Interconnected Neural Linear Contextual Bandits with UCB Exploration

Yang Chen[1(✉)], Miao Xie[2], Jiamou Liu[1], and Kaiqi Zhao[1]

[1] The University of Auckland, Auckland, New Zealand
{yang.chen,jiamou.liu,kaiqi.zhao}@auckland.ac.nz
[2] Kuaishou Technology Co. Ltd., Beijing, China

Abstract. Contextual multi-armed bandit algorithms are widely used to solve online decision-making problems. However, traditional methods assume linear rewards and low dimensional contextual information, leading to high regrets and low online efficiency in real-world applications. In this paper, we propose a novel framework called *interconnected neural-linear UCB* (InlUCB) that interleaves two learning processes: an offline representation learning part, to convert the original contextual information to low-dimensional latent features via non-linear transformation, and an online exploration part, to update a linear layer using upper confidence bound (UCB). These two processes produce an effective and efficient strategy for online decision-making problems with non-linear rewards and high dimensional contexts. We derive a general expression of the finite-time cumulative regret bound of InlUCB. We also give a tighter regret bound under certain assumptions on neural networks. We test InlUCB against state-of-the-art bandit methods on synthetic and real-world datasets with non-linear rewards and high dimensional contexts. Results demonstrate that InlUCB significantly improves the performance on cumulative regrets and online efficiency.

Keywords: Contextual bandits · Upper confidence bound · Neural networks · Regret bound

1 Introduction

Contextual multi-armed bandit algorithms are powerful solutions to online sequential decision making problems such as influence maximisation [17] and recommendation [20]. In its setting, an agent sequentially observes a feature vector associated with each arm (action), called the *context*. Based on the contexts, the agent selects an arm which provides a random reward that is assumed to follow some distribution. Since the underlying distribution is unknown and the reward can only be observed at run-time, the agent should balance exploration and exploitation to maximise total rewards or, equivalently, to minimise *regret*.

Well-established contextual bandit methods, e.g., linear upper confidence bound (LinUCB) [7] and linear Thompson sampling (LinTS) [2], were effective

J. Gama et al. (Eds.): PAKDD 2022, LNAI 13280, pp. 169–181, 2022.
https://doi.org/10.1007/978-3-031-05933-9_14

assuming that the reward is linear and the contexts are low-dimensional. However, these methods face two important challenges when applied to real-world scenarios such as online image classification [19] and online audio recognition [23]: First, these applications involve reward distributions that are non-linear w.r.t. the contexts. This violates the linear-reward assumption which is needed to achieve non-trivial regret bounds. Thus it is possible that these methods could result in high regrets. Second, most of these existing methods involve inverting a matrix online [4] whose dimension coincides with that of the contexts. However, the applications above usually involve high-dimensional contexts. Thus the existing methods will suffer from poor online efficiency. Although some recent methods relax the liner-reward assumption, they still rely on relatively restrictive modelling assumptions on rewards and/or cannot provide acceptable online efficiency. For instance, KernelUCB [16] relaxes the linear reward assumption by asserting that the reward function belongs to a reproducing kernel Hilbert space, but it incurs an even higher computation cost on matrix inversions as the dimension of the kernel matrix increases with time.

Recently, several new methods under the name of *neural contextual bandits* [24] are proposed to extend classical contextual bandit algorithms. Leveraging the expressive power of neural networks, these methods aim to learn richer non-linear reward function and latent features through representation learning. So far, two major neural contextual bandit paradigms have been proposed: Neural-Linear and NeuralUCB. The former uses neural networks to extract a dimension-reduced latent feature (representation learning) and conduct exploration on top of the latent features [15,22], while the latter uses neural networks as a reward predictor and use UCB for exploration [24]. Despite showing promise in certain empirical tasks, these methods still suffer from some significant shortcomings. (1) While Neural-Linear is time-efficient, the method often incurs high regrets. This is because that it trains networks end-to-end, failing to use the result of exploration to boost representation learning. Worse yet, its regret bound is still unknown [15]. (2) NeuralUCB, in contrast, can provide the theoretical guarantee on regret bound. But updating the entire network every step results in low online efficiency, which makes it infeasible in practice.

