Adversarial Attacks on Linear Contextual Bandits

Evrard Garcelon[⋆]
Facebook AI Research
evrard@fb.com

Baptiste Rozière[⋆]
Facebook AI Research
broz@fb.com

Laurent Meunier[⋆]
Facebook AI Research
laurentmeunier@fb.com

Jean Tarbouriech
Facebook AI Research
jtarbouriech@fb.com

Olivier Teytaud
Facebook AI Research
oteytaud@fb.com

Alessandro Lazaric
Facebook AI Research
lazaric@fb.com

Matteo Pirotta
Facebook AI Research
pirotta@fb.com

Abstract

Contextual bandit algorithms are applied in a wide range of domains, from advertising to recommender systems, from clinical trials to education. In many of these domains, malicious agents may have incentives to force a bandit algorithm into a desired behavior. For instance, an unscrupulous ad publisher may try to increase their own revenue at the expense of the advertisers; a seller may want to increase the exposure of their products, or thwart a competitor's advertising campaign. In this paper, we study several attack scenarios and show that a malicious agent can force a linear contextual bandit algorithm to pull any desired arm $T - o(T)$ times over a horizon of T steps, while applying adversarial modifications to either rewards or contexts with a cumulative cost that only grow logarithmically as $O(\log T)$. We also investigate the case when a malicious agent is interested in affecting the behavior of the bandit algorithm in a single context (e.g., a specific user). We first provide sufficient conditions for the feasibility of the attack and an efficient algorithm to perform an attack. We empirically validate the proposed approaches in synthetic and real-world datasets.

1 Introduction

Recommender systems are at the heart of the business model of many industries like e-commerce or video streaming [1, 2]. The two most common approaches for this task are based either on matrix factorization [3] or bandit algorithms [4], which both rely on a unaltered feedback loop between the recommender system and the user. In recent years, a fair amount of work has been dedicated to understanding how targeted perturbations in the feedback loop can fool a recommender system into recommending low quality items.

Following the line of research on adversarial attacks in supervised learning [5, 6, 7, 8, 9], attacks on recommender systems have been focused on filtering-based algorithms [10, 11] and offline contextual bandits [12]. The question of adversarial attacks for online bandit algorithms has only been studied quite recently [13, 14, 15, 16], and solely in the multi-armed stochastic setting. Although the idea of online adversarial bandit algorithms is not new (see EXP3 algorithm in [17]), the focus is different from what we are considering in this article. Indeed, algorithms like EXP3 or EXP4 [18] are designed to find optimal actions in hindsight in order to adapt to any rewards stream.

[⋆] indicates equal contribution

34th Conference on Neural Information Processing Systems (NeurIPS 2020), Vancouver, Canada.

The opposition between adversarial and stochastic bandit settings has sparked interests in studying a middle ground. In [19], the learning algorithm has no knowledge of the type of feedback it receives (either stochastic or adversarial). In [20, 21, 22, 23, 24], the rewards are assumed to be corrupted by adversarial rewards. The authors focus on building algorithms able to find the optimal actions even in the presence of some non-random perturbations. This setting is different from what is studied in this article because those perturbations are bounded and agnostic to arms pulled by the learning algorithm, i.e., the adversary corrupt the rewards before the algorithm chooses an arm.

In the broader Deep Reinforcement Learning (DRL) literature, the focus is placed on modifying the observations of different states to fool a DRL system at inference time [25, 26] or the rewards [27].

Contribution. In this work, we first follow the research direction opened by [13] where the attacker has the objective of fooling a learning algorithm into taking a specific action as much as possible. For example in a news recommendation problem, as described in [4], a bandit algorithm chooses between K articles to recommend to a user, based on some information about them, called context. We assume that an attacker sits between the user and the website, they can choose the reward (i.e., click or not) for the recommended article observed by the recommending algorithm. Their goal is to fool the bandit algorithm into recommending some articles to most users. The contributions of our work can be summarized as follows:

- We extend the work of [13, 14] to the contextual linear bandit setting showing how to perturb rewards for both stochastic and adversarial algorithms, forcing **any** bandit algorithms to pull a specific set of arms, $o(T)$ times for logarithmic cost for the attacker.
- We analyze, for the first time, the setting in which the attacker can only modify the context x associated with the current user (the reward is not altered). The goal of the attacker is to fool the bandit algorithm into pulling arms of a target set for most users (i.e., contexts) while minimizing the total norm of their attacks. We show that the widely known LinUCB algorithm [28, 18] is vulnerable to this new type of attack.
- We present a harder setting for the attacker, where the latter can only modify the context associated to a specific user. This situation may occur when a malicious agent has infected some computers with a Remote Access Trojan (RAT). The attacker can then modify the history of navigation of a specific user and, as a consequence, the information seen by the online recommender system.We show how the attacker can attack the two very common bandit algorithms LinUCB and Linear Thompson Sampling (LinTS) [29, 30] and, in certain cases, force them to pull a set of arms most of the time when a specific context (i.e., user) is presented to the algorithm (i.e., visits a website).

2 Preliminaries

We consider the standard contextual linear bandit setting with $K \in \mathbb{N}$ arms. At each time t, the agent observes a context $x_t \in \mathbb{R}^d$, selects an action $a_t \in [\![1, K]\!]$ and observes a reward: $r_{t,a_t} = \langle \theta_{a_t}, x_t \rangle + \eta_{a_t}^t$ where for each arm a, $\theta_a \in \mathbb{R}^d$ is a feature vector and $\eta_{a_t}^t$ is a conditionally independent zero-mean, σ^2-subgaussian noise. The contexts are assumed to be sampled *stochastically* except in App. D.

Assumption 1. *There exist $L > 0$ and $\mathcal{D} \subset \mathbb{R}^d$, such that for all t, $x_t \in \mathcal{D}$ and, $\forall x \in \mathcal{D}, \forall a \in [\![1, K]\!]$, $\|x\|_2 \leq L$ and $\langle \theta_a, x \rangle \in (0, 1]$. In addition, we assume that there exists $S > 0$ such that $\|\theta_a\|_2 \leq S$ for all arms a.*

The agent minimizes the cumulative regret after T steps $R_T = \sum_{t=1}^T \langle \theta_{a_t^\star}, x_t \rangle - \langle \theta_{a_t}, x_t \rangle$, where $a_t^\star := \operatorname{argmax}_a \langle \theta_a, x_t \rangle$. A bandit learning algorithm $\mathfrak{A}$ is said to be *no-regret* when it satisfies $R_T = o(T)$, i.e., the average expected reward received by $\mathfrak{A}$ converges to the optimal one. Classical bandit algorithms (e.g., LinUCB and LinTS) compute an estimate of the unknown parameters θ_a using past observations. Formally, for each arm $a \in [K]$ we define S_a^t as the set of times up to $t-1$ (included) where the agent played arm a. Then, the estimated parameters are obtained through regularized least-squares regression as $\widehat{\theta}_a^t = (X_{t,a} X_{t,a}^\top + \lambda I)^{-1} X_{t,a} Y_{t,a}$, where $\lambda > 0$, $X_{t,a} = (x_i)_{i \in S_a^t} \in \mathbb{R}^{d \times |S_a^t|}$ and $Y_{t,a} = (r_{i,a_i})_{i \in S_a^t} \in \mathbb{R}^{|S_a^t|}$. Denote by $V_{t,a} = \lambda I + X_{t,a} X_{t,a}^\top$ the design matrix of the regularized least-square problem and by $\|x\|_V = \sqrt{x^\top V x}$ the weighted norm w.r.t. any positive matrix $V \in \mathbb{R}^{d\times d}$. We define the confidence set:

$$\mathcal{C}_{t,a} = \left\{\theta \in \mathbb{R}^d \,:\, \left\|\theta - \widehat{\theta}_{t,a}\right\|_{V_{t,a}} \le \beta_{t,a}\right\} \tag{1}$$

where $\beta_{t,a} = \sigma\sqrt{d\log\left((1+L^2(1+|S_a^t|)/\lambda)/\delta\right)} + S\sqrt{\lambda}$, which guarantees that $\theta_a \in \mathcal{C}_{t,a}$, for all $t > 0$, w.p. $1-\delta$. This uncertainty is used to balance the exploration-exploitation trade-off either through optimism (e.g., LinUCB) or through randomization (e.g., LinTS).

3 Online Adversarial Attacks on Rewards

The ultimate goal of a malicious agent is to force a bandit algorithm to perform a desired behavior. An attacker may simply want to induce the bandit algorithm to perform poorly—ruining the users' experience—or to force the algorithm to suggest a specific arm. The latter case is particularly interesting in advertising where a seller may want to increase the exposure of its product at the expense of the competitors. Note that the users' experience is also compromised by the latter attack since the suggestions they will receive will not be tailored to their needs. Similarly to [14, 13], we focus on the latter objective, i.e., to fool the bandit algorithm into pulling arms in $A^\dagger$, a set of target arms, for $T - o(T)$ time steps (*independently of the user*).

A way to obtain this behavior is to dynamically modify the reward in order to make the bandit algorithm believe that $a^\dagger$ is optimal, for some $a^\dagger \in A^\dagger$. Clearly, the attacker has to pay a price in order to modify the perceived bandit problem and fool the algorithm. If there is no restriction on when and how the attacker can alter the reward, the attacker can easily fool the algorithm. However, this setting is not interesting since the attacker may pay a cost higher than the loss suffered by the attacked algorithm. An attack strategy is considered successful when the total cost of the attack is sublinear in T.

In this section, we show that under Assumption 1, there exists an attack algorithm that is successful against any bandit algorithm, stochastic or adverserial.

Setting. We assume that the attacker has the same knowledge as the bandit algorithm $\mathfrak{A}$ about the problem (i.e., knows σ and L). The attacker is assumed to be able to observe the context x_t, the arm a_t pulled by $\mathfrak{A}$, and can modify the reward received by $\mathfrak{A}$. When the attacker modifies the reward r_{t,a_t} into $\widetilde{r}_{t,a_t}$ the *instantaneous cost* of the attack is defined as $c_t := \left|r_{t,a_t} - \widetilde{r}_{t,a_t}\right|$. The goal of the attacker is to fool algorithm $\mathfrak{A}$ such that the arms in $A^\dagger$ are pulled $T - o(T)$ times and $\sum_{t=1}^T c_t = o(T)$. We also assume that the action for the arms in the target set is strictly positive for every context $x \in \mathcal{D}$. That is to say that $\Delta := \min_{x\in\mathcal{D}}\left\{\langle x, \theta_{a^\dagger_\star(x)}\rangle - \max_{a\in A^\dagger, a\neq a^\dagger_\star(x)}\langle x, \theta_a\rangle\right\} > 0$ where $a^\dagger_\star(x) = \arg\max_{a\in A^\dagger}\langle x, \theta_a\rangle$ for every $x \in \mathcal{D}$.

Attack idea. We leverage the idea presented in [14] and [13] where the attacker lowers the reward of arms $a \notin A^\dagger$ so that algorithm $\mathfrak{A}$ learns that an arm of the target set is optimal for every context. Since $\mathfrak{A}$ is assumed to be no-regret, the attacker only needs to modify the rewards $o(T)$ times to achieve this goal. Lowering the rewards has the effect of shifting the vectors $(\theta_a)_{a\notin A^\dagger}$ to new vectors $(\theta'_a)_{a\notin A^\dagger}$ such that for all arms $a \notin A^\dagger$ and all contexts $x \in \mathcal{D}$, there exists an arm $a^\dagger \in A^\dagger$ such that $\langle \theta'_a, x\rangle \le \langle \theta_{a^\dagger}, x\rangle$. Since rewards are assumed to be bounded (see Asm. 1), this objective can be achieved by simply forcing the reward of non-target arms $a \notin A^\dagger$ to the minimum value. Contextual ACE (see Fig. 2) implements a soft version of this idea by leveraging the knowledge of the reward distribution. At each round t, Contextual ACE modifies the reward perceived by $\mathfrak{A}$ as follows:

$$\widetilde{r}^1_{t,a_t} = \eta'_t \mathbb{1}_{\{a_t\notin A^\dagger\}} + r_{t,a_t}\mathbb{1}_{\{a_t\in A^\dagger\}} \tag{2}$$

where η'_t is a σ-subgaussian random variable generated by the attacker independently of all other random variables. Contextual ACE transforms the original problem into a *stationary* bandit problem in which there is a targeted arm that is optimal for all contexts and all non targeted arms have expected reward of 0. The following propostion shows that the cumulative cost of the attack is sublinear.

