IEEE Open Journal of Signal Processing

Received 8 December 2023; revised 13 March 2024; accepted 18 March 2024. Date of publication 8 April 2024; date of current version 24 April 2024. The review of this article was arranged by Associate Editor Byung-Jun Yoon.

Digital Object Identifier 10.1109/OJSP.2024.3386490

Non-Stationary Linear Bandits With Dimensionality Reduction for Large-Scale Recommender Systems

SAEED GHOORCHIAN [1], EVGENII KORTUKOV [2], AND SETAREH MAGHSUDI [1,3]

[1]Faculty of Electrical Engineering and Information Technology, Ruhr-University Bochum, 44801 Bochum, Germany
[2]Faculty of Mathematics and Natural Sciences, Tübingen University, 72074 Tübingen, Germany
[3]Fraunhofer Heinrich-Hertz Institute, 10587 Berlin, Germany

CORRESPONDING AUTHOR: SAEED GHOORCHIAN (e-mail: saeed.ghoorchian@uni-tuebingen.de).

This work was supported by the German Federal Ministry of Education and Research (BMBF) under Grant 01IS20051.
This article has supplementary downloadable material available at https://doi.org/10.1109/OJSP.2024.3386490, provided by the authors.

ABSTRACT Taking advantage of contextual information can potentially boost the performance of recommender systems. In the era of Big Data, such side information often has several dimensions. Thus, developing decision-making algorithms to cope with such a high-dimensional context in real time is essential. That is specifically challenging when the decision-maker has a variety of items to recommend. In addition, changes in items' popularity or users' preferences can hinder the performance of the deployed recommender system due to a lack of robustness to distribution shifts in the environment. In this paper, we build upon the linear contextual multi-armed bandit framework to address this problem. We develop a decision-making policy for a linear bandit problem with high-dimensional feature vectors, a large set of arms, and non-stationary reward-generating processes. Our Thompson sampling-based policy reduces the dimension of feature vectors using random projection and uses exponentially increasing weights to decrease the influence of past observations with time. Our proposed recommender system employs this policy to learn the users' item preferences online while minimizing runtime. We prove a regret bound that scales as a factor of the reduced dimension instead of the original one. To evaluate our proposed recommender system numerically, we apply it to three real-world datasets. The theoretical and numerical results demonstrate the effectiveness of our proposed algorithm in making a trade-off between computational complexity and regret performance compared to the state-of-the-art.

INDEX TERMS Decision-making, multi-armed bandit, non-stationary environment, online learning, recommender systems.

I. INTRODUCTION

Over the past decade, recommender systems have benefited the economy by guiding decision-makers in different roles, such as service providers, consumers, and producers, toward cost-effective and time-saving actions while retaining the constraints, such as safety, privacy, and quality-of-service satisfaction. Famous examples of success stories include the recommendation systems deployed in online shopping or streaming websites that provide personalized suggestions to the users [1], [2], [3]. A widely-used metric to evaluate a recommender system is the returned payoff, measured in terms of the users' responses to recommended items. One well-known example is the Click-Through Rate (CTR). Therefore, the decision-making algorithms driving a recommender system aim at maximizing the payoffs over time [4], [5], [6].

Due to the growing demand for online services, recommender systems must serve a large and diverse group of users by providing fast and accurate recommendations from a vast set of available items. To deliver real-time services that match the users' interests, recommender systems take advantage of side information. Thus, building efficient recommender systems becomes challenging in a large-scale scenario with high-dimensional side information and various items [4]. In addition, online recommender systems often

face distribution drifts in the environment where they are deployed. For instance, in personalized news recommendations, customer preferences over news can change over time and exhibit various seasonality patterns [7]. Hence, building robust recommender systems poses a significant challenge due to environmental changes. As the user's interests in items evolve, a learning agent must constantly adapt its decision-making strategy to comply faster with the environmental changes while attempting to keep the runtime as low as possible [8]. Hence, it is imperative to design adaptive and efficient algorithms, in contrast to the traditional offline models where the recommendation engine has to restart the learning from scratch regularly [9], [10].

In this paper, we take advantage of an online framework, namely Multi-Armed Bandit (MAB) [11], to build a recommender system and address the efficiency and robustness challenges mentioned above. The seminal MAB problem portrays a player who sequentially selects an arm from a finite set of arms. Upon being pulled, each arm produces an instantaneous reward following a reward-generating process. The player does not know the reward-generating processes of arms in advance, which might result in sub-optimal selection of arms, hence, losing rewards. Consequently, the decision-maker might try to improve his decisions over time by learning, that is, sampling seemingly sub-optimal arms to obtain some information that might result in optimal actions only in the future. A decision-making strategy shall assist the player in maximizing the cumulative reward by finding a balance between collecting information (exploration or learning) and accumulating rewards (exploitation or control).

The multi-armed bandit framework can accommodate different models of the reward-generating process of arms. In the most basic one, namely, *stationary multi-armed bandits*, the reward process follows a stationary distribution. That means that the expected reward does not change over time. Therefore, the player tries to find the arm with the highest average reward as efficiently as possible. Despite being simple, the stationary model does not suit many real-world applications, where the environment varies dynamically. To address that challenge, we resort to *piece-wise stationary multi-armed bandit models*, where the reward-generating process of each arm follows a specific distribution over some intervals and changes from one interval to another, so that the expected reward remains constant for some time but then changes at some unknown point. As a result, the optimal arm in the sense of highest average reward evolves over time, and the player must keep track of the changes. This type of multi-armed bandits is the building block of our contribution.

Besides the non-stationary model, in this paper, we also take advantage of the contextual setting. In the Contextual Multi-Armed Bandit (CMAB) problem [12], each arm associates with a context vector. At each round of decision-making, the player observes these contexts before selecting an arm. We consider a CMAB problem with high-dimensional context vectors and a large number of arms whose associated rewards follow a non-stationary linear model; the unknown model parameter can vary in time. The state-of-the-art methods that address such a problem [13], [14], [15], [16], [17] either suffer from excessive computational complexity and weak regret performance, e.g., their regret bound scales as a factor of the context vectors' dimension, or do not take into account the non-stationarity of the environment. To address these shortcomings, we propose a Thompson sampling (TS)-based policy that uses Random Projection (RP) to perform dimensionality reduction. We choose the random projection method for dimensionality reduction as it is computationally efficient [18], [19], [20]; for a matrix of size $k \times n$ with k observed data and n features, the time complexity of RP is $O(nkd)$ [21], [22], [23], where d is the reduced dimension size. Therefore, the primary benefit of random projection lies in accelerating computations, albeit with a potential minor decline in performance, as evident from our numerical results. In addition, RP is a memory-efficient method as we do not need to store the past data points to perform projection to a lower dimensional space. In addition, our algorithm uses weighted least-squares as an efficient method to estimate the reduced model parameter while gradually forgetting past interactions. Our proposed algorithm guarantees an upper regret bound that depends on the reduced dimension instead of the original dimension of context vectors. We use three real-world datasets to evaluate our proposed recommender system. Numerical results demonstrate the efficacy of our proposed algorithm in making a trade-off between computational complexity and regret performance in non-stationary environments compared to the state-of-the-art.

In the following, we present the problem setting and notations. We then compare our work with state-of-the-art. In Section II, we propose our decision-making strategy and introduce our algorithm, namely `D-LinTS-RP`. Section III includes the theoretical analysis of the regret performance of D-LinTS-RP. Section IV is dedicated to numerical evaluation. Section V concludes the paper.

A. PROBLEM SETTING AND NOTATIONS

We denote the set of arms by $\mathcal{A} = \{1, 2, \ldots, A\}$. For each arm $a \in \mathcal{A}$, $\boldsymbol{x}_{a,t} \in \mathbb{R}^n$ represents its corresponding random context vector at time t. Let $r_{a,t}$, $\forall a \in \mathcal{A}, \forall t$, represent the random reward corresponding to the arm a at time t. The instantaneous rewards of each arm a at each time t are independent random variables drawn from an unknown probability distribution. In this paper, we consider a non-stationary linear bandit model; that is, the reward $r_{a,t}$ for each arm $a \in \mathcal{A}$ is linear with respect to the context vector $\boldsymbol{x}_{a,t}$, and there exists an unknown time-varying parameter vector $\boldsymbol{\theta}_t^* \in \mathbb{R}^n$ such that

$$r_{a,t} = \boldsymbol{x}_{a,t}^\top \boldsymbol{\theta}_t^* + \eta_t \tag{1}$$

where η_t is a conditionally R-subGaussian zero-mean random noise, where $R \geq 0$ is a fixed constant. We assume that $\|\boldsymbol{x}_{a,t}\|_2 \leq 1$, $\forall a \in \mathcal{A}$, and $\|\boldsymbol{\theta}_t^*\|_2 \leq 1$. Therefore, $|\langle \boldsymbol{\theta}_t^*, \boldsymbol{x}_{a,t} \rangle| \leq 1$.

