshold of creating and abandoning a slave model matches and the algorithm only maintains one admissible slave model. When $\tilde{\delta_1} < \delta_1$ multiple slave models will be maintained. The intuition is that an environment change is very likely to happen when all active slave models face a high risk of being out-of-date (although they have not been abandoned yet). This corresponds to line 8, 16-17, and 22-26 in Algorithm 5.

Slave model selection and update. At each round, the master bandit model selects one active slave bandit model to interact with the environment, and updates all active slave models with the acquired feedback accordingly. As we mentioned before, with the model abandonment mechanism every active slave model is guaranteed to be admissible for taking acceptable actions; but they are associated with different levels of risk of being out of date. A well-designed model selection strategy can further reduce the overall regret, by minimizing this risk. Intuitively, when facing a changing environment, one should prefer a slave model with the lowest empirical error in the most recent period.

The uncertainty in assessing each slave model's 'badness' introduces another explore-exploit dilemma, when choosing the active slave models. Essentially, we prefer a slave model of lower 'badness' with a higher confidence. We realize this criterion by selecting a slave model according to its Lower Confidence Bound (LCB) of the estimated 'badness.' This corresponds to line 4 in Algorithm 5.

Once the feedback $(\mathbf{x}_{a_t}, r_t)$ is obtained from the environment on the selected arm a_t, the master algorithm can not only update the selected slave model but also all other active ones for both of their 'badness' estimation and model parameters (line 11-13 and line 15 in Algorithm 5 accordingly). This would reduce the sample complexity in each slave model's estimation. However, at this stage, it is important to differentiate those "about to be out-of-date" models from the "up-to-date" ones, as any unnecessary model update blurs the boundary between them. As a result, we only update the *perfect* slave models, i.e., those whose 'badness' is still zero at this round of interaction; and later we will prove this updating strategy is helpful to decrease the chance of late detection.

3.3.2 Regret Analysis

In this section, we provide a detailed regret analysis of our proposed dLinUCB algorithm. It is easy to prove that if a bandit algorithm does not model the change of environment, it may suffer from a linearly increasing regret: An optimal arm in the previous stationary period may become sub-optimal after the change; but the algorithm that does not model environment change will constantly choose this sub-optimal arm until its estimated reward falls behind that of the other arms. This leads to a linearly increasing regret in each new stationary period. Next, we first characterize the confidence bound of reward estimation in a linear bandit model in Theorem 7. Then we prove the regret upper bound of a variant of our dLinUCB algorithm in Theorem 8. Detailed proof of this theorem and the related lemmas are provided in Appendix B.

Theorem 7. *For a linear bandit model m specified in Algorithm 5, if the underlying environment is stationary, for any $\delta_1 \in (0,1)$ we have the following inequality with probability at least $1-\delta_1$,*

$$|\hat{r}_t(m) - r_t| \le B_t(m,a) + \epsilon \tag{3.7}$$

where $B_t(m,a) = \alpha_t \|\mathbf{x}_{a_t}\|_{\mathbf{A}_{t-1}^{-1}}$ with $\alpha_t = \big(\sigma^2\sqrt{d\ln(1+\frac{|\mathcal{I}_t(m)|}{\lambda\delta_1})} + \sqrt{\lambda}\big)$, $\epsilon = \sqrt{2}\sigma erf^{-1}(\delta_1 - 1)$, σ is the standard deviation of the Gaussian noise in the reward feedback, and erf$(\cdot)$ is the Gauss error function.

Denote $R_{\text{Lin}}(S)$ as the upper regret bound of a linear bandit model within a stationary period S. Based on Theorem 7, one can prove that $R_{\text{Lin}}(S) \le \sqrt{dS\log(\lambda + \frac{S}{d})}\Big(\sigma^2\sqrt{d\log(1+\frac{S}{\lambda\delta_1})} + \sqrt{\lambda}\Big)$ [37]. In the following, we provide an upper regret bound analysis for the basic version of dLinUCB, in which the size of the admissible slave models is restricted to one (i.e., by setting $\tilde{\delta_1} = \delta_1$).

Theorem 8. *When Assumption 2 is satisfied with $\Delta \ge 2\sqrt{\lambda} + 2\epsilon$, if δ_1 and τ in Algorithm 5 are set according to Lemma 10, and δ_2 is set to $\delta_2 \le \frac{1}{2S_{max}}$, with probability at least $(1-\delta_1)(1-\delta_2)(1-\frac{\delta_2}{1-\delta_2})$, the accumulated regret of dLinUCB satisfies,*

$$\boldsymbol{R}(T) \le 2\Gamma_T R_{Lin}(S_{max}) + \Gamma_T(\tau + \frac{4}{1-\delta_2}) \tag{3.8}$$

where S_{max} is the length of the longest stationary period up to T.

This theorem shows that the order of regret upper bound of dLinUCB is $O(\Gamma_T\sqrt{S_{\max}}\log S_{\max})$, which is the best upper regret bound a bandit algorithm can achieve in such a non-stationary environment [39], and it matches the lower bound up to a $\Gamma_T \log S_{\max}$ factor.

Lemma 7 (Bound the probability of early detection)**.** *For $\delta_2 \in (0,1)$ and any slave model in Algorithm 5,*

$$p_e = \mathbb{P}[\hat{e}_t(m) > \delta_1 + d_t(m) | \textit{stationary in past } \tilde{\tau}(m) \textit{rounds}] \le \delta_2.$$

The intuition behind Lemma 9 is that when the environment is stationary, the 'badness' of a slave model m should be small and bounded according to Eq (3.6).

Lemma 8 (Bound the probability of late detection)**.** *When the magnitude of change in the environment in Assumption 2 satisfies* $\Delta > 2\sqrt{\lambda}+2\epsilon$*, and the shortest stationary period length* S_{min} *satisfies* $S_{min} > \frac{\sqrt{\lambda}}{2\rho}(\Delta - 2\sqrt{\lambda} - 2\epsilon)$*, for any* $\delta_2 \in (0,1)$*, if* δ_1 *and* τ *in Algorithm 5 are set to* $\delta_1 \leq 1 - \frac{1}{\rho}\big(1 - \frac{\sqrt{\lambda}}{2S_{min}\rho}(\Delta - 2\sqrt{\lambda} - 2\epsilon)\big)$ *and* $\tau \geq \frac{2\ln\frac{2}{\delta_2}}{(\rho(1-\delta_1)-\delta_1)^2}$*, for any slave model* m *in Algorithm 5, we have,*

$$p_d = \mathbb{P}\big(\hat{e}_t(m) > \delta_1 + d_t(m) | \textit{ changed within past } \tilde{\tau}(m)\big) \geq 1 - \delta_2.$$

The intuition behind Lemma 10 is that when the environment has changed, with a high probability that Eq (3.7) will not be satisfied in an out-of-date slave model. It means that we will accumulate larger badness from this slave model. In both Lemma 9 and 10, δ_2 is a parameter controlling the confidence of the 'badenss' estimation in Chernoff Bound; and therefore an input to the algorithm.

Remark 2 (Discussion about how the environment assumption affect dLinUCB)**.** *1. The magnitude of environment change* Δ *affects whether a change is detectable by our algorithm. However, we need to emphasize that when* Δ *is very small, the additional regret from re-using an out-of-date slave model is also small. In this case, a similar scale of regret bound can still be achieved, which will be briefly proved in Appendix. 2. We require the shortest stationary period length* $S_{min} > \max\{\frac{\sqrt{\lambda}}{2\rho}(\Delta - 2\sqrt{\lambda} - 2\epsilon), \tau\}$*, which guarantees there are enough observations accumulated in a slave model to make an informed decision. 3. The portion of changed arms* ρ *will affect the probability of achieving our derived regret bound, as we require* $\delta_1 \leq 1 - \frac{1}{\rho}\big(1 - \frac{\sqrt{\lambda}}{2S_{min}\rho}(\Delta - 2\sqrt{\lambda} - 2\epsilon)\big)$*.* ρ *also interacts with* S_{min} *and* τ*: when* ρ *is small, more observations are needed for a slave model to detect the potential changes. The effect of* ρ *and* S_{min} *will also be studied in empirical evaluations.*

Remark 3 (Generalization of dLinUCB)**.** *Our theoretical analysis confirms that any contextual bandit algorithm can be used as the slave model in dLinUCB, as long as the its reward estimation error is bounded with a high probability, which corresponds* $B_t(m,a)$ *in Eq* (3.7)*.*

3.4 Dynamic Ensemble of Bandits

Existing bandit algorithms address the piecewise stationary environment by either introducing a forgetting mechanism to down weight historical observations [39,40], or creating a bandit model for each newly detected stationary period as that in dLinUCB and [41,42,79]. These strategies, nevertheless, failed to recognize an important property of this non-stationary environment: the changes of user interests can be *context dependent*. Even though one's interest could change frequently, his/her preference for a particular type of item might be stable over a longer period of time. For example, in the news recommendation scenario mentioned above, users' preferences over sports news may change with sports seasons. However, at the same time, their preferences over political news may stay stationary, independent of the sports season.

This phenomenon can be verified in Figure 3.2, where we collect real-time click-through-rate (CTR) of four sample news articles from the public Yahoo front-page user click log [2,3] over a week's period. In the figure, each point denotes average CTR over 4000 log records. We can clearly observe the changes of user interest on article 2 at time t_3 and on article 3 at time t_1, t_2 and t_4 in this period. In the meanwhile, the CTR of article 1 and 4 remain quite stable, i.e., the expected reward for recommending these two articles remain unchanged. As a result, the experience recorded in old models can still be used to make accurate reward estimations for these two articles. On the contrary, strategies that discount observations or abandon the "old" models must regain confidence in their newly estimated parameters, which may lead to higher regret due to redundant explorations. The key is thus to recognize the reward change in a per-arm basis regarding the context.

To capitalize on such a unique property of the changing environment, we develop our contextual bandit algorithm that adapts its arm selection and model update strategy concerning users' interest changes in a context-dependent manner. Our solution consists of a dynamic set of contextual bandit models, collectively

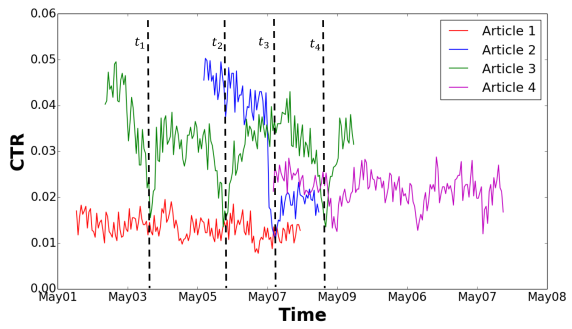


Figure 3.2: Real-time click-through-rate of four sample news articles collected from Yahoo user click log dataset [2, 3].

referred to as *bandit experts*, which are maintained to estimate the underlying reward distribution. Meanwhile, we monitor each bandit expert's reward estimation quality regarding specific context through another bandit model, referred to as *bandit auditors*. The auditor predicts whether a bandit expert is 'admissible' to a particular arm under the given context. At each round of interaction, an ensemble of admissible bandit experts is created to estimate the reward of each arm; and the arm with the highest upper confidence bound of estimated reward will be selected. The acquired feedback is used to update all admissible bandit experts for this chosen arm and their corresponding auditors. When no admissible bandit expert exists, a new bandit expert will be created and added to the set before evaluating the recommendation candidates.

3.4.1 Context-Dependent Changes in the Piecewise Stationary Environment

Under the piecewise stationary environment described in Section 3.2, we capitalize on the unique property that, in a contextual bandit setting, the changes of reward distribution are context-dependent. Thus we categorize arms into two types, *change-invariant* and *change-sensitive*, between any two stationary periods. For stationary periods i and j with their ground-truth bandit parameters $\boldsymbol{\theta}_i^*$ and $\boldsymbol{\theta}_j^*$, the arm that satisfies $|\mathbf{x}_a^\mathsf{T}\boldsymbol{\theta}_i^* - \mathbf{x}_a^\mathsf{T}\boldsymbol{\theta}_j^*| \leq \Delta_L$ (with $\Delta_L > 0$) is referred as a change-invariant arm; otherwise as a change-sensitive arm. Δ_L is a parameter to introduce flexibility and accommodate stochastic noise, which relaxes the requirement that the change has to be completely orthogonal to the context vector of a change-invariant arm. This makes our problem setting more general than those in [42, 77]. An illustration of these two types of arms is provided in Figure 3.3: when the ground-truth bandit parameter changes from $\boldsymbol{\theta}_c^*$ to $\boldsymbol{\theta}_{c+1}^*$, although there are *change-sensitive arms*, for example arm a_1, whose expected reward changes dramatically, there are also *change-invariant arms*, for example arm a_2, whose context vectors are orthogonal to the change, i.e., $\mathbf{x}_a^\mathsf{T}(\boldsymbol{\theta}_c^* - \boldsymbol{\theta}_{c+1}^*) = 0 < \Delta_L$, so that their expected reward does not change significantly even after the environment changes. For example, mapping it back to our previous news recommendation example, if the reward change in a user was caused by the change of sports season, the sports news and political news could be considered as the change-sensitive and change-invariant arms respectively for this user. To differentiate the actual reward change from stochastic noise, we impose the following assumption about the non-stationary environment, which characterizes the detectability of reward changes between the stationary period i and j,

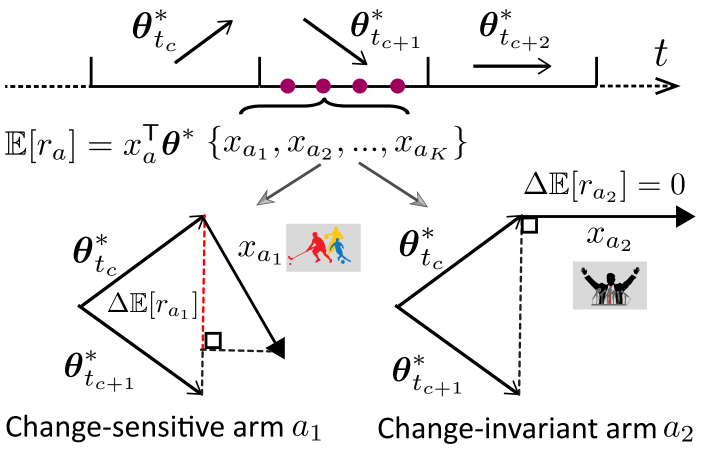
Assumption 2. *Among the change-sensitive arms between stationary period i and j, there are at least ρ portion of them satisfying $|\mathbf{x}_a^\mathsf{T}\boldsymbol{\theta}_i^* - \mathbf{x}_a^\mathsf{T}\boldsymbol{\theta}_j^*| > \Delta_H$.*

This assumption requires that between any two stationary periods, there are a number of change-sensitive arms undergo perceivable reward changes, which differentiate these two periods.

3.4.2 Dynamic Ensemble of Bandit Algorithm

In the non-stationary environment specified above, where the changes in users' interests occasionally happen at unknown time points, new bandit models should be rebuilt in accordance with the changes of underlying user interest. In addition, the possible existence of change-invariant arms urges us to reuse the bandit models estimated for those earlier periods, so that more accurate reward estimation on such arms can be achieved sooner so as to obtain reduced regret in a new stationary period.

In order to achieve these goals, three challenges have to be addressed: (1) as the changes of the environment are unknown to the learner, how to detect the potential change of user interests, and create new bandit models



$\boldsymbol{\theta}^*_{t_c}$
$\boldsymbol{\theta}^*_{t_{c+1}}$
$\boldsymbol{\theta}^*_{t_{c+2}}$
t
$\mathbb{E}[r_a] = x_a^\top \boldsymbol{\theta}^*$ $\{x_{a_1}, x_{a_2}, ..., x_{a_K}\}$
$\Delta\mathbb{E}[r_{a_2}] = 0$
x_{a_1}
$\Delta\mathbb{E}[r_{a_1}]$
x_{a_2}
Change-sensitive arm a_1
Change-invariant arm a_2

Figure 3.3: An illustrative example of context-dependent changes in a piecewise stationary environment. A linear reward assumption is postulated to simplify the illustration.

to account for the change-sensitive arms in a new environment? (2) as the reward changes in each arm are context-dependent, how to recognize the change-invariant arms at the current period such that experience from old models can be fully utilized? And (3) which arm to choose given multiple bandit models might exist at the same time? To avoid any potential ambiguity in our later discussions, we refer to the contextual bandit models created for reward estimation as *bandit experts*.

We address the first two challenges by creating a companion bandit model for each bandit expert to monitor its reward estimation quality. We refer to this companion bandit model as a *bandit auditor*. In a nutshell, a bandit expert, who works with the context features and reward collected from its chosen arms, is responsible for reward estimation regarding its corresponding environment. Its companion bandit auditor works with the context features of the expert's choices and the observed prediction errors from the bandit expert. The auditor is responsible for assessing the expert's prediction accuracy. At each round, every bandit auditor evaluates whether the monitored bandit expert is *admissible* to make an accurate reward estimation for a given arm, with respect to potential changes in the environment. A bandit expert being identified as admissible to a particular arm indicates with a high probability that, either 1) no change has happened since the creation of this bandit expert, or 2) the environment has changed, but the arm is change-invariant between this bandit expert's estimated reward distribution and the current period's underlying reward distribution. This addresses the second challenge. When no admissible bandit expert exists for a given arm, it is thus highly likely that a change has happened, and the arm is a change-sensitive arm. This can be considered as an indicator that a new bandit expert is needed, which addresses the first challenge.

To address the third challenge, at each round of the interactions, following the principle of optimism in the face of uncertainty [2, 37], an arm is chosen by the upper confidence bound of reward estimation based on an ensemble of all its admissible bandit experts. The acquired feedback for the selected arm/item is used to update all corresponding bandit experts and their auditors. We name the resulting bandit algorithm as Dynamic Ensemble of Bandit Experts, or DenBand in short. We describe DenBand in Algorithm 6 and discuss the key components of it in details as follows.

Bandit Expert. Define t_m as the time when bandit expert m is created. Each bandit expert m maintains an estimated bandit parameter $\hat{\boldsymbol{\theta}}_t(m)$ for the stationary period at t_m. Define $\mathcal{I}^\theta_t(m)$ as a set of timestamps when observation $(\mathbf{x}_{a_i}, r_{a_i,i})$ is assigned to the bandit expert m for model update till time t (in line 24-25 of Algorithm 6). $r_{a_i,i}$ is the observed reward on arm a_i at time i. Because of our linear reward structure, $\hat{\boldsymbol{\theta}}_t(m)$ can be readily estimated by $\hat{\boldsymbol{\theta}}_t(m) = \mathbf{A}_t^{-1}(m)\mathbf{b}_t(m)$, in which $\mathbf{A}_t(m) = \lambda\mathbf{I} + \sum_{i\in\mathcal{I}^\theta_t(m)} \mathbf{x}_{a_i}\mathbf{x}_{a_i}^\top$, $\mathbf{I}$ is a $d \times d$ identity matrix, λ is the regularization coefficient in the least square regression, and $\mathbf{b}_t(m) = \sum_{i\in\mathcal{I}^\theta_t(m)} \mathbf{x}_{a_i} r_{a_i,i}$.

Bandit Auditor. Denote the reward estimation error of bandit expert m on arm a at time t as $e_{a,t}(m) = \hat{r}_{a,t}(m) - r_{a,t}$, in which $\hat{r}_{a,t}(m) = \mathbf{x}_a^\top \hat{\boldsymbol{\theta}}_t(m)$. We have $\mathbb{E}[e_{a,t}(m)] = \mathbf{x}_a^\top(\hat{\boldsymbol{\theta}}_t(m) - \boldsymbol{\theta}^*_t)$, which is referred as 'badness' of bandit expert m on arm a at time t and leads to a linear structure for badness estimation. We

Algorithm 6 Dynamic Ensemble of Bandit Experts (DenBand)

```
1: **Inputs:** $\alpha \in \mathbb{R}_+, \lambda > 0, \delta_1, \delta_2 \in (0,1), \tau, \Delta_L$
2: **Initialize:** Create and initialize bandit expert $m$: $\mathbf{A}_1(m) = \lambda\mathbf{I}$, $\mathbf{b}_1(m) = \mathbf{0}$, $\hat{\boldsymbol{\theta}}_1(m) = \mathbf{0}$, and its auditor: $\mathbf{C}_1(m) = \lambda\mathbf{I}$, $\mathbf{d}_1(m) = \mathbf{0}$, $\hat{\boldsymbol{\beta}}_1(m) = \mathbf{0}$. Initialize the bandit expert set $\mathcal{M}_1 = \{m\}$
3: **for** $t = 1$ to $T$ **do**
4:     **for** $a \in \mathcal{A}_t$ **do**
5:         Create an admissible model set for arm $a$: $\mathcal{M}_t^a = \emptyset$
6:         **for** $m \in \mathcal{M}_t$ **do**
7:             $\hat{e}_{a,t}(m) = \mathbf{x}_a^\mathsf{T}\hat{\boldsymbol{\beta}}_t(m)$
8:             Compute $B^{\beta}_{a,t}(m)$ and $B^{\theta}_{a,t}(m)$ by Eq (3.9) and (3.10)
9:             **if** $|\hat{e}_{a,t}(m)| < B^{\theta}_{a,t}(m) + B^{\beta}_{a,t}(m) + \Delta_L$ **then**
10:                Add $m$ into $\mathcal{M}_t^a$
11:        **if** $|\mathcal{M}_t^a| = 0$ **then**
12:            Create and initialize a new bandit expert $m$ and its auditor as in Line 2; Add $m$ to $\mathcal{M}_t^a$ and $\mathcal{M}_t$
13:        Compute UCB$_{a,t}$ of arm $a$ with bandit experts in $\mathcal{M}_t^a$
14:    Select an arm by $a_t = \arg\max_{a\in\mathcal{A}} \text{UCB}_{a,t}$
15:    Observe reward $r_{a_t,t}$
16:    **for** $m \in \mathcal{M}_t^{a_t}$ **do**
17:        $\hat{r}_{a_t,t}(m) = \mathbf{x}_{a_t}^\mathsf{T}\hat{\boldsymbol{\theta}}_t(m)$, $e_{a_t,t}(m) = \hat{r}_{a_t,t}(m) - r_{a_t,t}$
18:        Update $\{e_{a_i,i}\}_{i\in\mathcal{I}_t^{\beta}(m)}$ according to $\hat{\boldsymbol{\theta}}_t(m)$, and keep the size of $\mathcal{I}_t^{\beta}(m)$ to $\tau$
19:        $\mathbf{C}_{t+1}(m) = \mathbf{C}_t(m) + \mathbf{x}_{a_t}\mathbf{x}_{a_t}^\mathsf{T}$, $\mathbf{d}_{t+1}(m) = \sum_{i\in\mathcal{I}_t^{\beta}(m)} \mathbf{x}_{a_i}e_{a_i,i}$
20:        **if** $m \in \mathcal{M}_t^{a_t}$ and $|\hat{r}_{a_t,t}(m) - r_{a_t,t}| \le B^{\theta}_{a_t,t}(m) + \Delta_L + \epsilon$ **then**
21:            $\mathbf{A}_{t+1}(m) = \mathbf{A}_t(m) + \mathbf{x}_{a_t}\mathbf{x}_{a_t}^\mathsf{T}$, $\mathbf{b}_{t+1}(m) = \mathbf{b}_t(m) + \mathbf{x}_{a_t}r_{a_t,t}$
22:        **else**
23:            $\mathbf{A}_{t+1}(m) = \mathbf{A}_t(m)$, $\mathbf{b}_{t+1}(m) = \mathbf{b}_t(m)$
24:        $\hat{\boldsymbol{\theta}}_{t+1}(m) = \mathbf{A}_{t+1}(m)^{-1}\mathbf{b}_{t+1}(m)$
25:        $\hat{\boldsymbol{\beta}}_{t+1}(m) = \mathbf{C}_{t+1}(m)^{-1}\mathbf{d}_{t+1}(m)$
```

create a new bandit model with the target parameter for estimation as $\boldsymbol{\beta}_t^*(m) = \hat{\boldsymbol{\theta}}_t(m) - \boldsymbol{\theta}_t^*$, and refer to it as the bandit auditor of bandit expert m. We maintain and update the bandit auditors in a similar manner as that in bandit experts. Denote $\mathcal{I}_t^{\beta}(m)$ as a set of timestamps when observation $(\mathbf{x}_{a_i}, e_{a_i,i}(m))$ is assigned to the bandit auditor for bandit expert m up to time t (line 21-23 in Algorithm 6). The bandit auditor estimates $\boldsymbol{\beta}_t^*(m)$ by $\hat{\boldsymbol{\beta}}_t(m) = \mathbf{C}_t^{-1}(m)\mathbf{d}_t(m)$, in which $\mathbf{C}_t(m) = \lambda\mathbf{I} + \sum_{i\in\mathcal{I}_t^{\beta}(m)} \mathbf{x}_{a_i}\mathbf{x}_{a_i}^\mathsf{T}$ and $\mathbf{d}_t(m) = \sum_{i\in\mathcal{I}_t^{\beta}(m)} \mathbf{x}_{a_i}e_{a_i,i}(m)$. Intuitively, the bandit auditor for expert m evaluates whether an arm a at time t is change-invariant to the reward distributions specified by $\boldsymbol{\theta}_t^*$ and $\boldsymbol{\theta}_{t_m}^*$. The definition of badness requires us to update $e_{a,t}(m)$ of all observations in $\mathcal{I}_t^{\beta}(m)$ whenever $\hat{\boldsymbol{\theta}}_t(m)$ is updated or $\boldsymbol{\theta}_t^*$ is changed. But as the environment change is unknown to the learner, this update is infeasible. We decide to only accumulate the most recent τ observations in $\mathcal{I}_t^{\beta}(m)$ for auditor update, and prove this still provides us a high probability bound of each bandit auditor's badness estimation in our regret analysis,

$$|\hat{e}_{a,t}(m) - \mathbb{E}[e_{a,t}(m)]| \le B^{\beta}_{a,t}(m) \tag{3.9}$$

where $B^{\beta}_{a,t}(m) = \big(\sigma^2\sqrt{d\ln(\frac{\lambda+|\mathcal{I}_t^{\beta}(m)|}{\lambda\delta_2})} + \sqrt{\lambda}\big)\|\mathbf{x}_a\|_{\mathbf{C}_t^{-1}(m)}$.