Contributions. This paper addresses the need for an *efficient* contextual bandit algorithm applicable to *non-linear rewards* and *high-dimensional contexts*. We summarise our main contributions as follows: **(1)** We propose a new neural contextual bandit framework, called *interconnected neural-linear upper confidence bound* (InlUCB) (see Sect. 4). To our knowledge, InlUCB is the first contextual bandit method that achieves high online efficiency with a theoretical guarantee on its regret bound. InlUCB uses neural networks with two parts: the lower layers transform raw contexts to a low-dimensional latent feature space; and the last linear layer represents a linear model that fits the observed reward in terms of the latent features. The key novelty of InlUCB lies in an *interconnected offline-online update mechanism* to train the two parts. The offline process (*representation learning*) updates lower layers subject to the current linear model, simplifying the task at hand. The online process (*exploration*) follows

UCB-based exploration to update only the last linear layer based on the proposed representation, thereby guaranteeing online efficiency. **(2)** We derive a general expression of the regret bound of InLUCB by decomposing the total regrets into regrets caused by representation learning and online exploration (see Theorem 1). Specifically, we present a tighter regret bound under certain assumptions on neural networks (see Corollary 1). **(3)** We test InLUCB against state-of-the-art contextual, non-linear contextual, non-parametric contextual and neural contextual bandit methods on synthetic dataset with high-dimensional contexts and non-linear rewards as well as on real-world datasets with audio and images as contextual information (see Sect. 6). Results demonstrate that InLUCB achieves much lower cumulative regrets than linear contextual bandit baselines and higher online efficiency than neural contextual bandit baselines.

2 Related Work

Classical Contextual Bandits. Both classical multi-armed bandits and contextual bandits have been studied extensively along with their variants. Classical bandit algorithms such as Upper Confidence Bound (UCB) and Thompson Sampling (TS) [1] achieve $\widetilde{O}(\sqrt{KT})$ regret, where K is the number of candidate arms, T is the number of steps, and $\widetilde{O}(\cdot)$ hides the logarithmic factors. Since this regret bound depends on K, they are inefficient in real-world applications when K is large. To alleviate this problem, one can assume that the reward of each arm is a function of some observed features (i.e., contexts), yielding the family of methods called the *contextual bandits* [9,13]. As two widely-adopted contextual bandit algorithms with the linear-reward assumption, LinUCB [7] and LinTS [2] have a regret bound of $\widetilde{O}(\sqrt{dT})$ and $\widetilde{O}(d\sqrt{T})$, respectively, which depends on the dimension d of features rather than K. However, contextual bandits may result in high regrets when the reward function is non-linear or d is large.

High-Dimensional Contexts and Non-linear Rewards. Despite works exist that attempt to extend contextual bandits to the setting of either high-dimensional contexts or non-linear rewards [6,11,18,21], no method so far can resolve these two challenges simultaneously with acceptable efficiency. Lasso regression is investigated for the sparse contexts [6,11,18]. Although its regret bound [11] is superior to LinUCB, optimising a lasso regression problem online makes it too time-consuming to be used in practice. CBRAP [21] adopts random projection to map the high-dimensional contexts onto a low-dimensional space. Although it improves efficiency, its performance heavily relies on a good initial projection matrix, leading to poor robustness. KernelUCB [16] adopts kernel functions to handle non-linear rewards but it uses matrix inversion which incurs high computation costs. Neural-Linear [22] and NeuralUCB [24] adopt neural networks to model rewards, each with issues mentioned above. In InLUCB, we propose a novel interconnected-update framework that makes our method unique and allows our method to overcome the shortcomings of the existing neural contextual bandit methods.