The agent's goal is to maximize its total accumulated reward over a finite time horizon T. Alternatively, the agent

aims to minimize the expected dynamic regret, defined as

$$\mathbb{E}[\mathcal{R}(T)] = \mathbb{E}\left[\sum_{t=1}^{T}\left[\boldsymbol{x}_{a_t^*,t}^\top \boldsymbol{\theta}_t^* - \boldsymbol{x}_{a_t,t}^\top \boldsymbol{\theta}_t^*\right]\right] \tag{2}$$

where $a_t^* = \arg\max_{a\in\mathcal{A}} \boldsymbol{x}_{a,t}^\top \boldsymbol{\theta}_t^*$ is the optimal arm at time t, and a_t denotes the played arm at time t under the applied policy.

By $\boldsymbol{I}_{d\times d}$ and $\boldsymbol{0}_d$, we denote an identity matrix of size $d \times d$ and a zero vector of dimension d, respectively. $\mu_{\min}(\boldsymbol{Z})$ represents the minimum eigenvalue of a positive definite matrix $\boldsymbol{Z}$. Moreover, for a positive definite matrix $\boldsymbol{Z} \in \mathbb{R}^{d\times d}$ and any vector $\boldsymbol{y} \in \mathbb{R}^d$, we define the norm $\|\boldsymbol{y}\|_{\boldsymbol{Z}} = \sqrt{\boldsymbol{y}^\top \boldsymbol{Z}\boldsymbol{y}}$.

B. RELATED WORKS

Online methods such as reinforcement learning and multi-armed bandit algorithms are popular bases to design recommender systems. Some examples include [24], [25], [26], [27]. The core concept is to design algorithms that balance exploration and exploitation to maximize the total payoff over time. In the context of recommender systems, exploration means learning the payoff of new items by recommending those items to users. Exploitation involves recommending the best item to users using the collected data. Besides exploration-exploitation balance, another important criterion is to maximize the total reward while keeping the runtime as low as possible. That results in faster services, and thereby a higher users' satisfaction level.

The contextual bandit framework serves as a conventional model to formalize and solve recommendation problems. Some recent works include [27], [28], [29], [30], [31], [32]. Despite being designed to solve large-scale problems, the performance of the state-of-the-art methods depends strongly on the number of items and the dimension of the context vectors. For instance, in [27], the authors consider the linear contextual bandit problem and propose the `BallExplore` algorithm to model and solve a recommendation problem with high-dimensional context vectors. They prove a regret bound that is proportional to the original dimension of context vectors. Also, the proposed algorithm runs in quadratic time regarding the original dimension n. As another example, in [26], the authors propose an algorithm for personalized news article recommendation that also runs in quadratic time w.r.t the original dimension n. In contrast, the time complexity of our proposed algorithm is linear concerning the original dimension of contextual data.

Other recent works investigate the high-dimensional CMAB problem. Some of these approaches achieve significant improvement for the regret bounds; nonetheless, they require additional knowledge or assumptions about the characteristics of the context vectors. For example, in [33], the authors consider a sparse linear bandit problem and propose an Upper Confidence Bound (UCB)-based policy that uses the algorithm developed in [34] as a subroutine. They establish an upper regret bound of order $\tilde{O}(\sqrt{nST})$, where S is the maximum number of non-zero components in the context vector. Furthermore, the authors in [35] propose a policy which achieves a regret bound of order $O(S\sqrt{T})$. The development and the analysis are based on the assumption that the set of context vectors is the unit ball in $\mathbb{R}^n$. In comparison with the research works described above, the authors in [36] make several additional assumptions, e.g., on the expected covariance matrix of the samples and on the distribution of the context vectors. In return, their proposed policy achieves a regret bound of order $O(S^2[\log(T) + \log(n)]^2)$. Besides, in [37], the authors study the high-dimensional linear contextual bandit problem assuming that the set of contexts are sparse; i.e., only a subset of contexts is correlated with the reward. The proposed algorithm achieves a regret bound that scales logarithmically with the original dimension n. Further, [38] uses a combinatorial bandit algorithm as a subroutine to select d entries of context vectors out of n, thereby reducing the dimension of each context vector at each decision-making round. The reduced context vectors are used to update the posterior distribution on the reward parameter. This work does not provide any theoretical analysis for the regret bound.

Our work extends the state-of-the-art research in the area of contextual bandits. In the following, we first review notably-related research works on stationary bandits and highlight the novelty of our approach. We then continue with reviewing the related works on non-stationary bandits. In [14], the authors study the CMAB problem with linear payoffs. They propose a UCB-based algorithm, namely `LinUCB`, that achieves a regret bound of order $O(\sqrt{Tn\ln^3(AT\frac{\ln(T)}{\delta})})$, $\delta \in (0, 1)$. Likewise, in [39], the authors develop the decision-making policy `LinRel` that achieves a regret bound similar to that in [14]. Reference [13] proposes `OFUL`, a UCB-based algorithm that achieves a regret bound of order $O(n\log(T)\sqrt{T} + \sqrt{nT\log(\frac{T}{\delta})})$. In [15], the authors utilize Thompson sampling to develop `LinTS` algorithm with a regret bound of order $O(n\sqrt{T}(\min\{\sqrt{n}, \sqrt{\ln(A)}\})(\ln(T) + \sqrt{\ln(T)\ln(\frac{1}{\delta})}))$. In [16], the authors propose a UCB-based algorithm `CBRAP` by using the random projection in combination with a UCB-based algorithm developed in [13]. The aforementioned algorithms either are not suitable for large-scale problems, i.e., they show poor regret or runtime performance in large-scale scenarios, or do not take into account the non-stationarity of the environment.

Real-world recommender systems often serve users whose preferences evolve over time. A recent line of research on linear contextual bandits is devoted to designing algorithms capable of handling this non-stationarity in the environments [17], [40], [41], [42]. For example, in [40], the authors study linear stochastic bandit in a drifting environment. They propose `SW-UCB` algorithm that uses a sliding window to estimate the unknown parameter of the linear bandit and achieves a regret bound of order $\tilde{O}(n^{2/3}(B_T + 1)^{1/3}T^{2/3})$, where B_T is the variation budget on the unknown parameter vector. Reference [41] examines the same problem in both slowly-varying and abruptly-changing environments. The authors propose `D-LinUCB`, a UCB-based algorithm that uses exponentially

increasing weights to gradually forget past observations and achieves a regret bound of order $O(n^{2/3}B_T^{1/3}T^{2/3})$. In [42], the authors show that a simple strategy based on periodically restarting a UCB-style algorithm is sufficient to achieve the same performance in terms of regret. In [17], two perturbation approaches based on LinUCB and LinTS algorithms are developed to address the non-stationary stochastic linear bandit problem. The proposed algorithms, namely `D-RandLinUCB` and `D-LinTS`, achieve regret bounds of order $O(n^{2/3}B_T^{1/3}T^{2/3})$ and $O(n^{2/3}(\log A)^{1/3}B_T^{1/3}T^{2/3})$, respectively. The aforementioned algorithms rely on the original context vectors; thus, they suffer high computational costs in large-scale scenarios. In contrast, our algorithm enjoys a regret bound that scales as a factor of a reduced dimension $d < n$ while it adapts to drifts in the environment.

In the following section, we describe our proposed decision-making strategy to minimize the expected dynamic regret defined in (2).