Bandit Expert Selection. Based on Eq (3.9) and our badness definition, we have $|\hat{e}_{a,t}(m)| \le \mathbb{E}[e_{a,t}(m)] + B^{\beta}_{a,t}(m) \le |\mathbf{x}_a^\mathsf{T}\hat{\boldsymbol{\theta}}_t(m) - \mathbf{x}_a^\mathsf{T}\boldsymbol{\theta}_{t_m}^*| + |\mathbf{x}_a^\mathsf{T}\boldsymbol{\theta}_{t_m}^* - \mathbf{x}_a^\mathsf{T}\boldsymbol{\theta}_t^*| + B^{\beta}_{a,t}(m)$. The first part on the right-hand side of this inequality is the reward estimation quality of bandit expert m, which can be bounded by,

$$|\mathbf{x}_a\hat{\boldsymbol{\theta}}_t(m) - \mathbf{x}_a\boldsymbol{\theta}_{t_m}^*| \le B^{\theta}_{a,t}(m,a) \tag{3.10}$$

where $B^{\theta}_{a,t}(m,a) = \Big(3\sigma^2\sqrt{d\ln(\frac{\lambda+|\mathcal{I}_t^{\theta}(m)|}{\lambda\delta_1})} + \sqrt{\lambda}\Big)\|\mathbf{x}_a\|_{\mathbf{A}_t^{-1}(m)}$. We should note that this bound is different from those for classical linear bandit algorithms (e.g., [37]), since it also has to account for the possible contamination from change-invariant arms (as we do not restrict Δ_L to zero). The second part on the right-hand side can be bounded by Δ_L, if no change has happened since t_m or the arm a is change-invariant between t_m and t. As a result, the condition $|\hat{e}_{a,t}(m)| < B^{\theta}_{a,t}(m) + B^{\beta}_{a,t}(m) + \Delta_L$ in line 9 of Algorithm 6 determines if the bandit expert m is admissible to arm a at time t (supported by Lemma 9 and 10 in Section 3.4.3). When there is no admissible bandit expert for arm a, a new bandit expert needs to be created to account for the detected change of environment (line 13-15 in Algorithm 6).

Arm Selection. We appeal to the Upper Confidence Bound (UCB) principle [2, 37] to select an arm from the candidate arm pool (line 18 of Algorithm 6). With the above bandit expert selection strategy, for arm a at time t, we might collect a set of admissible bandit experts $\mathcal{M}^a_t$, and each admissible bandit expert m gives us an upper confidence bound of reward estimation for arm a: $\text{UCB}_{a,t}(m) = \mathbf{x}_a^{\mathsf{T}}\hat{\boldsymbol{\theta}}_t(m) + B^{\theta}_{a,t}(m)$. Therefore, we need to integrate $\text{UCB}_{a,t}(m)$ from multiple bandit experts (line 16 of Algorithm 6). We propose two strategies for this purpose, each of which has its own advantages; but both of them lead to the same upper regret bound (proved in Theorem 10 of Section 3.4.3).

Option 1: Average ensemble. For each arm, as in principle every admissible bandit expert is 'legitimate' to give a reward prediction for this arm, we compute an average upper confidence bound of reward based on all admissible expert models: $\text{UCB}_{a,t} = \frac{1}{|\mathcal{M}^a_t|}\sum_{m\in\mathcal{M}^a_t}\text{UCB}_{a,t}(m)$. This ensemble helps reduce variance in reward estimation among the admissible bandit experts, but might be disturbed by some least qualified bandit experts.

Option 2: Lower confidence bound of badness. Although every admissible bandit expert is guaranteed to have small enough badness to make an accurate reward prediction on the chosen arm, the uncertainty of their auditors' badness estimation may differ, which introduces another level of trade-off between exploitation and exploration in bandit expert selection. By applying the Lower Confidence Bound (LCB) principle (as we want to minimize the badness of chosen bandit experts), we select a bandit expert from $\mathcal{M}^a_t$ by the LCB of its auditor's estimated badness: $\tilde{m}_{a,t} = \arg\min_{m\in\mathcal{M}^a_t}\big(|\hat{e}_{a,t}(m)| - B^{\beta}_{a,t}(m)\big)$, and compute the UCB of arm a using the selected bandit expert $\tilde{m}_{a,t}$.

Model Update. Once the feedback $(\mathbf{x}_{a_t}, r_{a_t})$ is obtained from the environment on the selected arm a_t, we update the bandit experts and auditors in this arm's admissible model set (line 20-31 in Algorithm 6). In particular, we compare the acquired feedback against each bandit expert's estimation (line 24) to decide whether we should update the bandit expert to improve its reward estimation or the bandit auditor to improve its badness estimation. This decision comes from two factors that cause the observed reward estimation error: 1) large noise from the environment; 2) the arm is actually not change-invariant. Large noise may happen, but with a very small probability, as it follows a Gaussian distribution. Define $\epsilon = \sqrt{2}\sigma\text{erf}^{-1}(\delta_1 - 1)$, in which σ is the standard deviation of the Gaussian noise in reward feedback, and $\text{erf}(\cdot)$ is the Gauss error function. Violating the condition $|\hat{r}_{a,t}(m) - r_{a,t}| \le B^{\theta}_{a,t}(m) + \Delta_L + \epsilon$ suggests that the chosen arm might not be change-invariant, such that the bandit expert should not be updated but the bandit auditor should be. This selective update strategy helps reduce erroneous observations in both bandit experts and auditors.

3.4.3 Regret Analysis

We first provide high probability bounds of bandit experts' reward estimation and bandit auditor's badness estimation in the following theorem.

Theorem 9 (Confidence Bounds). *We define S_{min} as the length of the shortest stationary period up to time T and $t_c(m)$ as the first change point in the environment after bandit expert m is created. When Assumption 2 is satisfied with $\Delta_H > 4\sqrt{\lambda} + \Delta_L$ and $\Delta_L \le \frac{\sigma^2\sqrt{d\lambda\ln\frac{\lambda+T}{\lambda\delta_1}}}{T}$, and $1 < \tau \le S_{min}\frac{\sigma^2\sqrt{d\lambda\ln\frac{\lambda+T}{\lambda\delta_1}}}{2\rho T}$, at time t for any $\delta_1 \in (0,1), \delta_2 \in (0,1)$ and $\delta_3 = 1 - (1-\delta_1)\big[(1-\delta_2)(1-\delta_1)\rho\big]^{\max\{t-t_c(m),0\}}$, with a probability at least $1-\delta_3$, for any arm a all the bandit experts in Algorithm 6 satisfy,*

$$|\mathbf{x}_a^{\mathsf{T}}\hat{\boldsymbol{\theta}}_t(m) - \mathbf{x}_a^{\mathsf{T}}\boldsymbol{\theta}^*_{t_m}| \le B^{\theta}_{a,t}(m,a) \tag{3.11}$$

If there is no environment change in the past τ iterations, with a probability at least $1-\delta_2$, all bandit auditors at time t satisfy,

$$|\mathbf{x}_a^\mathsf{T}\hat{\boldsymbol{\beta}}_t(m) - \mathbf{x}_a^\mathsf{T}\boldsymbol{\beta}_t^*| \le B_{a,t}^{\beta}(m,a) \tag{3.12}$$

And with a probability at least $1-\delta_3$, all the selected bandit experts in $\mathcal{M}_t^a$ for arm a satisfy,

$$|\mathbf{x}_a^\mathsf{T}\hat{\boldsymbol{\theta}}_t(m) - \mathbf{x}_a^\mathsf{T}\boldsymbol{\theta}_t^*| \le \Delta_L + B_{a,t}^{\theta}(m) \tag{3.13}$$

This theorem specifies the threshold of minimum reward change for change-sensitive arms, i.e., Δ_H, which decides whether a change is detectable by our algorithm. We can further relax the threshold to $2B_{a,t}^{\theta}(m) + 2B_{a,t}^{\beta}(m) + \Delta_L$, which is shrinking over time and related to the current model uncertainties of the bandit expert and auditor. This would allow us to recognize more subtle reward changes across different stationary periods so as to improve the model estimation quality on the fly. On the other hand, Δ_L is the threshold deciding whether an arm is change-invariant with respect to two stationary periods. It is thus the resolution of our bandit experts in recognizing the ground-truth bandit parameters of their designated periods. Note, Δ_L is a parameter of the environment, rather than a parameter of our model. The parameter τ determines the number of most recent observations used for estimating the bandit auditors. Although a larger τ naturally leads to better badness estimation quality in the auditors, it cannot be arbitrarily large as it brings in out-of-date observations to the auditors. Theorem 9 imposes an upper bound of τ to guide the practical use of our algorithm.

In the following two lemmas, we bound the probability of false-negative and false-positive selection of bandit experts, which proves the validity of our designed bandit expert selection strategy.

Lemma 9. *A false negative selection happens when the bandit expert m is not selected in its designated period or for the truly change-invariant arms to it in other periods. Denote the probability of a false negative selection as P_{FN}. At time t, we have $P_{FN} = \mathbb{P}\big(|\hat{e}_{a,t}(m)| > B_{a,t}^{\theta}(m) + B_{a,t}^{\beta}(m) + \Delta_L \mid |\mathbf{x}_a^\mathsf{T}\boldsymbol{\theta}_t^* - \mathbf{x}_a^\mathsf{T}\boldsymbol{\theta}_{t_m}^*| \le \Delta_L\big) \le (1-\delta_2)(1-\delta_3)$, in which δ_2 and δ_3 are defined in Theorem 9.*

Lemma 10. *A false positive selection happens when the environment has changed and the current arm is change-sensitive to a particular bandit expert m, but the bandit expert is mistakenly selected. Denote the probability of a false negative selection as P_{FP}. When Assumption 2 holds and $\Delta_H > 4\sqrt{\lambda} + \Delta_L$, we have at time $t > t_c(m)$, $P_{FP} = \mathbb{P}\big(|\hat{e}_{a,t}(m)| \le B_{a,t}^{\theta}(m) + B_{a,t}^{\beta}(m) + \Delta_L \mid |\mathbf{x}_a^\mathsf{T}\boldsymbol{\theta}_t^* - \mathbf{x}_a^\mathsf{T}\boldsymbol{\theta}_{t_m}^*| > \Delta_L\big) \le 1 - \big((1-\delta_1)(1-\delta_2)\rho\big)^{t-t_c(m)}$, in which δ_1 and δ_2 are defined in Theorem 9.*

Intuitively, a false negative selection of bandit experts happens when both the reward and badness estimations on a change-invariant arm exceed their confidence bounds; and a false positive selection happens when a change-sensitive arm undergoes substantial reward change but both the expert's reward estimation and the auditor's badness estimation still stay within their confidence bounds. Putting all these analyses together, we have the following regret bound for our proposed algorithm.

Theorem 10 (Regret bound of DenBand)**.** *Under the same condition as stated in Theorem 9, and $(\delta_2 + \delta_3 - \delta_2\delta_3) \le \frac{1}{2S_{max}}$, using any bandit expert in $\mathcal{M}_t^a$ for UCB-based arm selection guarantees that with a probability at least $(1-\delta_3)(1-\delta_2)$ the expected accumulated regret of DenBand up to time T can be bounded as,*

$$R(T) \le 2\Gamma_T\Big(3\sigma^2\sqrt{d\ln\frac{\lambda+U_T}{\lambda\delta_1}} + \sqrt{\lambda}\Big)\sqrt{S_{max}d\ln(\lambda + \frac{S_{max}}{d})} + (\frac{1}{\rho}+2)\sigma^2\sqrt{d\lambda\ln\frac{\lambda+T}{\lambda\delta_1}}$$

where S_{max} is the length of the longest stationary period up to time T, Γ_T is the total number of environment changes till T, and $U_T = \max\{|\mathcal{I}_T^{\theta}(m)|\}_{m\in\mathcal{M}_T}$.

U_T is the maximum number of updates taken among any of the bandit experts, and it is clearly smaller than T. Hence the accumulated regret is in the order of $\mathcal{O}(\Gamma_T\sqrt{S_{\max}\ln T\ln S_{\max}})$, which is arguably a very tight upper regret bound without any further assumption about the environment.

Theorem 10 provides a general upper regret bound of our DenBand in the order of $\mathcal{O}(\Gamma_T\sqrt{S_{\max}\ln T\ln S_{\max}})$. To better interpret this upper regret bound under different circumstances, we consider the following two special environment cases. **Case 1:** When the changes are evenly (or almost evenly) distributed, i.e., $S_{\max} = S = \frac{T}{\Gamma_T}$, the resulting regret bound can be rewritten as $\mathcal{O}(\sqrt{\Gamma_T T}\ln T)$, which matches the general lower regret bound in an abruptly changing environment proved in [39] without further assumptions about the environment. **Case 2:** When the distribution of changes are highly unbalanced such that there is a single very long stationary period and many very short stationary periods, by denoting the short periods with a superscript 's', the final upper regret bound can be rewritten as $\mathcal{O}(\sqrt{S_{\max}\ln T\ln S_{\max}} + \Gamma_T\sqrt{S^s_{\max}\ln T\ln S^s_{\max}})$ where $S^s_{\max} \ll S_{\max}$. For example, when $S^s_{\max} \ll \frac{S_{\max}}{\Gamma^2_T}$, the regret can be further upper bounded by $\mathcal{O}(\sqrt{S_{\max}\ln T\ln S_{\max}})$, which matches and is better than the upper regret bound of running a single contextual bandit over the whole period (as $S_{\max} < T$).

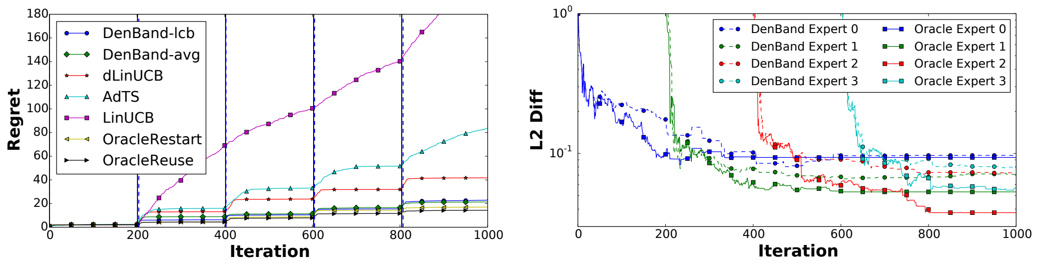
Our regret analysis in those two special cases supports the validity of DenBand, and we can further generalize our analysis of it under other reward assumptions. In particular, our theoretical analysis supports that any contextual bandit algorithm can be used as a bandit expert in DenBand, as long as its reward estimation error is bounded with a high probability, which corresponds to $B^\theta_t(m,a)$ in Eq (3.11). The overall regret of DenBand will only be a factor of the actual number of changes in the environment, which is arguably inevitable without further assumptions about the environment.

3.5 Experiments

We tested dLinUCB and DenBand on a comprehensive set of evaluation datasets, which include a synthetic dataset and three real-world recommendation datasets. We compared dLinUCB and DenBand against the following bandit baselines: LinUCB [2], a reference stationary linear contextual bandit algorithm; AdTS [41], WMDUCB1 [42], and Meta-Bandit [40], which are state-of-the-art bandit algorithms for piecewise stationary environment. For DenBand, we tested two variants of it: DenBand-lcb, which uses lower confidence bound of badness to select a bandit expert, and DenBand-avg, which takes the average reward UCBs from all admissible bandit experts.

Experiments on Synthetic Dataset

• **Simulation Settings.** In simulation, we generate a size-K ($K = 1000$) arm pool $\mathcal{A}$, in which each arm a is associated with a feature vector $\mathbf{x}_a \in \mathbb{R}^d$ with $d = 10$, $\|\mathbf{x}_a\|_2 \le 1$. Similarly, we create a set of ground-truth bandit parameters $\boldsymbol{\theta}^* \in \mathbb{R}^d$ with $\|\boldsymbol{\theta}^*\|_2 \le 1$, which are not disclosed to the learners. The standard deviation of Gaussian noise σ on the reward is set to 0.05 by default. To simulate an abruptly changing environment, after every S rounds, we randomize $\boldsymbol{\theta}^*$ with respect to the constraint that at least ρ portion of arms in $\mathcal{A}$ satisfy $|\mathbf{x}_a^\mathsf{T}\boldsymbol{\theta}^*_{t_{c_j}} - \mathbf{x}_a^\mathsf{T}\boldsymbol{\theta}^*_{t_{c_{j+1}}}| > \Delta_H$. And by default, we set S to 200, Δ_H to 0.7, and ρ to 0.2.



(a) Accumulated regret over time (b) Parameter estimation quality of bandit experts

Figure 3.4: Performance comparison on a synthetic dataset.

• **Empirical Regret Comparisons.** Under this simulation setting, we execute all algorithms 1000 iterations and report their accumulated regret in Figure 3.4 (a). Because LinUCB imposes a stationary assumption about the environment, it suffers almost linearly increasing regret after the first change point. AdTS can react to the environment changes, but it is slow in doing so and thus still tends to accumulate large regret after the changes happen. All of our proposed algorithms, dLinUCB, DenBand-lcb, and DenBand-avg, can quickly identify the changes and create corresponding bandit experts to capture the new reward distributions. The solid and

Table 3.1: Accumulated regret with different settings of the non-stationary environment.

(σ, Δ_H, S, ρ)	(.05, .7, 200, .8)	(.1, .7, 200, .8)	(.05, .7, 400, .8)	(.05, .5, 200, .8)	(.05, .5, 200, .5)	(.05, .5, 200, .2)
DenBand-lcb	**22.48 $\pm$ 3.12**	36.68 $\pm$ 6.64	16.55 $\pm$ 1.48	**23.24 $\pm$ 3.01**	**18.96 $\pm$ 1.96**	22.45 $\pm$ 2.32
DenBand-avg	25.54 $\pm$ 3.88	**33.87 $\pm$ 4.13**	**13.08 $\pm$ 2.12**	23.51 $\pm$ 2.55	20.40 $\pm$ 2.06	**19.87 $\pm$ 1.85**
dLinUCB	35.12 $\pm$ 2.20	54.58$\pm$ 3.56	22.33 $\pm$ 2.83	35.72 $\pm$ 2.05	34.45 $\pm$ 2.83	36.65 $\pm$ 2.10
AdTS	65.16$\pm$ 20.30	67.43 $\pm$ 21.49	38.61 $\pm$ 5.08	70.74 $\pm$ 26.60	58.59 $\pm$ 14.13	52.45 $\pm$ 14.21
LinUCB	253.22 $\pm$ 7.12	263.14 $\pm$ 11.76	257.50 $\pm$ 8.07	216.60 $\pm$ 11.32	206.29 $\pm$ 15.71	164.68 $\pm$ 8.98
WMDUCB1	519.73 $\pm$ 21.53	528.48 $\pm$ 23.63	473.13 $\pm$ 43.29	484.23 $\pm$ 17.73	404.17 $\pm$ 10.95	372.30 $\pm$ 9.68
Meta-Bandit	585.70 $\pm$ 3.72	585.97 $\pm$ 4.62	499.83 $\pm$ 5.14	454.56 $\pm$ 4.29	412.30 $\pm$ 2.68	362.96 $\pm$ 2.17

Table 3.2: Accumulated regret with different hyperparameter configurations.

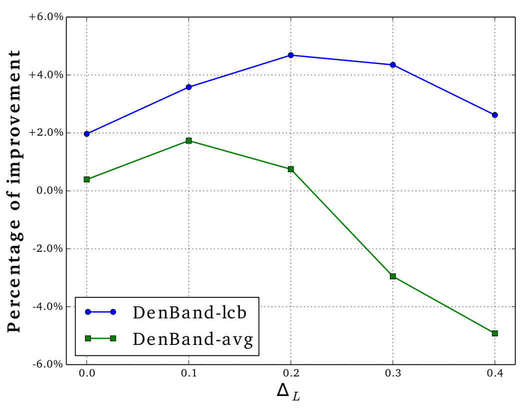
(τ, Δ_L)	(30, 0.0)	(50, 0.0)	(100, 0.0)	(100, 0.025)	(100, 0.05)	(100, 0.1)	(100, 0.15)
DenBand-lcb	24.36 $\pm$ 4.23	22.48 $\pm$ 3.12	18.36 $\pm$ 3.27	19.27 $\pm$ 1.74	19.58 $\pm$ 1.38	21.08 $\pm$ 3.17	23.75 $\pm$ 5.96
DenBand-avg	25.73 $\pm$ 1.53	21.19 $\pm$ 3.29	19.04 $\pm$ 1.65	19.52 $\pm$ 2.10	22.78 $\pm$ 2.19	26.84 $\pm$ 4.03	31.81 $\pm$ 6.08
dLinUCB	35.08 $\pm$ 2.59	35.12 $\pm$ 2.20	42.86 $\pm$ 2.23	-	-	-	-
AdTS	94.90 $\pm$ 13.08	65.16 $\pm$ 20.30	91.31 $\pm$ 20.08	-	-	-	-

dashed vertical lines in Figure 3.4 (a) show the actual and detected change points by DenBand-lcb, respectively. We can clearly notice that DenBand can almost immediately respond to the changes in the environment. To improve visibility of Figure 3.4 (a), WMDUCB and Meta-Bandit are excluded.