3 Problem Formulation and Background

Problem Setting. We consider the stochastic contextual bandit problem with K *arms* (actions) and T *steps*. At each step $t \leq T$, the agent observes the *context* (feature) $\mathbf{x}_{t,a} \in \mathbb{R}^d$ of each arm a with $\|\mathbf{x}_{t,a}\|_2 \leq 1$, where the *contextual dimension* d is usually very **large** in applications. An algorithm selects an action $a_t \in [K]$ at step t and receives a *reward* $r_{t,a_t} \in [0,1]$, where $[K]$ denotes the set $\{1, 2, \ldots, K\}$. The reward r_{t,a_t} is an independent random variable conditioned on context $\mathbf{x}_{t,a_t}$. The *regret* of the algorithm is defined as:

$$R_T \triangleq \sum_{t=1}^{T} r_{t,a_t^*} - \sum_{t=1}^{T} r_{t,a_t}, \tag{1}$$

where $a_t^* = \arg\max_{a \in [K]} \mathbb{E}[r_{t,a}|\mathbf{x}_{t,a}]$ is the optimal action at step t that maximises the expected reward. The goal is to find an algorithm to minimise R_T.

We focus on the cases where the reward function is **non-linear** in terms of contexts. To capture this fact, for each step t, we assume that the reward is generated by:

$$r_{t,a} \triangleq g(\mathbf{x}_{t,a}) + \xi_t, \tag{2}$$

where $g\colon \mathbb{R}^d \to \mathbb{R}$ is an unknown non-linear function satisfying $g(\mathbf{x}) \in [0,1]$ for any $\mathbf{x}$, and ξ_t is a sub-Gaussian noise satisfying $\mathbb{E}[\xi_t] = 0$. The sub-Gaussian noise is a standard assumption in the stochastic bandit literature, which can represent any bounded noise [14].

Neural Contextual Bandits. Neural contextual bandit methods [15,22,24] compute the rewards using a neural network. In this way, the method handles high dimensional contexts and non-linear rewards. Formally, the function g in Eq. (2) is realised by:

$$g(\mathbf{x}_{t,a}) = f_\star^\top(\mathbf{x}_{t,a})\boldsymbol{\theta}_\star, \tag{3}$$

where $f_\star\colon \mathbb{R}^d \to \mathbb{R}^p$ represents all layers except the last that satisfies $\|f_\star(\mathbf{x}_{t,a})\|_2 \leq 1$, $\boldsymbol{\theta}_\star$ represents the weights of the last linear layer that satisfies $\|\boldsymbol{\theta}_\star\|_2 \leq 1$, and $p \ll d$. We call f and $\boldsymbol{\theta}$ the *dimension reduction mapping* and *latent weight vector*, respectively. Intuitively, f serves as a non-linear transformation that converts raw contexts of a large dimension d to latent features of a much lower dimension p, and the reward function is linear in latent features. Since a neural network with suitable size and activation functions is a global function approximator [5], it is reasonable to assume that Eq. (3) expresses the underlying reward function, i.e., there exists a pair $(f_\star, \boldsymbol{\theta}_\star)$ that fulfils Eq. (3).

We introduce the two major neural contextual bandits: (1) NEURAL-LINEAR [15,22] trains $\boldsymbol{\theta}$ by applying linear contextual bandit methods (e.g., UCB or TS) on top of f for exploration. The training of f (representation learning) and $\boldsymbol{\theta}$ (exploration) are executed at different time-scales. Whenever the exploration is terminated, we turn to representation learning by training the entire model (both f and $\boldsymbol{\theta}$) end-to-end. Although online exploration quantifies uncertainties over rewards, end-to-end training makes NEURAL-LINEAR ignore this important

information in representation learning. This may lead to a low convergence speed and thereby result in high regrets. Notably, the regret bound of NEURAL-LINEAR is still unknown. (2) and NEURALUCB [24] provides a regret bound of $\widetilde{O}(\tilde{d}\sqrt{T})$ through leveraging the *neural tangent kernel* (NTK) [10] to characterise a fully connected neural network, where $\tilde{d}$ is the effective dimension of a NTK matrix. However, reformulating a neural network as a NTK matrix requires updating all parameters (both f and $\boldsymbol{\theta}$) of a neural network at once after each step of online decision-making, making NEURALUCB too inefficient to be used in practice.