II. DECISION-MAKING STRATEGY

As mentioned before, the agent's goal is to minimize the expected dynamic regret (2) via learning the unknown model parameter $\boldsymbol{\theta}_t^*$ from history up to time $t-1$, $\mathcal{H}_{t-1} = \{\boldsymbol{x}_{a_\tau,\tau}, r_{a_\tau,\tau}\}_{\tau=1}^{t-1}$. As discussed above, working with high-dimensional data points affects the runtime and regret performance of bandit algorithms. Our proposed decision-making strategy alleviates this effect by reducing the dimension of each context vector using the RP method. More specifically, we project the data points in the original space $\mathbb{R}^n$ to a random lower-dimensional space $\mathbb{R}^d$, $d < n$, using a randomly designed projection matrix $\boldsymbol{P} \in \mathbb{R}^{d\times n}$ whose columns are scaled to have unit length. It is common to design the matrix $\boldsymbol{P}$ such that each entry of $\boldsymbol{P}$ is a realization of independent and identically distributed (i.i.d.) zero-mean variables with Gaussian distribution [21], [23], [43]. Therefore, at time t, $\boldsymbol{z}_{a,t} = \boldsymbol{P}\boldsymbol{x}_{a,t}, \forall a \in \mathcal{A}$.

As we are now working in the lower-dimensional space $\mathbb{R}^d$, the player's goal is to learn the unknown parameter $\boldsymbol{\psi}_t^* = \boldsymbol{P}\boldsymbol{\theta}_t^*$, from history up to time $t-1$, $\mathcal{H}'_{t-1} = \{\boldsymbol{z}_{a_\tau,\tau}, r_{a_\tau,\tau}\}_{\tau=1}^{t-1}$. This means that, based on our model and solution, the player does not have access to the full context vectors and can only observe the d-dimensional vectors $\boldsymbol{z}_{a,t}, \forall a, t$. To learn the unknown parameter $\boldsymbol{\psi}_t^*$, we rely on the weighted l^2-regularized least-squares estimator with discount factor $\gamma \in (0,1)$. At each time step, the old observations are weighted by the discount factor $\gamma < 1$, which diminishes the effect of samples in the least-squares estimation as they become outdated. That mechanism allows the estimated parameter $\hat{\boldsymbol{\psi}}_t$ to track the changes in the true unknown parameter, enabling the algorithm to adapt to an evolving environment swiftly. Formally, the estimated parameter $\hat{\boldsymbol{\psi}}_t$ at time t is obtained as

$$\hat{\boldsymbol{\psi}}_t = \underset{\boldsymbol{\psi}\in\mathbb{R}^d}{\arg\min} \left(\sum_{\tau=1}^{t} \gamma^{t-\tau} \left(r_{a_\tau,\tau} - \boldsymbol{\psi}^\top \boldsymbol{z}_{a_\tau,\tau} \right)^2 + \frac{\lambda}{2} \|\boldsymbol{\psi}\|_2^2 \right). \tag{3}$$

Algorithm 1: D-LinTS-RP: Discounted Linear Thompson Sampling with Random Projection.

Input: Parameters $d, \kappa, \lambda \geq 1, \xi > 0$, and $0 < \gamma < 1$.
Initialize: $\boldsymbol{Z}_1 = \lambda \boldsymbol{I}_{d\times d}$, $\tilde{\boldsymbol{Z}}_1 = \lambda \boldsymbol{I}_{d\times d}$, and $\boldsymbol{b}_1 = \boldsymbol{0}_d$.
for $i = 1, \ldots, d$ **do**
for $j = 1, \ldots, n$ **do**
Generate $g_{i,j} \sim \mathcal{N}(0, \kappa^2)$ and assign $\boldsymbol{P}[i, j] = g_{i,j}$.
end for
end for
for $t = 1, \ldots, T$ **do**
Calculate $\hat{\boldsymbol{\psi}}_t = \boldsymbol{Z}_t^{-1}\boldsymbol{b}_t$.
Observe the new context vectors $\boldsymbol{x}_{a,t}, \forall a \in \mathcal{A}$.
Calculate $\boldsymbol{z}_{a,t} = \boldsymbol{P}\boldsymbol{x}_{a,t}, \forall a \in \mathcal{A}$.
Calculate $\tilde{\boldsymbol{\psi}}_t = \hat{\boldsymbol{\psi}}_t + \boldsymbol{Z}_t^{-1}\tilde{\boldsymbol{Z}}_t^{1/2}\boldsymbol{W}$, where $\boldsymbol{W} \sim \mathcal{N}(\boldsymbol{0}_d, \xi^2 \boldsymbol{I}_{d\times d})$.
Select arm $a_t = \arg\max_{a\in\mathcal{A}} \tilde{\boldsymbol{\psi}}_t^\top \boldsymbol{z}_{a,t}$ and observe reward $r_{a_t,t}$.
Update $\boldsymbol{Z}_{t+1} = \gamma \boldsymbol{Z}_t + \boldsymbol{z}_{a_t,t}\boldsymbol{z}_{a_t,t}^\top + (1-\gamma)\lambda \boldsymbol{I}_{d\times d}$.
Update $\tilde{\boldsymbol{Z}}_{t+1} = \gamma^2 \tilde{\boldsymbol{Z}}_t + \boldsymbol{z}_{a_t,t}\boldsymbol{z}_{a_t,t}^\top + (1-\gamma^2)\lambda \boldsymbol{I}_{d\times d}$.
Update $\boldsymbol{b}_{t+1} = \gamma \boldsymbol{b}_t + r_{a_t,t}\boldsymbol{z}_{a_t,t}$.
end for

where $\lambda > 0$ is a regularization parameter. At each time t, The closed form of the weighted least-squares estimator can be calculated as $\hat{\boldsymbol{\psi}}_t = \boldsymbol{Z}_t^{-1}\boldsymbol{b}_t$, where $\boldsymbol{Z}_t = \sum_{\tau=1}^{t-1} \gamma^{-\tau} \boldsymbol{z}_{a_\tau,\tau}\boldsymbol{z}_{a_\tau,\tau}^\top + \lambda\gamma^{-(t-1)}\boldsymbol{I}_{d\times d}$ and $\boldsymbol{b}_t = \sum_{\tau=1}^{t-1} \gamma^{-\tau} r_{a_\tau,\tau}\boldsymbol{z}_{a_\tau,\tau}$. In addition, at each time t, we define $\tilde{\boldsymbol{Z}}_t = \sum_{\tau=1}^{t-1} \gamma^{-2\tau} \boldsymbol{z}_{a_\tau,\tau}\boldsymbol{z}_{a_\tau,\tau}^\top + \lambda\gamma^{-2(t-1)}\boldsymbol{I}_{d\times d}$.

Our proposed decision-making strategy, D-LinTS-RP, is summarized inAlgorithm 1. As mentioned before, in the initial phase, D-LinTS-RP constructs the random projection matrix $\boldsymbol{P}$ as a random matrix whose elements are drawn from a normal distribution $\mathcal{N}(0, \kappa^2)$, where κ is a parameter of the algorithm. At each time t, similar to [17], our algorithm perturbs the estimated parameter $\hat{\boldsymbol{\psi}}_t$ via a multivariate Gaussian perturbation $\boldsymbol{W} \sim \mathcal{N}(\boldsymbol{0}_d, \xi^2 \boldsymbol{I}_{d\times d})$, with ξ being a tunable parameter. Afterward, D-LinTS-RP calculates the perturbed estimate $\tilde{\boldsymbol{\psi}}_t$ and selects the arm that has the highest value of $\tilde{r}_{a,t} = \tilde{\boldsymbol{\psi}}_t^\top \boldsymbol{z}_{a,t}$. Finally, it observes the corresponding reward value and updates the model parameter using the reduced context vector of the selected arm and the corresponding reward.

Our randomized algorithm can be efficiently implemented when the set of arms is large. Moreover, D-LinTS-RP adapts to parameter changes by using the discount factor, thereby reducing the influence of past observations with time. The computational complexity of D-LinTS-RP is polynomial w.r.t. the lower dimension d. We observe that for a fixed d, the computational complexity of D-LinTS-RP scales linearly w.r.t. the original dimension n. This is an improvement over the previous methods, such as the works proposed in [13], [14], [15], and [27].

III. THEORETICAL ANALYSIS

The following theorem states an upper bound on the expected dynamic regret of the decision policy D-LinTS-RP, summarized in Algorithm 1.