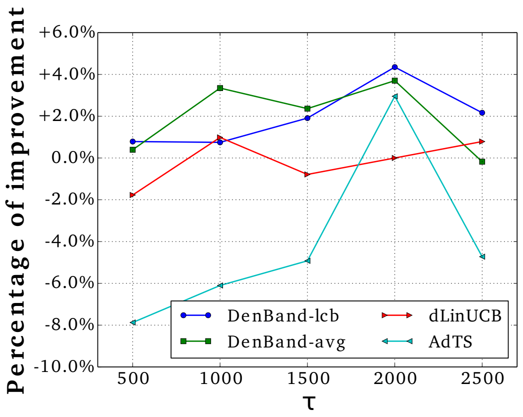
In addition, we also include two oracle algorithms in Figure 3.4 (a). One is to start a new bandit expert at each change point and only uses it in this period, named as *OracleRestart*. The other, named as *OracleReuse*, maintains one bandit expert for each stationary period and uses all ground-truth reusable experts for reward estimation in change-invariant arms. Both dLinUCB and DenBand perform quite closely to such optimal algorithms, although it does not get access to the ground-truth environment changes. Figure 3.4 (b) shows the parameter estimation quality (i.e., the L2 difference between the estimated parameter and the ground truth parameter) for four dynamically created bandit experts in DenBand-lcb. For comparison, we also include the estimation quality of experts from the OracleReuse algorithm. The good estimation quality of each bandit expert in DenBand further verifies our conclusion in Theorem 9 about the confidence bound of admissible bandit experts' estimation quality.

• **Sensitivity to Environment Settings.** According to our regret analysis, the performance of DenBand and dLinUCB depend on the environment settings: including standard deviation σ in the Gaussian noise, length S of the stationary period, proportion ρ of change-sensitive arms (used only in DenBand), and magnitude Δ_H of reward change. We varied them in simulation to investigate their influence on the algorithms. We ran all algorithms for 10 times and reported the mean and standard derivation of obtained accumulated regret in Table 3.1. DenBand consistently achieved the best performance against all baselines in all environment settings. As expected, a larger noise level σ leads to worse regret in almost all algorithms. Since T is fixed in our simulation, a smaller S leads to a larger Γ_T, which linearly scales DenBand's regret. When Δ_H becomes smaller, the regret of the two variants of DenBand and AdTS is further reduced. This is because once Δ_H satisfies the requirement in Theorem 9, the change can be confidently detected. Meanwhile, because of a smaller Δ_H, the added regret from using a false-positive bandit expert becomes smaller. Lastly, ρ does not seriously affect the performance of DenBand, which indicates that the algorithm is robust to ρ as long as Assumption 2 is satisfied.

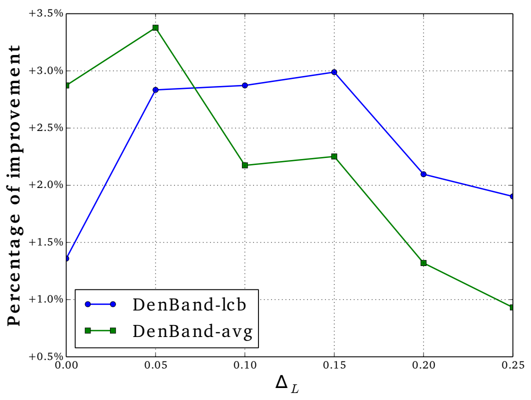
• **Sensitivity to Hyper-Parameters.** To verify the robustness of our proposed algorithm, we studied the effect of two important hyper-parameters in Table 3.2: τ, which determines the number of most recent observations used in bandit auditors, and Δ_L, which introduces flexibility and accounts for the existence of change-invariant arms with infinitesimal reward shift. As a comparison, we also studied the effect of τ on dLinUCB and AdTS, which also use a sliding window for change detection. From Table 3.2, we can find that both variants of DenBand are robust to its parameter τ as long as it falls into the required range. This result verified our theoretical analysis about the role of τ in Theorem 9, which defines an upper and lower bound of τ for the proved confidence bound. On the contrary, the baselines, especially AdTS, are very sensitive to the setting of τ. For the parameter Δ_L, DenBand-lcb is very robust to it, but DenBand-avg is not. This can be explained from two perspectives: 1). As Δ_L accounts for the existence of change-invariant arms with infinitesimal reward shift, it should be upper bounded as required in Theorem 9. 2). When Δ_L is set too large, inadmissible bandit experts may be mistakenly selected. In this case, DenBand-avg suffers from a contaminated average ensemble



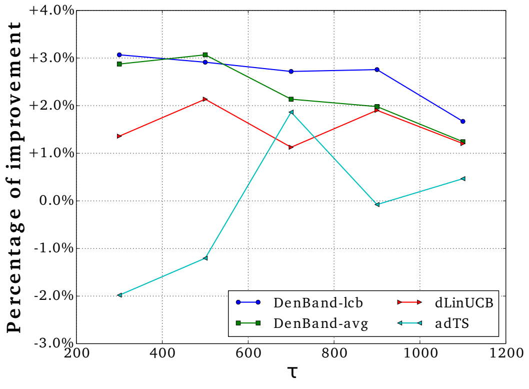
(a) Effect of Δ_L on Yahoo dataset



(b) Effect of τ on Yahoo dataset



(c) Effect of Δ_L on Snapchat dataset



(d) Effect of τ on Snapchat dataset

Figure 3.5: Effect of hyper-parameters on DenBand on two real-world datasets

of bandit experts in reward prediction. DenBand-lcb reduces the risk by selecting a bandit expert by its lower confidence bound of estimated badness.

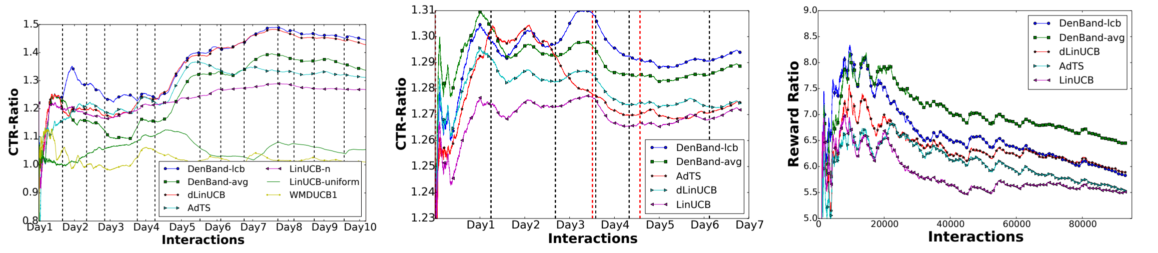
Experiments on Real-World Datasets

• **Datasets.** We tested all algorithms on the following three real-world recommendation datasets.

The first dataset is a large collection of Yahoo frontpage recommendation log, made available by the Yahoo Webscope program [2]. Unbiased offline evaluation is performed on this dataset following the offline evaluation protocol used in [2,3]. The second dataset is a user click log collected from the Snapchat lens recommendation platform over a one-week period. This dataset contains several hundred millions of anonymous observations related to 495 unique lenses and more than five hundred thousand randomly selected users. Each observation contains anonymous user id, a lens id which is recommended by the logging policy, and corresponding user actions on this lens. Each lens is associated with an 8-dimensional feature vector constructed by historical click statistics. The sharing-related actions on items are considered as positive user click feedback. Due to the sparsity of observations, users are grouped into 22 groups according to their demographic information and estimate one bandit model in each user group. The same offline evaluation protocol is used here as in the Yahoo news recommendation dataset.

The third data is LastFM dataset, which is introduced in Section 2.2. And we followed [40] to simulate a non-stationary environment: we ordered observations chronologically inside each user and built a single hybrid user by merging different users. Hence, the boundary between two consecutive batches of observations from two original users is treated as the preference change of the hybrid user.

• **Recommendation Quality** In Figure 3.6 (a), we report normalized Click-Through Rate (CTR) from different algorithms based on the corresponding logged random strategy's CTR on the Yahoo dataset. We set τ to 2,000 and Δ_L to 0.3 based on the hyper-parameter tuning results. From Figure 3.6 (a), we can find that DenBand-lcb



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

Figure 3.6: Normalized CTR comparison on three real-world datasets.

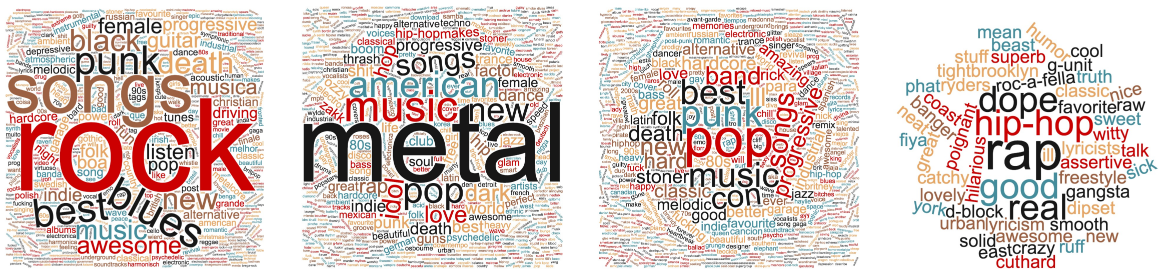


Figure 3.7: Word cloud of tags from four identified user groups in dLinUCB on LastFM dataset.

achieves significant improvement compared with all baselines, especially at the beginning of the testing period. WMDUCB1 performs the worst as it cannot utilize any available context information. We also looked into the detected change points by DenBand-lcb, and found that it detected 25 changes in total, and we plotted 12 of them in Figure 3.6 (a) using vertical lines (to increase the visibility of the figure). We can find a close correspondence between its detected change points and its performance improvement.

Using the same evaluation protocol, we report the CTR ratio between different algorithms and the logging policy on the Snap dataset in Figure 3.6 (b). We set τ to 500 and Δ_L to 0.1. On this dataset, the context-free algorithms perform significantly worse than the contextual ones, so that we excluded them from the comparisons. The results show that our algorithms achieve a 29% - 31% improvement comparing to the logging policy and a 3.1% improvement against LinUCB in a per-user basis. Comparing to AdTS, although our proposed algorithm performs similarly at the beginning, it caught up very quickly later. We plot the detected change points for the two largest groups of users (users in the same group share the same set of bandit experts) in Figure 3.6 (b) using vertical lines of different colors, which well correlate with performance improvement during the adaptive recommendation process.

The ratio between reward from the bandit algorithms and that from a random selection policy on the LastFM dataset is reported in Figure 3.6 (c), where Δ_L is set to 0.1 and τ to 300. From this result, we can see that DenBand, especially DenBand-avg outperforms all the baselines. LinUCB performs the worst as it failed to capture the non-stationarity of the environment. Since the distribution of items is highly skewed [15], the context-free bandits perform very poorly on this dataset. We, therefore, decided to exclude them from the comparison in the figure.

Qualitative Analysis. To illustrate dLinUCB's effectiveness in change detection and reveal how DenBand recognizes the changes in users' interests in a context-dependent manner, we did a Quantitive Analysis on the LastFM dataset.

For dLinUCB, We qualitatively studied those created slave models to investigate what kind of stationarity they have captured. On the LastFM dataset, each user is associated with a list of tags he/she gave to the artists. The tags are usually descriptive and reflect users' preference for music genres or artist styles. In each slave model, we use all the tags from the users being served by this model to generate a word cloud. Figure 3.8 are four representative groups identified on LastFM, which clearly correspond to four different music genres – rock music, metal music, pop music, and hip-hop music. dLinUCB recognizes those meaningful clusters purely from user click feedback.

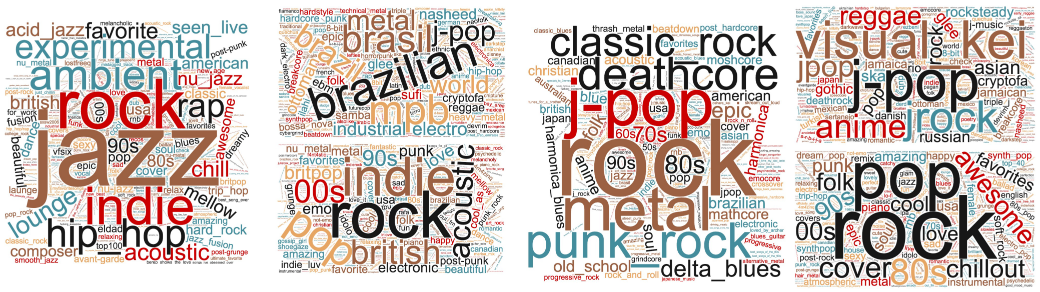


Figure 3.8: Word cloud visualization of two bandit experts and their auditors from DenBand on LastFM. In each of the word cloud groups, the left one shows the tags from high reward items selected by the bandit expert; the upper right one shows tags from change-invariant items, and the lower right one shows tags collected from change-sensitive items according to the bandit auditor, respectively.

We looked into the high reward items, change-invariant items, and change-sensitive items according to our sequentially created bandit experts and their corresponding auditors on the LastFM dataset. For each bandit expert and auditor pair, we used the learned models in the bandit expert to get the top 500 high reward items, and used its auditor to get the top 500 change-invariant items and top 500 change-sensitive items separately. As each item is associated with some short text descriptions provided by the users, we then generated word clouds for each group of those selected items to summarize the learned bandit models in Figure 3.8. It is interesting to find that change-invariant items tend to be related to geographical regions. For example, the change-invariant items in group 1 are mostly about Brazilian music, and those in group 2 are related to Japanese music. The change-sensitive items are mostly related to those more common genres of music. And the high reward items are a mix of these two types. Hence recognizing the changing and stable users' interest is essential in making satisfactory recommendations.

3.6 Unifying Non-stationary Bandit With Clustering of Bandit

As introduced in previous sections of this chapter, in stochastic non-stationary environments, for example, the piecewise stationary environment studied in this dissertation, the reward mapping function becomes time-variant. A working solution needs to either properly discount historical observations [80–82] or detect the change points and reset the model estimation accordingly [83–86] as that in dLinUCB and DenBand. In online clustering of bandits, grouping structures of bandit models are assumed in a population of users, e.g., users in a group share the same bandit model. But instead of assuming an explicit dependency structure, e.g., leveraging existing social network among users [65, 87], online clustering of bandits aim to simultaneously cluster and estimate the bandit models during the sequential interactions with users [45, 52, 88, 89]. Its essence is thus to measure the relatedness between different users' bandit models. Typically, confidence bound of model parameter estimation [45] or reward estimation [88] is used for this purpose.

So far these two problems have been studied in parallel, but the key principles to solve them overlap considerably. On the one hand, mainstream solutions for piecewise stationary bandits detect change points in the underlying reward distribution by comparing the observed rewards [84] or the quality of estimated rewards [83, 85, 86] in a window of consecutive observations. If a change happens in the window, the designed statistics of interest will exceed a threshold with a high probability. This is essentially a sequential hypothesis testing of a model's fitness [90]. On the other hand, existing solutions for the online clustering of bandits evaluate if two bandit models share the same set of parameters [45, 52] or the same reward estimation on a particular arm [88]. This can also be understood as a goodness-of-fit test between models.

In this work, we take the first step to unify these two parallel strands of bandit research under the notion of *test of homogeneity*. We address both problems by testing whether the collection of observations in a bandit model follows the same distribution as that of new observations (i.e., change detection in non-stationary bandits) or of those in another bandit model (i.e., cluster identification in online clustering of bandits). Built upon our solution framework, bandit models can operate on individual users with much stronger flexibility, so that new bandit learning problems can be created and addressed. For example, learning in a clustered non-stationary environment, where the individual models are reset when a change of reward distribution is detected and merged when they are determined as identical. Our rigorous regret analysis and extensive empirical evaluations

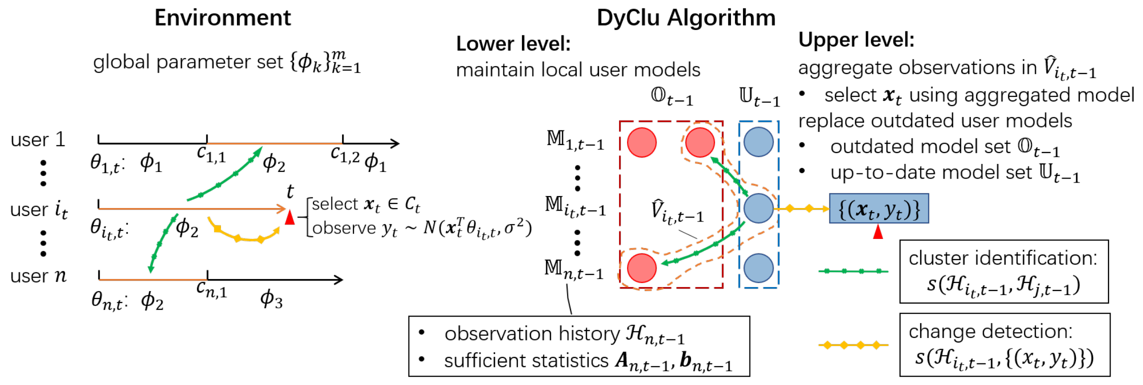

Environment
global parameter set $\{\phi_k\}_{k=1}^m$
user 1
user i_t
user n
$\theta_{1,t}$: ϕ_1 $c_{1,1}$ ϕ_2 $c_{1,2}$ ϕ_1
$\theta_{i_t,t}$: ϕ_2
$\theta_{n,t}$: ϕ_2 $c_{n,1}$ ϕ_3
t
select $x_t \in C_t$
observe $y_t \sim N(x_t^T\theta_{i_t,t}, \sigma^2)$
DyClu Algorithm
Lower level:
maintain local user models
$\mathbb{M}_{1,t-1}$
$\mathbb{M}_{i_t,t-1}$
$\mathbb{M}_{n,t-1}$
$\mathbb{O}_{t-1}$
$\mathbb{U}_{t-1}$
$\hat{V}_{i_t,t-1}$
observation history $\mathcal{H}_{n,t-1}$
sufficient statistics $A_{n,t-1}, b_{n,t-1}$
Upper level:
aggregate observations in $\hat{V}_{i_t,t-1}$
select x_t using aggregated model
replace outdated user models
outdated model set $\mathbb{O}_{t-1}$
up-to-date model set $\mathbb{U}_{t-1}$
$\{(x_t, y_t)\}$
cluster identification: $s(\mathcal{H}_{i_t,t-1}, \mathcal{H}_{j,t-1})$
change detection: $s(\mathcal{H}_{i_t,t-1}, \{(x_t, y_t)\})$

Figure 3.9: Online bandit learning in a non-stationary and clustered environment. The environment setting is shown on the left side of the figure, where each user's reward mapping function undergoes a piecewise stationary process; and the reward mapping functions are globally shared across users. The proposed DyClu algorithm is illustrated on the right side of the figure. The model has a two-level hierarchy: at the lower level, individual users' bandit models are dynamically maintained; and at the upper level, a unified test of homogeneity is performed for the purpose of change detection and cluster identification among the lower-level user models.

demonstrate the value of this unified solution, especially its advantages in handling various assumptions about the environment.

We first formulate the problem setup studied in this work. Then we describe two key components pertaining to non-stationary bandits and online clustering of bandits, and pinpoint the essential equivalence between them under the notion of homogeneity test, which becomes the cornerstone of our unified solution. Based on our construction of the homogeneity test, we explain the proposed solution to the problem, followed by our theoretical analysis of the resulting upper regret bound of the proposed solution.

Problem formulation

To offer a unified approach that addresses the two target problems, we formulate a general bandit learning setting that encompasses both non-stationarity in individual users and existence of clustering structure among users. We consider the contextual bandit problem introduced in Section 1.1 in the following environment: At each time $t = 1, 2, ..., T$, the learner interacts with an arbitrary user indexed by i_t from a set of n users $\mathcal{U} = \{1, ..., n\}$. Denote the set of time steps when user $i \in \mathcal{U}$ is served up to time T as $\mathcal{N}_i(T) = \{1 \le t \le T : i_t = i\}$. Among time steps $t \in \mathcal{N}_i(T)$, user i's parameter $\theta_{i,t}$ changes abruptly at arbitrary time steps $\{c_{i,1}, ..., c_{i,\Gamma_i(T)-1}\}$, but remain constant between any two consecutive change points. $\Gamma_i(T)$ denotes the total number of stationary periods in $\mathcal{N}_i(T)$. The set of unique parameters that $\theta_{i,t}$ takes for any user at any time is denoted as $\{\phi_k\}_{k=1}^m$ and their frequency of occurrences in T is $\{p_k\}_{k=1}^m$. Note that the ground-truth linear parameters, the set of change points, the number and frequencies of unique parameters are unknown to the learner. Moreover, the number of users, i.e., n, and the number of unique bandit parameters across users, i.e., m, are finite but arbitrary.

Our problem setting defined above is general. The non-stationary and clustering structure of an environment can be specified by different associations between $\{\theta_{i,t}\}_{i=1}^n$ and $\{\phi_k\}_{k=1}^m$ across users over time $t = 1, 2, ..., T$. For instance, by setting $n > m$ and $\Gamma_i(T) = 1, \forall i \in \mathcal{U}$, the problem naturally reduces to the online clustering of bandits problem, which assumes sharing of bandit models among users with stationary reward distributions. By setting $n = 1$, $m > 1$ and $\Gamma_i(T) > 1, \forall i \in \mathcal{U}$, it reduces to the piecewise stationary bandits problem, which only concerns a single user with non-stationary reward distributions in isolation.

To make our solution compatible with existing work in non-stationary bandits and clustered bandits, we introduce the following three commonly made assumptions about the environment.

Assumption 3 (Change detectability [85])**.** *For any user $i \in \mathcal{U}$ and any change point c in user i, there exists $\Delta > 0$ such that at least ρ portion of arms satisfy:* $|\mathbf{x}^\top\theta_{i,c-1} - \mathbf{x}^\top\theta_{i,c}| > \Delta$.

Assumption 4 (Separateness among $\{\phi_k\}_{k=1}^m$ [45, 88, 89])**.** *For any two different unique parameters $\phi_i \neq \phi_j$, we have* $\|\phi_i - \phi_j\| \ge \gamma > 0$.

Assumption 5 (Context regularity [45, 88, 89]). *At each time t, arm set C_t is generated i.i.d. from a sub-Gaussian random vector $X \in \mathbb{R}^d$, such that $\mathbb{E}[XX^\top]$ is full-rank with minimum eigenvalue $\lambda' > 0$; and the variance ς^2 of the random vector satisfies $\varsigma^2 \leq \frac{\lambda'^2}{8\ln 4K}$.*

The first assumption establishes the detectability of change points in each individual user's bandit models over time. The second assumption ensures separation within the global unique parameter set shared by all users, and the third assumption specifies the property of context vectors. Based on these assumptions, we establish the problem setup in this work and illustrate it on the left side of Figure 3.9.