Proposition 1. *For any $\delta \in (0, 1/K]$, when using Contextual ACE algorithm (Fig. 1) with perturbed rewards $\widetilde{r}^1$, with probability at least $1 - K\delta$, algorithm $\mathfrak{A}$ pulls an arm in $A^\dagger$ for $T - o(T)$ time steps and the total cost of attacks is $o(T)$.*

The proof of this proposition is provided in App. A.1. While Prop. 1 holds for any no-regret algorithm $\mathfrak{A}$, we can provide a more precise bound on the total cost by inspecting the algorithm. For example, we can show (see App. E), that, with probability at least $1-K\delta$, the number of times LINUCB [28] pulls arms not in $A^\dagger$ is at most $\sum_{j\notin A^\dagger} N_j(T) \leq \frac{64K\sigma^2\lambda S^2}{\Delta^2}\left(d\log\left(\frac{\lambda+\frac{TL^2}{d}}{\delta^2}\right)\right)^2$. This directly translates into a bound on the total cost.

Comparison with ACE [14]. In the stochastic setting, the ACE algorithm [14] leverages a bound on the expected reward of each arm in order to modify the reward. However, the perturbed reward process seen by algorithm $\mathfrak{A}$ is non-stationary and in general there is no guarantee that an algorithm minimizing the regret in a stationary bandit problem keeps the same performance when the bandit problem is not stationary anymore. Nonetheless, transposing the idea of the ACE algorithm to our setting would give an attack of the following form, where at time t, Alg. $\mathfrak{A}$ pulls arm a_t and receives rewards $\widetilde{r}^2_{t,a_t}$:

$$\widetilde{r}^2_{t,a_t} = (r_{t,a_t} + \max(-1, \min(0, C_{t,a_t})))\mathbb{1}_{\{a_t\notin A^\dagger\}} + r_{t,a_t}\mathbb{1}_{\{a_t\in A^\dagger\}}$$

with $C_{t,a_t} = (1-\gamma)\min_{a^\dagger\in A^\dagger}\min_{\theta\in\mathcal{C}_{t,a^\dagger}}\langle\theta, x_t\rangle - \max_{\theta\in\mathcal{C}_{t,a_t}}\langle\theta, x_t\rangle$. Note that $\mathcal{C}_{t,a}$ is defined as in Eq. 1 using the *non-perturbed* rewards, i.e., $Y_{t,a} = (r_{i,a_i})_{i\in S^t_a}$.

Bounded Rewards. The bounded reward assumption is necessary in our analysis to prove a formal bound on the total cost of the attacks for *any* no-regret bandit algorithm, otherwise we need more information about the attacked algorithm. In practice, the second attack on the rewards, $\widetilde{r}^2$, can be used in the case of unbounded rewards for any algorithms. The difficulty for unbounded reward is that the attacker has to adapt to the environment reward but in order to do so the reward process observed by the bandit algorithm becomes non-stationary under the attack. Thus, there is no guarantee that an algorithm like LINUCB will pull a target arm as the proof relies on the environment observed by the bandit algorithm being stationary. We observe empirically that the total cost of attack is sublinear when using $\widetilde{r}^2$.

[13] does not assume that rewards are bounded but focus on attacking algorithms in the stochastic multi-armed setting. That is to say they study attacks only designed for ε-greedy and UCB while we provide an efficient attack for any algorithms in the linear contextual case. We can extend their work, and thus remove the bounded reward assumption, in the linear contextual case by using the following attack, designed only for LINUCB:

$$\widetilde{r}^3_{t,a_t} = \left(r_{t,a_t} + \min_{a^\dagger\in A^\dagger}\min_{\theta\in\mathcal{C}_{t,a^\dagger}}\langle\theta, x_t\rangle - \max_{\theta\in\mathcal{C}_{t,a_t}}\langle\theta, x_t\rangle\right)\mathbb{1}_{\{a_t\notin A^\dagger\}} + r_{t,a_t}\mathbb{1}_{\{a_t\in A^\dagger\}} \tag{3}$$

with $C_{t,a}$ defined as in Eq. (1). Although, the attack $\widetilde{r}^3$ is not stationary, it is possible to prove that the total cost of attack is $\mathcal{O}(\log(T))$ because we know that the attacked bandit algorithm is LINUCB.

Constrained Attack. When the attacker has a constraint on the instantaneous cost of the attack, using the perturbed reward $\widetilde{r}^1$ may not be possible as the cost of the attack at time t is not decreasing over time. Using the perturbed reward $\widetilde{r}^2$ offers a more flexible type of attack with more control on the instantaneous cost thanks to the parameter γ. But it still suffers from a minimal cost of attack from lowering rewards of arms not in $A^\dagger$.

Defense mechanism. The attack based on reward $\widetilde{r}_1$ is hardly detectable without prior knownledge about the problem. In fact, the reward process associated to $\widetilde{r}_1$ is stationary and compatible with the assumption about the true reward (e.g., subgaussian). While having very low rewards is reasonable in advertising, it can make the attack easily detectable in some other problems. On the other hand, the fact that $\widetilde{r}_2$ is a non-stationary process makes this attack easier to detect. When some data are already available on each arm, the learner can monitor the difference between the average rewards per action computed on new and old data.

4 Online Adversarial Attacks on Contexts

In this section, we consider the attacker to be able to alter the context x_t perceived by the algorithm rather than the reward. The attacker is now restricted to change the type of users presented to the learning algorithm $\mathfrak{A}$, hence changing its perception of the environment. We show that under the assumption that the attacker knows a lower-bound to the reward of the target set, it is possible to fool LINUCB.

```
For time $t = 1, 2, ..., T$ do
  1. Alg. $\mathfrak{A}$ chooses arm $a_t$ based on context $x_t$
  2. Environment generates reward: $r_{t,a_t} = \langle \theta_{a_t}, x_t \rangle + \eta_t$ with $\eta^t_{a_t}$ conditionally $\sigma^2$-subgaussian
  3. Attacker observes reward $r_{t,a_t}$ and feeds the perturbed reward $\widetilde{r}^1_{t,a_t}$ (or $\widetilde{r}^2_{t,a_t}$) to $\mathfrak{A}$
```

Figure 1: Contextual ACE algorithm

```
Input: attack parameter: $\alpha$
For time $t = 1, 2, ..., T$ do
  1. Attacker observes the context $x_t$, computes potential arm $a'_t$ and sets $\widetilde{x}_t = x_t + (\alpha(x_t) - 1)x_t \mathbb{1}_{\{a'_t \notin A^\dagger\}}$
  2. Alg. $\mathfrak{A}$ chooses arm $a_t$ based on context $\widetilde{x}_t$
  3. Environment generates reward: $r_{t,a_t} = \langle \theta_{a_t}, x_t \rangle + \eta_t$ with $\eta_t$ conditionally $\sigma^2$-subgaussian
  4. Alg. $\mathfrak{A}$ observes reward $r_{t,a_t}$
```

Figure 2: ConicAttack algorithm.

Setting. As in Sec. 3, we consider the attacker to have the same knowledge about the problem as $\mathfrak{A}$. The main difference with the previous setting is that the attacker attacks before the algorithm. We adopt a *white-box* [6] setting attacking LinUCB. The goal of the attacker is unchanged: they aim at forcing the algorithm to pull arms in $A^\dagger$ for $T - o(T)$ time steps while paying a sublinear total cost. We denote by $\widetilde{x}_t$ the context after the attack and by $c_t = \|x_t - \widetilde{x}_t\|_2$ the instantaneous cost.

Difference between attacks on contexts and rewards. Perturbing contexts is fundamentally different from perturbing the rewards. The attacker only modifies the context that is shown to the bandit algorithm. The true context, which is used to compute the reward, remains unchanged. In other words, the attacker cannot modify the reward observed by the bandit algorithm. Instead, the attack algorithm described in this section fools the bandit algorithm by making the rewards appear small relative to the contexts and requires more assumptions on the bandit algorithm than in Sec. 3.

Attack Idea. The idea of the attack in this setting is similar to the attack of Sec. 3. The attacker builds a bandit problem where arm an $a^\dagger \in A^\dagger$ is optimal for all contexts by lowering the perceived value of all other arms not in $A^\dagger$. The attacker cannot modify the reward but, thanks to the linear reward assumption, they can scale the contexts to decrease the predicted rewards in the original context.

At time t, the attacker receives the context x_t and computes the attack. Thanks to the white-box setting, it computes the arm a_t that algorithm $\mathfrak{A}$ would pull if presented with context x_t. If $a_t \notin A^\dagger$ then the attacker changes the context to $\widetilde{x}_t = \alpha_{a_t} x_t$ with $\alpha_{a_t} > \max_{x \in \mathcal{D}} \min_{a^\dagger \in A^\dagger} \langle \theta_{a_t}, x \rangle / \langle \theta_{a^\dagger}, x \rangle$.This factor is chosen such that for a ridge regression computed on the dataset $(\alpha x_i, \langle \theta, x_i \rangle)_i$ outputs a parameter close to θ/α therefore the attacker needs to choose α such that for every context $x \in \mathcal{D}$, $\langle x, \theta/\alpha \rangle \leq \max_{a^\dagger \in A^\dagger} \langle x, \theta_{a^\dagger}, x \rangle$. In other words, the attacker performs a dilation of the incoming context every time algorithm $\mathfrak{A}$ does not pull an arm in $A^\dagger$. The fact that the decision rule used by LinUCB is invariant by dilation guarantees that the attacker will not inadvertently lower the perceived rewards for arms in $A^\dagger$. Because the rewards are assumed to be linear, presenting a large context αx and receiving the reward associated with the normal context x will skew the estimated rewards of LinUCB. The attack protocol is summarized in Fig. 2.

In order to compute the parameter α used in the attack, we make the following assumption concerning the performance of the arms in the target set:

Assumption 2. *For all $x \in \mathcal{D}$, there exists $a^\dagger \in A^\dagger$, such that $0 < \nu \leq \langle x, \theta_{a^\dagger} \rangle$ and ν is known to the attacker.*

Knowing ν. For advertising and recommendation systems, knowing ν is not problematic. Indeed in those cases, the reward is the probability of impression of the ad ($r \in [0, 1]$). The attacker has the freedom to choose one of multiple target arms with strictly positive click probability in every context. This freedom is an important aspect for the attacker since it allows the attacker to cherry pick the target ad(s). In particular, the attacker can estimate ν based on data from previous campaigns (only for the target ad). For instance, a company could have run many ad campaigns for one of their products and try to get the defender's system to advertise it.

An issue is that the norm of the attacked context can be greater that the upper bound L of Assumption 1. To prevent this issue, we choose a context-dependent multiplicative constant $\alpha(x) = \min\{2/\nu, L/\|x\|_2\}$ which amounts to clip the norm of the attacked context to L. In Sec. 6, we show that this attack is effective for different size of target arms sets. We also show that in the case of contexts such that $\|x\|_2 \leq \nu L/2$ that the cost of attacks is logarithmic in the horizon T.

Proposition 2. *Using the attack described in Fig. 2 and assuming that $\|x\|_2 \leq \nu L/2$ for all contexts $x \in \mathcal{D}$, for any $\delta \in (0, 1/K]$, with probability at least $1 - K\delta$, the number of times* LinUCB *does not* *pull an arm in $A^\dagger$ before time T is at most $\sum_{j\notin A^\dagger} N_j(T) \leq 32K^2\left(\frac{\lambda}{\alpha^2} + \sigma^2 d\log\left(\frac{\lambda d + TL^2\alpha^2}{d\lambda\delta}\right)\right)^3$ with $N_j(T)$ the number of times arm j has been pulled during the first T steps, The total cost for the attacker is bounded by: $\sum_{t=1}^{T} c_t \leq \frac{64K^2}{\nu}\left(\frac{\lambda}{\alpha^2} + \sigma^2 d\log\left(\frac{\lambda d + TL^2\alpha^2}{d\lambda\delta}\right)\right)^3$ with $\alpha = 2/\nu$.*

The proof of Proposition 2 (see App. A.2) assumes that the attacker can attack at any time step, and that they can know in advance which arm will be pulled by Alg. $\mathfrak{A}$ in a given context. Thus it is not applicable to random exploration algorithms like LinTS [29] and ε-greedy. We also observed empirically that thowe two randomized algorithms are more robust to attacks (see Sec. 6) than LinUCB.

Norm Clipping. Clipping the norm of the attacked contexts is not beneficial for the attacker. Indeed, this means that an attacked context was violating the assumption (used by the bandit algorithm) that contexts are bounded by L. The attack could then be easily detectable and may succeed only because it is breaking an underlying assumption used by the bandit algorithm. Prop. 2 provides a theoretical grounding for the proposed attack when contexts are bounded by $\nu L/2$ and not only L. Although, we can not prove a bound on the cumulative cost of attacks in general, we show in Sec. 6 that attacks are still successful for multiple datasets where contexts are not bounded by $\nu L/2$.