Theorem 1: Let $\kappa^2 = \frac{1}{d}$ and $B_T = \sum_{t=1}^{T-1} \|\boldsymbol{\theta}_t^* - \boldsymbol{\theta}_{t+1}^*\|_2$. For any $\varepsilon \in (0, 1)$ and $\lambda \geq 1$, with probability at least $1 - 2\exp(-\frac{d\varepsilon^2}{8})$, the expected dynamic regret of D-LinTS-RP is upper bounded as

$$\mathbb{E}[\mathcal{R}(T)] = O\left(\frac{\log(T)B_T}{1-\gamma} + T\left(\exp\left(-d\varepsilon^2\right) + \varepsilon\left(1 + \sqrt{\frac{d}{T}\log(A)}\right)\right)\right). \tag{4}$$

Proof: See Section III-A of supplementary material.■

The original dimension n does not appear in our regret bound, which is an improvement over the previous works that directly scale with n. Note that although the regret bound (4) depends on the reduced dimension d, choosing a small d does not necessarily reduce the regret as, in this case, the obtained regret bound holds with a low probability. Indeed, choosing d to be too small might even increase the regret due to the excessive information loss. Also, as mentioned before, choosing a smaller value of d improves the running time of our proposed algorithm. Therefore, selecting a suitable value for the reduced parameter d is crucial for achieving a low computational complexity while ensuring negligible regret. We elaborate on this trade-off in our numerical analysis in the next section. Our algorithm does not require the knowledge of the total variation budget B_T. However, as we will see in our numerical analysis, it requires a suitable reduced dimension and a tuned discount factor as the input to achieve efficient runtime and regret performance.

IV. NUMERICAL ANALYSIS

In this section, we provide more insights into the effects of high-dimensional features and environmental changes on the performance of learning algorithms. Besides, we clarify how our proposed algorithm mitigates the adverse effects on runtime and regret performance by reducing the feature dimensions and adapting to drifts, respectively. We also compare the performance of our algorithm with conventional benchmarks using three real-world datasets. In particular, we study the following issues through numerical experiments: (i) The performance of our proposed decision-making policy compared to benchmark algorithms in terms of runtime, Click-Through-Rate (CTR), and Normalized Discounted Cumulative Gain (NDCG); (ii) the effect of the reduced dimension d on the performance of our algorithm; (iii) the trade-off between computational complexity and regret bound together with the balance found by our algorithm, in particular, in comparison with the theoretical results. The source code for our algorithm and experiments in this paper are publicly available.[1]

[1]Source code: https://github.com/saeedghoorchian/D-LinTS-RP.git

TABLE 1. Summary of Datasets' Characteristics

Dataset	#Arms (A)	#Features (n)	#Unique Users
MovieLens 10M	1000	120	2885
Jester	140	300	59 132
Amazon Books	400	200	7000

Baselines: We compare our algorithm with state-of-the-art context-aware and context-agnostic algorithms. Context-aware benchmarks in our experiment can be divided into four categories. First, we consider **D-LinTS** [17], which is designed for non-stationary environments and uses the original context vector with dimension n to select arms. Second, we consider **CBRAP** [16], which is designed for bandit problems with high dimensions in stationary environments. Similar to our algorithm, CBRAP can reduce the dimension of original features at each time of play. As a result, we expect that they enjoy lower computational costs compared to other benchmarks. Third, we consider **LinTS** [15] that is neither designed for changing environments nor high-dimensional features. It utilizes the original context vectors with dimension n to select arms in stationary environments and has a Bayesian approach similar to our algorithm. As the last context-aware benchmark, we consider **DeepFM** [10], which is a state-of-the-art algorithm designed for CTR prediction, a technique widely employed when designing offline recommender systems. This is in contrast to the online nature of our proposed algorithm.

As the context-agnostic benchmark, we choose ε-**Greedy** [44], which is a standard method despite its weakness due to being blind to contextual information. It does not incur a high computational cost as it does not rely on feature observations and works only based on collected rewards. In contrast, LinTS, D-LinTS, and DeepFM always observe all features. Hence, they incur a higher computational cost compared to other benchmarks. We also consider a **random** policy that selects an action uniformly at random at each time.

A. SETTINGS AND DATA PREPARATION

We evaluate the performance of our algorithm using three real-world datasets. In the following, we introduce each dataset individually and explain the data preparation steps for our experiments. Table 1 presents a summary of the datasets used in our experiment.

MovieLens 10M: This dataset contains users' ratings and tag applications applied to a set of movies from the MovieLens website [45]. The ratings have a 5-star scale, with half-star increments. Thus, the possible values for rating are $0.5, 1, 1.5, \ldots, 5$. In our experiment, we select the top $A = 1000$ movies based on the number of ratings given by the users. We form the context vector for each user by using the movies that the user has watched together with the tags he applied to those movies. Tags in the MovieLens dataset correspond to movie genres. There are 20 possible tags in total. For example, the movie "Toy Story (1995)" has the

following tags: "Adventure | Animation | Children | Comedy | Fantasy". As each movie can have multiple genres, we enumerate the genre tags and turn them into multi-hot vectors. To achieve contextual attributes for each user, we average the multi-hot vectors corresponding to all the movies the user has watched that fall outside the top 1000 ones considered in the experiment. That way, each column in the user's contextual information corresponds to the estimated probability with which the user watches movies of a particular genre. Afterward, we extract latent context vectors for each arm (movie), using a low-rank matrix factorization with 100 latent contexts. Then, we represent the context vector of the movie-user pair by concatenating the user and the movie context vectors. The dimension of the final context vectors is $n = 120$. We generate a user stream by considering only the users that have rated any of the $A = 1000$ movies. We take the users in the order of their appearances in the data. Hence, it is possible that a specific user appears more than once in our experiment. In this case, we sort the appearances of this specific user according to the timestamps. Our experiment with MovieLens dataset contains 2,885 unique users. At each time, one user from the user stream arrives, the environment reveals the context vectors, and the algorithm needs to recommend one of $A = 1000$ movies to the user. We employ the implicit feedback model to generate rewards for the benchmark algorithms; if there is a rating present in the dataset, this indicates user interest, and we assign a reward equal to 1. Otherwise, the reward is 0. Hence, the reward is 0 for an unwatched movie.

Jester: It consists of more than 1.7 million joke ratings on a continuous scale from -10 to 10 for $A = 140$ jokes [46]. We extract latent contexts for representing the users and arms (jokes), using a low-rank matrix factorization with 150 latent contexts. We then concatenate these context vectors to create the context vector of each joke-user pair. Hence, the dimension of the final context vectors is $n = 300$. In the Jester dataset, the time of user-item interaction is unavailable. Hence, to create a user stream, we sample users from the original dataset uniformly at random with replacement. This procedure results in 59,132 unique users in our experiment. The algorithm recommends a joke to an incoming user and receives a reward of 1 if the corresponding rating is greater than 0. If the rating is less than 0 or no rating for a joke by a user exists, then the algorithm collects a reward equal to 0. This pre-processing step rests on an assumption that a missing rating corresponds to a user not being interested in a joke. During the creation of the Jester dataset, the jokes were shown to users sequentially. A user not rating a joke means they stopped using the Jester website before seeing it. We interpret this as the user losing interest; thereby, we disincentivize the algorithm from recommending such jokes to the user.

Amazon Books: This dataset is a subset of the 2018 Amazon Review Data [47] that contains 51,311,621 book ratings on a discrete scale from 0 to 5. The original data spans a period from May 1996 to October 2018. As the ratings in the original data are highly sparse, we use a subset of the original rating data from 15 December 2012 to 15 June 2013 to form a user stream for our experiment. From this subset of ratings, we extract $A = 400$ items that have the most reviews. Then, we pick the 7,000 most active users (with the most number of rated books) amongst those users that have rated the 400 items in the experiment. We sample users from this set uniformly at random with replacement to form the final user stream. We extract latent contexts for representing the users and books using a low-rank matrix factorization with latent space dimension 100. We concatenate these latent vectors to create the final context vector of each book-user pair with dimension $n = 200$. Similar to Movielens 10 M dataset, we consider an implicit feedback recommendation system model. When the algorithm recommends an item to a user, if the rating for this user-item pair is present in the original data, the reward is 1. Otherwise, the reward is 0.