3.6.1 Test Statistic for Homogeneity

As discussed in Section 3.3 and 3.4, the key problem in non-stationary bandits is to detect changes in the underlying reward distribution, and the key problem in online clustering of bandits is to measure the relatedness between different models. We view both problems as testing homogeneity between two sets of observations to unify these two seemingly distinct problems. For change detection, we test homogeneity between recent and past observations to evaluate whether there has been a change in the underlying bandit parameters for these two consecutive sets of observations. For cluster identification, we test homogeneity between observations of two different users to verify whether they share the same parameters. On top of the test results, we can operate the bandit models accordingly for model selection, model aggregation, arm selection, and model update.

We use $\mathcal{H}_1 = \{(\mathbf{x}_i, y_i)\}_{i=1}^{t_1}$ and $\mathcal{H}_2 = \{(\mathbf{x}_j, y_j)\}_{j=1}^{t_2}$ to denote two sets of observations, where $t_1, t_2 \geq 1$ are their cardinalities. $(\mathbf{X}_1, \mathbf{y}_1)$ and $(\mathbf{X}_2, \mathbf{y}_2)$ denote design matrices and feedback vectors of $\mathcal{H}_1$ and $\mathcal{H}_2$ respectively, where each row of $\mathbf{X}$ is the context vector of a selected arm and the corresponding element in $\mathbf{y}$ is the observed reward for this arm. Under our linear reward assumption, $\forall(\mathbf{x}_i, y_i) \in \mathcal{H}_1$, $y_i \sim N(\mathbf{x}_i^\top \boldsymbol{\theta}_1, \sigma^2)$, and $\forall(\mathbf{x}_j, y_j) \in \mathcal{H}_2$, $y_j \sim N(\mathbf{x}_j^\top \boldsymbol{\theta}_2, \sigma^2)$. The test of homogeneity between $\mathcal{H}_1$ and $\mathcal{H}_2$ can thus be formally defined as testing whether $\boldsymbol{\theta}_1 = \boldsymbol{\theta}_2$, i.e., whether observations in $\mathcal{H}_1$ and $\mathcal{H}_2$ come from a homogeneous population.

Because $\boldsymbol{\theta}_1$ and $\boldsymbol{\theta}_2$ are not observable, the test has to be performed on their estimates, for which maximum likelihood estimator (MLE) is a typical choice. Denote MLE for $\boldsymbol{\theta}$ on a dataset $\mathcal{H}$ as $\vartheta = (\mathbf{X}^\top\mathbf{X})^{-}\mathbf{X}^\top\mathbf{y}$, where $(\cdot)^{-}$ stands for generalized matrix inverse. A straightforward approach to test homogeneity between $\mathcal{H}_1$ and $\mathcal{H}_2$ is to compare $\|\vartheta_1 - \vartheta_2\|$ against the estimation confidence on ϑ_1 and ϑ_2. The clustering methods used in [45, 88] essentially follow this idea. However, theoretical guarantee on the false negative probability of this method only exists when the minimum eigenvalues of $\mathbf{X}_1^\top\mathbf{X}_1$ and $\mathbf{X}_2^\top\mathbf{X}_2$ are larger than a predefined threshold. In other words, only when *both* $\mathcal{H}_1$ and $\mathcal{H}_2$ have sufficient observations, this test is effective.

We appeal to an alternative test statistic that has been proved to be *uniformly most powerful* for this type of problems [91–93]:

$$s(\mathcal{H}_1, \mathcal{H}_2) = \frac{||\mathbf{X}_1(\vartheta_1 - \vartheta_{1,2})||^2 + ||\mathbf{X}_2(\vartheta_2 - \vartheta_{1,2})||^2}{\sigma^2} \tag{3.14}$$

where $\vartheta_{1,2}$ denotes the estimator using data from both $\mathcal{H}_1$ and $\mathcal{H}_2$. The knowledge about σ^2 can be relaxed by replacing it with empirical estimate, which leads to Chow test that has F-distribution [91].

When $s(\mathcal{H}_1, \mathcal{H}_2)$ is above a chosen threshold υ, it suggests the pooled estimator deviates considerably from the individual estimators on two datasets. Thus, we conclude $\boldsymbol{\theta}_1 \neq \boldsymbol{\theta}_2$; otherwise, we conclude $\mathcal{H}_1$ and $\mathcal{H}_2$ are homogeneous. The choice of υ is critical, as it determines the type-I and type-II error probabilities of the test. Upper bounds of these two error probabilities are given below and their proofs are provided in Appendix D.

Theorem 11. *The test statistic $s(\mathcal{H}_1, \mathcal{H}_2)$ follows a non-central χ^2 distribution $s(\mathcal{H}_1, \mathcal{H}_2) \sim \chi^2(df, \psi)$, where the degree of freedom $df = rank(\mathbf{X}_1) + rank(\mathbf{X}_2) - rank(\begin{bmatrix}\mathbf{X}_1\\ \mathbf{X}_2\end{bmatrix})$, and the non-centrality parameter*

$$\psi = \frac{\begin{bmatrix}\mathbf{X}_1\boldsymbol{\theta}_1\\ \mathbf{X}_2\boldsymbol{\theta}_2\end{bmatrix}^\top \left[\mathbf{I}_{t_1+t_2} - \begin{bmatrix}\mathbf{X}_1\\ \mathbf{X}_2\end{bmatrix}(\mathbf{X}_1^\top\mathbf{X}_1 + \mathbf{X}_2^\top\mathbf{X}_2)^{-}\begin{bmatrix}\mathbf{X}_1^\top & \mathbf{X}_2^\top\end{bmatrix}\right]\begin{bmatrix}\mathbf{X}_1\boldsymbol{\theta}_1\\ \mathbf{X}_2\boldsymbol{\theta}_2\end{bmatrix}}{\sigma^2}.$$

Algorithm 7 Dynamic Clustering of Bandits (DyClu)

```
1: Input: sliding window size $\tau$, $\delta, \delta_e \in (0,1)$, threshold for change detection and neighbor identification
   $\upsilon^e$ and $\upsilon^c$, and regularization parameter $\lambda$
2: Initialization: for each user model $\mathbb{M}_{i,0}, \forall i \in \mathcal{U}$: $\mathbf{A}_{i,0} = \mathbf{0} \in \mathbb{R}^{d\times d}$, $\mathbf{b}_{i,0} = \mathbf{0} \in \mathbb{R}^d$, $\mathcal{H}_{i,0} = \emptyset$, $\hat{e}_{i,0} = 0$;
   the set of outdated user models $\mathbb{O}_0 = \emptyset$, and up-to-date user models $\mathbb{U}_0 = \{\mathbb{M}_{i,0}\}_{i\in\mathcal{U}}$
3: for $t = 1, 2, ..., T$ do
4:     Observe user $i_t \in \mathcal{U}$, and set of available arms $C_t = \{x_{t,1}, ..., x_{t,K}\}$
5:     Choose arm $\mathbf{x}_t \in C_t$ by Eq 3.15: $\arg\max_{\mathbf{x}\in C_t} \mathbf{x}^\top \hat{\boldsymbol{\theta}}_{\hat{V}_{i_t,t-1}} + CB_{\hat{V}_{i_t,t-1}}(x)$
6:     Observe reward $y_t$ from user $i_t$
7:     Compute $e_{i_t,t} = \mathbf{1}\left\{S(\mathcal{H}_{i_t,t-1}, (\mathbf{x}_t^\top, y_t)) > \upsilon^e\right\}$
8:     Update $\hat{e}_{i_t,t} = \sum_{\tilde{t}_{i_t}(\tau)<j\le t: i_j=i_t} e_{i_t,j}$
9:     if $\hat{e}_{i_t,t} \le 1 - F(\upsilon^e; 1, 0) + \sqrt{\frac{\log 1/\delta_e}{2\tau}}$ then
10:        if $e_{i_t,t} = 0$ then
11:            $\mathbb{M}_{i_t,t}$: $\mathcal{H}_{i_t,t} = \mathcal{H}_{i_t,t-1} \cup (\mathbf{x}_t, y_t)$, $\mathbf{A}_{i_t,t} = \mathbf{A}_{i_t,t-1} + \mathbf{x}_t\mathbf{x}_t^\top$, $\mathbf{b}_{i_t,t} = \mathbf{b}_{i_t,t-1} + \mathbf{x}_t y_t$
12:    else
13:        $\mathbb{O}_t = \mathbb{O}_{t-1} \cup \mathbb{M}_{i_t,t-1}$, $\hat{e}_{i_t,t} = 0$
14:        Replace $\mathbb{M}_{i_t,t-1}$ with $\mathbb{M}_{i_t,t} = (A_{i_t,t} = \mathbf{0}, b_{i_t,t} = \mathbf{0}, \mathcal{H}_{i,t} = \emptyset)$ in $\mathbb{U}_t$
15:    Update user $i_t$'s neighborhood: $\hat{V}_{i_t,t} = \{\mathbb{M} \in \mathbb{U}_t \cup \mathbb{O}_t : S(\mathcal{H}_{i_t,t}, \mathcal{H}) \le \upsilon^c\}$
```

Lemma 11. *When $\boldsymbol{\theta}_1 = \boldsymbol{\theta}_2$, $\psi = 0$; the type-I error probability can be upper bounded by:*

$$P\big(s(\mathcal{H}_1, \mathcal{H}_2) > \upsilon | \boldsymbol{\theta}_1 = \boldsymbol{\theta}_2\big) \le 1 - F(\upsilon; df, 0),$$

where $F(\upsilon; df, 0)$ denotes the accumulated density function of distribution $\chi^2(df, 0)$ evaluated at υ.

Lemma 12. *When $\boldsymbol{\theta}_1 \ne \boldsymbol{\theta}_2$, $\psi \ge 0$; the type-II error probability can be upper bounded by,*

$$P\big(s(\mathcal{H}_1, \mathcal{H}_2) \le \upsilon | \boldsymbol{\theta}_1 \ne \boldsymbol{\theta}_2\big) \le \begin{cases} F\Big(\upsilon; d, \frac{||\boldsymbol{\theta}_1 - \boldsymbol{\theta}_2||^2/\sigma^2}{1/\lambda_{min}(\mathbf{X}_1^\top \mathbf{X}_1) + 1/\lambda_{min}(\mathbf{X}_2^\top \mathbf{X}_2)}\Big), & \textit{if } \mathbf{X}_1 \textit{ and } \mathbf{X}_2 \textit{ are full-rank.} \\ F(\upsilon; df, 0), & \textit{otherwise.} \end{cases}$$

These error probabilities are the key concerns in our problem: in change detection, they correspond to the early and late detection of change points [85]; and in cluster identification, they correspond to missing a user model in the neighborhood and placing a wrong user model in the neighborhood [45]. The uniformly most powerful property of the test defined in Eq (3.14) guarantees its sensitivity is optimal at any level of specificity.

3.6.2 Dynamic Clustering of Bandits

In the environment specified in Section 3.6, the user's reward mapping function is piecewise stationary (e.g., the line segments on each user's interaction trace in Figure 3.9), which calls for the learner to actively detect changes and re-initialize the estimator to avoid distortion from outdated observations [83–86, 94]. A limitation of these methods is that they do not attempt to reuse outdated observations because they implicitly assume each stationary period has an unique parameter. Our setting relaxes this by allowing for existence of identical reward mappings across users and time (e.g., the orange line segments in Figure 3.9), which urges the learner to take advantage of this situation by identifying and aggregating observations with the same parameter to obtain a more accurate reward estimation.

Since neither the change points nor the grouping structure is known, in order to reuse past observations while avoiding distortion, the learner needs to accurately detect change points, stores observations in the interval between two consecutive detection together, and then correctly identify intervals with the same parameter as the current one. in this work, we propose to unify these two operations using the test in Section 3.6.1, which leads to the algorithm Dynamic Clustering of Bandits, or DyClu in short. DyClu forms a two-level hierarchy as shown in Figure 3.9: at the lower level, it stores observations in each interval and their sufficient statistics in a user model; at the upper level, it detects change in user's reward function to decide when to create new user models and clusters individual user models for arm selection. Detailed steps of DyClu are explained in Algorithm 7.

The lower level of DyClu manages observations associated with each user $i \in \mathcal{U}$ in user models, denoted by $\mathbb{M}_{i,t}$. Each user model $\mathbb{M}_{i,t} = (\mathbf{A}_{i,t}, \mathbf{b}_{i,t}, \mathcal{H}_{i,t})$ stores:

1. $\mathcal{H}_{i,t}$: a set of observations associated with user i since the initialization of $\mathbb{M}_{i,t}$ up to time t, where each element is a context vector and reward pair $(\mathbf{x}_k, y_k)$.
2. Sufficient statistics: $\mathbf{A}_{i,t} = \sum_{(\mathbf{x}_k,\cdot)\in\mathcal{H}_{i,t}} \mathbf{x}_k\mathbf{x}_k^\top$ and $\mathbf{b}_{i,t} = \sum_{(\mathbf{x}_k,y_k)\in\mathcal{H}_{i,t}} \mathbf{x}_k y_k$.

Every time DyClu detects change in a user's reward mapping function, a new user model is created to replace the previous one (line 15 in Algorithm 7). We refer to the replaced user models as outdated models and the others up-to-date ones. We denote the set of all outdated user models at time t as $\mathbb{O}_t$ and the up-to-date ones as $\mathbb{U}_t$. In Figure 3.9, the row of circles next to $\mathbb{M}_{1,t-1}$ represents all the user models for user 1, red ones denote outdated models and the blue one denotes up-to-date model.

The upper level of DyClu is responsible for managing the user models via change detection and model clustering. It replaces outdated models in each user and aggregates models across users and time for arm selection.

• **Change detection.** A one-sample homogeneity test is used to construct a test variable

$$e_{i_t,t} = \mathbf{1}\left\{s(\mathcal{H}_{i_t,t-1}, \{(\mathbf{x}_t, y_t)\}) > \upsilon^e\right\}$$

to measure whether the user model $\mathbb{M}_{i_t,t-1}$ is 'admissible' to the new observation $(\mathbf{x}_t, y_t)$. υ^e is a chosen threshold for change detection.

To make more reliable change detection, we use the empirical mean of $e_{i_t,t}$ in a sliding window of size $\min(|\mathcal{H}_{i_t,t-1}|, \tau)$ as the test statistic, denoted as $\hat{e}_{i_t,t} = \frac{1}{\min(|\mathcal{H}_{i_t,t-1}|,\tau)} \sum_k e_{i_t,k}$.

Lemma 13 specifies the upper bound of early detection probability using $\hat{e}_{i,t}$, which is used for selecting threshold for it.

Lemma 13. *From Lemma 11, type-1 error probability* $P(e_{i,t} = 1) \le 1 - F(\upsilon^e; 1, 0)$*, and thus* $\mathbb{E}[e_{i,t}] \le 1 - F(\upsilon^e; 1, 0)$*. Applying Hoeffding inequality gives,*

$$P\Big(\hat{e}_{i,t} > 1 - F(\upsilon^e; 1, 0) + \sqrt{\frac{\log 1/\delta_e}{2\tau}}\Big) \le \delta_e$$

At any time step t, DyClu only updates $\mathbb{M}_{i_t,t-1}$ when $e_{i_t,t} = 0$ (line 10-12 in Algorithm 7). This guarantees that if the underlying reward distribution has changed, with a high probability we have $e_{i_t,t} = 1$, and thus the user model $\mathbb{M}_{i_t,t-1}$ will not be updated. This prevents any distortion in $\mathcal{H}_{i_t,t}$ by observations from different reward distributions. When $\hat{e}_{i_t,t}$ exceeds the threshold specified by Lemma 13, DyClu will inform the lower level to move $\mathbb{M}_{i_t,t-1}$ to the outdated model set $\mathbb{O}_t = \mathbb{O}_{t-1} \cup \{\mathbb{M}_{i_t,t-1}\}$; and then create a new model $\mathbb{M}_{i_t,t} = (A_{i_t,t} = \mathbf{0}, b_{i_t,t} = \mathbf{0}, \mathcal{H}_{i,t} = \emptyset)$ for user i_t as shown in line 13-16 in Algorithm 7.

• **Clustering of user models.** In this step, DyClu finds the set of "neighborhood" user models $\hat{V}_{i_t,t}$ of current user model $\mathbb{M}_{i_t}, t$, where $\hat{V}_{i_t,t-1} = \{\mathbb{M} = (\mathbf{A}, \mathbf{b}, \mathcal{H}) \in \mathbb{U}_t \cup \mathbb{O}_t : s(\mathcal{H}_{i_t,t}, \mathcal{H}) \le \upsilon^c\}$. Basically, DyClu executes homogeneity test between $\mathbb{M}_{i_t,t}$ and all other user models $\mathbb{M} \in \mathbb{U}_t \cup \mathbb{O}_t$ (both outdated and up-to-date) with a given threshold υ^c (line 17 in Algorithm 7). Lemma 11 and 12 again specify error probabilities of each decision.

When selecting an arm for user i_t at time t, DyClu aggregates the sufficient statistics of user models in neighborhood $\hat{V}_{i_t,t-1}$. Then it adopts the popular UCB strategy [95,96] to balance exploitation and exploration. Specifically, DyClu selects arm $\mathbf{x}_t$ that maximizes the UCB score computed by aggregated sufficient statistics as follows (line 5 in Algorithm 7),

$$\mathbf{x}_t = \arg\max_{\mathbf{x}\in C_t} \mathbf{x}^\top \hat{\boldsymbol{\theta}}_{\hat{V}_{i_t,t-1}} + CB_{\hat{V}_{i_t,t-1}}(\mathbf{x}) \tag{3.15}$$

In Eq (3.15), $\hat{\boldsymbol{\theta}}_{\hat{V}_{i_t,t-1}} = \mathbf{A}^{-1}_{\hat{V}_{i_t,t-1}} \mathbf{b}_{\hat{V}_{i_t,t-1}}$ is the ridge regression estimator using aggregated statistics $\mathbf{A}_{\hat{V}_{i_t,t-1}} = \lambda \mathbf{I}_d + \sum_{(\mathbf{A}_j,\mathbf{b}_j,\mathcal{H}_j)\in\hat{V}_{i_t,t-1}} \mathbf{A}_j$ and $\mathbf{b}_{\hat{V}_{i_t,t-1}} = \sum_{(\mathbf{A}_j,\mathbf{b}_j,\mathcal{H}_j)\in\hat{V}_{i_t,t-1}} \mathbf{b}_j$; the confidence bound of reward estimation for arm $\mathbf{x}$ is $CB_{\hat{V}_{i_t,t-1}}(\mathbf{x}) = \alpha_{\hat{V}_{i_t,t-1}}\sqrt{\mathbf{x}^\top \mathbf{A}^{-1}_{\hat{V}_{i_t,t-1}}\mathbf{x}}$, where

$$\alpha_{\hat{V}_{i_t,t-1}} = \sigma\sqrt{d\log\left(1+\frac{\sum_{(A_j,b_j,\mathcal{H}_j)\in\hat{V}_{i_t,t-1}}|\mathcal{H}_j|}{d\lambda}\right)+2\log\frac{1}{\delta}} + \sqrt{\lambda}.$$

3.6.3 Regret Analysis

Our regret analysis relies on the high probability results in [97] and decomposition of "good" and "bad" events according to change detection and clustering results. The full proof, along with ancillary results, are given in the appendix.

Theorem 12. *Under Assumptions 3, 4 and 5, the regret of DyClu is upper bounded by:*

$$R_T = O\Big(\sigma d\sqrt{T\log^2 T}\big(\sum_{k=1}^{m}\sqrt{p_k}\big) + \sum_{i\in\mathcal{U}}\Gamma_i(T)\cdot C\Big)$$

where $C = \frac{1}{1-\delta^e} + \frac{\sigma^2}{\gamma^2\lambda'^2}\log\frac{d}{\delta'}$, *with a probability at least* $(1-\delta)(1-\frac{\delta_e}{1-\delta_e})(1-\delta')$.

This upper regret bound depends both on the intrinsic clustering and non-stationary property of user set $\mathcal{U}$. To better understand it, we discuss in the appendix how it compares with state-of-the-art bandit solutions in settings like non-stationary environment only or clustered environment only.

3.6.4 Experiments

We investigate the empirical performance of the proposed algorithm by comparing with a list of state-of-the-art baselines for both non-stationary bandits and clustered bandits on synthetic and real-world recommendation datasets.

Experiment setup and baselines

• **Synthetic dataset.** We create a set of unique bandit parameters $\{\phi_k\}_{k=1}^m$ and arm pool $\{\mathbf{x}_j\}_{j=1}^K$ ($K = 1000$), where ϕ_k and $\mathbf{x}_j$ are first sampled from $N(\mathbf{0}_d, \mathbf{I}_d)$ with $d = 25$ and then normalized so that $\forall k, j$, $\|\phi_k\| = 1$ and $\|\mathbf{x}_j\| = 1$. When sampling $\{\phi_k\}_{k=1}^m$, the separation margin γ is set to 0.9 and enforced via rejection sampling. n users are simulated. In each user, we sample a series of time intervals from (S_{min}, S_{max}) uniformly; and for each time interval, we sample a unique parameter from $\{\phi_k\}_{k=1}^m$ as the ground-truth bandit parameter for this period. This creates asynchronous changes and clustering structure in users' reward functions. The users are served in a round-robin fashion. At time step $t = 1, 2, \dots, T$, a subset of arms are randomly chosen and disclosed to the learner. Reward of the selected arm is generated by the linear function governed by the corresponding bandit parameter and context vector, with additional Gaussian noise sampled from $N(0, \sigma^2)$.

• **LastFM dataset.** We used the same way of preprocessing as that in the experiments for dLinUCB and DenBand introduced in Section 3.5. We simulate a clustered non-stationary environment by creating 20 'hybrid users'. We create hybrid users by sampling three real users uniformly and then concatenating their associated data points together. Hence, data points of the same real user would appear in different hybrid users, which is analogous to stationary periods that share the same unique bandit parameters across different users and time.

Table 3.3: Comparison of accumulated regret under different environment settings.

	n	m	S_{min}	S_{max}	T	σ	oracle.	LinUCB	adTS	dLin.	CLUB	DyClu
1	100	10	400	2500	2500	0.09	115	19954	9872	2432	20274	853
2	100	50	400	2500	2500	0.09	489	20952	9563	2420	21205	1363
3	100	100	400	2500	2500	0.09	873	21950	10961	2549	22280	1958
4	100	10	200	400	2500	0.09	112	39249	36301	10831	39436	3025
5	100	10	800	1000	2500	0.09	113	34385	13788	3265	34441	1139
6	100	10	1200	1400	2500	0.09	112	24769	8124	2144	24980	778
7	100	10	400	2500	2500	0.12	166	22453	10567	3301	22756	1140
8	100	10	400	2500	2500	0.15	232	19082	10000	5872	19427	1487
9	100	10	400	2500	2500	0.18	307	23918	11255	9848	24050	1956

• **Baselines.** We compare DyClu with a set of state-of-the-art bandit algorithms: linear bandit LinUCB [97], non-stationary bandit dLinUCB developed in Section 3.3 [85] and adTS [98], as well as online clustering bandit CLUB [45]. In simulation based experiments, we also include oracle-LinUCB for comparison, which runs an instance of LinUCB for each unique bandit parameter. Comparing with it helps us understand the added regret from errors in change detection and clustering.