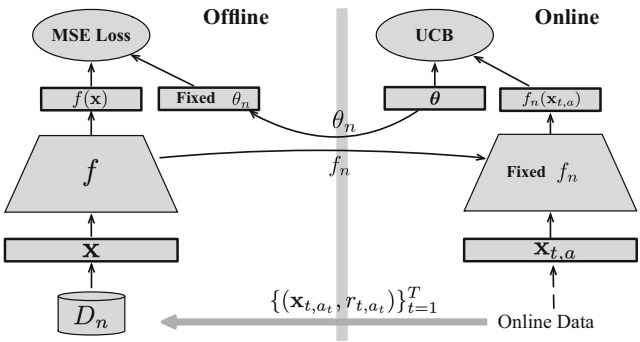

Fig. 1. The process flow of INLUCB framework. Solid and dashed arrows represent input/output and sampling, respectively.

4 The Interconnected Neural-Linear UCB Framework

To address the need for novel contextual bandit methods with non-linear rewards and high-dimensional contexts, we propose a new contextual bandit framework called *interconnected neural-linear UCB* (INLUCB). Following the neural contextual bandits regime, INLUCB alternates between the training of f and $\boldsymbol{\theta}$.

The key to INLUCB is an interconnected online-offline mechanism rather than end-to-end training. Fixing f, the online process tunes $\boldsymbol{\theta}$ using UCB to balance exploration and exploration. In turn, freezing $\boldsymbol{\theta}$, the offline process updates f based on samples collected by online exploration. Figure 1 depicts this mechanism. This interconnected update mechanism overcomes the shortcoming of NEURAL-LINEAR in the sense that representation learning and online exploration are alternatively performed to *boost each other*. Besides, the method has two extra advantages: (1) online exploration is an effective way to sample data since initially data is often too scarce to train the entire model offline, i.e., the *cold-start problem*; (2) moving the heavy workload of updating hidden layers offline can significantly improve online efficiency. Formally, let $n \in [N]$ denote the index of iterations, and denote by $\boldsymbol{\theta}_n$ and f_n the values of $\boldsymbol{\theta}$ and f after the nth iteration, respectively. Let D_n denote the offline dataset at the nth iteration. Initially, we assume $D_0 = \varnothing$. We next formally introduce the two processes.

Online Exploration. Each iteration starts from the online exploration. In the nth iteration, we fix f_{n-1} to extract a latent feature $f_{n-1}(\mathbf{x}_{t,a})$ for each context $\mathbf{x}_{t,a}$. As for updating $\boldsymbol{\theta}$, we apply LinUCB on top of the extracted latent features for exploration. The basic idea is to maintain a *reward predictor* (i.e., predicted expected reward) $\hat{r}_{t,a}$ and a confidence interval around it with *width* $w_{t,a}$ that captures the variance of rewards. Then, at each step t, we choose the action with the highest upper confidence bound $\hat{r}_{t,a} + w_{t,a}$. Formally, we use $\boldsymbol{\theta}_{n,t}$ to denote the estimation of $\boldsymbol{\theta}$ at the tth step of the nth iteration. For each action a, the reward predictor and the width of the confidence interval are given by

$$\hat{r}_{t,a} \triangleq f_{n-1}^{\top}(\mathbf{x}_{t,a})\boldsymbol{\theta}_{n,t}, \text{ and } w_{t,a} \triangleq \alpha\sqrt{f_{n-1}^{\top}(\mathbf{x}_{t,a})\mathbf{A}_{n,t}^{-1}f_{n-1}(\mathbf{x}_{t,a})}, \tag{4}$$

where $\alpha > 0$ is a given constant and

$$\mathbf{A}_{n,t} \triangleq \mathbf{I}_p + \sum\nolimits_{\tau=1}^{t-1} f_{n-1}(\mathbf{x}_{\tau,a_\tau})f_{n-1}^{\top}(\mathbf{x}_{\tau,a_\tau}). \tag{5}$$