Using the aforementioned setup, we create two user streams for each dataset as the validation and evaluation data with 30,000 and 100,000 time steps, respectively. We use the validation data to tune the hyperparameters of each benchmark algorithm by performing a grid search. In Section IV of the supplementary material, we present a detailed explanation of tuning the parameters and their corresponding grids. Table 3 lists the tuned parameters of algorithms used in our experiments. We use the training data to evaluate the algorithms on $T = 100,000$ user appearances in each user stream. In order to simulate a non-stationary environment during evaluation, we introduce change points at times $\{5000, 10000, 20000, 35000, 50000, 65000, 80000, 90000\}$. At every change point, we cyclically shift the arms backward by one-third of the size of the arms' set. For example, for MovieLens 10 M dataset, we shift the arm indices by 333 at each change point. This means if the algorithm chooses arm k after a change point, it receives the reward it would get from arm $(k + 333 \bmod 1000)$ before the change point. This ensures a piece-wise stationary expected reward for each arm throughout the experiment. This way of introducing non-stationarity can correspond to a shift in users' preferences or in items' popularity.

Remark 1: Assuming abrupt changes is common in the literature concerning piece-wise stationary multi-armed bandits; nonetheless, it is necessary to mention that in most real-world applications, the environment evolves gradually. Although the state-of-the-art algorithms, including ours, work also in case of gradual changes, their performance degrades as naturally, detecting small changes is more troublesome and associated with frequent false alarms and missed detections. Still, note that in some applications, detecting small changes is not essential, because they degrade the performance only slightly. As such, the system would intentionally ignore small changes to maintain efficiency, leading to a model similar to the one considered here, i.e., abrupt changes.

To deploy the DeepFM in an online recommender system setting, we proceed as follows. During an initial exploration phase of length 1000, arms are chosen uniformly at random. After the exploration phase, we re-train the model from scratch every 1000 time steps on all the already observed

TABLE 2. Comparison of Cumulative Reward, Cumulative NDCG@5, and Time Consumption of Different Policies Corresponding to Different Datasets and Context Dimensions

Dataset	Policy	Context Dimension	Runtime (second)	Cumulative Reward	Cumulative NDCG@5
MovieLens 10M	D-LinTS	120	1739.8	**74771.6**	**48499.1**
	LinTS	120	1370.9	53814.0	33572.3
	D-LinTS-RP (Proposed)	24	**1109.2**	70130.0	39509.2
		60	1231.5	74294.6	44014.6
	CBRAP	24	1917.0	57314.0	32521.0
		60	1989.3	59425.0	34382.5
	DeepFM	120	4174.4	25419.8	23624.2
Jester	D-LinTS	300	6800.4	**44134.6**	**33895.4**
	LinTS	300	2769.7	36695.2	30340.7
	D-LinTS-RP (Proposed)	60	610.6	43382.8	32116.6
		150	1554.3	43693.2	33030.9
	CBRAP	60	**521.3**	26171.0	22018.0
		150	673.6	25081.0	22004.3
	DeepFM	300	3912.7	19275.0	22513.9
Amazon Books	D-LinTS	200	2549.3	**35018.2**	**12929.0**
	LinTS	200	1315.6	9513.6	4594.4
	D-LinTS-RP (Proposed)	40	**586.2**	34118.0	12257.4
		100	973.2	34654.0	12603.2
	CBRAP	40	833.8	12500.0	4948.5
		100	918.5	5779.0	3073.5
	DeepFM	200	4377.9	5562.8	3402.3

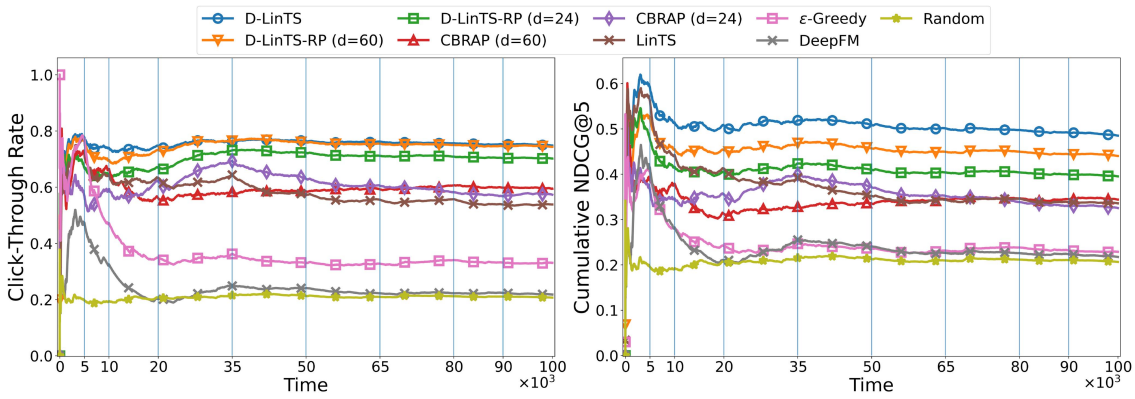
The reported values are averaged over five repetitions. The bold numbers represent the best results for each dataset. The underlined numbers are the results for our proposed algorithm.

context-reward pairs. After that, we use the DeepFM model to estimate the expected reward of each user-item pair using the given context vector and then choose the arm with the highest estimated reward. After the reward is revealed by the environment, it is saved in the model memory to be used later for re-training the model. This way, we deploy a CTR prediction model in our experiments such that it can make use of newly gathered data over time.

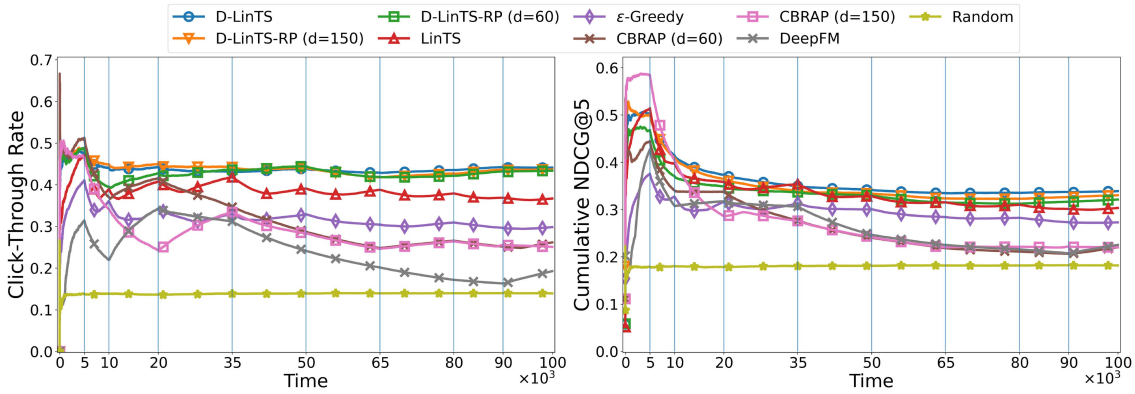
We run the algorithms on the evaluation data for each dataset using the aforementioned setup for 5 repetitions and report the results by averaging over the repetitions. Random projection matrix, if used, stays the same for each repetition. We report the average runtime, the average cumulative reward, and the average cumulative NDCG@5 of each policy. For D-LinTS-RP and CBRAP, we reduce the context dimension to 5%, 10%, 20%, and 50% of the original context dimension to analyze the effect of the reduced dimension d on the algorithms' performance. For the sake of presentation, in Table 2, we list some selected results corresponding to context-aware benchmarks and reduced dimensions equal to 20% and 50% of the original dimension, and defer the full results to Section V of the supplementary material. All the policies are evaluated on a single compute node with 64 Intel Xeon Gold 6226R CPUs and 64 G of RAM. To simplify the presentation, we show the best results for each dataset with bold numbers and the results for our proposed algorithm with underlined numbers.

CTR and NDCG Comparison: We depict the average CTR and the average cumulative NDCG@5 of different policies for the MovieLens 10 M, Jester, and Amazon Books datasets in Fig. 1(a)–(c), respectively. The results show the importance of adapting to a non-stationary environment; algorithms that adapt to changes in the reward-generating processes, i.e., D-LinTS and D-LinTS-RP, achieve higher CTR and NDCG than policies that do not recommend items adaptively. As we see, D-LinTS and D-LinTS-RP achieve comparable results; however, D-LinTS-RP uses the reduced context vectors, performing more efficiently in terms of runtime compared to D-LinTS. Note that, DeepFM performs poorly in terms of achieved reward in our experiments. This is due to the fact that DeepFM policy chooses arms based on estimated rewards, effectively doing only exploitation and no exploration. In addition, re-training the model from scratch with newly collected data does not help the DeepFM to improve its performance compared to other online benchmark methods.