Experiment results

• **Empirical comparisons on synthetic dataset.** We compare accumulated regret of all bandit algorithms under three environment settings, and the results are reported in Figure 3.10. Environment 1 simulates the online clustering setting in [45], where *no* change in the reward function is introduced. DyClu outperformed other baselines, including CLUB, demonstrating the quality of its identified clustering structure. Specifically, compared with adTS that incurs high regret as a result of too many false detections, the change detection in DyClu and dLinUCB has much less false positives, as there is no change in each user's reward distribution. Environment 2 simulates the piecewise stationary setting in [85]. Algorithms designed for stationary environment, e.g., CLUB and LinUCB suffer from a linear regret after the first change point. DyClu achieved the best performance, with a wide margin with second best, dLinUCB, which is designed for this environment. Environment 3 combines previous two settings with both non-stationarity and clustering structure. DyClu again outperformed others. It is worth noting that regret of all algorithms increased compared with Environment 1 due to the nonstationarity, but the increase in DyClu is the smallest. And in all settings, DyClu's performance is closest to the oracle LinUCB's, which shows that DyClu can correctly cluster and aggregate observations from the dynamically changing users.

• **Sensitivity to environment settings.** According to our regret analysis, the performance of DyClu depends on environment parameters like the number of unique bandit parameters m, the number of stationary periods $\Gamma_i(T)$ for $i \in \mathcal{U}$, and variance of Gaussian noise σ^2. We investigate their influence on DyClu and baselines, by varying these parameters while keeping the others fixed. The accumulated regret under different settings are reported in Table 3.3. DyClu outperformed other baselines in all 9 different settings, and the changes of its regret align with our theoretical analysis. A larger number of unique parameters m leads to higher regret of DyClu as shown in setting 1, 2 and 3, since observations are split into more clusters with smaller size each. In addition, larger number of stationary periods incurs more errors in change detection, leading to an increased

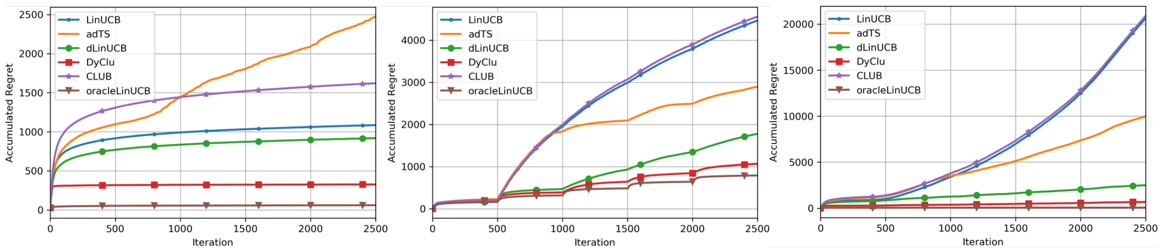


Figure 3.10: Accumulated regret on synthetic datasets with three different environment settings. Environment 1: $n = 100$ users share a global set of $m = 5$ unique bandit parameters, and each user remains stationary all the time. Environment 2: $n = 20$ user with fixed stationary period length 500; each period sample a unique bandit parameter. Environment 3: $n = 100$ users share a global set of $m = 5$ unique bandit parameters, and each user changes in a asynchronous manner.

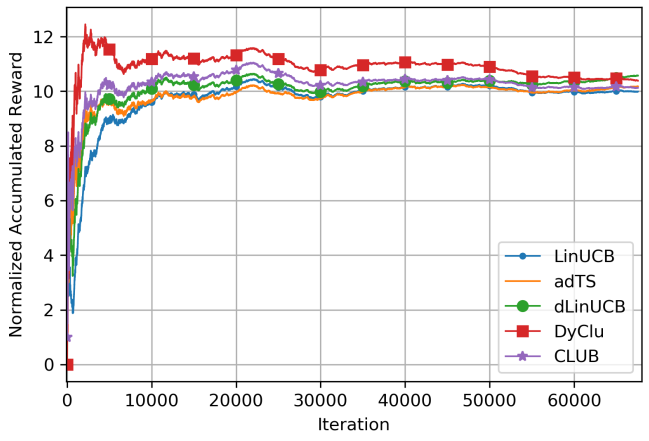


Figure 3.11: Comparison of accumulated reward normalized by a random policy on LastFM dataset.

regret. This is confirmed by results in setting 4, 5 and 6. Lastly, as shown in setting 7, 8 and 9, larger Gaussian noise leads to a higher regret, as it slows down convergence of reward estimation and change detection.

• **Empirical comparisons on LastFM.** We report normalized accumulated reward (ratio between baselines and uniformly random arm selection strategy [86]) on LastFM in Figure 3.11. In this environment, realizing both non-stationarity and clustering structure is important for an online learning algorithm to perform. DyClu's improvement over other baselines confirms its quality in partitioning and aggregating relevant data points across users. The advantage of DyClu is more apparent at the early stage of learning, where each local user model has not collected sufficient amount of observations for individualized reward estimation; and thus change detection and clustering are more difficult there.

3.7 Conclusion

In this chapter, we studied contextual bandits for in a piece-wise stationary environment, which is very common in many important real-world applications but insufficiently investigated in existing works. We develop a suite of contextual bandit solutions including dLinUCB [85], DenBand [86] and DyClu for such a non-stationary environment. dLinUCB is able to adaptively decide when to create a new bandit model and when to abandon an out-of-date bandit model depending on the current environment it is interacting with. DenBand capitalizes on the context-dependent property of environment changes and dynamically forms an ensemble of multiple bandit models, to conquer the non-stationary environment. DyClu unifies the efforts in non-stationary bandits and online clustering of bandits via homogeneity test. It adaptively detects changes in the underlying reward distribution and clusters bandit models for aggregated arm selection. Rigorous regret analysis validates the convergence of the proposed solutions. Extensive empirical evaluations on simulation and three large real-world datasets verified the effectiveness and reliability of the proposed solutions.

Research in the previous chapter indicates that sharing information among multiple learning agents can accelerate learning. Information sharing could be particularly helpful if learners operate in a changing environment because a learner could benefit from the previous experience of another learner to adapt to its new environment. In addition, sociologists have long converged that the evolution of user preferences in a social network is driven by the interplay between users' preferences and the social network structure [99]. There are two social theories that explain this evolution: the *social influence effect* states that users' future preferences are affected by the social network around them, and the *homophily effect* suggests that people tend to associate and bond with others that have similar preferences [100]. These social psychology perspectives further indicate the possibility and benefits of collaborative learning in a non-stationary environment. With such insights, it is worth studying how collaborative bandit learning can help better conquer a non-stationary environment. One the one hand, as suggested by DyClu bandit, the learner's experience on some users can be reused for some other users if some users share the same interest/preference.

Chapter 4

Conclusion & Future Work

This dissertation aims at equipping modern intelligent systems with interactive online learning solutions. We identify two important challenges toward realizing this goal using multi-armed bandit solutions. We develop a series of solutions to conquer these two challenges from the perspective of collaborative bandit learning and non-stationary bandit learning, respectively.

From the collaborative bandit learning perspective, we first developed a collaborative bandit learning algorithm CoLin. 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 fewer observations to achieve satisfactory prediction performance) and leads to reduced overall regret. Then, we developed a factorization-based bandit algorithm. Both observable dependency and implicit dependency among users are leveraged to improve the algorithm's convergence rate and help conquer cold-start in recommendation tasks. A high probability sublinear upper regret bound is proved, where considerable regret reduction is achieved in on both user and item sides. Lastly, we studied the problem of protecting global and local differential privacy for collaborative bandits. Our solution framework allows the privacy mechanism to calibrate the noise scale with respect to the user dependency graph. Our theoretical analysis proves the desired privacy guarantee in the collaborative bandit learning setting. We also rigorously proved the corresponding upper regret bound of the derived private algorithms.

Under the non-stationary bandit learning environment, we develop two contextual bandit solutions dLinUCB and DenBand for piecewise stationary environments. By maintaining multiple contextual bandit models and tracking their reward estimation quality over time, dLinUCB adaptively updates its strategy for interacting with a changing environment. We rigorously prove an $O(\Gamma_T\sqrt{S_T}\ln S_T)$ upper regret bound, which is arguably the tightest upper regret bound any algorithm can achieve in such an environment without further assumptions about the environment. By capitalizing on the context-dependent property of environment changes, in DenBand bandit models are dynamically ensembled and reused to conquer the non-stationary environment. Finally, we unify the efforts in non-stationary bandits and the online clustering of bandits via homogeneity test. The proposed solution DyClu adaptively detects changes in the underlying reward distribution and clusters bandit models for aggregated arm selection. The resulting upper regret bound matches with the ideal algorithm's only up to a constant; and extensive empirical evaluations validate its effectiveness in learning in a non-stationary and clustered environment.

4.1 Future Work

Research in this dissertation address two main important challenges when developing interactive online learning solutions for intelligent systems where humans are an integral part. There are more challenges to be conquered in such scenarios. Besides, we also recognize the emerging need to take humans out of the loop of many tedious tasks.

4.1.1 Human-in-the-loop Interactive Online Learning

In addition to the two main challenges addressed by this dissertation, we would like to point out several important questions worth thinking about and exploring when humans are an integral part of the learning process. Interactive online learning solutions interact with the environment, which consists of human users by taking actions and learning from the corresponding reward. We thus summarize the challenges systematically from the perspective of reward formulation, action-taking, and the learning process.

Accurate interpretation of the observed user-system interactions. (What is the reward?) Interactive online learning relies on feedback or reward to further improve its decision-making strategy. Although interaction data can be plentiful in real-world intelligent systems, it remains a challenge to accurately interpret the observed user interactions and understand what can be learned from them. First of all, with the existence of various forms of interactions in an intelligent system, it is worth thinking about how to define 'reward' for interactive online learning. For example, in recommendation systems, user click is typically considered as the reward. However, such immediate user response is only one aspect of user engagement and is only a partial reflection of the recommendation service quality. In fact, metrics purely based on immediate feedback signals such as user-item ratings and click-through rates have been increasingly criticized for being insufficient to measure and represent the real engagement of users [101]. So in order to accurately interpret human users' interest from interaction data, a more comprehensive definition of 'reward' is needed such that the decisions of the interactive online learning algorithm can be optimized in the right direction. For example, in one of our work, we set the optimization of users' long-term engagement as the learning target and consider two types of feedback as the reward, including one immediate user response and one long-term user response, to improve the interactive online learning agent. Second, the observed user interaction data can be biased due to various reasons, for example, presentation bias or user behavior, so when translating user interaction into the reward signal, it is critical to consider and correct the potential bias in them. For example, the examination hypothesis [102], which is a fundamental assumption in click modeling, postulates that a user clicks on a system's returned result if and only if that result has been examined by the user and it is relevant to the user's information need at the moment. It indicates that the observation of no click is not necessarily negative feedback. To deal with this specific issue, we performed contextual bandit learning with implicit feedback by modeling the feedback as a composition of user result examination and relevance judgment [103]. We acknowledge that accurate interpretation of the observed human interaction should be considered together with the specific application scenarios. For example, in ranking scenarios, where multiple items need to be selected and presented, the existence position bias needs to be considered, while position bias may not exist in recommendation scenarios where only one item is presented.

Diverse forms of user-system interactions. (What are the actions?) The forms of actions in a particular intelligent system can be obvious at first sight. For example, making recommendations is the action of the interactive online learning agent in recommendation scenarios. However it is not necessarily the case with the emerging development of intelligent systems for sophisticated application scenarios, where much more diverse forms of user-system interactions may be needed. For example, in conversational recommendation systems, the system may ask questions occasionally in addition to making recommendations. In an intelligent tutoring system, the system may first recommend a video and then recommend exercise questions accordingly to the students. On the one hand, these diverse forms of actions need to be considered for the seamless application of interactive online learning agent to handle sophisticated tasks; on the other hand, since different forms of interactions can correspond to information acquisition of different resolutions, joint modeling of them can make the learning process more efficient.

Efficient learning from observed user-system interactions. (Can we learn more efficiently?) User-system interactions can be structured. For example, the existence of dependency among users or items indicates the existence of low-rank structures, which opens the possibility of more efficient collaborative learning, as described in Chapter 2. Structures may also exist in reward formulation and action-taking, which can be utilized to improve learning efficiency. For example, because of position bias, the users would only examine a very small number of documents under each query, and users only perceive utility from the documents that they actually examine. In one of our work, this property is utilized in dueling bandit based online learning to rank solutions to reduce the variance of online gradient estimate [104]. The key insight is that the *true gradient* is only revealed by features playing an essential role in ranking those examined documents (i.e., document space) under this query and hence components of the proposed gradient which are orthogonal to the document space can be safely removed to achieve variance reduction. Intuitively speaking, this work leveraged a special structure in the reward function in online learning to rank problems to improve the online learning efficiency. In the future, it is worth thinking about how to leverage the diverse forms of interactions to improve learning efficiency.

4.1.2 Human-out-of-the-loop Interactive Online Learning

In addition to the scenarios where humans are part of the intelligent systems as users or service consumers, there are also scenarios where humans are the decision-makers. For example, design choices are pervasive in both scientific and industrial endeavors: engineers and scientists design machines or programs to execute tasks more efficiently, and pharmaceutical researchers design new drugs to cure disease. Such design choices inevitably involve extensive human experts' trial and error to find the right combinations. Interactive online learning has the potential to make advances in automating such processes, which can result in immediate productivity improvement and innovation in a wide area of domains, including but not limited to e-commerce, healthcare, cyber-physical systems, computing systems, and education. So it is worth studying how to use interactive online learning techniques to move humans out of those tedious trial and error processes. Such a long-term goal is not an unachievable tale, and promising progress has been made in areas such as automated machine learning (AutoML), and autonomous systems using machine learning and reinforcement learning techniques. However, such innovations are not mature enough to be widely adopted in real-world scenarios. A lot of challenges are yet to be addressed. For example, achieving this goal requires the interactive online learning agent to be efficient, robust, and accountable.

4.1.3 Ethical Considerations in Interactive Online Learning

With the vast increase of both diversity and the importance of the applications to which machine learning is applied, ethical considerations are becoming critical to algorithmic designs of machine learning solutions. It is especially important to consider the ethical aspects of interactive online learning solutions. The inevitable involvement of uncertainty in such solutions could sow public distrust, especially with their integration into various infrastructure. In addition to the privacy concerns studied in this dissertation, there are multiple other aspects of ethical considerations worth exploring in the context of interactive online learning.

Robustness. In many real-world scenarios, the environment with which the learning agent is interacting is not always benign. Various attacks, such as data poisoning attack and extract attack, exist in many real-world situations. Prior work on data poisoning attack focused almost exclusively on the batch setting, where the attacker poisons a batch training set, and then the victim learns from the batch. However, the batch setting misses the threats posed by the attacker on sequential learners. For example, in an online ranking system or recommendation system, malicious users can take advantage of the model updating mechanisms in online learning to launch poisoning attacks, e.g., strategically submitting results to manipulate the rankings/recommendation. Thus it is necessary to consider the online learning process in an adversarial setting.

Accountability. According to the ACM code of ethics and professional conduct, 'If you can not predict what your system will do, and you can not or won't monitor it, you should not deploy it.' However, the performance of an interactive online learning algorithm, such as multi-armed bandit and reinforcement learning, can vary drastically during online learning. Conventional online learning algorithms provide little information about the quality of their current policies before deployment, which directly limits their usage in high-stake applications like healthcare and may violate the ACM code of ethics and professional conduct. Accountable online learning frameworks are needed to avoid this severe limitation.

Fairness Fairness is another important ethical constraint for online learning algorithms, especially when humans are in the loop, and the online decisions have important consequences on people's lives, such as hiring, policing, and even criminal sentencing. It is thus critically important to study fairness in online decision making. Despite the volume and velocity of published work about fairness in machine learning, the understanding of the fundamental questions related to fairness and machine learning, especially in the interactive online learning setting, remains in its infancy. First, because of the online learning agents' uncertainty about the environment, the definition of 'fairness' needs to be carefully studied, and it needs to be defined in a problem-dependent manner in the online learning setting. Second, due to the sequential nature of many online learning solutions, such as reinforcement learning, the impact of decisions may be delayed, which requires us to study their delayed impact on fairness. Third, it is necessary to study the trade-off and reconciliation between fairness and utility maximization during online decision making.

4.2 Broader Impact

Researchers working in areas like online learning and bandit learning, and practitioners developing online information service systems, such as recommender systems and online learning to rank systems, will benefit from research in this dissertation. The interactive online learning paradigm considered in this dissertation is general and applicable to various types of intelligent systems where sequential decision making is involved. In addition, new applications and intelligent systems can be made possible, which can result in immediate productivity improvement and user benefits. Systems equipped with the developed solutions are expected to provide better overall utility to the general public who are using them. For example, the collaborative bandit learning approaches developed in this dissertation can make the learning process more efficient when correlations between users exist; the non-stationary bandit learning approaches can make the systems more adaptive and robust to the change of user interest; the differentially private collaborative bandit learning solution developed in this dissertation can help protect users' privacy in a real-world intelligent system. The insights and discussions about other ethical considerations, including robustness, accountability, and fairness, in this dissertation, foster the awareness and understanding of a more responsible interactive online learning paradigm.

Bibliography

[1] Robert B Cialdini and Melanie R Trost. Social influence: Social norms, conformity and compliance. 1998.

[2] Lihong Li, Wei Chu, John Langford, and Robert E Schapire. A contextual-bandit approach to personalized news article recommendation. In *Proceedings of 19th WWW*, pages 661–670. ACM, 2010.

[3] Lihong Li, Wei Chu, John Langford, and Xuanhui Wang. Unbiased offline evaluation of contextual-bandit-based news article recommendation algorithms. In *Proceedings of 4th WSDM*, pages 297–306. ACM, 2011.

[4] Shuo Tian, Jehane Michael Le Grange, Peng Wang, Wei Huang, and Zhewei Ye. Smart healthcare: making medical care more intelligent. *Global Health Journal*, 3, 10 2019.

[5] Ali Alkhatlan and Jugal Kalita. Intelligent tutoring systems: A comprehensive historical survey with recent developments. *International Journal of Computer Applications*, 181(43):1–20, Mar 2019.

[6] Benjamin Nye. Intelligent tutoring systems by and for the developing world: A review of trends and approaches for educational technology in a global context. *International Journal of Artificial Intelligence in Education*, 25, 06 2015.

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

[8] Badrul Sarwar, George Karypis, Joseph Konstan, and John Riedl. Item-based collaborative filtering recommendation algorithms. In *Proceedings of the 10th international conference on World Wide Web*, pages 285–295. ACM, 2001.

[9] Charu C. Aggarwal. *Recommender Systems: The Textbook*. Springer Publishing Company, Incorporated, 1st edition, 2016.

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

[11] Tze Leung Lai and Herbert Robbins. Asymptotically efficient adaptive allocation rules. *Advances in applied mathematics*, 6(1):4–22, 1985.

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

[13] P. Auer, N. Cesa-Bianchi, Y. Freund, and Robert 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.

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

[15] Nicolò Cesa-Bianchi, Claudio Gentile, and Giovanni Zappella. A gang of bandits. *In Pro. NIPS*, 2013.

[16] Sarah Filippi, Olivier Cappe, Aurélien Garivier, and Csaba Szepesvári. Parametric bandits: The generalized linear case. In *NIPS*, pages 586–594, 2010.

[17] Djallel Bouneffouf, Amel Bouzeghoub, and Alda Lopes Gançarski. A contextual-bandit algorithm for mobile context-aware recommender system. In *Neural Information Processing*, pages 324–331. 2012.

[18] Olivier Chapelle and Lihong Li. An empirical evaluation of thompson sampling. In *NIPS 2011*, pages 2249–2257, 2011.

[19] Wei Li, Xuerui Wang, Ruofei Zhang, Ying Cui, Jianchang Mao, and Rong Jin. Exploitation and exploration in a performance based contextual advertising system. In *Proceedings of 16th SIGKDD*, pages 27–36. ACM, 2010.

[20] Peter Brusilovsky, Alfred Kobsa, and Wolfgang Nejdl, editors. *The Adaptive Web, Methods and Strategies of Web Personalization*, volume 4321 of *Lecture Notes in Computer Science*. Springer, 2007.

[21] Andrew I Schein, Alexandrin Popescul, Lyle H Ungar, and David M Pennock. Methods and metrics for cold-start recommendations. In *Proceedings of 25th SIGIR*, pages 253–260. ACM, 2002.

[22] Joseph A Calandrino, Ann Kilzer, Arvind Narayanan, Edward W Felten, and Vitaly Shmatikov. " you might also like:" privacy risks of collaborative filtering. In *2011 IEEE Symposium on Security and Privacy*, pages 231–246. IEEE, 2011.

[23] Aleksandra Korolova. Privacy violations using microtargeted ads: A case study. In *ICDM 2010*, pages 474–482. IEEE, 2010.

[24] Abhradeep Guha Thakurta and Adam Smith. (nearly) optimal algorithms for private online learning in full-information and bandit settings. In *Advances in Neural Information Processing Systems*, pages 2733–2741, 2013.

[25] Naman Agarwal and Karan Singh. The price of differential privacy for online learning. In *ICML 2017*, pages 32–40. JMLR. org, 2017.

[26] Aristide Charles Yedia Tossou and Christos Dimitrakakis. Achieving privacy in the adversarial multi-armed bandit. In *AAAI 2017*, 2017.

[27] Roshan Shariff and Or Sheffet. Differentially private contextual linear bandits. In *Advances in Neural Information Processing Systems*, pages 4296–4306, 2018.

[28] Prateek Jain, Pravesh Kothari, and Abhradeep Thakurta. Differentially private online learning. In *Conference on Learning Theory*, pages 24–1, 2012.

[29] Seth Neel and Aaron Roth. Mitigating bias in adaptive data gathering via differential privacy. *arXiv preprint arXiv:1806.02329*, 2018.

[30] Yehuda Koren, Robert Bell, and Chris Volinsky. Matrix factorization techniques for recommender systems. *Computer*, (8):30–37, 2009.

[31] Yehuda Koren. Factorization meets the neighborhood: a multifaceted collaborative filtering model. In *Proceedings of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 426–434. ACM, 2008.

[32] Xiaoyuan Su and Taghi M Khoshgoftaar. A survey of collaborative filtering techniques. *Advances in artificial intelligence*, 2009:4, 2009.

[33] Emmanuel J Candès and Benjamin Recht. Exact matrix completion via convex optimization. *Foundations of Computational mathematics*, 9(6):717–772, 2009.

[34] Emmanuel J Candès and Terence Tao. The power of convex relaxation: Near-optimal matrix completion. *IEEE Transactions on Information Theory*, 56(5):2053–2080, 2010.

[35] Cynthia Dwork, Frank McSherry, Kobbi Nissim, and Adam Smith. Calibrating noise to sensitivity in private data analysis. In *Theory of cryptography conference*, pages 265–284. Springer, 2006.

[36] John C Duchi, Michael I Jordan, and Martin J Wainwright. Local privacy and statistical minimax rates. In *2013 IEEE 54th Annual Symposium on Foundations of Computer Science*, pages 429–438. IEEE, 2013.