5 Offline attacks on a Single Context

Previous sections focused on the man-in-the-middle (MITM) attack either on reward or context. The MITM attack allows the attacker to arbitrarily change the information observed by the recommender system at each round. This attack may be hardly feasible in practice, since the exchange channels are generally protected by authentication and cryptographic systems. In this section, we consider the scenario where the attacker has control over a single user u. As an example, consider the case where the device of the user is infected by a malware (e.g., Trojan horse), giving full control of the system to the malicious agent. The attacker can thus modify the context of the specific user (e.g., by altering the cookies) that is perceived by the recommender system. We believe that changes to the context (e.g., cookies) are more subtle and less easily detectable than changes to the reward (e.g., click). Moreover, if the reward is a purchase, it cannot be altered easily by taking control of the user's device. Clearly, the impact of the attacker on the overall performance of the recommender system depends on the frequency of the specific user, that is out of the attacker's control. It may be thus difficult to obtain guarantees on the cumulative regret of algorithm $\mathfrak{A}$. For this reason, we mainly focus on the study of the feasibility of the attack.

The attacker targets a specific user (i.e., the infected user) associated to a context $x^\dagger$. Similarly to Sec. 4, the objective of the attacker is to find the minimal change to the context presented to the recommender system $\mathfrak{A}$ such that $\mathfrak{A}$ selects an arm in $A^\dagger$. $\mathfrak{A}$ observes a modified context $\widetilde{x}$ instead of $x^\dagger$. After selecting an arm a_t, $\mathfrak{A}$ observes the true noisy reward $r_{t,a_t} = \langle \theta_{a_t}, x^\dagger\rangle + \eta^t_{a_t}$. We still study a white-box setting: the attacker can access all the parameters of $\mathfrak{A}$.

In this section, we show under which condition it is possible for an attacker to fool both an optimistic and posterior sampling algorithm.

5.1 Optimistic Algorithm: LinUCB

We consider the LinUCB algorithm which chooses the arm to pull by maximizing an upper-confidence bound on the expected reward. For each arm a and context x, the UCB value is given by $\max_{\theta\in\mathcal{C}_{t,a}}\langle x, \theta\rangle = \langle x, \hat{\theta}^t_a\rangle + \beta_{t,a}\|x\|_{\widetilde{V}^{-1}_{t,a}}$ (see Sec. 2). The objective of the attacker is to force LinUCB to pull an arm in $A^\dagger$ once presented with context $x^\dagger$. This means to find a perturbation of context $x^\dagger$ that makes any arm in $A^\dagger$ the most optimistic arm. Clearly, we would like to keep the perturbation as small as possible to reduce the cost for the attacker and the probability of being detected. Formally, the attacker needs to solve the following *non-convex* optimization problem:

$$\min_{y\in\mathbb{R}^d} \quad \|y\|_2 \qquad \text{s.t} \qquad \max_{a\notin A^\dagger}\max_{\theta\in\widetilde{\mathcal{C}}_{t,a}}\langle x^\dagger + y, \theta\rangle + \xi \leq \max_{a^\dagger\in A^\dagger}\max_{\theta\in\widetilde{\mathcal{C}}_{t,a^\dagger}}\langle x^\dagger + y, \theta\rangle \tag{4}$$

where $\xi > 0$ is a parameter of the attacker and $\widetilde{\mathcal{C}}_{t,a} := \left\{\theta \mid \|\theta - \hat{\theta}^t_a\|_{\widetilde{V}_{t,a}} \leq \beta_{t,a}\right\}$ is the confidence set constructed by LinUCB. We use the notation $\widetilde{\mathcal{C}}, \widetilde{V}$ to stress the fact that LinUCB observes only the modified context. In contrast to Sec. 3 and 4, the attacker may not be able to force the algorithm to pull any of the target arms in $A^\dagger$. In other words, Problem 4 may not be feasible. However, we are able to characterize the feasibility of (4).

Theorem 1. *Problem* (4) *is feasible at time t* iff.

$$\exists \theta \in \cup_{a^\dagger \in A^\dagger} \widetilde{\mathcal{C}}_{t,a^\dagger},\ \theta \notin \mathit{Conv}\Big(\cup_{a \notin A^\dagger} \widetilde{\mathcal{C}}_{t,a} \Big) \tag{5}$$

The condition given by Theorem 1 says that this attack can be done when there exists a vector x for which an arm in $A^\dagger$ is assumed to be optimal according to LINUCB. The condition mainly stems from the fact that optimizing a linear product on a convex compact set will reach its maximum on the edge of this set. In our case this set is the convex hull of the confidence ellipsoids of LINUCB. Although it is possible to use an optimization algorithm for this class of non-convex problems—e.g., DC programming [31]—they are still slow compared to convex algorithms. Therefore, we present a simple convex relaxation of the previous problem for a single target arm $a^\dagger \in A^\dagger$ that still enjoys some empirical performance compared to Problem (4). The final attack can then be computed as the minimum of the attacks obtained for each $a^\dagger \in A^\dagger$. The relaxed problem is the following for each $a^\dagger \in A^\dagger$:

$$\min_{y \in \mathbb{R}^d} \quad \|y\|_2 \qquad \text{s.t} \qquad \max_{a \neq a^\dagger, a \notin A^\dagger} \ \max_{\theta \in \mathcal{C}_{t,a}} \langle x^\dagger + y, \theta - \hat{\theta}^t_{a^\dagger} \rangle \leq -\xi \tag{6}$$

Since the RHS of the constraint in Problem (4) can be written as $\max_{\theta \in \mathcal{C}_{t,a^\dagger}} \langle \theta, x^\dagger + y \rangle$ for any y, the relaxation here consists in using $\langle \theta, x^\dagger + y \rangle$ as a lower-bound to this maximum for any $\theta \in \mathcal{C}_{t,a^\dagger}$.

For the relaxed Problem (6), the same type of reasoning as for Problem (4) gives that Problem (6) is feasible if and only if $\hat{\theta}_{a^\dagger}(t) \notin \mathrm{Conv}\left(\bigcup_{a \neq a^\dagger, a \notin A^\dagger} \mathcal{C}_{t,a} \right)$.

If Condition (5) is not met, no arm $a^\dagger \in A^\dagger$ can be pulled by LINUCB. Indeed, the proof of Theorem 1 shows that the upper-confidence of every arm in $A^\dagger$ is always dominated by another arm for any context. In other words, if any arm in $A^\dagger$ is optimal for some contexts then the condition is satisfied a linear number of times for LINUCB (for formal proof of this fact see App. A.4).

5.2 Random Exploration Algorithm: LINTS

The previous subsection focused on LINUCB, however we can obtain similar guarantees for algorithms with random exploration such as LINTS. In this case, it is not possible to guarantee that a specific arm will be pulled for a given context because of the randomness in the arm selection process. The objective is to guarantee that an arm from $A^\dagger$ is pulled with probability at least $1 - \delta$. Similarly to the previous subsection, the problem of the attacker can be written as:

$$\min_{y \in \mathbb{R}^d} \quad \|y\| \qquad \text{s.t} \qquad \mathbb{P}\left(\exists a^\dagger \in A^\dagger,\ \forall a \notin A^\dagger,\ \langle x^\dagger + y, \widetilde{\theta}_a - \widetilde{\theta}_{a^\dagger} \rangle \leq -\xi \right) \geq 1 - \delta \tag{7}$$

where the $\widetilde{\theta}_a$ for different arms a are independently drawn from a normal distribution with mean $\hat{\theta}_a(t)$ and covariance matrix $\upsilon^2 \bar{V}_a^{-1}(t)$ with $\upsilon = \sigma\sqrt{9d\ln(T/\delta)}$. Solving this problem is not easy and in general not possible, even for a single arm. For a given x and arm a, the random variable $\langle x, \widetilde{\theta}_a \rangle$ is normally distributed with mean $\mu_a(x) := \langle \hat{\theta}_a(t), x \rangle$ and variance $\sigma_a^2(x) := \nu^2 ||x||^2_{\bar{V}_a^{-1}(t)}$. We can then write $\langle x, \widetilde{\theta}_a \rangle = \mu_a(x) + \sigma_a(x) Z_a$ with $(Z_a)_a \sim \mathcal{N}(0, I_K)$. For the sake of clarity, we drop the variable x when writing $\mu_a(x)$ and $\sigma_a(x)$.

Let's imagine (just for this paragraph) that $A^\dagger = \{a^\dagger\}$, then the constraint in Problem (7) becomes $\left[1 - \mathbb{E}_{Z_{a^\dagger}} \left(\Pi_{a \notin A^\dagger} \Phi \left(\frac{\sigma_{a^\dagger} Z_{a^\dagger} + \mu_{a^\dagger} - \mu_a}{\sigma_a} \right) \right) \right] \leq \delta$ where Φ is the cumulative distribution function of a normally distributed Gaussian random variable. Unfortunately, computing exactly this expectation is an open problem.

In the more general case where $|A^\dagger| \geq 1$, rewriting the constraints of Problem (7) is not possible. Following the idea of [14], for every single target arm $a^\dagger \in A^\dagger$, a possible relaxation of the constraint in Problem (7) is, to ensure that there exists an arm $a^\dagger \in A^\dagger$ such that for every arm $a \notin A^\dagger$, $1 - \Phi\left((\mu_{a^\dagger} - \mu_a - \xi)/(\sqrt{\sigma_a^2 + \sigma_{a^\dagger}^2}) \right) \leq \frac{\delta}{K - |A^\dagger|}$, where $|A^\dagger|$ is the cardinal of $A^\dagger$. Thus the relaxed version of the attack on LINTS for a single arm $a^\dagger$ is:

$$\min_{y\in\mathbb{R}^d} \|y\| \quad \text{s.t.} \quad \forall a \notin A^\dagger, \langle x^\dagger + y, \hat{\theta}_{a^\dagger} - \hat{\theta}_a\rangle - \xi \geq \nu\Phi^{-1}\left(1 - \tfrac{\delta}{K-|A^\dagger|}\right)\left\|x^\dagger + y\right\|_{\bar{V}_a^{-1}+\bar{V}_{a^\dagger}^{-1}} \quad (8)$$

Problem (8) is similar to Problem (6) as the constraint is also a Second Order Cone Program but with different parameters (see App. C). As in section 5.1, we compute the final attack as the minimum of the attacks computed for each arm in $A^\dagger$.

6 Experiments

In this section, we conduct experiments on the attacks on contextual bandit problems with simulated data and two real-word datasets: MovieLens25M [32] and Jester [33]. The synthetic dataset and the data preprocessing step are presented in App. B.1.

6.1 Attacks on Rewards

We study the impact of the reward attack for 4 contextual algorithms: LINUCB, LINTS, ε-GREEDY and EXP4. As parameters, we use $L = 1$ for the maximal norm of the contexts, $\delta = 0.01$, $\upsilon = \sigma\sqrt{d\ln(t/\delta))/2}$, $\varepsilon_t = 1/\sqrt{t}$ at each time step t and $\lambda = 0.1$. We choose only a *unique target arm* $a^\dagger$. For EXP4, we use $N = 10$ experts with $N - 2$ experts returning a random arm at each time, one expert choosing arm $a^\dagger$ every time and one expert returning the optimal arm for every context. With this set of experts the regret of bandits with expert advice is the same as in the contextual case. To test the performance of each algorithm, we generate 40 random contextual bandit problems and run each algorithm for $T = 10^6$ steps on each. We report the average cost and regret for each of the 40 problems. Figure 3 (Top) shows the attacked algorithms using the attacked reward $\widetilde{r}^1$ (reported as "stationary CACE") and the rewards $\widetilde{r}^2$ (reported as CACE).

These experiments show that, even though the reward process is non-stationary, usual stochastic algorithms like LINUCB can still adapt to it and pull the optimal arm for this reward process (which is arm $a^\dagger$). The true regret of the attacked algorithms is linear as $a^\dagger$ is not optimal for all contexts. In the synthetic case, for the algorithms attacked with the rewards $\widetilde{r}^2$, over 1M iterations and $\gamma = 0.22$, the target arm is drawn more than 99.4% of the time on average for every algorithm and more than 97.8% of the time for the stationary attack $\widetilde{r}^1$ (see Table 2 in App. B.2). The dataset-based environments (see Figure 3 (Left)) exhibit the same behavior: the target arm is pulled more than 94.0% of the time on average for all our attacks on Jester and MovieLens and more than 77.0% of the time in the worst case (for LINTS attacked with the stationary rewards) (see Table 2).