Here, $\mathbf{I}_p$ denotes the identity matrix of size p which is to guarantee the non-singularity of $\mathbf{A}_{n,t}$. After choosing an action with the highest $\hat{r}_{t,a} + w_{t,a}$, we update $\boldsymbol{\theta}$ as follows:

$$\boldsymbol{\theta}_{n,t} = \mathbf{A}_{n,t}^{-1}\mathbf{b}_{n,t}, \text{ where } \mathbf{b}_{n,t} \triangleq \sum\nolimits_{\tau=1}^{t-1} f_{n-1}(\mathbf{x}_{\tau,a_\tau})r_{\tau,a_\tau}. \tag{6}$$

Online training terminates after T steps (T is a predefined constant). Then, accumulated online samples $\{(x_{t,a_t}, r_{t,a_t})\}_{t=1}^{T}$ are appended to the offline dataset D_{n-1}, yielding D_n.

Offline Representation Learning. In the nth iteration of offline learning, we fix $\boldsymbol{\theta}_n$ and train f_n on D_n by minimising the mean square error (MSE) loss:

$$\mathcal{L}_{D_n}(f; \boldsymbol{\theta}_n) \triangleq \mathbb{E}_{(\mathbf{x},r)\sim D_n}\left[(f^{\top}(\mathbf{x})\boldsymbol{\theta}_n - r)^2\right], \tag{7}$$

Since the sub-Gaussian noise on rewards has zero mean, the minimiser of Eq. (7) is an unbiased estimator of the optimal f w.r.t. $\boldsymbol{\theta}_n$. Algorithm 1 presents the pseudocode of InLUCB.

5 Regret Analysis

This section studies the regret bound of InLUCB. By Eq. (1), the total regret of InLUCB with N iterations, each of T online steps, can be written as

$$R_{N,T} = \sum_{n=1}^{N} R_{n,T} \triangleq \sum_{n=1}^{N}\left[\sum_{t=1}^{T} r_{n,t,a_t^*} - \sum_{t=1}^{T} r_{n,t,a_t}\right], \tag{8}$$

where $R_{n,T}$ denotes the regret at the nth iteration. We study the per-iteration regret $R_{n,T}$. The total regret can then be obtained by summing $R_{n,T}$ over all N iterations. For simplicity, we will omit the iteration index n in some notations.

Algorithm 1. InLUCB

Input: $\alpha \in \mathbb{R}^+, N, T, K, d \in \mathbb{N}, p < d \in \mathbb{N}$
Output: $f : \mathbb{R}^d \to \mathbb{R}^p$ and $\boldsymbol{\theta} \in \mathbb{R}^p$.
Initialisation: $D_0 = \varnothing$, random f_0, $\mathbf{A}' \leftarrow \mathbf{I}_p$ and $\mathbf{b}' \leftarrow \mathbf{0}_p$