[37] Yasin Abbasi-yadkori, Dávid Pál, and Csaba Szepesvári. Improved algorithms for linear stochastic bandits. In *NIPS*, pages 2312–2320. 2011.

[38] Wei Chu, Lihong Li, Lev Reyzin, and Robert E Schapire. Contextual bandits with linear payoff functions. In *AISTATS'11*, pages 208–214, 2011.

[39] Aurélien Garivier and Eric Moulines. On upper-confidence bound policies for non-stationary bandit problems. In *arXiv preprint arXiv:0805.3415 (2008)*.

[40] Cedric Hartland, Sylvain Gelly, Nicolas Baskiotis, Olivier Teytaud, and Michele Sebag. Multi-armed Bandit, Dynamic Environments and Meta-Bandits. November 2006.

[41] Negar Hariri, Bamshad Mobasher, and Robin Burke. Adapting to user preference changes in interactive recommendation. In *Proceedings of the 24th International Conference on Artificial Intelligence*, IJCAI'15, pages 4268–4274. AAAI Press, 2015.

[42] Jia Yuan Yu and Shie Mannor. Piecewise-stationary bandit problems with side observations. In *Proceedings of the 26th Annual International Conference on Machine Learning*, ICML '09, pages 1177–1184, New York, NY, USA, 2009. ACM.

[43] Swapna Buccapatnam, Atilla Eryilmaz, and Ness 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.

[44] Swapna Buccapatnam, Atilla Eryilmaz, and Ness B. Shroff. Stochastic bandits with side observations on networks. *SIGMETRICS Perform. Eval. Rev.*, 42(1):289–300, June 2014.

[45] Claudio Gentile, Shuai Li, and Giovanni Zappella. Online clustering of bandits. In *ICML'14*, pages 757–765, 2014.

[46] S. Kar, H.V. Poor, and Shuguang 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.

[47] Kareem Amin, Michael Kearns, and Umar Syed. Graphical models for bandit problems. *Proceedings of UAI 2011*, 2011.

[48] Aleksandrs Slivkins. Contextual bandits with similarity information. *The Journal of Machine Learning Research*, 15(1):2533–2568, 2014.

[49] Jaya Kawale, Hung H Bui, Branislav Kveton, Long Tran-Thanh, and Sanjay Chawla. Efficient thompson sampling for online matrix-factorization recommendation. In *Advances in Neural Information Processing Systems*, pages 1297–1305, 2015.

[50] Xiaoxue Zhao, Weinan Zhang, and Jun Wang. Interactive collaborative filtering. In *Proceedings of the 22nd CIKM*, pages 1411–1420. ACM, 2013.

[51] Atsuyoshi Nakamura. A ucb-like strategy of collaborative filtering. In *ACML*, 2014.

[52] Shuai Li, Alexandros Karatzoglou, and Claudio Gentile. Collaborative filtering bandits. In *Proceedings of the 39th International ACM SIGIR conference on Research and Development in Information Retrieval*, pages 539–548. ACM, 2016.

[53] Deepak Agarwal and Bee-Chung Chen. Regression-based latent factor models. In *Proceedings of the 15th ACM SIGKDD*, pages 19–28. ACM, 2009.

[54] Steffen Rendle. Factorization machines with libfm. *ACM Transactions on Intelligent Systems and Technology (TIST)*, 3(3):57, 2012.

[55] Hao Ma, Dengyong Zhou, Chao Liu, Michael R Lyu, and Irwin King. Recommender systems with social regularization. In *Proceedings of the fourth ACM international conference on Web search and data mining*, pages 287–296. ACM, 2011.

[56] Hao Ma, Haixuan Yang, Michael R Lyu, and Irwin King. Sorec: social recommendation using probabilistic matrix factorization. In *Proceedings of the 17th ACM conference on Information and knowledge management*, pages 931–940. ACM, 2008.

[57] Frank McSherry and Ilya Mironov. Differentially private recommender systems: Building privacy into the netflix prize contenders. In *Proceedings of the 15th ACM SIGKDD*, pages 627–636. ACM, 2009.

[58] Tianqing Zhu, Gang Li, Yongli Ren, Wanlei Zhou, and Ping Xiong. Differential privacy for neighborhood-based collaborative filtering. In *ASONAM 2013*, pages 752–759. ACM, 2013.

[59] Ziqi Liu, Yu-Xiang Wang, and Alexander Smola. Fast differentially private matrix factorization. In *RecSys 2015*, pages 171–178. ACM, 2015.

[60] Prateek Jain, Om Dipakbhai Thakkar, and Abhradeep Thakurta. Differentially private matrix completion revisited. In *ICML*, pages 2215–2224, 2018.

[61] Cynthia Dwork, Moni Naor, Toniann Pitassi, and Guy N Rothblum. Differential privacy under continual observation. In *Proceedings of the forty-second ACM symposium on Theory of computing*, pages 715–724. ACM, 2010.

[62] T-H Hubert Chan, Elaine Shi, and Dawn Song. Private and continual release of statistics. *ACM Transactions on Information and System Security (TISSEC)*, 14(3):26, 2011.

[63] Jacob Abernethy, Chansoo Lee, Audra McMillan, and Ambuj Tewari. Online learning via differential privacy. *arXiv preprint arXiv:1711.10019*, 2017.

[64] Nikita Mishra and Abhradeep Thakurta. (nearly) optimal differentially private stochastic multi-arm bandits. In *Proceedings of the Thirty-First Conference on Uncertainty in Artificial Intelligence*, pages 592–601. AUAI Press, 2015.

[65] Qingyun Wu, Huazheng Wang, Quanquan Gu, and Hongning Wang. Contextual bandits in a collaborative environment. In *Proceedings of the 39th International ACM SIGIR conference on Research and Development in Information Retrieval*, pages 529–538. ACM, 2016.

[66] Steffen Rendle, Zeno Gantner, Christoph Freudenthaler, and Lars Schmidt-Thieme. Fast context-aware recommendations with factorization machines. In *Proceedings of the 34th International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '11, pages 635–644, New York, NY, USA, 2011. ACM.

[67] André Uschmajew. Local convergence of the alternating least squares algorithm for canonical tensor approximation. *SIAM Journal on Matrix Analysis and Applications*, 33(2):639–652, 2012.

[68] Kamalika Chaudhuri, Claire Monteleoni, and Anand D Sarwate. Differentially private empirical risk minimization. *Journal of Machine Learning Research*, 12(Mar):1069–1109, 2011.

[69] Úlfar Erlingsson, Vasyl Pihur, and Aleksandra Korolova. Rappor: Randomized aggregatable privacy-preserving ordinal response. In *SIGSAC 2014*, pages 1054–1067. ACM, 2014.

[70] Jun Tang, Aleksandra Korolova, Xiaolong Bai, Xueqiang Wang, and Xiaofeng Wang. Privacy loss in apple's implementation of differential privacy on macos 10.12. *arXiv preprint arXiv:1709.02753*, 2017.

[71] Cynthia Dwork, Aaron Roth, et al. The algorithmic foundations of differential privacy. *Foundations and Trends® in Theoretical Computer Science*, 9(3–4):211–407, 2014.

[72] Huazheng Wang, Qingyun Wu, and Hongning Wang. Factorization bandits for interactive recommendation. In *AAAI*, pages 2695–2702, 2017.

[73] Shuai Li, Alexandros Karatzoglou, and Claudio Gentile. Collaborative filtering bandits. In *Proceedings of the 39th International ACM SIGIR*, SIGIR '16, pages 539–548, New York, NY, USA, 2016. ACM.

[74] Claudio Gentile, Shuai Li, Purushottam Kar, Alexandros Karatzoglou, Giovanni Zappella, and Evans Etrue. On context-dependent clustering of bandits. In Doina Precup and Yee Whye Teh, editors, *ICML'17*, volume 70 of *Proceedings of Machine Learning Research*, pages 1253–1262, International Convention Centre, Sydney, Australia, 06–11 Aug 2017. PMLR.

[75] Alex Slivkins and Eli Upfal. Adapting to a changing environment: the brownian restless bandits. In *COLT08'*, pages 343–354, July 2008.

[76] Avishek Ghosh, Sayak Ray Chowdhury, and Aditya Gopalan. Misspecified linear bandits. *CoRR*, abs/1704.06880, 2017.

[77] Fang Liu, Joohyun Lee, and Ness Shroff. A change-detection based framework for piecewise-stationary multi-armed bandit problem. AAAI'18, 2018.

[78] Haipeng Luo, Alekh Agarwal, and John Langford. Efficient contextual bandits in non-stationary worlds. *arXiv preprint arXiv:1708.01799*, 2017.

[79] Yang Cao, Zheng Wen, Branislav Kveton, and Yao Xie. Nearly optimal adaptive procedure for piecewise-stationary bandit: a change-point detection approach. *https://arxiv.org/abs/1802.03692*, 2018.

[80] Cédric Hartland, Sylvain Gelly, Nicolas Baskiotis, Olivier Teytaud, and Michéle Sebag. Multi-armed bandit, dynamic environments and meta-bandits. 2006.

[81] Aurélien Garivier and Eric Moulines. On upper-confidence bound policies for switching bandit problems. In *Proceedings of the 22nd International Conference on Algorithmic Learning Theory*, ALT'11, pages 174–188, Berlin, Heidelberg, 2011. Springer-Verlag.

[82] Yoan Russac, Claire Vernade, and Olivier Cappé. Weighted linear bandits for non-stationary environments. In *Advances in Neural Information Processing Systems*, pages 12017–12026, 2019.

[83] Jia Yuan Yu and Shie Mannor. Piecewise-stationary bandit problems with side observations. In *Proceedings of the 26th Annual International Conference on Machine Learning*, pages 1177–1184. ACM, 2009.

[84] Yang Cao, Wen Zheng, Branislav Kveton, and Yao Xie. Nearly optimal adaptive procedure for piecewise-stationary bandit: a change-point detection approach. *AISTATS,(Okinawa, Japan)*, 2019.

[85] Qingyun Wu, Naveen Iyer, and Hongning Wang. Learning contextual bandits in a non-stationary environment. In *The 41st International ACM SIGIR Conference on Research & Development in Information Retrieval*, pages 495–504. ACM, 2018.

[86] Qingyun Wu, Huazheng Wang, Yanen Li, and Hongning Wang. Dynamic ensemble of contextual bandits to satisfy users' changing interests. In *The World Wide Web Conference*, pages 2080–2090, 2019.

[87] Nicolo Cesa-Bianchi, Claudio Gentile, and Giovanni Zappella. A gang of bandits. In *Advances in Neural Information Processing Systems*, pages 737–745, 2013.

[88] Claudio Gentile, Shuai Li, Purushottam Kar, Alexandros Karatzoglou, Giovanni Zappella, and Evans Etrue. On context-dependent clustering of bandits. In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, pages 1253–1262. JMLR. org, 2017.

[89] Shuai Li, Wei Chen, and Kwong-Sak Leung. Improved algorithm on online clustering of bandits. *arXiv preprint arXiv:1902.09162*, 2019.

[90] David Siegmund. *Sequential analysis: tests and confidence intervals*. Springer Science & Business Media, 2013.

[91] Gregory C Chow. Tests of equality between sets of coefficients in two linear regressions. *Econometrica: Journal of the Econometric Society*, pages 591–605, 1960.

[92] R Stephen Cantrell, Peter M Burrows, and Quang H Vuong. Interpretation and use of generalized chow tests. *International Economic Review*, pages 725–741, 1991.

[93] AL Wilson. When is the chow test ump? *The American Statistician*, 32(2):66–68, 1978.

[94] Lilian Besson and Emilie Kaufmann. The generalized likelihood ratio test meets klucb: an improved algorithm for piece-wise non-stationary bandits. *arXiv preprint arXiv:1902.01575*, 2019.

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

[96] Lihong Li, Wei Chu, John Langford, and Robert E Schapire. A contextual-bandit approach to personalized news article recommendation. In *Proceedings of the 19th international conference on World wide web*, pages 661–670. ACM, 2010.

[97] Yasin Abbasi-Yadkori, Dávid Pál, and Csaba Szepesvári. Improved algorithms for linear stochastic bandits. In *Advances in Neural Information Processing Systems*, pages 2312–2320, 2011.

[98] Aleksandrs Slivkins and Eli Upfal. Adapting to a changing environment: the brownian restless bandits. In *COLT*, pages 343–354, 2008.

[99] L. Wu, Y. Ge, Q. Liu, E. Chen, R. Hong, J. Du, and M. Wang. Modeling the evolution of users #8217; preferences and social links in social networking services. *IEEE Transactions on Knowledge and Data Engineering*, 29(6):1240–1253, June 2017.

[100] Sinan Aral, Lev Muchnik, and Arun Sundararajan. Distinguishing influence-based contagion from homophily-driven diffusion in dynamic networks. *Proceedings of the National Academy of Sciences*, 106(51):21544–21549, 2009.

[101] Xing Yi, Liangjie Hong, Erheng Zhong, Nanthan Nan Liu, and Suju Rajan. Beyond clicks: Dwell time for personalization. In *Proceedings of the 8th ACM Conference on Recommender Systems*, RecSys '14, page 113–120, New York, NY, USA, 2014. Association for Computing Machinery.

[102] Nick Craswell, Onno Zoeter, Michael Taylor, and Bill Ramsey. An experimental comparison of click position-bias models. In *Proceedings of the 2008 International Conference on Web Search and Data Mining*, WSDM '08, page 87–94, New York, NY, USA, 2008. Association for Computing Machinery.

[103] Yi Qi, Qingyun Wu, Hongning Wang, Jie Tang, and Maosong Sun. Bandit learning with implicit feedback. In S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett, editors, *Advances in Neural Information Processing Systems 31*, pages 7276–7286. Curran Associates, Inc., 2018.

[104] Huazheng Wang, Sonwoo Kim, Eric McCord-Snook, Qingyun Wu, and Hongning Wang. Variance reduction in gradient exploration for online learning to rank. In *Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 835–844, 2019.

[105] Herbert Robbins. Some aspects of the sequential design of experiments. *Bull. Amer. Math. Soc.*, 58(5):527–535, 09 1952.

[106] T.L. Lai and H. Robbins. Asymptotically efficient adaptive allocation rules. *Advances in Applied Mathematics*, 6:4–22, 1985.

[107] Peter Auer, Nicolò Cesa-Bianchi, Yoav Freund, and Robert E. Schapire. The nonstochastic multiarmed bandit problem. *SIAM J. Comput.*, 32(1):48–77, January 2003.

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

[109] Naoki Abe, Alan W Biermann, and Philip M Long. Reinforcement learning with immediate rewards and linear hypotheses. *Algorithmica*, 37(4):263–293, 2003.

[110] Ido Guy, Naama Zwerdling, David Carmel, Inbal Ronen, Erel Uziel, Sivan Yogev, and Shila Ofek-Koifman. Personalized recommendation of social software items based on social relations. In *Proceedings of the Third ACM Conference on Recommender Systems*, RecSys '09, pages 53–60, New York, NY, USA, 2009. ACM.

[111] Stephen M. Stigler. The asymptotic distribution of the trimmed mean. *Ann. Statist.*, 1(3):472–477, 05 1973.

[112] Paul Resnick, Neophytos Iacovou, Mitesh Suchak, Peter Bergstrom, and John Riedl. Grouplens: An open architecture for collaborative filtering of netnews. In *Proceedings of the 1994 ACM Conference on Computer Supported Cooperative Work*, CSCW '94, pages 175–186, New York, NY, USA, 1994. ACM.

[113] Eshcar Hillel, Zohar S Karnin, Tomer Koren, Ronny Lempel, and Oren Somekh. Distributed exploration in multi-armed bandits. In C.J.C. Burges, L. Bottou, M. Welling, Z. Ghahramani, and K.Q. Weinberger, editors, *Advances in Neural Information Processing Systems 26*, pages 854–862. Curran Associates, Inc., 2013.

[114] Baruch Awerbuch and Robert Kleinberg. Competitive collaborative learning. *J. Comput. Syst. Sci.*, 74(8):1271–1288, December 2008.

[115] J. Ben Schafer, Joseph Konstan, and John Riedl. Recommender systems in e-commerce. In *Proceedings of the 1st ACM Conference on Electronic Commerce*, EC '99, pages 158–166, New York, NY, USA, 1999. ACM.

[116] Benjamin Marlin and Richard S. Zemel. The multiple multiplicative factor model for collaborative filtering. In *Proceedings of the Twenty-first International Conference on Machine Learning*, ICML '04, pages 73–, New York, NY, USA, 2004. ACM.

[117] John S. Breese, David Heckerman, and Carl Kadie. Empirical analysis of predictive algorithms for collaborative filtering. Technical Report MSR-TR-98-12, Microsoft Research, May 1998.

[118] Julien Delporte, Alexandros Karatzoglou, Tomasz Matuszczyk, and Stéphane Canu. Socially enabled preference learning from implicit feedback data. In *Machine Learning and Knowledge Discovery in Databases*, pages 145–160. Springer, 2013.

[119] Alon Zweig and Gal Chechik. Group online adaptive learning. *Machine Learning*, 106(9):1747–1770, Oct 2017.

[120] Thomas J Walsh, István Szita, Carlos Diuk, and Michael L Littman. Exploring compact reinforcement-learning representations with linear regression. In *Proceedings of the Twenty-Fifth Conference on Uncertainty in Artificial Intelligence*, pages 591–598. AUAI Press, 2009.

[121] Giovanni Cavallanti, Nicolò Cesa-Bianchi, and Claudio Gentile. Linear algorithms for online multitask classification. *J. Mach. Learn. Res.*, 11:2901–2934, December 2010.

[122] Paul M Weichsel. The kronecker product of graphs. *Proceedings of the American Mathematical Society*, 13(1):47–52, 1962.

[123] Filip Radlinski, Robert Kleinberg, and Thorsten Joachims. Learning diverse rankings with multi-armed bandits. In *Proceedings of 25th ICML*, pages 784–791. ACM, 2008.

[124] Yisong Yue and Thorsten Joachims. Interactively optimizing information retrieval systems as a dueling bandits problem. In *Proceedings of 26th ICML*, pages 1201–1208. ACM, 2009.

[125] S.S. Villar, J. Bowden, and J. Wason. Multi-armed bandit models for the optimal design of clinical trials: Benefits and challenges. In *Statist. Sci., 30 (2) (2015)*, pages 199–215, 2015.

[126] Shweta Jain, Balakrishnan Narayanaswamy, and Y. Narahari. A multiarmed bandit incentive mechanism for crowdsourcing demand response in smart grids. In *Proceedings of the Twenty-Eighth AAAI Conference on Artificial Intelligence*, AAAI'14, pages 721–727. AAAI Press, 2014.

[127] Theodoros Evgeniou and Massimiliano Pontil. Regularized multi–task learning. In *Proceedings of the 10th ACM SIGKDD*, pages 109–117. ACM, 2004.

[128] Theodoros Evgeniou, Charles A Micchelli, and Massimiliano Pontil. Learning multiple tasks with kernel methods. In *Journal of Machine Learning Research*, pages 615–637, 2005.

[129] Peter Auer and Chao-Kai Chiang. An algorithm with nearly optimal pseudo-regret for both stochastic and adversarial bandits. In Vitaly Feldman, Alexander Rakhlin, and Ohad Shamir, editors, *29th Annual Conference on Learning Theory*, volume 49 of *Proceedings of Machine Learning Research*, pages 116–120, Columbia University, New York, New York, USA, 23–26 Jun 2016. PMLR.

[130] Rich Caruana. Multitask learning. *Machine Learning*, 28(1):41–75, Jul 1997.

[131] Sebastien Bubeck and Nicolo Cesa-Bianchi. Regret analysis of stochastic and nonstochastic multi-armed bandit problems. *Foundations and Trends in Machine Learning*, 5(1):1–122, 2012.

[132] Zohar S Karnin and Oren Anava. Multi-armed bandits: Competing with optimal sequences. In D. D. Lee, M. Sugiyama, U. V. Luxburg, I. Guyon, and R. Garnett, editors, *NIPS'16*, pages 199–207. Curran Associates, Inc., 2016.

[133] Omar Besbes, Yonatan Gur, and Assaf Zeevi. Stochastic multi-armed-bandit problem with non-stationary rewards. In Z. Ghahramani, M. Welling, C. Cortes, N. D. Lawrence, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems 27*, pages 199–207. Curran Associates, Inc., 2014.

[134] Sébastien Bubeck and Aleksandrs Slivkins. The best of both worlds: stochastic and adversarial bandits. *CoRR*, abs/1202.4473, 2012.

[135] Peter Auer and Chao-Kai Chiang. An algorithm with nearly optimal pseudo-regret for both stochastic and adversarial bandits. In *Conference on Learning Theory*, pages 116–120, 2016.

[136] Thorsten Joachims, Laura Granka, Bing Pan, Helene Hembrooke, and Geri Gay. Accurately interpreting clickthrough data as implicit feedback. In *ACM SIGIR Forum*, volume 51, pages 4–11. Acm, 2017.

[137] Qingyun Wu, Naveen Iyer, and Hongning Wang. Learning contextual bandits in a non-stationary environment. In *Proceedings of the 41th International ACM SIGIR*. ACM, 2018.

[138] Ruslan Salakhutdinov and Andriy Mnih. Probabilistic matrix factorization. Citeseer, 2011.

[139] Chih-Chun Wang, Sanjeev R Kulkarni, and H Vincent Poor. Bandit problems with side observations. *Automatic Control, IEEE Transactions on*, 50(3):338–355, 2005.

[140] Shuang-Hong Yang, Bo Long, Alexander J Smola, Hongyuan Zha, and Zhaohui Zheng. Collaborative competitive filtering: learning recommender using context of user choice. In *Proceedings of the 34th international ACM SIGIR conference on Research and development in Information Retrieval*, pages 295–304. ACM, 2011.

[141] Naoki Abe and Atsuyoshi Nakamura. Learning to optimally schedule internet banner advertisements. In *ICML*, volume 99, pages 12–21, 1999.

[142] Wei Chen, Yajun Wang, and Yang Yuan. Combinatorial multi-armed bandit: General framework and applications. In *International Conference on Machine Learning*, pages 151–159, 2013.

[143] Tommi S Jaakkola and Michael I Jordan. Bayesian parameter estimation via variational methods. *Statistics and Computing*, 10(1):25–37, 2000.

[144] Shipra Agrawal and Navin Goyal. Thompson sampling for contextual bandits with linear payoffs. In *International Conference on Machine Learning*, pages 127–135, 2013.

[145] Lihong Li, Wei Chu, John Langford, and Xuanhui Wang. Unbiased offline evaluation of contextual-bandit-based news article recommendation algorithms. In *Proceedings of the fourth ACM international conference on Web search and data mining*, pages 297–306. ACM, 2011.

[146] Masrour Zoghi, Tomas Tunys, Mohammad Ghavamzadeh, Branislav Kveton, Csaba Szepesvari, and Zheng Wen. Online learning to rank in stochastic click models. In *International Conference on Machine Learning*, pages 4199–4208, 2017.

[147] Branislav Kveton, Csaba Szepesvari, Zheng Wen, and Azin Ashkan. Cascading bandits: Learning to rank in the cascade model. In *International Conference on Machine Learning*, pages 767–776, 2015.

[148] Nick Craswell, Onno Zoeter, Michael Taylor, and Bill Ramsey. An experimental comparison of click position-bias models. In *Proceedings of the 2008 international conference on web search and data mining*, pages 87–94. ACM, 2008.