6.2 Attacks on Contexts

We now illustrate the effectiveness of the attack in Alg. 2. We study the behavior of attacked LINUCB, LINTS, ε-GREEDY with different size of target arms set ($|A^\dagger|/K \in \{0.3, 0.6, 0.9\}$ with K the total number of arms). We test the performance of LINUCB with the same parameters as in the previous experiments. Yet since the variance is much smaller in this case, we generate a random problem and run 20 simulations for each algorithm. The target arms are chosen randomly and we use the exact lower-bound on the reward of those arms to compute ν.

Table 1: Percentage of iterations for which the algorithm pulled an arm in the target set $A^\dagger$ (with a target set size of $0.3K$ arms) (**Left**) Online attacks using ContextualConic (CC) algorithm. Percentages are averaged over 20 runs of 1M iterations. (**Right**) Offline attacks with exact (Full) and Relaxed optimization problem. Percentages are averaged over 40 runs of 1M iterations.

	Synthetic	Jester	Movilens
LINUCB	28.91%	26.59%	31.13%
CC LinUCB	98.55%	98.36%	99.61%
ε-GREEDY	25.7%	25.85%	31.78%
CC ε-GREEDY	89.71%	99.85%	99.92%
LINTS	27.2%	26.10%	33.24%
CC LINTS	30.93%	97.26%	98.82%

	Synthetic	Jester	MovieLens
LINUCB	0.07%	0.01%	0.39%
LINUCB Relaxed	13.76%	97.81%	4.09%
LINUCB Full	88.30%	99.98%	99.99%
ε-GREEDY	0.01%	0.00%	0.03%
ε-GREEDY Full	99.98%	99.95%	99.97%
LINTS	0.02%	0.01%	0.05%
LINTS Relaxed	18.21%	80.48%	5.56%

Table 1 (Left) shows the percentage of times an arm in $A^\dagger$, for $|A^\dagger| = 0.3K$, has been selected by the attacked algorithm. We see that, as expected, CC LINUCB reaches a ratio of almost 1, meaning the

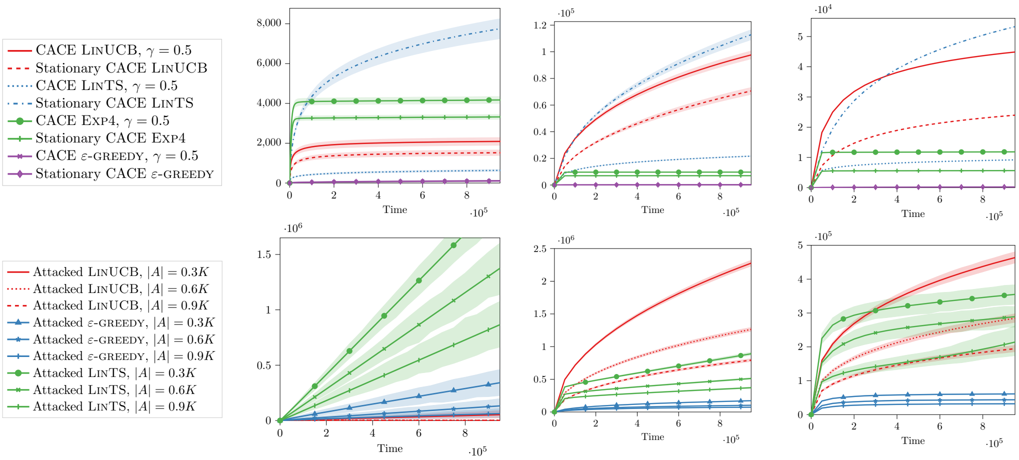


Figure 3: Total cost of attacks on rewards for the synthetic (Left, $\gamma = 0.22$), Jester (Center, $\gamma = 0.5$) and MovieLens (Right, $\gamma = 0.5$) environments. Bottom, total cost of ContextualConic attacks on the synthetic (Left), Jester (Center) and MovieLens (Right) environments.

target arms are indeed pulled a linear number of times. A more surprising result (at least not covered by the theory) is that ε-GREEDY exhibits the same behavior. Similarly to LINTS, ε-GREEDY exhibits some randomness in the action selection process. It can cause an arm $a^\dagger \in A^\dagger$ to be chosen when the context is attacked and interfere with the principle of the attack. We suspect that is what happens for LINTS. Fig. 3 (Bottom) shows the total cost of the attacks for the attacked algorithms . Despite the fact that the estimate of $\theta_{a^\dagger}$ can be polluted by attacked samples, it seems that LINTS can still pick up $a^\dagger$ as being optimal for some dataset like MovieLens and Jester but not on the simulated dataset.

6.3 Offline attacks on a Single Context

We now move to the setting described in Sec. 5 and test the same algorithms as in Sec. 6.2. We run 40 simulations for each algorithm and each attack type. The target context $x^\dagger$ is chosen randomly and the target arm as the arm minimizing the expected reward for $x^\dagger$. The attacker is only able to modify the incoming context for the target context (which corresponds to the context of one user) and the incoming contexts are sampled uniformly from the set of all possible contexts (of size 100). Table 1 (Right) shows the percentage of success for each attack. We observe that the non-relaxed attacks on ε-GREEDY and LINUCB work well across all datasets. However, the relaxed attack for LINUCB and LINTS are not as successful, on the synthetic dataset and MovieLens25M. The Jester dataset seems to be particularly suited to this type of attacks because the true feature vectors are well separated from the convex hull formed by the feature vectors of the other arms: only 5% of Jester's feature vectors are within the convex hull of the others versus 8% for MovieLens and 20% for the synthetic dataset. As expected, the cost of the attacks is linear on all the datasets (see Figure 6 in App. B.4). The cost is also lower for the non-relaxed than for the relaxed version of the attack on LINUCB. Unsurprisingly, the cost of the attacks on LINTS is the highest due to the need to guarantee that $a^\dagger$ will be chosen with high probability (95% in our experiments).

7 Conclusion

We presented several settings for online attacks on contextual bandits. We showed that an attacker can force any contextual bandit algorithm to almost always pull an arbitrary target arm $a^\dagger$ with only sublinear modifications of the rewards. When the attacker can only modify the contexts, we prove that LINUCB can still be attacked and made to almost always pull an arm in $A^\dagger$ by adding sublinear perturbations to the contexts. When the attacker can only attack a single context, we derive a feasibility condition for the attacks and we introduce a method to compute some attacks of small instantaneous cost for LINUCB, ε-GREEDY and LINTS. To the best of our knowledge, this paper is the first to describe effective attacks on the contexts of contextual bandit algorithms. Our numerical experiments, conducted on both synthetic and real-world data, validate our results and show that the attacks on all contexts are actually effective on several algorithms and with more permissible settings.

Broader Impact

Adversarial attacks have been a major concerns in the machine learning community for some time [5, 6, 7, 8, 9] as they delve deeply into the robustness of such machine learning systems. Although, adversarial attacks have only been recently studied for bandits and reinforcement learning algorithms [12, 25]. Those settings are applied to a wide range of applications such as recommender systems or cooling down data centers [34].

In adversarial attacks on supervised algorithms and cryptography, it is well-accepted that the study and publication of attack schemes helps build trustful secure systems [35]. While there is a risk that our methods could be used by malicious attackers, we believe that they will also prompt some practitioners to ensure such modifications of the rewards or contexts of their data can be detected or even prevented.

References

[1] James Davidson, Benjamin Liebald, Junning Liu, Palash Nandy, Taylor Van Vleet, Ullas Gargi, Sujoy Gupta, Yu He, Mike Lambert, Blake Livingston, et al. The youtube video recommendation system. In *Proceedings of the fourth ACM conference on Recommender systems*, pages 293–296, 2010.

[2] Carlos A Gomez-Uribe and Neil Hunt. The netflix recommender system: Algorithms, business value, and innovation. *ACM Transactions on Management Information Systems (TMIS)*, 6(4):1–19, 2015.

[3] Haekyu Park, Jinhong Jung, and U Kang. A comparative study of matrix factorization and random walk with restart in recommender systems. In *2017 IEEE International Conference on Big Data (Big Data)*, pages 756–765. IEEE, 2017.

[4] 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.

[5] Battista Biggio, Blaine Nelson, and Pavel Laskov. Poisoning attacks against support vector machines. *arXiv preprint arXiv:1206.6389*, 2012.

[6] Ian J Goodfellow, Jonathon Shlens, and Christian Szegedy. Explaining and harnessing adversarial examples. *arXiv preprint arXiv:1412.6572*, 2014.

[7] Matthew Jagielski, Alina Oprea, Battista Biggio, Chang Liu, Cristina Nita-Rotaru, and Bo Li. Manipulating machine learning: Poisoning attacks and countermeasures for regression learning. In *2018 IEEE Symposium on Security and Privacy (SP)*, pages 19–35. IEEE, 2018.

[8] Bo Li, Yining Wang, Aarti Singh, and Yevgeniy Vorobeychik. Data poisoning attacks on factorization-based collaborative filtering. In *Advances in neural information processing systems*, pages 1885–1893, 2016.

[9] Chang Liu, Bo Li, Yevgeniy Vorobeychik, and Alina Oprea. Robust linear regression against training data poisoning. In *Proceedings of the 10th ACM Workshop on Artificial Intelligence and Security*, pages 91–102, 2017.

[10] Konstantina Christakopoulou and Arindam Banerjee. Adversarial attacks on an oblivious recommender. In *Proceedings of the 13th ACM Conference on Recommender Systems*, RecSys '19, page 322–330, New York, NY, USA, 2019. Association for Computing Machinery.

[11] Bhaskar Mehta and Wolfgang Nejdl. Attack resistant collaborative filtering. In *Proceedings of the 31st annual international ACM SIGIR conference on Research and development in information retrieval*, pages 75–82, 2008.

[12] Yuzhe Ma, Kwang-Sung Jun, Lihong Li, and Xiaojin Zhu. Data poisoning attacks in contextual bandits. In *International Conference on Decision and Game Theory for Security*, pages 186–204. Springer, 2018.

[13] Kwang-Sung Jun, Lihong Li, Yuzhe Ma, and Jerry Zhu. Adversarial attacks on stochastic bandits. In *Advances in Neural Information Processing Systems*, pages 3640–3649, 2018.

[14] Fang Liu and Ness Shroff. Data poisoning attacks on stochastic bandits. *arXiv preprint arXiv:1905.06494*, 2019.

[15] Nicole Immorlica, Karthik Abinav Sankararaman, Robert E. Schapire, and Aleksandrs Slivkins. Adversarial bandits with knapsacks. *2019 IEEE 60th Annual Symposium on Foundations of Computer Science (FOCS)*, pages 202–219, 2018.

[16] Ziwei Guan, Kaiyi Ji, Donald J Bucci Jr, Timothy Y Hu, Joseph Palombo, Michael Liston, and Yingbin Liang. Robust stochastic bandit algorithms under probabilistic unbounded adversarial attack. *arXiv preprint arXiv:2002.07214*, 2020.

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

[18] Tor Lattimore and Csaba Szepesvári. Bandit algorithms. Pre-publication version, 2018.

[19] Sébastien Bubeck and Aleksandrs Slivkins. The best of both worlds: Stochastic and adversarial bandits. In *Conference on Learning Theory*, pages 42–1, 2012.

[20] Thodoris Lykouris, Vahab Mirrokni, and Renato Paes Leme. Stochastic bandits robust to adversarial corruptions. In *Proceedings of the 50th Annual ACM SIGACT Symposium on Theory of Computing*, pages 114–122, 2018.

[21] Yingkai Li, Edmund Y Lou, and Liren Shan. Stochastic linear optimization with adversarial corruption. *arXiv preprint arXiv:1909.02109*, 2019.

[22] Anupam Gupta, Tomer Koren, and Kunal Talwar. Better algorithms for stochastic bandits with adversarial corruptions. *arXiv preprint arXiv:1902.08647*, 2019.

[23] Thodoris Lykouris, Max Simchowitz, Aleksandrs Slivkins, and Weidong Sun. Corruption robust exploration in episodic reinforcement learning. *ArXiv*, abs/1911.08689, 2019.

[24] Sayash Kapoor, Kumar Kshitij Patel, and Purushottam Kar. Corruption-tolerant bandit learning. *Machine Learning*, 108(4):687–715, 2019.

[25] Léonard Hussenot, Matthieu Geist, and Olivier Pietquin. Targeted attacks on deep reinforcement learning agents through adversarial observations. *arXiv preprint arXiv:1905.12282*, 2019.

[26] Jianwen Sun, Tianwei Zhang, Xiaofei Xie, Lei Ma, Yan Zheng, Kangjie Chen, and Yang Liu. Stealthy and efficient adversarial attacks against deep reinforcement learning. *To appear in Proceedings of the AAAI Conference on Artificial Intelligence*, 2020.