while $n = 1, 2, \ldots, N$ **do**
  ▷ ***Online Exploration***
  $\mathbf{A} \leftarrow \mathbf{A}'$, $\mathbf{b} \leftarrow \mathbf{b}'$
  for $t = 1, 2, \ldots, T$ **do**
    $\boldsymbol{\theta}_{n,t} \leftarrow \mathbf{A}^{-1}\mathbf{b}$
    Observe K features $\mathbf{x}_{t,1}, \mathbf{x}_{t,2}, \ldots, \mathbf{x}_{t,K} \in \mathbb{R}^d$
    for each arm $a = 1, 2, \ldots, K$ **do**
      Compute $p_{t,a} \leftarrow f_{n-1}^\top(\mathbf{x}_{t,a})\boldsymbol{\theta}_{n,t} + \alpha\sqrt{f_{n-1}^\top(\mathbf{x}_{t,a})\mathbf{A}^{-1}f_{n-1}(\mathbf{x}_{t,a})}$
    end for
    Choose action $a_t \leftarrow \arg\max_{a\in[K]} p_{t,a}$
    Observe payoff $r_{t,a_t} \in [0, 1]$
    $\mathbf{A} \leftarrow \mathbf{A} + f_{n-1}(\mathbf{x}_{t,a})f_{n-1}^\top(\mathbf{x}_{t,a})$
    $\mathbf{b} \leftarrow \mathbf{b} + f_{n-1}(\mathbf{x}_{t,a})r_{t,a_t}$
  end for
  $\boldsymbol{\theta}_n \leftarrow \boldsymbol{\theta}_{n,T}$, $D_n \leftarrow D_{n-1} \cup \{(\mathbf{x}_{t,a_t}, r_{t,a_t})\}_{t=1}^T$
  ▷ ***Offline Representation Learning***
  Fix $\boldsymbol{\theta}_n$, train $f_{(n)}$ on D_n through gradient descent on the loss defined in Eq. (7)
end while

Recall that the agent always pulls the arm with the highest UCB which is a sum of the reward predictor $\hat{r}_{t,a}$ and a width term $w_{t,a}$. Therefore, to bound $R_{n,T}$, we need to know the *error in reward prediction*:

$$|\hat{r}_{t,a} - f_\star^\top(\mathbf{x}_{t,a})\boldsymbol{\theta}_\star| = |f_{n-1}^\top(\mathbf{x}_{t,a})\boldsymbol{\theta}_{n,t} - f_\star^\top(\mathbf{x}_{t,a})\boldsymbol{\theta}_\star|.$$

Same as LinUCB, the reward predictors $\hat{r}_{t,a}$ in InLUCB are sums of *dependent* variables since predictions in later steps are made using previous outcomes, which prevents us from applying Azuma-Hoeffding inequality to control the error in reward prediction. Thus, directly analysing regret bound of InLUCB is intractable. To sidestep this problem, we use the construction in [3] to modify the online learning of InLUCB into BaseInLUCB which assumes statistical independence among samples. We then use a master algorithm SupInLUCB to pull arms in a way that ensures this assumption holds. The pseudocode for both algorithms can be found in Appendix A. In the literature of contextual bandits, due to the intractability of the regret bound of the original algorithm, the convention is to instead analyse the regret bound of the master algorithm [3,7,16], which can be viewed as an appropriate modification of the original algorithm. Following this convention, we next analyse the regret bound of SupInLUCB.

However, although the above technique ensures independence among samples, directly calculating the error in reward prediction is still intractable due to the coupling between the estimation errors of $\boldsymbol{\theta}_{n,t}$ and f_{n-1} in the total error of reward prediction. One of our main contributions is proposing a method to separate them by defining the *offline error*:

$$\epsilon_n \triangleq \max_{\mathbf{x}\in\mathbb{R}^d} \left|f_{n-1}^\top(\mathbf{x})\boldsymbol{\theta}_\star - f_\star^\top(\mathbf{x})\boldsymbol{\theta}_\star\right| \in [0, 1], \tag{9}$$

and the *online error*: $\gamma_n(\mathbf{x}_{t,a}) \triangleq \left|f_{n-1}^\top(\mathbf{x}_{t,a})\boldsymbol{\theta}_{n,t} - f_{n-1}^\top(\mathbf{x}_{t,a})\boldsymbol{\theta}_\star\right|$.