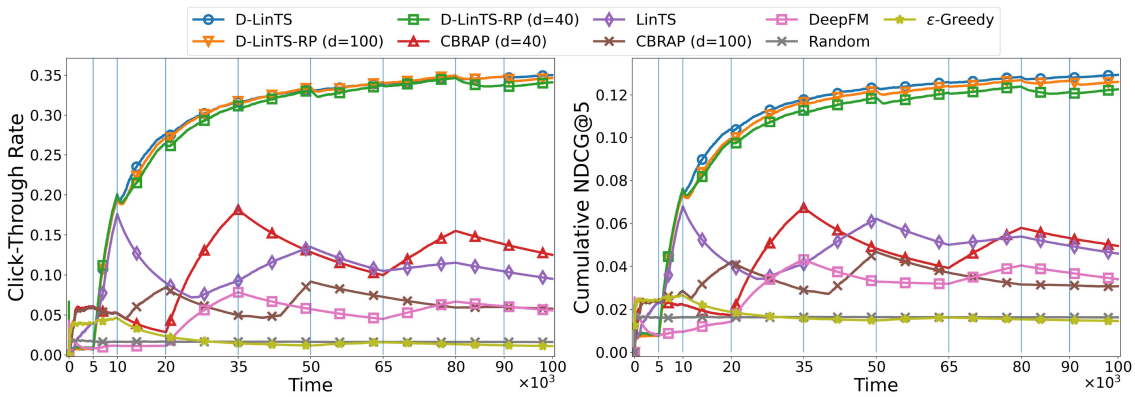
The figures depicting the cumulative NDCG@5 provide additional insight into the relative performance of the algorithms. The NDCG metric assesses the ability of the



(a) Results for MovieLens 10M dataset.



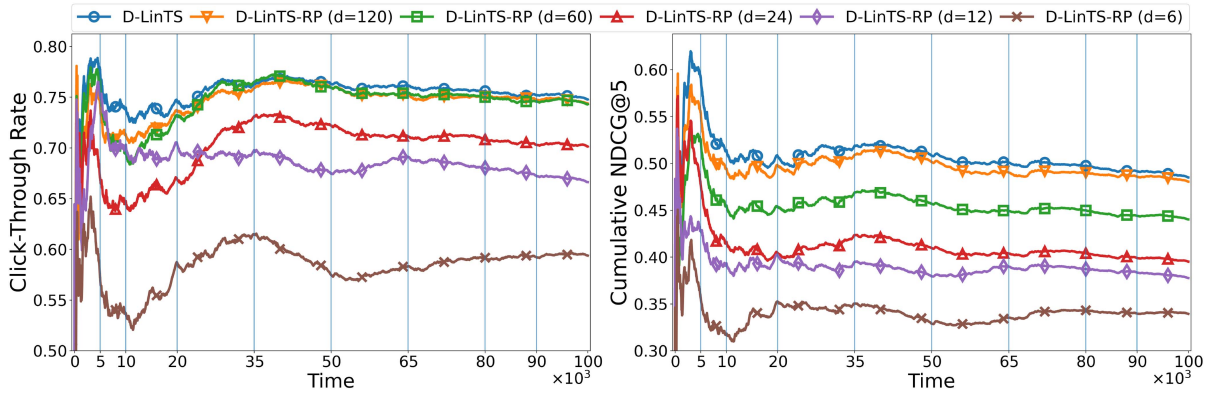
(b) Results for Jester dataset.



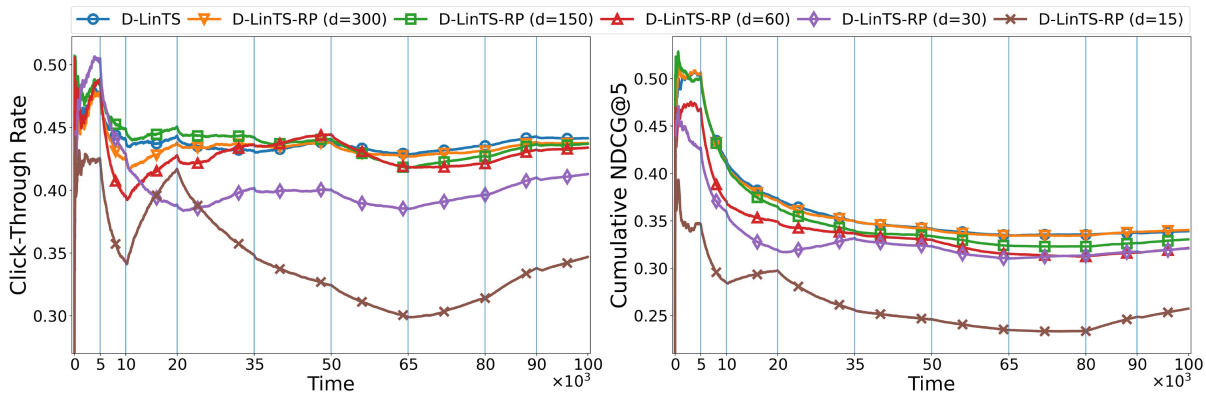
(c) Results for Amazon Books dataset.

FIGURE 1. **CTR and NDCG@5 of different policies over time. Vertical lines show the change points.**

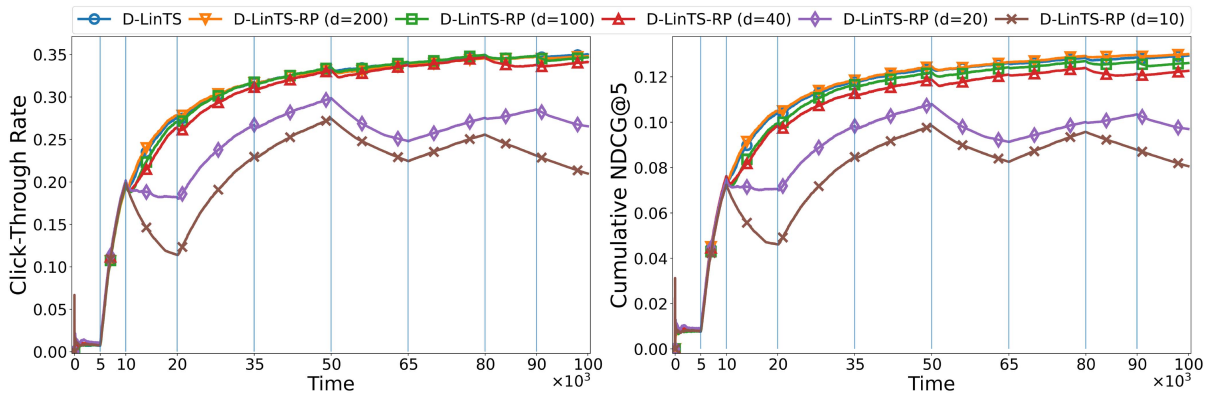
evaluated policies to rank the items. NDCG@5 is defined as $\frac{DCG@5}{IDCG@5}$ where $DCG@5 = \sum_{i=1}^{5} \frac{rel_i}{\log_2 i+1}$ and $IDCG@5$ is Ideal DCG@5, or the highest achievable DCG@5 for the given collection of items. Moreover, rel_i denotes the true relevance of the item at position i, and the items' positions are obtained by sorting the items according to the predicted relevance. In words, NDCG quantifies how well an algorithm can predict the relevance of each item. In our experiments, instantaneous arm rewards are used as true relevance scores for NDCG computation. We adapt the bandit policies to output the best arm and a score for each arm at each decision-making round. The scores are defined based on the decision-making strategy used by an algorithm and lead to a natural ranking of arms induced by that policy. Therefore, we use these scores as predicted relevance. For D-LinTS, LinTS, and D-LinTS-RP, the score of each arm $a \in \mathcal{A}$ is the estimated expected reward $\tilde{r}_{a,t} = \tilde{\boldsymbol{\psi}}_t^\top z_{a,t}$. For CBRAP, the upper confidence bound is used as the predicted relevance. For DeepFM and ε-Greedy, estimated rewards are used as the predicted relevance. For the random policy, the predicted relevance values are sampled



(a) Results for MovieLens 10M dataset.