[27] Yuzhe Ma, Xuezhou Zhang, Wen Sun, and Jerry Zhu. Policy poisoning in batch reinforcement learning and control. In *Advances in Neural Information Processing Systems*, pages 14543–14553, 2019.

[28] 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.

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

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

[31] Hoang Tuy. Dc optimization: theory, methods and algorithms. In *Handbook of global optimization*, pages 149–216. Springer, 1995.

[32] F Maxwell Harper and Joseph A Konstan. The movielens datasets: History and context. *Acm transactions on interactive intelligent systems (tiis) DOI=http://dx.doi.org/10.1145/2827872*, 5(4):1–19, 2015.

[33] Ken Goldberg, Theresa Roeder, Dhruv Gupta, and Chris Perkins. Eigentaste: A constant time collaborative filtering algorithm. *information retrieval*, 4(2):133–151, 2001.

[34] Nevena Lazic, Craig Boutilier, Tyler Lu, Eehern Wong, Binz Roy, MK Ryu, and Greg Imwalle. Data center cooling using model-predictive control. In *Advances in Neural Information Processing Systems*, pages 3814–3823, 2018.

[35] Nicholas Carlini Athalye Anish and David Wagner. Obfuscated gradients give a false sense of security: Circumventing defenses to adversarial examples. In *International Conference on Machine Learning*, 2018.

[36] Akshay Agrawal, Robin Verschueren, Steven Diamond, and Stephen Boyd. A rewriting system for convex optimization problems. *Journal of Control and Decision*, 5(1):42–60, 2018.

[37] Pauli Virtanen, Ralf Gommers, Travis E. Oliphant, Matt Haberland, Tyler Reddy, David Cournapeau, Evgeni Burovski, Pearu Peterson, Warren Weckesser, Jonathan Bright, Stéfan J. van der Walt, Matthew Brett, Joshua Wilson, K. Jarrod Millman, Nikolay Mayorov, Andrew R. J. Nelson, Eric Jones, Robert Kern, Eric Larson, CJ Carey, İlhan Polat, Yu Feng, Eric W. Moore, Jake Vand erPlas, Denis Laxalde, Josef Perktold, Robert Cimrman, Ian Henriksen, E. A. Quintero, Charles R Harris, Anne M. Archibald, Antônio H. Ribeiro, Fabian Pedregosa, Paul van Mulbregt, and SciPy 1. 0 Contributors. SciPy 1.0–Fundamental Algorithms for Scientific Computing in Python. *arXiv e-prints*, page arXiv:1907.10121, Jul 2019.

[38] J.M. Varah. A lower bound for the smallest singular value of a matrix. *Linear Algebra and its Applications*, 11(1):3 – 5, 1975.

A Proofs

In this appendix, we present the proofs of different theoretical results presented in the paper.

A.1 Proof of Proposition 1

Proposition. *For any $\delta \in (0, 1/K]$, when using Contextual ACE algorithm (Alg. 1) with perturbed rewards $\tilde{r}^1$, with probability at least $1 - K\delta$, algorithm $\mathfrak{A}$ pulls an arm in $A^\dagger$ for $T - o(T)$ time steps and the total cost of attacks is $o(T)$.*

Proof. Let us consider the contextual bandit problem $\mathcal{A}_1$, with K arms with contexts $x \in \mathcal{D}$ such that every arm in $a^\dagger \in A^\dagger$ has mean reward $\langle \theta_{a^\dagger}, x\rangle$ and all other arms has mean 0. Then the regret of algorithm $\mathfrak{A}$ for this bandit problem is upper-bounded with probability at least $1 - \delta$ by a function $f_{\mathfrak{A}}(T)$ such that $f_{\mathfrak{A}}(T) = o(T)$. In addition, the reward process fed to Alg. $\mathfrak{A}$ by the attacker is a stationary reward process with σ^2-subgaussian noise. Therefore, the number of times algorithm $\mathfrak{A}$ pulls an arm not in $A^\dagger$ is upper-bounded by $f_{\mathfrak{A}}(T)/\min_{x\in\mathcal{D}} \Delta(x)$ where for every context $x \in \mathcal{D}$, let $a^\dagger_\star(x) := \arg\max_{a\in A^\dagger} \langle x, \theta_a\rangle$ and $\Delta(x) = \langle x, \theta_{a^\dagger_\star(x)}\rangle - \max_{a\in A^\dagger, a\neq a^\dagger_\star(x)} \langle x, \theta_a\rangle$.

In addition, the total cost of the attack is upper-bounded by $\max_{a\in[\![1,K]\!]} \max_{x\in\mathcal{D}} |\langle x, \theta_a\rangle|(T - N_{A^\dagger}(T))$ where $N_{A^\dagger}(T)$ is the number of times an arm in $A^\dagger$ has been pulled up to time T. Thanks to the previous argument, $T - N_{A^\dagger}(T) \leq f_{\mathfrak{A}}(T)/\min_{x\in\mathcal{D}} \Delta(x)$. □

A.2 Proof of Proposition 2

Proposition. *Using the attack described in Alg. 2, for any $\delta \in (0, 1/K]$, with probability at least $1 - K\delta$, the number of times* LinUCB *does not pull an arm in $A^\dagger$ is at most:*

$$\sum_{j\notin A^\dagger} N_j(T) \leq 32K^2 \left(\frac{\lambda}{\alpha^2} + \sigma^2 d \log\left(\frac{\lambda d + TL^2\alpha^2}{d\lambda\delta}\right)\right)^3$$

with $N_j(T)$ the number of times arm j has been pulled after T steps, $||\theta_a|| \leq S$ for all arms a, λ the regularization parameter of LinUCB *and for all $x \in \mathcal{D}$, $||x||_2 \leq L$. The total cost for the attacker is bounded by:*

$$\sum_{t=1}^{T} c_t \leq \frac{64K^2}{\nu} \left(\frac{\lambda}{\alpha^2} + \sigma^2 d \log\left(\frac{\lambda d + TL^2\alpha^2}{d\lambda\delta}\right)\right)^3$$

Proof. Let a_t be the arm pulled by LinUCB at time t. For each arms a, let $\tilde{\theta}_a(t)$ be the result of the linear regression with the attacked context and $\hat{\theta}_a(t, \lambda/\alpha^2)$ the one with the unattacked context and a regularization of $\frac{\lambda}{\alpha^2}$. At any time step t, we can write, for all $a \notin A^\dagger$:

$$\begin{aligned}
\tilde{\theta}_a(t) &= \left(\lambda I_d + \sum_{l=0, a_l=a}^{t} \alpha^2 x_l x_l^\intercal\right)^{-1} \sum_{k=0, a_k=a}^{t} r_k \alpha x_k \\
&= \frac{1}{\alpha}\left(\frac{\lambda}{\alpha^2} I_d + \sum_{k=0, a_k=a}^{t} x_k x_k^\intercal\right)^{-1} \sum_{k=0, a_k=a}^{t} r_k x_k \\
&= \frac{\hat{\theta}_a(t, \lambda/\alpha^2)}{\alpha}
\end{aligned}$$

We also note that, since the contexts are not modified for arms in $a^\dagger \in A^\dagger$: $\tilde{\theta}_{a^\dagger}(t) = \hat{\theta}_{a^\dagger}(t, \lambda)$. In addition, for any context x and arm $a \notin A^\dagger$, the exploration term used by LinUCB becomes:

$$||x||_{\tilde{V}^{-1}_{a,t}} = \frac{1}{\alpha} ||x||_{\hat{V}^{-1}_{a,t}} \tag{9}$$

where $\tilde{V}_{a,t} = \lambda I_d + \sum_{l=0,a_l=a}^{t} \alpha^2 x_l x_l^\intercal$ and $\hat{V}_{a,t}^{-1} = \lambda/\alpha^2 I_d + \sum_{k=0,a_k=a}^{t} x_k x_k^\intercal$. For a time t, if presented with context x_t LINUCB pulls arm $a_t \notin A^\dagger$, we have:

$$\alpha\left(\left\langle \hat{\theta}_{a^\dagger}(t), x_t \right\rangle + \beta_{a^\dagger}(t)||x_t||_{V_{a^\dagger,t}^{-1}}\right) \leq \left\langle \hat{\theta}_{a_t}(t, \lambda/\alpha^2), x_t \right\rangle + \beta_{a_t}(t)||x_t||_{\hat{V}_{a_t,t}^{-1}}$$

As $\alpha = \frac{2}{\nu} \geq \min_{a^\dagger \in A^\dagger} \frac{2}{\langle \theta_{a^\dagger}, x_t \rangle}$, we deduce that on the event that the confidence sets (Theorem 2 in [28]) hold for arm $a^\star$:

$$2 \leq \left\langle \hat{\theta}_{a_t}(t, \lambda/\alpha^2), x_t \right\rangle + \beta_{a_t}(t)||x_t||_{\hat{V}_{a_t,t}^{-1}} \leq \langle \theta_{a_t}, x_t \rangle + 2\beta_{a_t}(t)||x_t||_{\hat{V}_{a_t,t}^{-1}}$$

Thus, $1 \leq 2 - \langle \theta_{a_t}, x_t \rangle \leq 2\beta_{a_t}(t)||x_t||_{\hat{V}_{a_t,t}^{-1}}$. Therefore,

$$\begin{aligned}\sum_{t=1}^{T} \mathbb{1}_{\{a_t \notin A^\dagger\}} &\leq \sum_{t=1}^{T} \min(2\beta_{a_t}(t)||x_t||_{\hat{V}_{a_t,t}^{-1}}, 1)\mathbb{1}_{\{a_t \notin A^\dagger\}} \\ &\leq \sum_{j \notin A^\dagger} 2\beta_j(T)\sqrt{\sum_{t=1}^{T} \mathbb{1}_{\{a_t = j\}} \sum_{t=1,a_t=j}^{T} \min(1, ||x_t||_{\hat{V}_{j,t}^{-1}}^2)}\end{aligned}$$

But using Lemma 11 from [28] and the bound on the $\beta_j(T)$ for all arms j, we have with Jensen inequality:

$$\begin{aligned}\sum_{t=1}^{T} \mathbb{1}_{\{a_t \notin A^\dagger\}} \leq& 4\sqrt{K \sum_{t=1}^{T} \mathbb{1}_{\{a_t \notin A^\dagger\}} d \log\left(1 + \frac{\alpha^2 T L^2}{\lambda d}\right)} \\ &\times \left(\sqrt{\lambda/\alpha^2} S + \sigma\sqrt{2\log(1/\delta) + d\log(1 + \frac{\alpha^2 T L^2}{\lambda d})}\right)\end{aligned}$$

□

A.3 Proof of Theorem 1

Theorem. *For any $\xi > 0$, Problem (4) is feasible if and only if:*

$$\exists \theta \in \bigcup_{a^\dagger \in A^\dagger} \mathcal{C}_{t,a^\dagger}, \qquad \theta \notin \mathit{Conv}\left(\bigcup_{a \notin A^\dagger} \mathcal{C}_{t,a}\right) \tag{10}$$

where for every arm a, $\mathcal{C}_{t,a} := \left\{\theta \mid ||\theta - \hat{\theta}_a(t)||_{\tilde{V}_{a,t}} \leq \beta_a(t)\right\}$ with $\hat{\theta}_a(t)$ the least squares estimate for arm a built by LINUCB *and*

$$\tilde{V}_{a,t} = \lambda I_d + \sum_{l=1,x_l \neq x^\dagger}^{t} \mathbb{1}_{\{a_l = a\}} x_l x_l^\intercal + \sum_{l=1,x_l = x^\dagger}^{t} \mathbb{1}_{\{a_l = a\}} \tilde{x}_l \tilde{x}_l^\intercal$$

the design matrix of LINUCB *at time t for all arms a (where $\tilde{x}_l$ is the modified context)*

Proof. The proof of Theorem 1 is decomposed in two parts.

First, let us assume that Equation (10) is satisfied. Then, let us define $a^\dagger \in A^\dagger$ such that $\theta \in \mathcal{C}_{t,a^\dagger} \setminus \text{Conv}\left(\bigcup_{a \notin A^\dagger} \mathcal{C}_{t,a}\right)$, then by the theorem of separation of convex sets applied to $\mathcal{C}_{t,a^\dagger}$ and $\{\theta\}$. There exists a vector v and $c_1 < c_2$ such that for all $y \in \text{Conv}\left(\bigcup_{a \neq a^\dagger} \mathcal{C}_{t,a}\right)$:

$$\langle y, v \rangle \leq c_1 < c_2 \leq \langle \theta, v \rangle\,.$$

Hence, for $\xi > 0$ we have that for $\tilde{v} = \frac{\xi}{c_2 - c_1} v$ that:

$$\langle y, \tilde{v} \rangle + \xi \leq \langle \theta, \tilde{v} \rangle$$

So the problem is feasible.