[149] Huazheng Wang, Qingyun Wu, and Hongning Wang. Learning hidden features for contextual bandits. In *Proceedings of the 25th ACM International on Conference on Information and Knowledge Management*, pages 1633–1642. ACM, 2016.

[150] Richard S Sutton and Andrew G Barto. *Reinforcement learning: An introduction*, volume 1. MIT press Cambridge, 1998.

[151] Marc Abeille, Alessandro Lazaric, et al. Linear thompson sampling revisited. *Electronic Journal of Statistics*, 11(2):5165–5197, 2017.

[152] Daniel Russo and Benjamin Van Roy. Learning to optimize via posterior sampling. *Mathematics of Operations Research*, 39(4):1221–1243, 2014.

[153] Daniel Russo and Benjamin Van Roy. An information-theoretic analysis of thompson sampling. *The Journal of Machine Learning Research*, 17(1):2442–2471, 2016.

[154] Lijun Zhang, Tianbao Yang, Rong Jin, Yichi Xiao, and Zhi-hua Zhou. Online stochastic linear optimization under one-bit feedback. In *International Conference on Machine Learning*, pages 392–401, 2016.

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

[156] Yisong Yue and Carlos Guestrin. Linear submodular bandits and their application to diversified retrieval. In *NIPS*, pages 2483–2491, 2011.

[157] Diane Kelly and Jaime Teevan. Implicit feedback for inferring user preference: a bibliography. In *Acm Sigir Forum*, volume 37, pages 18–28. ACM, 2003.

[158] Yifan Hu, Yehuda Koren, and Chris Volinsky. Collaborative filtering for implicit feedback datasets. In *Data Mining, 2008. ICDM'08. Eighth IEEE International Conference on*, pages 263–272. Ieee, 2008.

[159] Aleksandr Chuklin, Ilya Markov, and Maarten de Rijke. Click models for web search. *Synthesis Lectures on Information Concepts, Retrieval, and Services*, 7(3):1–115, 2015.

[160] Sumeet Katariya, Branislav Kveton, Csaba Szepesvari, and Zheng Wen. Dcm bandits: Learning to rank with multiple clicks. In *International Conference on Machine Learning*, pages 1215–1224, 2016.

[161] Fan Guo, Chao Liu, and Yi Min Wang. Efficient multiple-click models in web search. In *Proceedings of the Second ACM International Conference on WSDM*, pages 124–131. ACM, 2009.

[162] Konstantina Christakopoulou, Filip Radlinski, and Katja Hofmann. Towards conversational recommender systems. In *Proceedings of the 22Nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '16, pages 815–824, New York, NY, USA, 2016. ACM.

[163] Robert M Bell and Yehuda Koren. Lessons from the netflix prize challenge. *ACM SIGKDD Explorations Newsletter*, 9(2):75–79, 2007.

[164] J Ben Schafer, Joseph A Konstan, and John Riedl. E-commerce recommendation applications. In *Applications of Data Mining to Electronic Commerce*, pages 115–153. Springer, 2001.

[165] Paul Resnick, Neophytos Iacovou, Mitesh Suchak, Peter Bergstrom, and John Riedl. Grouplens: an open architecture for collaborative filtering of netnews. In *Proceedings of the 1994 ACM conference on Computer supported cooperative work*, pages 175–186. ACM, 1994.

[166] Nathan Srebro, Tommi Jaakkola, et al. Weighted low-rank approximations. In *ICML*, volume 3, pages 720–727, 2003.

[167] Liangjie Hong, Aziz S Doumith, and Brian D Davison. Co-factorization machines: modeling user interests and predicting individual decisions in twitter. In *Proceedings of the sixth ACM international conference on Web search and data mining*, pages 557–566. ACM, 2013.

[168] Muzafer Sherif. The psychology of social norms. 1936.

[169] Noam Koenigstein, Gideon Dror, and Yehuda Koren. Yahoo! music recommendations: modeling music ratings with temporal dynamics and item taxonomy. In *Proceedings of the fifth ACM conference on Recommender systems*, pages 165–172. ACM, 2011.

[170] Yunhong Zhou, Dennis Wilkinson, Robert Schreiber, and Rong Pan. Large-scale parallel collaborative filtering for the netflix prize. In *International Conference on Algorithmic Applications in Management*, pages 337–348. Springer, 2008.

[171] Cynthia Dwork, Guy N Rothblum, and Salil Vadhan. Boosting and differential privacy. In *2010 IEEE 51st Annual Symposium on Foundations of Computer Science*, pages 51–60. IEEE, 2010.

[172] Peter Kairouz, Sewoong Oh, and Pramod Viswanath. The composition theorem for differential privacy. *IEEE Transactions on Information Theory*, 63(6):4037–4049, 2017.

[173] Kobbi Nissim, Sofya Raskhodnikova, and Adam Smith. Smooth sensitivity and sampling in private data analysis. In *Proceedings of the thirty-ninth annual ACM symposium on Theory of computing*, pages 75–84. ACM, 2007.

[174] Kentaro Minami, HItomi Arai, Issei Sato, and Hiroshi Nakagawa. Differential privacy without sensitivity. In *Advances in Neural Information Processing Systems*, pages 956–964, 2016.

[175] Michael Kearns, Mallesh Pai, Aaron Roth, and Jonathan Ullman. Mechanism design in large games: Incentives and privacy. In *Proceedings of the 5th conference on Innovations in theoretical computer science*, pages 403–410. ACM, 2014.

[176] Aristide CY Tossou and Christos Dimitrakakis. Algorithms for differentially private multi-armed bandits. In *AAAI 2016*, 2016.

[177] Herbert Robbins. Some aspects of the sequential design of experiments. *Bulletin of the American Mathematical Society*, 58(5):527–535, 1952.

[178] William R Thompson. On the likelihood that one unknown probability exceeds another in view of the evidence of two samples. *Biometrika*, 25(3/4):285–294, 1933.

[179] Sébastien Bubeck, Nicolo Cesa-Bianchi, et al. Regret analysis of stochastic and nonstochastic multi-armed bandit problems. *Foundations and Trends® in Machine Learning*, 5(1):1–122, 2012.

[180] Tor Lattimore and Csaba Szepesvári. Bandit algorithms.

[181] Audrey Durand, Charis Achilleos, Demetris Iacovides, Katerina Strati, Georgios D Mitsis, and Joelle Pineau. Contextual bandits for adapting treatment in a mouse model of de novo carcinogenesis. In *Machine Learning for Healthcare Conference*, pages 67–82, 2018.

[182] Lihong Li, Yu Lu, and Dengyong Zhou. Provably optimal algorithms for generalized linear contextual bandits. In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, pages 2071–2080. JMLR. org, 2017.

[183] Huazheng Wang, Qingyun Wu, and Hongning Wang. Factorization bandits for interactive recommendation. In *AAAI 2017*, 2017.

[184] Graham Cormode, Somesh Jha, Tejas Kulkarni, Ninghui Li, Divesh Srivastava, and Tianhao Wang. Privacy at scale: Local differential privacy in practice. In *Proceedings of the 2018 International Conference on Management of Data*, pages 1655–1658. ACM, 2018.

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

[186] Jianqiang Li, Ji-Jiang Yang, Yu Zhao, Bo Liu, Mengchu Zhou, Jing Bi, and Qing Wang. Enforcing differential privacy for shared collaborative filtering. *IEEE Access*, 5:35–49, 2016.

[187] Inderjit S Dhillon, Yuqiang Guan, and Brian Kulis. Weighted graph cuts without eigenvectors a multilevel approach. *IEEE transactions on pattern analysis and machine intelligence*, 29(11):1944–1957, 2007.

[188] Peter Auer, Nicolo Cesa-Bianchi, Yoav Freund, and Robert E Schapire. Gambling in a rigged casino: The adversarial multi-armed bandit problem. In *Proceedings of IEEE 36th Annual Foundations of Computer Science*, pages 322–331. IEEE, 1995.

[189] Peter Auer, Nicolo Cesa-Bianchi, and Paul Fischer. Finite-time analysis of the multiarmed bandit problem. *Machine learning*, 47(2-3):235–256, 2002.

[190] Varsha Dani, Thomas P Hayes, and Sham M Kakade. Stochastic linear optimization under bandit feedback. 2008.

[191] Eric Moulines. On upper-confidence bound policies for non-stationary bandit problems. 1985.

[192] Negar Hariri, Bamshad Mobasher, and Robin Burke. Adapting to user preference changes in interactive recommendation. In *Twenty-Fourth International Joint Conference on Artificial Intelligence*, 2015.

[193] Yifang Chen, Chung-Wei Lee, Haipeng Luo, and Chen-Yu Wei. A new algorithm for non-stationary contextual bandits: Efficient, optimal, and parameter-free. *arXiv preprint arXiv:1902.00980*, 2019.

[194] Odalric-Ambrym Maillard and Shie Mannor. Latent bandits. In *International Conference on Machine Learning*, pages 136–144, 2014.

[195] Robert Kleinberg, Aleksandrs Slivkins, and Eli Upfal. Multi-armed bandits in metric spaces. In *Proceedings of the fortieth annual ACM symposium on Theory of computing*, pages 681–690, 2008.

[196] Daniel Weitekamp, Erik Harpstead, and Ken R. Koedinger. An interaction design for machine teaching to develop ai tutors. In *Proceedings of the 2020 CHI Conference on Human Factors in Computing Systems*, page 1–11, New York, NY, USA, 2020. Association for Computing Machinery.

[197] Qingyun Wu, Chi Wang, and Silu Huang. Cost effective optimization for cost-related hyperparameters, 2020.

Appendix

A Proof of Lemma 1 and Theorem 1

Proof of Lemma 1. Consider the objective function of ridge regression defined in Eq (2.2). By taking the gradient of $L(\mathbf{\Theta})$ with respect to $\mathbf{\Theta}$ and applying our model assumption specified in Eq (2.15), 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 [37],

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

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

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

□

Proof of Theorem 1:

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

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

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,

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

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

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

and therefore, we have

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

And according to Lemma 11 in [37], we have,

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

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

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

□

B Proof of Lemma 9, 10 and Theorem 8

If the training instances $\{(\mathbf{x}_i,r_i)\}_{i\in\mathcal{I}_{m,t}}$ in a linear bandit model come from multiple distributions/environments, we separate the training instances in $\mathcal{I}_{m,t}$ into two sets $\mathcal{H}_{m,t}$ and $\tilde{\mathcal{H}}_{m,t}$ so that instances from $\mathcal{H}_{m,t}$ are from the target stationary distribution, while instances in $\tilde{\mathcal{H}}_{m,t}$ are not. In this case, we provide the confidence bound for the reward estimation in Theorem 13.

Theorem 13 (LinUCB with contamination)**.** *In the LinUCB model with contaminated instances set $\tilde{\mathcal{H}}_{m,t}$, with probability at least $1-\delta_1$, we have,*

$$|\hat{r}_t(m)-\mathbb{E}[r_t]|\leq\tilde{B}_t$$

where $\tilde{B}_t(m,a)=\tilde{\alpha}_t\|\mathbf{x}_{a_t}\|_{\mathbf{A}_{t-1}^{-1}}$ with $\tilde{\alpha}_t=\sigma^2\sqrt{d\ln(1+\frac{|\mathcal{I}_{m,t}|}{\lambda\delta_1})}+\sqrt{\lambda}+C_t(m)$, in which $C_t=\sum_{i\in\tilde{\mathcal{H}}_{m,t}}(\mathbf{x}_{a_i}(\boldsymbol{\theta}_i^*-\boldsymbol{\theta}_{t_c}^*))$

Comparing $\tilde{B}_t(m,a)$ with $B_t(m,a)$, we can see that when the reward derivation of an arm (the $1-\rho$ portion of arms that do not satisfy Eq (2) in Assumption 2) is small with $(1-\rho)\Delta_{\text{small}} \leq \frac{\sigma^2\sqrt{d\ln(1+\frac{|\mathcal{I}_{m,t}|}{\lambda\delta_1})}}{\tilde{\mathcal{H}}_{m,t}}$, same confidence bound scaling can be achieved.

Theorem 14 (Chernoff Bound). *Let $Z_1, Z_2, ..., Z_n$ be random variables on $\mathbb{R}$ such that $a \leq Z_i \leq b$. Define $W_n = \sum_{i=1}^n Z_i$ then for all $c > 0$,*

$$\mathbb{P}(|W_n - \mathbb{E}(W_n)| > c\mathbb{E}(W_n)) \leq 2\exp\Big(\frac{-2c^2\mathbb{E}(W_n)^2}{n(b-a)^2}\Big)$$

Proof of Theorem 8. **Step 1:** If change points can be perfectly detected, the regret of dLinUCB can be bounded by $\sum_{j=0}^{\Gamma_T-1} R_{\text{Lin}}(S_{c_j})$. However, additional regret may accumulate if early detections or late detections happen. In the following two steps, we will bound the possible additional regret from early detection, denoted as R_{early}, and that from late detection, denoted as R_{late}.

Step 2: Define k_{c_j} as the number of early detection within this stationary period $[t_{c_j}, t_{c_{j+1}}]$, with $S_{c_j} = t_{c_{j+1}} - t_{c_j}$. Define p_e as the probability of early detection in the stationary period, we have $\mathbb{P}(k_{c_j} = k) = \binom{S_{c_j}}{k} p_e^k (1-p_e)^{S_{c_j}-k}$. According to Lemma 9, we have $p_e \leq \delta_2$. Combining the property of binomial distribution $B(S_{c_j}, p_e)$ and Chebyshev's concentration inequality, we have $k_{c_j} \leq 2S_{c_j}\delta_2$ with probability at least $1 - \frac{1-\delta_2}{2S_{c_j}\delta_2}$. Hence, with the probability $(1-\delta_3)\times(1-\delta_1)^{kS_{\max}}$, we have $R_{\text{early}} \leq \sum_{j=1}^{\Gamma_T} k_{c_j} R_{\text{Lin}}(\frac{s_{c_j}}{k_{c_j}}) \leq \sum_{j=1}^{\Gamma_T} 2S_{c_j}\delta_2 R_{\text{Lin}}(\frac{1}{2\delta_2}) \leq 2\Gamma_T S_T \delta_2 R_{\text{Lin}}(\frac{1}{2\delta_2})$. Considering the calculation of R_{Lin}, when $\delta_2 \leq \frac{1}{2S_{\max}}$ we have $R_{\text{early}} \leq \Gamma_T R_{\text{Lin}}(S_{\max})$ with a probability at least $(1-\delta_2)(1-\delta_1)$. This upper bounds the additional regret from any possible early detection, and maintains it in the same order as the slave model's.

Step 3. Define $\tilde{k}_{c_j}$ as the number of interactions where the environment has changed (comparing to $\boldsymbol{\theta}^*_{c_j}$) but the change is not detected by the algorithm. The additional regret from this late detection can be bounded by $2\tilde{k}_{c_j}$ (i.e., maximum regret in each round of interaction). Define p_d as the probability of detection after the change happens, we have $\mathbb{P}(\tilde{k}_{c_j} = k) = (1-p_d)^{k-1}p_d$, i.e., a geometric distribution. According to Lemma 10, $p_d \geq 1-\delta_2$. Based on the property of Geometric distribution $G(p_d)$ and Chebyshev's inequality, we have $\tilde{k}_{c_j} \leq \frac{2}{1-\delta_2}$ with probability $1 - \frac{\delta_2}{1-\delta_2}$. If we consider the case where the change point locates inside the sliding window τ, we may have at most another τ delay after each change point. Therefore, the additional regret from late detection can be bounded by $R_{\text{late}} \leq \Gamma_T\big(\tau + \frac{4}{1-\delta_2}\big)$, which is not directly related to the length of any stationary period.

Combining the above three steps concludes the proof. □

Proof sketch of Theorem 7 and Theorem 13. The proof of Eq (3.7) in Theorem 7 and Theorem 13 are mainly based on the proof of Theorem 2 in [37] and the concentration property of Gaussian noise. □

Proof of Lemma 9. According to Chernoff Bound, we have $P(\hat{e}_t(m) \leq \delta_1 + \frac{\ln(1/\delta_2)}{2\tilde{\tau}(m)}) \geq 1-\delta_2$, which finishes the proof. □

Proof of Lemma 10. At time $i \geq t_{c_{j+1}}$, which means the environment has already changed from $\boldsymbol{\theta}^*_{c_j}$ to $\boldsymbol{\theta}^*_{c_{j+1}}$, we have,

$$\begin{aligned}
&\mathbb{P}(e_i(m)=1) = \mathbb{P}(|\hat{r}_i - r_i| > B_i(m,a_i)+\epsilon) \qquad (2)\\
=&\mathbb{P}(|(\mathbf{x}_i^{\mathsf{T}}\hat{\boldsymbol{\theta}}_i - \mathbf{x}_i^{\mathsf{T}}\boldsymbol{\theta}^*_{t_{c_j}} - \eta_i) + (\mathbf{x}_i^{\mathsf{T}}\boldsymbol{\theta}^*_{t_c} - \mathbf{x}_i^{\mathsf{T}}\boldsymbol{\theta}^*_i)| > B_i(m,a_i)+\epsilon)\\
\geq&\mathbb{P}(|\mathbf{x}_i^{\mathsf{T}}\hat{\boldsymbol{\theta}}_i - \mathbf{x}_i^{\mathsf{T}}\boldsymbol{\theta}^*_{t_{c_j}} - \eta_i| \leq \tilde{B}_i(m,a_i)+\epsilon)\\
&\times \mathbb{P}(|\mathbf{x}_i^{\mathsf{T}}\boldsymbol{\theta}^*_{t_{c_j}} - \mathbf{x}_i^{\mathsf{T}}\boldsymbol{\theta}^*_i| > \tilde{B}_i(m,a_i) + B_i(m,a_i) + 2\epsilon)
\end{aligned}$$

According to Theorem 13, we have $\mathbb{P}\big(|\mathbf{x}_i^\mathsf{T}\hat{\boldsymbol{\theta}}_i - \mathbf{x}_i^\mathsf{T}\boldsymbol{\theta}^*_{t_{c_j}} - \eta_i| \leq \tilde{B}_i(m, a_i) + \epsilon\big) \geq 1 - \delta_1$. Define U_{c_j} as the upper bound of $\tilde{B}_i(m, a_i) + B_{t_c}(m, a_i) + 2\epsilon$. If the change gap Δ_{c_j} satisfies $\Delta_{c_j} \geq U_{c_j}$, we have $\mathbb{P}(e_i(m) = 1) \geq \rho(1 - \delta_1)$.

Next, we will prove that $\Delta_{c_j} \geq U_{c_j}$ can be achieved by a properly set δ_1. Similar as the proof in Step 2 of Theorem 8 where we bound k_c, we have with a high probability that $\tilde{B}_i(m, a_i) + B_{t_{c_j}}(m, a_i) + 2\epsilon \leq 2\epsilon + 2\sqrt{\lambda} + \frac{2}{\sqrt{\lambda}} S_{c_j}\rho\big(1 - \rho(1 - \delta_1)\big) = U_{c_j}$. When $\Delta_{c_j} > 2\sqrt{\lambda} + 2\epsilon$, $S_{\text{min}} > \frac{\sqrt{\lambda}}{2\rho}(\Delta_{c_j} - 2\sqrt{\lambda} - 2\epsilon)$ and $\delta_1 \leq 1 - \frac{1}{\rho}(1 - \frac{\sqrt{\lambda}}{2S_{\text{min}}\rho}\big(\Delta_{c_j} - 2\sqrt{\lambda} - 2\epsilon)\big)$, $\Delta_{c_j} > U_{c_j}$ can be achieved.

Eq (2) indicates when the environment has changed for a slave model m, then with high probability $e_i(m) = 1$ and slave model m will not be updated, which avoids possible contamination in m. According to the concentration inequality in Theorem 14, with a probability at least $1 - \delta_2$, we have,

$$\sum_{i=t-\tau}^{t} e_i(m) \geq \mathbb{E}[\sum_{i=t-\tau}^{t} e_i(m)] - \sqrt{\frac{\tau}{2}\ln\frac{1}{\delta_2}} \geq \rho(1 - \delta_1)\tau - \sqrt{\frac{\tau}{2}\ln\frac{1}{\delta_2}}$$

With simple rewriting, we have when $\tau \geq \frac{2\ln\frac{2}{\delta_2}}{(\rho(1-\delta_1)-\delta_1)^2}$, $\rho(1 - \delta_1)\tau - \sqrt{\frac{\tau}{2}\ln\frac{1}{\delta_2}} \geq \delta_1\tau + \sqrt{\frac{\tau}{2}\ln\frac{1}{\delta_2}}$, which means that with probability at least $1 - \delta_2$, $\hat{e}_t(m) = \frac{\sum_{i=t-\tau}^{t} e_i}{\tau} \geq \delta_1 + \sqrt{\frac{1}{2\tau}\ln\frac{1}{\delta_2}}$ □

C Proof of Theorem 9 and Theorem 10

Proof of Theorem 9. **Proof of Eq** (3.11)**:** For the bandit expert m created at time t_m, we split its training instances in $\mathcal{I}_t^\theta(m)$ up to time t into two sets $\mathcal{H}_t(m)$ and $\tilde{\mathcal{H}}_t(m)$. The instances in $\mathcal{H}_t(m)$ are all from the reward distribution governed by $\boldsymbol{\theta}^*_{t_m}$, while the instances in $\tilde{\mathcal{H}}_t(m)$ are not. According to the first order optimum condition in ridge regression and the definition of $\mathbf{A}_t(m)$, the following equation can be obtained,

$$\hat{\boldsymbol{\theta}}_t(m) - \boldsymbol{\theta}^*_{t_m} = \mathbf{A}_t^{-1}\big(\sum_{i\in\mathcal{I}_t^\theta(m)} \mathbf{x}_i\eta_i - \lambda\boldsymbol{\theta}^*_{t_m} - \sum_{i\in\tilde{\mathcal{H}}_t(m)} \mathbf{x}_i\mathbf{x}_i^\mathsf{T}(\boldsymbol{\theta}^*_{t_m} - \boldsymbol{\theta}^*_i)\big)$$

Based on the self-normalized martingale bound in Theorem 1 of [37], for any $\delta_1 \in (0, 1)$, with a probability at least $1 - \delta_1$, we have

$$\|\hat{\boldsymbol{\theta}}_t(m) - \boldsymbol{\theta}^*_{t_m}\|_{\mathbf{A}_t} \leq \sigma^2\sqrt{d\ln\frac{\lambda + |\mathcal{I}_t^\theta(m)|}{\lambda\delta_1}} + \sqrt{\lambda} + \|\sum_{i\in\tilde{\mathcal{H}}_t(m)} \mathbf{x}_i\mathbf{x}_i^\mathsf{T}(\boldsymbol{\theta}^*_t - \boldsymbol{\theta}^*_i)\|_{\mathbf{A}_t^{-1}}$$

To simplify notations, we define $F_t(m) = \sum_{i\in\tilde{\mathcal{H}}_t(m)} \mathbf{x}_i\mathbf{x}_i^\mathsf{T}(\boldsymbol{\theta}^*_t - \boldsymbol{\theta}^*_i)$. Then we have,

$$\begin{aligned} |\mathbf{x}_a^\mathsf{T}\hat{\boldsymbol{\theta}}_t(m) - \mathbf{x}_a^\mathsf{T}\boldsymbol{\theta}^*_{t_m}| &\leq \|\hat{\boldsymbol{\theta}}_t(m) - \boldsymbol{\theta}^*_{t_m}\|_{\mathbf{A}_t}\|\mathbf{x}_{a_t}\|_{\mathbf{A}_t^{-1}} \\ &\leq \big(\sigma^2\sqrt{d\ln\frac{\lambda + |\mathcal{I}_t^\theta(m)|}{\lambda\delta_1}} + \sqrt{\lambda} + \|F_t(m)\|_{\mathbf{A}_t^{-1}}\big)\|\mathbf{x}_{a_t}\|_{\mathbf{A}_t^{-1}} \end{aligned} \tag{3}$$

According to the definition of $F_t(m)$ and $\tilde{H}_t(m)$, $F_t(m)$ can be understood as a form of contamination in bandit expert m's estimation of the ground-truth bandit parameter $\boldsymbol{\theta}^*_{t_m}$. Denote $t_c(m)$ as the time index of the first change point after the bandit expert m is created. When $t > t_c(m)$, essentially, the contamination in $F_t(m)$ comes from two sources: First, erroneous updates from change-sensitive arms, which is referred as false positive selection of bandit experts. In Lemma 10, we proved that with a high probability there will not be any false positive selection up to time t when the auditor's estimation is not contaminated (best case scenario). But we may have additional erroneous updates when the auditor's observations contain change-sensitive arms collected after the change points. Therefore, the possible erroneous updates are at most $2\frac{(t-t_m)}{S_{\text{min}}}\tau$. This also explains why the sliding observation window τ of the bandit auditors cannot be arbitrarily large when we explained in the requirement of this Theorem. The second source of error is the small contamination from change-invariant arms, which can be bounded by $\big(t - t_c(m)\big)\Delta_L$, due to the fact that for change-invariant arms $|\mathbf{x}_a^\top(\boldsymbol{\theta}_{t_m}^* - \boldsymbol{\theta}_i^*)| \leq \Delta_L$. As a result, we have $F_t(m) \leq 2\frac{t-t_m}{S_{\min}}\tau\rho + \big(t - t_c(m)\big)\Delta_L$. Thus when $\Delta_L \leq \frac{\sigma^2\sqrt{d\lambda \ln \frac{\lambda+|\mathcal{I}_t^\theta(m)|}{\lambda\delta_1}}}{t-t_c(m)}$ and $\tau \leq S_{\min}\frac{\sigma^2\sqrt{d\lambda \ln \frac{\lambda+|\mathcal{I}_t^\theta(m)|}{\lambda\delta_1}}}{2\rho(t-t_m)}$, we have $\|F_t(m)\|_{\mathbf{A}_t^{-1}} \leq 2\sigma^2\sqrt{d \ln \frac{\lambda+|\mathcal{I}_t^\theta(m)|}{\lambda\delta_1}}$. Substituting this inequaity into Eq (3) concludes the proof.