(b) Results for Jester dataset.



(c) Results for Amazon Books dataset.

FIGURE 2. CTR and NDCG@5 of D-LinTS and D-LinTS-RP policies over time. Vertical lines show the change points.

uniformly at random from the [0, 1] interval. This approach allows us to assess the ability of the evaluated policies to not only identify the best item to recommend to a user but also to rank a set of relevant items in a manner that accurately reflects their relevance. As can be seen from the figures, D-LinTS-RP, with 50% and 80% reduction in context dimension, exhibits a performance close to that of D-LinTS and outperforms all the other benchmark policies on this ranking metric. As expected, the ability to rank the items gradually diminishes as we consider smaller values for the reduced context dimension.

Effect of Reduced Dimension d: To further elaborate on the impact of the reduced dimension d, we compare the CTR and NDCG performance of the D-LinTS-RP with various reduced dimensions to those of the D-LinTS algorithm. In Fig. 2(a)–(c), we depict the results for MovieLens 10 M, Jester, and Amazon Books datasets, respectively. When we increase

the reduced dimension d, the performance of D-LinTS-RP approaches that of D-LinTS in terms of CTR and NDCG, with D-LinTS-RP matching D-LinTS eventually by using d equal to the original context dimension. Therefore, for large-scale recommender systems with changing users' interests, D-LinTS-RP is a better choice than D-LinTS, provided that we use a suitable reduced dimension as the input to the algorithm.

As evident from the theoretical and numerical results, the choice of the reduced dimension d impacts the performance of the D-LinTS-RP algorithm. When we increase the reduced dimension d, the regret of D-LinTS-RP decreases, and the performance of D-LinTS-RP approaches that of D-LinTS in terms of both runtime and reward. Although larger values of d expand the regret bound, this does not necessarily mean that the achieved cumulative reward will be different in practice. That is the case for our experiment on the Jester dataset, where the cumulative reward is not affected much as we decrease the value of reduced dimension d.

Trade-off between Computational Complexity and Regret Bound: The discussion above suggests that the reduced dimension d makes a trade-off between the computational complexity of our algorithm and its regret performance. As expected, large d increases the runtime, while choosing a small d might yield information loss, thereby harming the performance w.r.t. the accumulated rewards. However, the results show that there are reduced dimensions using which D-LinTS-RP's runtime drops significantly while the algorithm continues to enjoy a high cumulative reward (See Tables 4, 5, and 6 in the supplementary material for the full results). For example, for the MovieLens dataset, reducing the context dimension to 50% results in just a 0.1% drop in achieved reward while reducing the runtime by 29.7%. For the Jester dataset, reducing the context dimension to 20% results in only a 0.8% drop in accumulated reward, while the runtime of the algorithm decreases significantly by 90%. Finally, for the Amazon Books dataset, reducing the context dimension to 20% results in a 2% reduction in accumulated reward but leads to a 77% decrease in the runtime.

V. CONCLUSION

We developed a decision-making policy, namely D-LinTS-RP, for the linear CMAB problem that is implementable in recommender systems. D-LinTS-RP is specifically suitable for scenarios with a large set of items, high-dimensional side information, and non-stationary environments. The policy utilizes a weighted least-squares estimator and takes advantage of random projection and exponentially increasing weights to reduce the dimension of the context vectors and the influence of past observations, respectively. We theoretically analyzed D-LinTS-RP and proved an upper regret bound that depends on the reduced dimension of the context vector. For numerical evaluation, we apply D-LinTS-RP on real-world datasets for content recommendation. The results demonstrate its effectiveness in making a trade-off between computational complexity and regret performance in non-stationary environments. Besides developing online content recommender systems, our work fits several real-world application domains, such as edge computing, medical applications, and stock trading.

REFERENCES

[1] P. Resnick and H. R. Varian, "Recommender systems," *Commun. ACM*, vol. 40, no. 3, pp. 56–58, Mar. 1997, doi: 10.1145/245108.245121.

[2] G. Adomavicius and A. Tuzhilin, "Toward the next generation of recommender systems: A survey of the state-of-the-art and possible extensions," *IEEE Trans. Knowl. Data Eng.*, vol. 17, no. 6, pp. 734–749, Jun. 2005.

[3] M. Hu, D. Wu, R. Wu, Z. Shi, M. Chen, and Y. Zhou, "RAP: A light-weight privacy-preserving framework for recommender systems," *IEEE Trans. Serv. Comput.*, vol. 15, no. 5, pp. 2969–2981, Sep./Oct. 2022.

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

[5] J. Louëdec, M. Chevalier, J. Mothe, A. Garivier, and S. Gerchinovitz, "A multiple-play bandit algorithm applied to recommender systems," in *Proc. Twenty-Eighth Int. FLAIRS Conf.*, 2015, pp. 67–72.

[6] Z. Cui et al., "Personalized recommendation system based on collaborative filtering for IoT scenarios," *IEEE Trans. Serv. Comput.*, vol. 13, no. 4, pp. 685–695, Jul./Aug. 2020.

[7] Q. Wu, H. Wang, Y. Li, and H. Wang, "Dynamic ensemble of contextual bandits to satisfy users' changing interests," in *Proc. World Wide Web Conf.*, 2019, pp. 2080–2090, doi: 10.1145/3308558.3313727.

[8] J. Vinagre, A. M. Jorge, and J. Gama, "An overview on the exploitation of time in collaborative filtering," *Wiley Interdiscipl. Rev.: Data Mining Knowl. Discov.*, vol. 5, no. 5, pp. 195–215, Aug. 2015, doi: 10.1002/widm.1160.

[9] M. Al-Ghossein, T. Abdessalem, and A. Barré, "A survey on stream-based recommender systems," *ACM Comput. Surv.*, vol. 54, no. 5, pp. 1–36, May 2021, doi: 10.1145/3453443.

[10] H. Guo, R. Tang, Y. Ye, Z. Li, and X. He, "DeepFM: A factorization-machine based neural network for CTR prediction," in *Proc. 26th Int. Joint Conf. Artif. Intell.*, 2017, pp. 1725–1731.

[11] H. Robbins, "Some aspects of the sequential design of experiments," *Bull. Amer. Math. Soc.*, vol. 58, no. 5, pp. 527–535, 1952. [Online]. Available: http://www.projecteuclid.org/DPubS/Repository/1.0/Disseminate?view=body&id=pdf_1&handle=euclid.bams/1183517370

[12] S. Maghsudi and E. Hossain, "Multi-armed bandits with application to 5G small cells," *IEEE Wireless Commun.*, vol. 23, no. 3, pp. 64–73, Jun. 2016, doi: 10.1109/MWC.2016.7498076.

[13] Y. Abbasi-Yadkori, D. Pál, and C. Szepesvári, "Improved algorithms for linear stochastic bandits," in *Proc. Annu. Conf. Neural Inf. Process. Syst.*, 2011, pp. 2312–2320.

[14] W. Chu, L. Li, L. Reyzin, and R. Schapire, "Contextual bandits with linear payoff functions," in *Proc. 14th Int. Conf. Artif. Intell. Statist.*, 2011, pp. 208–214.

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

[16] X. Yu, M. R. Lyu, and I. King, "CBRAP: Contextual bandits with random projection," in *Proc. 31st AAAI Conf. Artif. Intell.*, 2017, pp. 2859–2866.

[17] B. Kim and A. Tewari, "Randomized exploration for non-stationary stochastic linear bandits," in *Proc. 36th Conf. Uncertainty Artif. Intell.*, 2020, pp. 71–80. [Online]. Available: https://proceedings.mlr.press/v124/kim20a.html

[18] I. K. Fodor, "A survey of dimension reduction techniques," Lawrence Livermore National Lab., Livermore, CA, USA, Tech. Rep. UCRL-ID-148494, 2002.

[19] W. Zhang, L. Zhang, R. Jin, D. Cai, and X. He, "Accelerated sparse linear regression via random projection," in *Proc. 13th AAAI Conf. Artif. Intell.*, 2016, pp. 2337–2343.

[20] X. Z. Fern and C. E. Brodley, "Random projection for high dimensional data clustering: A cluster ensemble approach," in *Proc. 20th Int. Conf. Mach. Learn.*, 2003, pp. 186–193.