Secondly, let us assume that an attack is feasible. Then there exists a vector y such that:

$$\max_{a^\dagger\in A^\dagger}\max_{\theta\in\mathcal{C}_{t,a^\dagger}}\langle y,\theta\rangle > c_1 := \max_{a\notin A^\dagger}\max_{\theta\in\mathcal{C}_{t,a}}\langle y,\theta\rangle \tag{11}$$

Let us reason by contradiction. We assume that $\bigcup_{a\in A^\dagger}\mathcal{C}_{t,a^\dagger}\subset \text{Conv}\left(\bigcup_{a\notin A^\dagger}\mathcal{C}_{t,a}\right)$ and consider

$$\theta^*\in\bigcup_{a\in A^\dagger}\mathcal{C}_{t,a^\dagger} \text{ such that } \langle y,\theta^*\rangle = \max_{a^\dagger\in A^\dagger}\max_{\theta\in\mathcal{C}_{t,a^\dagger}}\langle y,\theta\rangle$$

As we assumed $\bigcup_{a\in A^\dagger}\mathcal{C}_{t,a^\dagger}\subset \text{Conv}\left(\bigcup_{a\notin A^\dagger}\mathcal{C}_{t,a}\right)$, there exists $n\in\mathbb{N}^\star$, $\lambda_1,\cdots,\lambda_n\geq 0$ and $\theta_1,\cdots,\theta_n\in\bigcup_{a\notin A^\dagger}\mathcal{C}_{t,a}$ such that

$$\theta^* = \sum_{i=1}^n \lambda_i\theta_i \text{ and } \sum_{i=1}^n\lambda_i = 1$$

Thus

$$\langle y,\theta^*\rangle = \sum_i \lambda_i\langle y,\theta_i\rangle \leq c_1\sum_{i=1}^n\lambda_i = c_1 \tag{12}$$

We assumed that the problem is feasible, so $c_1 < \langle y,\theta^*\rangle$ according to Eq. 11. It contradicts Eq. 12.
□

A.4 Condition of Sec. 5

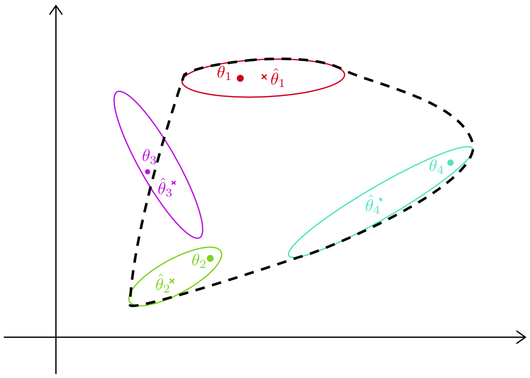


Figure 4: Illustrative example of condition (5). The target arm is arm 3 or 5 and the dashed black line is the convex hull of the other confidence sets. The ellipsoids are the confidence sets $\mathcal{C}_{t,a}$ for each arm a. If we consider only arms $\{1,2,4,5\}$, and we use 5 as the target arm, the condition (5) is satisfied as there is a θ outside the convex hull of the other confidence sets. On the other hand, if we consider arms $\{1,2,3,4\}$ and we use 3 as the target arm, the condition is not satisfied anymore.

Let us assume that there is an arm in $a^\dagger\in A^\dagger$ which is optimal for some contexts. More formally, there exists a subspace $V\subset\mathcal{D}$ such that:

$$\forall x\in V, \exists a^\dagger_\star(x)\in A^\dagger, \forall a\in [\![1,K]\!]\setminus\{a^\dagger_\star(x)\} \qquad \langle x,\theta_{a^\dagger_\star(x)}\rangle > \langle x,\theta_a\rangle\,.$$

We also assume that the distribution of the contexts is such that, for all t, $\mu := \mathbb{P}(x_t\in V) > 0$. Then, the regret is lower-bounded in expectation by:

$$\mathbb{E}(R_T) = \mathbb{E}\left(\sum_{t=1}^T \mathbb{1}_{\{x_t\in V\}}\left(\left\langle x_t,\theta_{a^\dagger_\star(x_t)}-\theta_{a_t}\right\rangle\right)\right) \geq \mu m(T)\min_{x\in V}\max_{a\neq a^\dagger_\star(x)}\langle\theta_{a^\dagger_\star(x)}-\theta_a,x\rangle$$

where $m(T)$ is the expected number of times $t\leq T$ such that condition (5) is not met. LinUCB guarantees that $\mathbb{E}(R_T)\leq\mathcal{O}(\sqrt{T})$ for every T. Hence, $m(T)\leq\mathcal{O}\left(\frac{\sqrt{T}}{\mu\min_{x\in V}\max_{a\neq a^\dagger_\star(x)}\langle\theta_{a^\dagger_\star(x)}-\theta_a,x\rangle}\right)$. This means that, in an unattacked problem, condition (5) is met $T-\mathcal{O}(\sqrt{T})$ times. On the other hand, when the algorithm is attacked the regret of LinUCB is not sub-linear as the confidence bound for the target arm is not valid anymore. Hence we cannot provide the same type of guarantees for the attacked problem.

B Experiments

B.1 Datasets and preprocessing

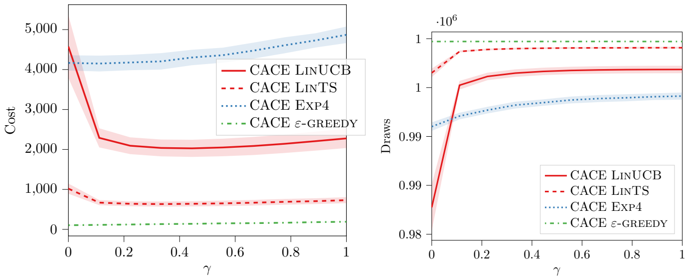
We present here the datasets used in the article and how we preprocess them for numerical experiments conducted in Section 6.

We consider two types of experiments, one on synthetic data with a contextual MAB problems with $K = 10$ arms such that for every arm a, θ_a is drawn from a folded normal distribution in dimension $d = 30$. We also use a finite number of contexts (10), each of them is drawn from a folded normal distribution projected on the unit circle multiplied by a uniform radius variable (i.i.d. across all contexts). Finally, we scale the expected rewards in $(0, 1]$ and the noise is drawn from a centered Gaussian distribution $\mathcal{N}(0, 0.01)$.

The second type of experiments is conducted in the real-world datasets Jester [33] and MovieLens25M [32]. Jester consists of joke ratings on a continuous scale from -10 to 10 for 100 jokes from a total of 73421 users. We use the features extracted via a low-rank matrix factorization ($d = 35$) to represent the actions (i.e., the jokes). We consider a complete subset of 40 jokes and 19181 users . Each user rates all the 40 jokes. At each time, a user is randomly selected from the 19181 users and mean rewards are normalized in $[0, 1]$. The reward noise is $\mathcal{N}(0, 0.01)$. The second dataset we use is MovieLens25M. It contains 25000095 ratings created by 162541 users on 62423 movies. We perform a low-rank matrix factorization to compute users features and movies features. We keep only movies with at least 1000 ratings, which leave us with 162539 users and 3794 movies. At each time step, we present a random user, and the reward is the scalar product between the user feature and the recommend movie feature. All rewards are scaled to lie in $[0, 1]$ and a Gaussian noise $\mathcal{N}(0, 0.01)$ is added to the rewards.

B.2 Attacks on Rewards

In this appendix, we present empirical evolution of the total cost and the number of draws for a unique target arm as a function of the attack parameter γ for the Contextual ACE attack with perturbed rewards $\tilde{r}^2$ on generated data.



(a) Total cost (b) Number of draws

Figure 5: Total cost of attacks and number of draws of the target arm at $T = 10^6$ as a function of γ on synthetic data

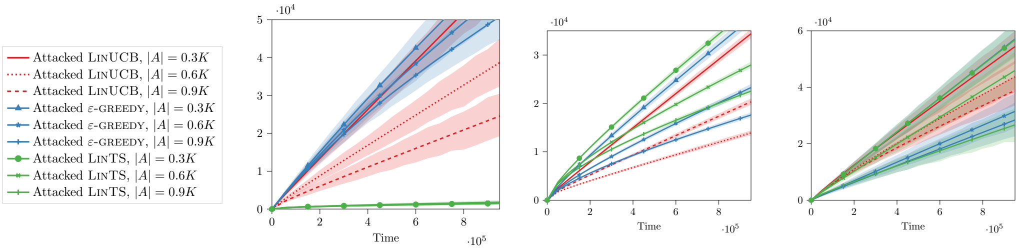
Fig. 5 (left) shows that the total cost of attacks seems to be quite invariant w.r.t. γ except when $\gamma \to 0$ because the difference between the target arm and the other becomes negligible. This is also depicted by the total number of draws (Fig. 5, Right) as the number of draws plummets when $\gamma \to 0$.

B.3 Attacks on all Contexts

Fig. B.3 shows the regret for all the attacks. This figure shows that even though the total cost of attacks is linear for algorithms like LinTS in the synthetic dataset, the regret is linear. More generally, we observe that the regret is linear for all attacked algorithms on all datasets.

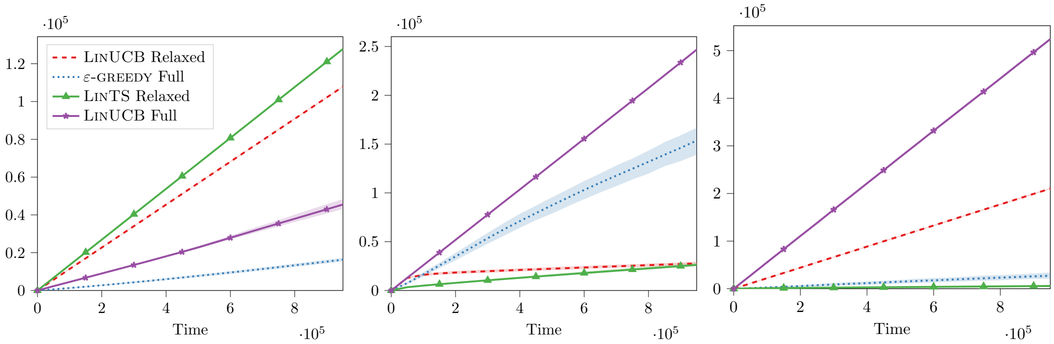
Table 2: Number of draws of the target arm $a^\dagger$ at $T = 10^6$, for the synthetic data, $\gamma = 0.22$ for the Contextual ACE algorithm and for the Jester and MovieLens datasets $\gamma = 0.5$.

	Synthetic	Jester	Movilens
LinUCB	$86,731.6$	$23,548.16$	$25,017.31$
CACE LinUCB	$996,238.6$	$921,083.69$	$944,721.28$
Stationary CACE LinUCB	$995,578.88$	$862,095.67$	$931,531.6$
ε-GREEDY	$111,380.44$	$21,911.54$	$3,165.81$
CACE ε-GREEDY	$999,812.92$	$999,755.72$	$999,776.82$
Stationary CACE ε-GREEDY	$999,806.32$	$999,615.98$	$999,316.76$
LinTS	$91,664.8$	$23,398.3$	$30,189.84$
CACE LinTS	$998,997.04$	$976,708.9$	$990,250.67$
Stationary CACE LinTS	$977,850.96$	$784,715.62$	$845,512.98$
Exp4	$93,860.4$	$29,147.01$	$17,985.78$
CACE Exp4	$992,793.36$	$989,214.36$	$936,230.4$
Stationary CACE Exp4	$993,673.24$	$988,463.56$	$934,304.23$



B.4 Attack on a single context

The attacks are computed by solving the optimization problems 4 and 6 (Sec. 5). We choose the libraries according to their efficiency for each problem we need to solve. For Problem (6) and Problem (8) we use CVXPY [36] and the ECOS solver. For Problem (4) we use the SLSQP method from the Scipy optimize library [37] to solve the full LinUCB problem (Equation 4) and QUADPROG to solve the quadratic problem to attack ε-GREEDY.



(a) Synthetic data

(b) Jester Dataset

(c) MovieLens Dataset

Figure 6: Total cost of the attacks for the attacks one one context on our synthetic dataset, Jester and MovieLens. As expected, the total cost is linear.