Proof of Eq (3.12)**:** The training instances for $\hat{\boldsymbol{\beta}}_t(m)$ are determined by the bandit expert's estimation about the ground-truth reward distribution in the environment. Every time when $\hat{\boldsymbol{\theta}}_t(m)$ gets updated, we will revise the historical training instances in $\hat{\boldsymbol{\beta}}_t(m)$, i.e., compute $e_{a,t}(m)$ by the updated reward estimation. In addition, the bandit auditors only accumulate badness observations in the most recent τ interactions. When there is no environment change in this time window, Eq (3.12) can be easily derived based on [37]. For the case in which there is environment change among the τ observations, wrong selection and update of bandit experts may happen but the number is relatively small, since the number of mistakes is at most τ in each stationary period. The effect of these mistakes will be taken into account in both the bandit expert's reward estimation and the final regret in Theorem 10.

Proof of Eq (3.13)**:** When $t \leq t_c(m)$, Eq (3.13) is equivalent to Eq (3.11). When $t > t_c(m)$, we have $|\mathbf{x}_a^\top\hat{\boldsymbol{\theta}}_t(m) - \mathbf{x}_a^\top\boldsymbol{\theta}_t^*| \leq |\mathbf{x}_a^\top\hat{\boldsymbol{\theta}}_t(m) - \mathbf{x}_a^\top\boldsymbol{\theta}_{t_m}^*| + |\mathbf{x}_a^\top\boldsymbol{\theta}_{t_m}^* - \mathbf{x}_a^\top\boldsymbol{\theta}_t^*|$, in which the first term can be bounded by $B_{a,t}^\theta(m)$ according to Eq (3.11). With Lemma 10, when the bandit expert m is selected for arm a, with a high probability the arm is change-invariant, which means that $|\mathbf{x}_a^\top\boldsymbol{\theta}_{t_m}^* - \mathbf{x}_a^\top\boldsymbol{\theta}_t^*| \leq \Delta_L$. Putting them together concludes the proof. □

Proof of Theorem 10. According to the reward confidence bound proved in Eq (3.13) of Theorem 9, and the UCB arm selection strategy in Algorithm 6, the regret at time t can be upper bounded by,

$$\begin{aligned}\mathbb{E}[r_{a_t^*,t}] - \mathbb{E}[r_{a_t,t}] \leq & \mathbf{x}_{a_t}\hat{\boldsymbol{\theta}}_t(m_{a_t,t}) + B_{t,a_t}^\theta(m_{a_t,t}) + \Delta_L - \mathbf{x}_{a_t}\boldsymbol{\theta}_t^* \\ \leq & 2\Delta_L + 2B_{t,a_t}^\theta(m_{a_t,t}) \end{aligned} \tag{4}$$

Combining with the additional regret caused by the events when a bandit auditor's most recent τ observations contain environment changes, the cumulative regret can be upper bounded by $R(T) \leq 2\Gamma_T\tau + 2T\Delta_L + \sum_{m\in\mathcal{M}_T}\sum_{i\in\Omega_m} 2B_{i,a_i}^\theta(m)$, in which Ω_m is the set of time indices when the bandit expert m is used for arm selection up to time T. According to the proof of Theorem 3 in [37], we have,

$$\sum_{i\in\Omega_m} B_{i,a_i}^\theta(m) \leq \Big(3\sigma^2\sqrt{d\ln\frac{\lambda+|\mathcal{I}_T^\theta(m)|}{\lambda\delta_1}} + \sqrt{\lambda}\Big)\sqrt{|\Omega_m| d \ln(\lambda + \frac{|\Omega_m|}{d})}$$

Since $\sum_{i\in\Omega_m} 2B_{a_i,i}^\theta(m)$ is a concave function with respect to $|\Omega_m|$, according to the Jensen's inequality $\sum_{m\in\mathcal{M}_T}\sum_{i\in\Omega_m} 2B_{i,a_i}^\theta(m) \leq |\mathcal{M}_T|\Big(3\sigma^2\sqrt{d\ln\frac{\lambda+U_T}{\lambda\delta_1}} + \sqrt{\lambda}\Big)\sqrt{S_{\max} d\ln(\lambda + \frac{S_{\max}}{d})}$, in which $U_T = \max\{|\mathcal{I}_T^\theta(m)|\}_{m\in\mathcal{M}_T}$.

Regarding to the number of bandit experts in $\mathcal{M}_T$: denote the possible number of false negative detection as k_{FN}, which follows a binomial distribution $B(T, P_{FN})$. According to Lemma 9 and the tail bound of Binomial distribution, it can be proved that with a high probability, $|\mathcal{M}_T| \leq \Gamma_T + k_{FN} \leq \Gamma_T + 1$. Combining all the conclusions above we have,

$$\begin{aligned} R(T) \leq & 2\Gamma_T\tau + 2T\Delta_L + \sum_{m\in\mathcal{M}_T}\sum_{i\in\mathcal{S}_m} 2B_{i,a_i}^\theta(m) \leq (\frac{1}{\rho} + 2)\sigma^2\sqrt{d\lambda\ln\frac{\lambda+T}{\lambda\delta_1}} \\ & + (\Gamma_T + 1)\Big(3\sigma^2\sqrt{d\ln\frac{\lambda+U_T}{\lambda\delta_1}} + \sqrt{\lambda}\Big)\sqrt{S_T d\ln(\lambda + \frac{S_T}{d})} \end{aligned}$$

which finishes the proof. □

Proofs of Lemma 9 and Lemma 10 are provided in [86].

D Proof of Lemma 11, Lemma 12 and Theorem 12

The statistical test introduced in Section 3.6.1 falls under the category of χ^2 test of homogeneity. Specifically, it is used to test whether the parameters of linear regression models associated with two datasets are the same, assuming equal variance. The test statistic follows noncentral χ^2-distribution $s(D_1, D_2) \sim \chi^2(df, \psi)$, where $df = rank(\mathbf{X}_1) + rank(\mathbf{X}_2) - rank(\begin{bmatrix}\mathbf{X}_1\\ \mathbf{X}_2\end{bmatrix})$ denotes the degree of freedom, and non-centrality parameter $\psi = \frac{1}{\sigma^2}\begin{bmatrix}\mathbf{X}_1\theta_1\\ \mathbf{X}_2\theta_2\end{bmatrix}^\top \left[\mathbf{I}_{m+n} - \begin{bmatrix}\mathbf{X}_1\\ \mathbf{X}_2\end{bmatrix}\left(\mathbf{X}_1^\top\mathbf{X}_1 + \mathbf{X}_2^\top\mathbf{X}_2\right)^- \begin{bmatrix}\mathbf{X}_1^\top & \mathbf{X}_2^\top\end{bmatrix}\right]\begin{bmatrix}\mathbf{X}_1\theta_1\\ \mathbf{X}_2\theta_2\end{bmatrix}$. Proof of this is beyond the scope of this paper. We refer the interested readers to statistics or econometrics literature like [91,92].

Proof of Lemma 11.

When datasets D_1 and D_2 are homogeneous, which means $\theta_1 = \theta_2$, the non-centrality parameter becomes:

$$\begin{aligned}
\psi &= \frac{1}{\sigma^2}\begin{bmatrix}\mathbf{X}_1\theta_1\\ \mathbf{X}_2\theta_1\end{bmatrix}^\top \left[\mathbf{I}_{m+n} - \begin{bmatrix}\mathbf{X}_1\\ \mathbf{X}_2\end{bmatrix}\left(\mathbf{X}_1^\top\mathbf{X}_1 + \mathbf{X}_2^\top\mathbf{X}_2\right)^- \begin{bmatrix}\mathbf{X}_1^\top & \mathbf{X}_2^\top\end{bmatrix}\right]\begin{bmatrix}\mathbf{X}_1\theta_1\\ \mathbf{X}_2\theta_1\end{bmatrix}\\
&= \frac{1}{\sigma^2}\begin{bmatrix}\mathbf{X}_1\theta_1\\ \mathbf{X}_2\theta_1\end{bmatrix}^\top \begin{bmatrix}\mathbf{X}_1\theta_1\\ \mathbf{X}_2\theta_1\end{bmatrix} - \frac{1}{\sigma^2}\begin{bmatrix}\mathbf{X}_1\theta_1\\ \mathbf{X}_2\theta_1\end{bmatrix}^\top \begin{bmatrix}\mathbf{X}_1\\ \mathbf{X}_2\end{bmatrix}\left(\mathbf{X}_1^\top\mathbf{X}_1 + \mathbf{X}_2^\top\mathbf{X}_2\right)^- \begin{bmatrix}\mathbf{X}_1^\top & \mathbf{X}_2^\top\end{bmatrix}\begin{bmatrix}\mathbf{X}_1\theta_1\\ \mathbf{X}_2\theta_1\end{bmatrix}\\
&= \frac{\theta_1^\top(\mathbf{X}_1^\top\mathbf{X}_1 + \mathbf{X}_2^\top\mathbf{X}_2)\theta_1 - \theta_1^\top(\mathbf{X}_1^\top\mathbf{X}_1 + \mathbf{X}_2^\top\mathbf{X}_2)(\mathbf{X}_1^\top\mathbf{X}_1 + \mathbf{X}_2^\top\mathbf{X}_2)^-(\mathbf{X}_1^\top\mathbf{X}_1 + \mathbf{X}_2^\top\mathbf{X}_2)\theta_1}{\sigma^2}\\
&= \frac{\theta_1^\top(\mathbf{X}_1^\top\mathbf{X}_1 + \mathbf{X}_2^\top\mathbf{X}_2)\theta_1 - \theta_1^\top(\mathbf{X}_1^\top\mathbf{X}_1 + \mathbf{X}_2^\top\mathbf{X}_2)\theta_1}{\sigma^2} = 0
\end{aligned}$$

Therefore, when $\theta_1 = \theta_2$, the test statistic $s(D_1, D_2) \sim \chi^2(df, 0)$. The Type-I error probability can be upper bounded by $P(s(D_1, D_2) > \upsilon^e | \theta_1 = \theta_2) \le 1 - F(\upsilon^e; df, 0)$, which finishes the proof of Lemma 11.

Proof of Lemma 12.

Similarly, using the cumulative density function of $\chi^2(df, \psi)$, we can show that the Type-II error probability $P\big(s(D_1, D_2) \le \upsilon | \theta_1 \ne \theta_2\big) \le F(\upsilon; df, \psi)$. As mentioned in Section 3.6.1, the value of ψ depends on the values of unknown parameters θ_1 and θ_2. From the definition of ψ, we know that $\theta_1 = \theta_2$ is only a sufficient condition for $\psi = 0$. The necessary and sufficient condition for $\psi = 0$ is that $\begin{bmatrix}\mathbf{X}_1\theta_1\\ \mathbf{X}_2\theta_2\end{bmatrix}$ is in the column space of $\begin{bmatrix}\mathbf{X}_1\\ \mathbf{X}_2\end{bmatrix}$, e.g., there exists θ such that $\begin{bmatrix}\mathbf{X}_1\theta_1\\ \mathbf{X}_2\theta_2\end{bmatrix} = \begin{bmatrix}\mathbf{X}_1\theta\\ \mathbf{X}_2\theta\end{bmatrix}$. Only when both $\mathbf{X}_1$ and $\mathbf{X}_2$ have a full column rank, $\theta_1 = \theta_2$ becomes the necessary and sufficient condition for $\psi = 0$. This means whenever either $\mathbf{X}_1$ or $\mathbf{X}_2$ is rank-deficient, there always exists θ_1 and θ_2 where $\theta_1 \ne \theta_2$ that would make $\psi = 0$. For example, assuming $\mathbf{X}_1$ is rank-sufficient and $\mathbf{X}_2$ is rank-deficient, then $\psi = 0$ as long as $\theta_1 - \theta_2$ is in the null space of $\mathbf{X}_2$.

To obtain a non-trivial upper bound of the Type-II error probability, or equivalently a non-zero lower bound of the non-centrality parameter ψ, both $\mathbf{X}_1$ and $\mathbf{X}_2$ need to be rank-sufficient. Under this assumption, we can rewrite ψ in the following way to derive its lower bound.

Denote $\epsilon = \theta_2 - \theta_1$. Then $\theta_2 = \theta_1 + \epsilon$. We can decompose $\sigma^2\psi$ as:

$$\begin{aligned}
\sigma^2\psi &= \begin{bmatrix}\mathbf{X}_1\theta_1 \\ \mathbf{X}_2(\theta_1+\epsilon)\end{bmatrix}^T \left[\mathbf{I}_{m+n} - \begin{bmatrix}\mathbf{X}_1 \\ \mathbf{X}_2\end{bmatrix}\left(\mathbf{X}_1^T\mathbf{X}_1 + \mathbf{X}_2^T\mathbf{X}_2\right)^{-1}\begin{bmatrix}\mathbf{X}_1^T & \mathbf{X}_2^T\end{bmatrix}\right]\begin{bmatrix}\mathbf{X}_1\theta_1 \\ \mathbf{X}_2(\theta_1+\epsilon)\end{bmatrix} \\
&= \begin{bmatrix}\mathbf{X}_1\theta_1 \\ \mathbf{X}_2\theta_1\end{bmatrix}^T \left[\mathbf{I}_{m+n} - \begin{bmatrix}\mathbf{X}_1 \\ \mathbf{X}_2\end{bmatrix}\left(\begin{bmatrix}\mathbf{X}_1^T & \mathbf{X}_2^T\end{bmatrix}\begin{bmatrix}\mathbf{X}_1 \\ \mathbf{X}_2\end{bmatrix}\right)^{-1}\begin{bmatrix}\mathbf{X}_1^T & \mathbf{X}_2^T\end{bmatrix}\right]\begin{bmatrix}\mathbf{X}_1\theta_1 \\ \mathbf{X}_2\theta_1\end{bmatrix} \\
&+ \begin{bmatrix}\mathbf{X}_1\theta_1 \\ \mathbf{X}_2\theta_1\end{bmatrix}^T \left[\mathbf{I}_{m+n} - \begin{bmatrix}\mathbf{X}_1 \\ \mathbf{X}_2\end{bmatrix}\left(\begin{bmatrix}\mathbf{X}_1^T & \mathbf{X}_2^T\end{bmatrix}\begin{bmatrix}\mathbf{X}_1 \\ \mathbf{X}_2\end{bmatrix}\right)^{-}\begin{bmatrix}\mathbf{X}_1^T & \mathbf{X}_2^T\end{bmatrix}\right]\begin{bmatrix}0 \\ \mathbf{X}_2\epsilon\end{bmatrix} \\
&+ \begin{bmatrix}0 \\ \mathbf{X}_2\epsilon\end{bmatrix}^T \left[\mathbf{I}_{m+n} - \begin{bmatrix}\mathbf{X}_1 \\ \mathbf{X}_2\end{bmatrix}\left(\begin{bmatrix}\mathbf{X}_1^T & \mathbf{X}_2^T\end{bmatrix}\begin{bmatrix}\mathbf{X}_1 \\ \mathbf{X}_2\end{bmatrix}\right)^{-1}\begin{bmatrix}\mathbf{X}_1^T & \mathbf{X}_2^T\end{bmatrix}\right]\begin{bmatrix}\mathbf{X}_1\theta_1 \\ \mathbf{X}_2\theta_1\end{bmatrix} \\
&+ \begin{bmatrix}0 \\ \mathbf{X}_2\epsilon\end{bmatrix}^T \left[\mathbf{I}_{m+n} - \begin{bmatrix}\mathbf{X}_1 \\ \mathbf{X}_2\end{bmatrix}\left(\begin{bmatrix}\mathbf{X}_1^T & \mathbf{X}_2^T\end{bmatrix}\begin{bmatrix}\mathbf{X}_1 \\ \mathbf{X}_2\end{bmatrix}\right)^{-1}\begin{bmatrix}\mathbf{X}_1^T & \mathbf{X}_2^T\end{bmatrix}\right]\begin{bmatrix}0 \\ \mathbf{X}_2\epsilon\end{bmatrix}
\end{aligned}$$

Since $\begin{bmatrix}\mathbf{X}_1\theta_1 \\ \mathbf{X}_2\theta_1\end{bmatrix}$ is in the column space of $\begin{bmatrix}\mathbf{X}_1 \\ \mathbf{X}_2\end{bmatrix}$, the first term in the above result is zero. The second and third terms can be shown equal to zero as well using the property that matrix product is distributive w.r.t. matrix addition, which leaves us only the last term. Therefore by substituting $\epsilon = \theta_2 - \theta_1$ back, we obtain:

$$\begin{aligned}
\psi &= \frac{(\theta_1-\theta_2)^T\mathbf{X}_2^T\mathbf{X}_2(\mathbf{X}_1^T\mathbf{X}_1 + \mathbf{X}_2^T\mathbf{X}_2)^{-1}\mathbf{X}_1^T\mathbf{X}_1(\theta_1-\theta_2)}{\sigma^2} \\
&\geq \frac{||\theta_1-\theta_2||^2\lambda_{min}\Big(\mathbf{X}_2^T\mathbf{X}_2(\mathbf{X}_1^T\mathbf{X}_1 + \mathbf{X}_2^T\mathbf{X}_2)^{-1}\mathbf{X}_1^T\mathbf{X}_1\Big)}{\sigma^2} \\
&\geq \frac{||\theta_1-\theta_2||^2/\sigma^2}{\frac{1}{\lambda_{min}(\mathbf{X}_1^T\mathbf{X}_1)} + \frac{1}{\lambda_{min}(\mathbf{X}_2^T\mathbf{X}_2)}}
\end{aligned}$$

The first inequality uses Rayleigh-Ritz theorem and the second inequality uses the relation $\mathbf{Y}(\mathbf{X}+\mathbf{Y})^{-1}\mathbf{X} = (\mathbf{X}^{-1}+\mathbf{Y}^{-1})^{-1}$ where $\mathbf{X}$ and $\mathbf{Y}$ are both invertible matrices. This relation can be derived by taking inverse on both sides of the equation $\mathbf{X}^{-1}(\mathbf{X}+\mathbf{Y})\mathbf{Y}^{-1} = \mathbf{X}^{-1}\mathbf{X}\mathbf{Y}^{-1} + \mathbf{X}^{-1}\mathbf{Y}\mathbf{Y}^{-1} = \mathbf{Y}^{-1} + \mathbf{X}^{-1}$.

The results above show that the type-II error probability of the homogeneity test is trivially upper bounded by $F(v; df, 0)$ unless both $\mathbf{X}_1$ and $\mathbf{X}_2$ are rank-sufficient. Intuitively, to have smaller Type-II error probability in the worst case, we need the gap between underlying parameters to be large, noise in the observations to be small, and the number of observations to be large.

E Proof of Lemma 13

To prove Lemma 13, we need the following lemma.

Lemma 14. *When user model $\mathbb{M}_{i_t,t-1}$ is up-to-date, the probability that $e_{i_t,t} = 1$ corresponds to the type-I error probability of an one-sample homogeneity test in Lemma 11. In this case, we have:*

$$\begin{aligned}
P\big(e_{i_t,t} = 1 | \mathbb{M}_{i_t,t-1} \textit{ is up-to-date}\big) &= P\big(s(\mathcal{H}_{i_t,t-1}, \{(\mathbf{x}_t, y_t)\}) > v^e | \mathbb{M}_{i_t,t-1} \textit{ is up-to-date}\big) \\
&\leq 1 - F(v^e; 1, 0)
\end{aligned}$$

Proof of Lemma 13.

As the test variable $e_{i,t} \in \{0,1\}$, it is $\frac{1}{2}$-sub-Gaussian. By applying Hoeffding inequality, we have:

$$P\big(\tau\hat{e}_{i,t} - \tau E[e_{i,t}] \geq h\big) \leq \exp\Big(-\frac{2h^2}{\tau}\Big)$$

Then setting $\delta_e = \exp(-\frac{2h^2}{\tau})$ gives $h = \sqrt{\frac{\tau \log 1/\delta_e}{2}}$. Substituting this back gives us:

$$P\Big(\hat{e}_{i,t} \geq E[e_{i,t}] + \sqrt{\frac{\log 1/\delta_e}{2\tau}}\Big) \leq \delta_e$$

From Lemma 14, we know when $\mathbb{M}_{i,t-1}$ is up-to-date, $E[e_{i,t}] \leq 1 - F(\upsilon^e; 1, 0)$. Therefore, we obtain the following inequality for $\hat{e}_{i,t}$:

$$P\Big(\hat{e}_{i,t} \geq 1 - F(\upsilon^e; 1, 0) + \sqrt{\frac{\log 1/\delta_e}{2\tau}}\Big) \leq \delta_e$$