[21] E. Bingham and H. Mannila, "Random projection in dimensionality reduction: Applications to image and text data," in *Proc. 7th ACM SIGKDD Int. Conf. Knowl. Discov. Data Mining*, 2001, pp. 245–250.
[22] S. Dasgupta, "Experiments with random projection," in *Proc. 16th Conf. Uncertainty Artif. Intell.*, 2000, pp. 143–151.
[23] A. Blum, "Random projection, margins, kernels, and feature-selection," in *Proc. Subspace, Latent Struct. Feature Selection*, 2005, pp. 52–68.
[24] G. Shani, D. Heckerman, R. I. Brafman, and C. Boutilier, "An MDP-based recommender system," *J. Mach. Learn. Res.*, vol. 6, no. 9, 2005, 1265–1295.
[25] I. Munemasa, Y. Tomomatsu, K. Hayashi, and T. Takagi, "Deep reinforcement learning for recommender systems," in *Proc. Int. Conf. Inf. Commun. Technol.*, 2018, pp. 226–233.
[26] L. Li, W. Chu, J. Langford, and R. E. Schapire, "A contextual-bandit approach to personalized news article recommendation," in *Proc. 19th Int. Conf. World Wide Web*, 2010, pp. 661–670, doi: 10.1145/1772690.1772758.
[27] Y. Deshpande and A. Montanari, "Linear bandits in high dimension and recommendation systems," in *Proc. 50th Annu. Allerton Conf. Commun., Control, Comput.*, 2012, pp. 1750–1754.
[28] L. Tang, R. Rosales, A. Singh, and D. Agarwal, "Automatic ad format selection via contextual bandits," in *Proc. 22nd ACM Int. Conf. Inf. Knowl. Manage.*, 2013, pp. 1587–1594.
[29] K. Mahadik, Q. Wu, S. Li, and A. Sabne, "Fast distributed bandits for online recommendation systems," in *Proc. 34th ACM Int. Conf. Supercomputing*, 2020, doi: 10.1145/3392717.3392748.
[30] N. Korda, B. Szorenyi, and S. Li, "Distributed clustering of linear bandits in peer to peer networks," in *Proc. Int. Conf. Mach. Learn.*, 2016, pp. 1301–1309.
[31] S. Li, W. Chen, S. Li, and K.-S. Leung, "Improved algorithm on online-clustering of bandits," in *Proc. 28th Int. Joint Conf. Artif. Intell.*, 2019, pp. 2923–2929.
[32] S. Li, A. Karatzoglou, and C. Gentile, "Collaborative filtering bandits," in *Proc. 39th Int. ACM SIGIR Conf. Res. Develop. Inf. Retrieval*, 2016, pp. 539–548.
[33] Y. Abbasi-Yadkori, D. Pal, and C. Szepesvari, "Online-to-confidence-set conversions and application to sparse stochastic bandits," in *Proc. 15th Int. Conf. Artif. Intell. Statist.*, 2012, pp. 1–9.
[34] S. Gerchinovitz, "Sparsity regret bounds for individual sequences in online linear regression," *J. Mach. Learn. Res.*, vol. 14, no. 1 pp. 729–769, 2013.
[35] A. Carpentier and R. Munos, "Bandit theory meets compressed sensing for high dimensional stochastic linear bandit," in *Proc. Int. Conf. Artif. Intell. Statist.*, 2012, pp. 190–198.
[36] H. Bastani and M. Bayati, "Online decision making with high-dimensional covariates," *Operations Res.*, vol. 68, pp. 276–294, 2019.
[37] G.-S. Kim and M. C. Paik, "Doubly-robust lasso bandit," in *Proc. Annu. Conf. Neural Inf. Process. Syst.*, 2019, pp. 5877–5887.
[38] D. Bouneffouf, I. Rish, G. A. Cecchi, and R. Feraud, "Context attentive bandits: Contextual bandit with restricted context," in *Proc. Int. Joint Conf. Artif. Intell.*, 2017. [Online]. Available: https://api.semanticscholar.org/CorpusID:3093677
[39] P. Auer, "Using confidence bounds for exploitation-exploration tradeoffs," *J. Mach. Learn. Res.*, vol. 3, pp. 397–422, 2002.
[40] W. C. Cheung, D. Simchi-Levi, and R. Zhu, "Learning to optimize under non-stationarity," in *Proc. 22nd Int. Conf. Artif. Intell. Statist.*, 2019, pp. 1079–1087. [Online]. Available: https://proceedings.mlr.press/v89/cheung19b.html
[41] Y. Russac, C. Vernade, and O. Cappé, "Weighted linear bandits for non-stationary environments," in *Proc. Annu. Conf. Neural Inf. Process. Syst.*, 2019. [Online]. Available: https://proceedings.neurips.cc/paper_files/paper/2019/hash/263fc48aae39f219b4c71d9d4bb4aed2-Abstract.html
[42] P. Zhao, L. Zhang, Y. Jiang, and Z.-H. Zhou, "A simple approach for non-stationary linear bandits," in *Proc. 23rd Int. Conf. Artif. Intell. Statist.*, 2020, pp. 746–755. [Online]. Available: https://proceedings.mlr.press/v108/zhao20a.html
[43] B. Ghojogh, A. Ghodsi, F. Karray, and M. Crowley, "Johnson-lindenstrauss lemma, linear and nonlinear random projections, random fourier features, and random kitchen sinks: Tutorial and survey," 2021, *arXiv:2108.04172*.
[44] P. Auer, N. Cesa-Bianchi, and P. Fischer, "Finite-time analysis of the multiarmed bandit problem," *Mach. Learn.*, vol. 47, no. 2/3, pp. 235–256, 2002.
[45] F. M. Harper and J. A. Konstan, "The movielens datasets: History and context," *ACM Trans. Interactive Intell. Syst.*, vol. 5, no. 4, pp. 1–19, Dec. 2015, doi: 10.1145/2827872.
[46] K. Goldberg, T. Roeder, D. Gupta, and C. Perkins, "Eigentaste: A constant time collaborative filtering algorithm," *Inf. Retrieval*, vol. 4, no. 2, pp. 133–151, 2001, doi: 10.1023/a:1011419012209.
[47] J. Ni, J. Li, and J. McAuley, "Justifying recommendations using distantly-labeled reviews and fine-grained aspects," in *Proc. Conf. Empirical Methods Natural Lang. Process. 9th Int. Joint Conf. Natural Lang. Process.*, 2019, pp. 188–197. [Online]. Available: https://aclanthology.org/D19-1018

SAEED GHOORCHIAN received the B.Sc. degree in pure mathematics from the Iran University of Science and Technology, Tehran, Iran, in 2014, and the joint M.Sc. degree in applied mathematics from the University of Hamburg, Hamburg, Germany, in 2017. He was a Guest Researcher with the Technical University of Hamburg, Hamburg, in 2018. In 2023, he successfully defended his Ph.D. thesis with the University of Tübingen. During 2023–2024, he held a Postdoctoral position with the University of Tübingen and was a visiting Researcher with SAP. He is a Postdoctoral Fellow with the Ruhr-University Bochum, Bochum, Germany. His research interests include machine learning, multi-armed bandits, reinforcement learning, and generative AI.

EVGENII KORTUKOV received the B.Sc. degree in informatics from Moscow State University, Moscow, Russia, in 2020. He is currently working toward the M.Sc. degree in machine learning with the University of Tübingen, Tübingen, Germany. His research interests include sequential decision-making and machine learning.

SETAREH MAGHSUDI received the Ph.D. degree (*summa cum laude*) from the Technical University of Berlin, Berlin, Germany, in 2015. She is currently an Adjunct Lecturer with Technical University of Berlin. She is currently an Assistant Professor with the University of Tübingen, Tübingen, Germany, and a Senior Project Manager with the Fraunhofer Heinrich-Hertz-Institute, Berlin. During 2015–2017, she held Postdoctoral positions with the University of Manitoba, Winnipeg, MB, Canada, and Yale University, New Haven, CT, USA. Her main research interests lie at the intersection of network analysis and optimization, game theory, machine learning, and data science. She was the recipient of several competitive fellowships and career awards, including from the German Ministry of Education and Research, the German Research Foundation, and the Japan Society for the Promotion of Science.