C Problem (8) as a Second Order Cone (SOC) Program

Problem (6) and Problem (8) are both SOC programs. We can see the similarities between both problems as follows. Let us define for every arm $a \notin A^\dagger$, the ellipsoid:

$$\mathcal{C}'_{t,a} := \left\{ y \in \mathbb{R}^d \mid ||y - \hat{\theta}_a(t)||_{A_a^{-1}(t)} \leq \upsilon \Phi^{-1}\left(1 - \frac{\delta}{K - |A^\dagger|}\right) \right\}$$

with $A_a(t) = \tilde{V}_a^{-1}(t) + \tilde{V}_{a^\dagger}^{-1}(t)$ with $\tilde{V}_a(t)$ and $\tilde{V}_{a^\dagger}(t)$ the design matrix built by LinTS and $\hat{\theta}_a(t)$ the least squares estimate of θ_a at time t. Therefore for an arm a, the constraint in Problem (8) can be written for any $y \in \mathbb{R}^d$ and some arm $a^\dagger \in A^\dagger$ as:

$$\left\langle x^\star + y, \hat{\theta}_{a^\dagger}(t) \right\rangle - \xi \geq \max_{z \in \mathcal{C}'_{t,a}} \langle z, x^\star + y \rangle$$

Indeed for any $x \in \mathbb{R}^d$,

$$\begin{aligned}
\max_{y \in \mathcal{C}'_{t,a}} \langle y, x \rangle &= \left\langle x, \hat{\theta}_a(t) \right\rangle + \upsilon \Phi^{-1}\left(1 - \frac{\delta}{K - |A^\dagger|}\right) \times \max_{||A_a^{-1/2}(t) u||_2 \leq 1} \langle u, x \rangle \\
&= \left\langle x, \hat{\theta}_a(t) \right\rangle + \upsilon \Phi^{-1}\left(1 - \frac{\delta}{K - |A^\dagger|}\right) \max_{||z||_2 \leq 1} \left\langle z, A_a^{1/2}(t) x \right\rangle \\
&= \left\langle x, \hat{\theta}_a(t) \right\rangle + \upsilon \Phi^{-1}\left(1 - \frac{\delta}{K - |A^\dagger|}\right) \|A_a^{1/2}(t) x\|_2
\end{aligned}$$

Thus, the constraint is feasible if and only if:

$$\hat{\theta}_{a^\dagger}(t) \notin \mathrm{Conv}\left(\bigcup_{a \notin A^\dagger} \mathcal{C}'_{t,a} \right)$$

D Attacks on Adversarial Bandits

In the previous sections, we studied algorithms with sublinear regret R_T, i.e., mainly bandit algorithms designed for stochastic stationary environments. Adversarial algorithms like Exp4 do not provably enjoy a sublinear **stochastic** regret R_T (as defined in the introduction) [1]. In addition, because this type of algorithms are, by design, robust to non-stationary environments, one could expect them to induce a linear cost on the attacker. In this section, we show that this is not the case for most contextual adversarial algorithms. Contextual adversarial algorithms are studied through the reduction to the bandit with expert advice problem. This is a bandit problem with K arms where at every step, N experts suggest a probability distribution over the arms. The goal of the algorithm is to learn which expert gets the best expected reward in hindsight after T steps. The regret in this type of problem is defined as $R_T^{\text{exp}} = \mathbb{E}\left(\max_{m \in [\![1,N]\!]} \sum_{t=1}^T \sum_{j=1}^K E_{m,j}^{(t)} r_{t,j} - r_{t,a_t} \right)$ where $E_{m,j}^{(t)}$ is the probability of selecting arm j for expert m. In the case of contextual adversarial bandits, the experts first observe the context x_t before recommending an expert m. Assuming the current setting with linear rewards, we can show that if an algorithm $\mathfrak{A}$, like Exp4, enjoys a sublinear regret R_T^{exp}, then, using the Contextual ACE attack with either $\tilde{r}^1$ or $\tilde{r}^2$, the attacker can fool the algorithm into pulling arm $a^\dagger$ a linear number of times under some mild assumptions. However, attacking contexts for this type of algorithm is difficult because, even though the rewards are linear, the experts are not assumed to use a specific model for selecting an action.

Proposition 3. *Suppose an adversarial algorithm $\mathfrak{A}$ satisfies a regret R_T^{exp} of order $o(T)$ for any bandit problem and that there exists an expert $m^\star$ such that $T - \sum_{t=1}^T \mathbb{E}\left(E_{m^\star, a^\dagger_{t,\star}}^{(t)} \right) = o(T)$ with $a^\dagger_{t,\star}$ the optimal arim in $A^\dagger$ at time t. Then attacking alg. $\mathfrak{A}$ with Contextual ACE leads to pulling arm $a^\dagger$, $T - o(T)$ of times in expectation with a total cost of $o(T)$ for the attacker.*

Proof. Similarly to the proof of Proposition 1, let's define the bandit with expert advice problem, $\mathcal{A}_i$, such that at each time t the reward vector is $(\tilde{r}^i_{t,a})_a$ (with $i \in \{1, 2\}$). The regret of this

[1] Exp4 enjoys a sublinear hindsight regret though. Showing a sublinear upper-bound for the stochastic regret of Exp4 is still an open problem (see Section 29.1 in [18])

algorithm is: $\tilde{R}_T^{i,\text{exp}} = \mathbb{E}\left(\max_{m\in[\![1,N]\!]} \sum_{t=1}^T E_m^{(t)} \tilde{r}_t^i - \tilde{r}_{t,a_t}^i\right) \in o(T)$. The regret of the learner is: $\mathbb{E}\left(\max_{m\in[\![1,N]\!]} \sum_{t=1}^T E_m^{(t)} r_t - r_{t,a_t}\right)$ where a_t are the actions taken by algorithm $\mathcal{A}_i$ to minimize $\tilde{R}_T^{i,\text{exp}}$. Then we have:

$$\tilde{R}_T^{i,\text{exp}} \geq \mathbb{E}\left(\sum_{t=1}^T \sum_{j=1}^K (E_{m^\star,j}^{(t)} - \mathbb{1}_{\{j=a_{t,\star}^\dagger\}})\tilde{r}_{t,j}^i + \sum_{t=1}^T \tilde{r}_{t,a_{t,\star}^\dagger}^i - \tilde{r}_{t,a_t}^i\right)$$

Therefore,

$$\begin{aligned}
\mathbb{E}\left(\sum_{t=1}^T \tilde{r}_{t,a_{t,\star}^\dagger}^i - \tilde{r}_{t,a_t}^i\right) &\leq \tilde{R}_T^{i,\text{exp}} + \mathbb{E}\left(\sum_{t=1}^T \sum_{j=1}^K (\mathbb{1}_{\{j=a_{t,\star}^\dagger\}} - E_{m^\star,j}^{(t)})\tilde{r}_{t,j}^i\right) \\
&\leq \tilde{R}_T^{i,\text{exp}} + \mathbb{E}\left(\sum_{t=1}^T (1 - E_{m^\star,a_{t,\star}^\dagger}^{(t)})\tilde{r}_{t,j}^i\right) \\
&\leq \tilde{R}_T^{i,\text{exp}} + \mathbb{E}\left(\sum_{t=1}^T (1 - E_{m^\star,a_{t,\star}^\dagger}^{(t)})\right)
\end{aligned}$$

For strategy $i = 1$, we have:

$$\mathbb{E}\left(\sum_{t=1}^T \tilde{r}_{t,a_{t,\star}^\dagger}^1 - \tilde{r}_{t,a_t}^1\right) = \sum_{t=1}^T \mathbb{E}\left(r_{t,a_{t,\star}^\dagger} - \mathbb{1}_{\{a_t\in A^\dagger\}}\right) \geq \left(T - \mathbb{E}\left(\sum_{t=1}^T \mathbb{1}_{\{a_t=a_{t,\star}^\dagger\}}\right)\right)\Delta$$

where $\Delta := \min_{x\in\mathcal{D}}\left\{\langle\theta_{a^\dagger(x)}, x\rangle - \max_{a\in A^\dagger, a\neq a^\dagger(x)}\langle\theta_{a'}, x\rangle\right\}$ with $a^\dagger(x) := \arg\max_{a\in A^\dagger}\langle\theta_a, x\rangle$. Then, as $\tilde{R}_T^{1,\text{exp}} \in o(T)$ and $\mathbb{E}\left(\sum_{t=1}^T (1 - E_{m^\star,a_{t,\star}^\dagger}^{(t)})\right) \in o(T)$, we deduce that $\mathbb{E}(\sum_t \mathbb{1}_{\{a_t=a_{t,\star}^\dagger\}}) = T - o(T)$.

For strategy $i = 2$, and $\delta > 0$, let us denote by E_δ the event that all confidence intervals hold with probability $1-\delta$. But on the event E_δ, for a time t where $a_t \neq a_{t,\star}^\dagger$ and such that $-1 \leq C_{t,a_t} \leq 0$:

$$\begin{aligned}
\tilde{r}_{t,a_t}^2 = r_{t,a_t} + C_{t,a_t} &= (1-\gamma)\min_{a^\dagger\in A^\dagger}\min_{\theta\in\mathcal{C}_{t,a^\dagger}}\langle\theta, x_t\rangle + \eta_{a_t,t} + \langle\theta_a, x_t\rangle - \max_{\theta\in\mathcal{C}_{t,a_t}}\langle\theta, x_t\rangle \\
&\leq (1-\gamma)\langle\theta_{a_{t,\star}^\dagger}, x_t\rangle + \eta_{a_t,t}
\end{aligned}$$

when $C_{t,a_t} > 0$ (still on the event E_δ):

$$\tilde{r}_{t,a_t}^2 = r_{t,a_t} \leq (1-\gamma)\langle\theta_{a_{t,\star}^\dagger}, x_t\rangle + \eta_{a_t,t}$$

because $C_{t,a_t} > 0$ means that $(1-\gamma)\langle\theta_{a_{t,\star}^\dagger}, x_t\rangle \geq (1-\gamma)\min_{a^\dagger\in A^\dagger}\min_{\theta\in\mathcal{C}_{t,a^\dagger}}\langle\theta, x_t\rangle \geq \max_{\theta\in\mathcal{C}_{t,a_t}}\langle\theta, x_t\rangle \geq \langle\theta_a, x_t\rangle$. But finally, when $C_{t,a_t} \leq -1$, $\tilde{r}_{t,a_t}^2 = r_{t,a_t} - 1 \leq \eta_{a_t,t} \leq (1-\gamma)\langle\theta_{a_{t,\star}^\dagger}, x_t\rangle + \eta_{a_t,t}$. But on the complementary event E_δ^c, $\tilde{r}_{t,a_t}^2 \leq r_{t,a_t}$. Thus, given that the expected reward is assumed to be bounded in $(0,1]$ (Assumption 1):

$$\begin{aligned}
\mathbb{E}\left(\sum_{t=1}^T \tilde{r}_{t,a_{t,\star}^\dagger}^2 - \tilde{r}_{t,a_t}^2\right) &= \mathbb{E}\left(\sum_{t=1}^T (r_{t,a^\dagger} - \tilde{r}_{t,a_t}^2)\mathbb{1}_{\{a_t\neq a_{t,\star}^\dagger\}}\right) \\
&\geq \mathbb{E}\left(\sum_{t=1}^T \min\{\gamma\min_{x\in\mathcal{D}}\langle x, \theta_{a_{t,\star}^\dagger}\rangle, \Delta\}\mathbb{1}_{\{a_t\neq a_{t,\star}^\dagger\}}\mathbb{1}_{\{E_\delta\}}\right) - T\delta
\end{aligned}$$

Finally, putting everything together we have:

$$\mathbb{E}\left(\sum_{t=1}^T \gamma\min_{x\in\mathcal{D}}\langle x, \theta_{a_{t,\star}^\dagger}\rangle\mathbb{1}_{\{a_t\neq a_{t,\star}^\dagger\}}\right) \leq \tilde{R}_T^{2,\text{exp}} + \mathbb{E}\left(\sum_{t=1}^T (1 - E_{m^\star,a_{t,\star}^\dagger}^{(t)})\right) + \delta T\left(\min\{\gamma\min_{a^\dagger\in A^\dagger}\min_{x\in\mathcal{D}}\langle x, \theta_{a^\dagger}\rangle, \Delta\} + 1\right)$$

Hence, because $\tilde{R}_T^{1,\text{exp}} = o(T)$ and $\mathbb{E}\left(\sum_{t=1}^T (1 - E_{m^\star, a^\dagger}^{(t)})\right) = o(T)$ we have that for $\delta \leq 1/T$, the expected number of pulls of the optimal arm in $A^\dagger$ is of order $o(T)$. In addition, the cost for the attacker is bounded by:

$$\mathbb{E}\left(\sum_{t=1}^T c_t\right) = \mathbb{E}\left(\sum_{t=1}^T \mathbb{1}_{\{a_t \neq a_{t,\star}^\dagger\}} \left|\max(-1, \min(C_{t,a_t}, 0))\right|\right) \leq \mathbb{E}\left(\sum_{t=1}^T \mathbb{1}_{\{a_t \neq a_{t,\star}^\dagger\}}\right)$$

□

The proof is similar to the one of Prop. 1. The condition on the expert in Prop. 3 means that there exists an expert which believes an arm $a^\dagger \in A^\dagger$ is optimal most of the time. The adversarial algorithm will then learn that this expert is optimal. Algorithm Exp4 has a regret R_T^{exp} bounded by $\sqrt{2TK\log(N)}$, thus the total number of pulls of arms not in $A^\dagger$ is bounded by $\sqrt{2TK\log(M)}/\gamma$. This result also implies that for adversarial algorithms like Exp3 [17], the same type of attacks could be used to fool $\mathfrak{A}$ into pulling arms in $A^\dagger$ because the MAB problem can be seen as a reduction of the contextual bandit problem with a unique context and one expert for each arm.

E Contextual Bandit Algorithms

In this appendix, we present the different bandit algorithms studied in this paper. All algorithms we consider except Exp4 uses disjoint models for building estimate of the arm feature vectors $(\theta_a)_{a\in[\![1,K]\!]}$. Each algorithm (except Exp4) builds least squares estimates of the arm features.

Algorithm 1 Contextual LinUCB

```
Input: regularization $\lambda$, number of arms $K$, number of rounds $T$, bound on context norms: $L$, bound on norms $\theta_a$: $D$
Initialize for every arm $a$, $\bar{V}_a^{-1}(t) = \frac{1}{\lambda} I_d$, $\hat{\theta}_a(t) = 0$ and $b_a(t) = 0$
for $t = 1, ..., T$ do
    Observe context $x_t$
    Compute $\beta_a(t) = \sigma\sqrt{d\log\left(\frac{1+N_a(t)L^2/\lambda}{\delta}\right)}$ with $N_a(t)$ the number of pulls of arm $a$
    Pull arm $a_t = \text{argmax}_a \langle\hat{\theta}_a(t), x_t\rangle + \beta_a(t)||x_t||_{\bar{V}_a^{-1}(t)}$
    Observe reward $r_t$ and update parameters $\hat{\theta}_a(t)$ and $\bar{V}_a^{-1}(t)$ such that:
    $\bar{V}_{a_t}(t+1) = \bar{V}_{a_t}(t) + x_t x_t^\intercal, \quad b_{a_t}(t+1) = b_{a_t}(t) + r_t x_t, \quad \theta_{a_t}(t+1) = \bar{V}_{a_t}^{-1}(t+1) b_{a_t}(t+1)$
end for
```

Algorithm 2 Linear Thompson Sampling with Gaussian prior

```
Input: regularization $\lambda$, number of arms $K$, number of rounds $T$, variance $\upsilon$
Initialize for every arm $a$, $\bar{V}_a^{-1}(t) = \lambda I_d$ and $\hat{\theta}_a(t) = 0$, $b_a(t) = 0$
for $t = 1, ..., T$ do
    Observe context $x_t$
    Draw $\tilde{\theta}_a \sim \mathcal{N}(\hat{\theta}_a(t), \upsilon^2 \bar{V}_a^{-1}(t))$
    Pull arm $a_t = \text{argmax}_{a\in[\![1,K]\!]} \left\langle \tilde{\theta}_a, x_t \right\rangle$
    Observe reward $r_t$ and update parameters $\hat{\theta}_a(t)$ and $\bar{V}_a^{-1}(t)$
    $\bar{V}_{a_t}(t+1) = \bar{V}_{a_t}(t) + x_t x_t^\intercal, \quad b_{a_t}(t+1) = b_{a_t}(t) + r_t x_t, \quad \theta_{a_t}(t+1) = \bar{V}_{a_t}^{-1}(t+1) b_{a_t}(t+1)$
end for
```

F Semi-Online Attacks

[14] studies what they call the offline setting for adversarial attacks on stochastic bandits. They consider a setting where a bandit algorithm is successively updated with mini-batches of fixed size B.

Algorithm 3 ε-GREEDY

```
Input: regularization $\lambda$, number of arms $K$, number of rounds $T$, exploration parameter $(\varepsilon)_t$
Initialize, for all arms $a$, $\bar{V}_a^{-1}(t) = \lambda I_d$ and $\hat{\theta}_a(t) = 0$, $\varepsilon_t = 1$, $b_a(t) = 0$
for $t = 1, ..., T$ do
    Observe context $x_t$
    With probability $\varepsilon_t$, pull $a_t \sim \mathcal{U}(\llbracket 1, K \rrbracket)$, or pull $a_t = \mathrm{argmax}\langle \theta_a, x_t \rangle$
    Observe reward $r_t$ and update parameters $\hat{\theta}_a(t)$ and $\bar{V}_a^{-1}(t)$
        $\bar{V}_{a_t}(t+1) = \bar{V}_{a_t}(t) + x_t x_t^\intercal, \quad b_{a_t}(t+1) = b_{a_t}(t) + r_t x_t,$
        $\theta_{a_t}(t+1) = \bar{V}_{a_t}^{-1}(t+1) b_{a_t}(t+1)$
end for
```

Algorithm 4 EXP4

```
Input: number of arms $K$, experts: $(E_m)_{m \in \llbracket 1, N \rrbracket}$, parameter $\eta$
Set $Q_1 = (1/N)_{j \in \llbracket 1, N \rrbracket}$
for $t = 1, ..., T$ do
    Observe context $x_t$ and probability recommendation $(E_m^{(t)})_{m \in \llbracket 1, N \rrbracket}$
    Pull arm $a_t \sim P_t$ where $P_{t,j} = \sum_{k=1}^N Q_{t,k} E_{j,k}^{(t)}$
    Observe reward $r_t$ and define for all arms $i$ $\hat{r}_{t,i} = 1 - \mathbb{1}_{\{a_t = i\}}(1 - r_t)/P_{t,i}$
    Define $\tilde{X}_{t,k} = \sum_a E_{k,a}^{(t)} \hat{r}_{t,a}$
    Update $Q_{t+1,j} = \exp(\eta Q_{t,i}) / \sum_{j=1}^N \exp(\eta Q_{t,j})$ for all experts $i$
end for
```

The attacker can tamper with some of the incoming mini-batches. More precisely, they can modify the context, the reward and even the arm that was pulled for any entry of the attacked mini-batches. The main difference between this type of attacks and the online attacks we considered in the main paper is that we do not assume that we can attack from the start of the learning process: the bandit algorithm may have already converged by the time we attack.

We can still study the cumulative cost for the attacker to change the mini-batch in order to fool a bandit algorithm to pull a target arm $a^\dagger$ (here we take $A^\dagger = \{a^\dagger\}$). Contrarily to [14], we call this setting semi-online. We first study the impact of an attacker on LINUCB where we show that, by modifying only $(K-1)d$ entries from the batch $\mathcal{B}$, the attacker can force LINUCB to pull arm $a^\dagger$, $M'B - o(M'B)$ times with M' the number of remaining batches updates. The cost of our attack is $\sqrt{MB}$ with M the total number of batches.

Cost of an attack: If presented with a mini-batch $\mathcal{B}$, with elements (x_t, a_t, r_t) composed of the context x_t presented at time t, the action taken a_t and the reward received r_t, the attacker modifies element i, namely (x_t^i, a_t^i, r_t^i) into $(\tilde{x}_t^i, \tilde{a}_t^i, \tilde{r}_t^i)$. The cost of doing so is $c_t^i = ||x_t^i - \tilde{x}_t^i||_2 + \left|\tilde{r}_t^i - r_t^i\right| + \mathbb{1}_{\{a_t^i \neq \tilde{a}_t^i\}}$ and the total cost for mini-batch $\mathcal{B}$ is defined as $c_{\mathcal{B}} = \sum_{i \in \mathcal{B}} c_t^i$. Finally, we consider the cumulative cost of the attack over M different mini-batches $\mathcal{B}_1, \ldots, \mathcal{B}_M$, $\sum_{l=1}^M c_{\mathcal{B}_l}$. The interaction between the environment, the attacker and the learning algorithm is summarized in Alg. 5.

The attack presented here is based on the Ahlberg–Nilson–Varah bound [38], which gives a control on the sup norm of a matrix with dominant diagonal elements. More precisely, when presented with a mini-batch $\mathcal{B}$, the attacker needs to modify the contexts and the rewards. We assume that the attacker knows the number of mini-batch updates M and has access to a lower-bound on the reward of the target arm, ν as in Assumption 2.

The attacker changes $(K-1) \times d$ rows of the first mini-batch to rewards of 0 with a context $\delta_a e_i$ for each arm $a \neq a^\dagger$ with (e_i) the canonical basis of $\mathbb{R}^d$. Moreover, δ_a is chosen such that:

$$\delta_a > \max\left(\sqrt{\frac{2MBL^2 d}{\nu} + dMB}, \sqrt{\frac{4\beta_{max}^2 L^2 d}{\nu^2} + dMB}\right) \tag{13}$$

Algorithm 5 Semi-Online Attack Setting.

```
Input: Bandit alg. $\mathfrak{A}$, size of a mini-batch: $B$
Set $t = 0$
while True do
  $\mathfrak{A}$ observe context $x_t$
  $\mathfrak{A}$ pulls arm $a_t$ and observes reward $r_t$
  Interaction $(x_t, a_t, r_t)$ is saved in mini-batch $\mathcal{B}$
  if $\left|\mathcal{B}\right| = B$ then
    Attacker modifies mini-batch $\mathcal{B}$ into $\tilde{\mathcal{B}}$
    Update alg. $\mathfrak{A}$ with poisoned mini-batch $\tilde{\mathcal{B}}$
  end if
end while
```

with $\beta_{max} = \max_{t=0}^{MB} \beta_a(t)$ and M the number of mini-batch updates.

Proposition 4. *After the first attack, with probability* $1 - \delta$*,* LinUCB *always pulls arm* $a^\dagger$*,*

Proof. After having poisoned the first mini-batch $\mathcal{B}$, the latter can be partitioned into two subsets, $\mathcal{B}_c$ (with non-perturbed rows) and $\mathcal{B}_{nc}$ (with the poisoned rows). The design matrix of arm $a \neq a^\dagger$ for every time t after the poisoning is:

$$V_{t,a} = \lambda I_d + \sum_{l=1, a_l = a}^{t} x_l x_l^\intercal + \delta_a^2 \sum_{i=1}^{d} e_i e_i^\intercal \tag{14}$$

For every time t, non diagonal elements of $V_{t,a} = (v_{i,j})_{i,j}$ are bounded by:

$$\forall i, r_i := \sum_{j \neq i} v_{i,j} \leq \sum_{j \neq i} \sum_{l=1, a_l = a}^{t} \|x_l x_l^\intercal\|_\infty \leq d N_a(kB) \tag{15}$$

Whereas for all diagonal elements, $v_{i,i} \geq \delta_a^2 > r_i$. Thus $V_{t,a}$ is strictly diagonal dominant and by the Ahlberg–Nilson–Varah bound [38]:

$$\|V_{t,a}^{-1}\|_\infty \leq \frac{1}{\min_i \left(\|v_{i,i}\| - r_i\right)} \leq \frac{1}{\delta_a^2 - dMB} \tag{16}$$

Then, for every arm $a \neq a^\dagger$ and any context $x \in \mathcal{D}$ and any time t after the attack:

$$\begin{aligned}
\langle \hat{\theta}_{a,t}, x \rangle + \beta_a(t) ||x||_{V_{t,a}^{-1}} &\leq \sum_{l=1, a_l = a}^{t} r_t (V_{t,a}^{-1} x_t)^\intercal x + \beta_a(t) ||x||_1 \sqrt{\|V_{t,a}^{-1}\|_\infty} \\
&\leq \|V_{t,a}^{-1}\|_\infty d N_t(a) \sup_{y \in \mathcal{D}} ||y||_2^2 + \beta_{\max} \sqrt{d} \sup_{y \in \mathcal{D}} ||y||_2 \sqrt{\|V_{t,a}^{-1}\|_\infty} < \nu
\end{aligned}$$

We have shown that for any arm $a \neq a^\dagger$ and any time step t after the attack, the upper confidence bound computed by LinUCB is upper-bounded bu ν the arm $a^\dagger$. Then, with probability $1 - \delta$, the confidence set for arm $a^\dagger$ holds and, for all $x \in \mathcal{D}$, arm $a^\dagger$ is chosen by LinUCB. The total cost of this attack is $d \sum_{a \neq a^\dagger} \delta_a L = O(\sqrt{MB})$ □