Intuitively, the offline error and online error capture the effects of the estimation error of f_{n-1} and $\boldsymbol{\theta}_{n,t}$ arising from representation learning and exploration, respectively. By applying a triangle inequality, we derive an upper bound of the error in reward prediction by splitting it into online and offline errors:

$$|\hat{r}_{t,a} - f_\star^\top(\mathbf{x}_{t,a})\boldsymbol{\theta}_\star| \leq \gamma_n(\mathbf{x}_{t,a}) + \epsilon_n.$$

Our main result given below is derived by bounding $\gamma_n(\mathbf{x}_{t,a})$, and leaving ϵ_n as a factor in the total regret. The proof is given in Appendix B.

Theorem 1. *If* SupInLUCB *is run with* $\alpha = \sqrt{\frac{1}{2}\ln\frac{2NTK}{\delta}}$*, with probability at least* $1-\delta$*, the regret of the algorithm is*

$$O\left(\left(N + T\sum_{n=1}^{N}\epsilon_n\right)\sqrt{Tp\ln^3(NTK\ln(T)/\delta)}\right). \tag{10}$$

Remark 1. Theorem 1 provides a general expression of the regret bound which implies that the rate of convergence of the sequence of offline errors $\{\epsilon_n\}_{n=1}^N$ determines the order of the regret bound. Generally, we know $\sum_{n=1}^N \epsilon_n \leq N$ as $\epsilon_n \leq 1$. But substituting N for $\sum_{n=1}^N \epsilon_n$ leads to a loose bound $\widetilde{O}(NT\sqrt{Tp})$. In general, the bound of ϵ_n depends on the complexity of the underlying dimension reduction mapping $f_\star$ and the error of estimating $f_\star$ using the neural network. Thus, we cannot derive a universal non-trivial upper bound for ϵ_n as we cannot guarantee that the neural network attains global minimum. While, if we discard the error of neural networks and assume the latent feature is in a simple form (e.g., linear in raw contexts), we can derive a tighter regret bound by further bounding ϵ_n (see Corollary 1). Also, empirically, we show that ϵ_n decreases fast with the number of iteration n increases (see next section).

Remark 2. We relate our regret analysis of InLUCB to that of LinUCB [7]. As for InLUCB, if we known in davance that the reward function degenerates to a linear mapping, offline representation learning is no longer needed, which means that only online exploration remains (i.e., $N = 1$) and the offline error would be zero (i.e., $\epsilon_n = 0$). Then in this case, the regret bound in Theorem 1 reduces to $\sqrt{Td\ln^3(TK\ln(T)/\delta)}$, which is the same as that of LinUCB [7, Theorem 1]. This suggests that InLUCB recovers LinUCB as a special case.

Corollary 1. *Assume that* $f_\star$ *is linear, i.e.,* $f_\star(\mathbf{x}) = \mathbf{Q}_\star\mathbf{x}$ *where* $\mathbf{Q}_\star \in \mathbb{R}^{p\times d}$*. Let* InLUCB *use a fully connected network of three layers, each of size* d, p *and* 1*. Also assume for all* $n \in [N]$*,* $(f_n, \boldsymbol{\theta}_n)$ *minimises* $\mathcal{L}_{D_n}(f;\boldsymbol{\theta})$*. If* SupInLUCB *is run with* $\alpha = \sqrt{\frac{1}{2}\ln\frac{2NTK}{\delta}}$*, then there exist constants* $\sigma \in [0,1]$ *and* $C_\sigma \geq 0$ *such that with probability at least* $(1-\delta)(1-\sigma)$*, the regret is*

$$O\left(\left(N + T + C_\sigma\sqrt{NT}\right)\sqrt{Tp\ln^3(NTK\ln(T)/\delta)}\right).$$

Remark 3. The regret of random selection is $O(NT)$. Thus, the regret bound in Corollary 1 is non-trivial as the magnitude of $\widetilde{O}((N+T+C_\sigma\sqrt{NT})\sqrt{Tp})$ (p is constant) is smaller than that of $O(NT)$. On the other hand, the non-trivial regret bound of other neural contextual bandit methods, e.g., NeuralUCB [24], also relies on the assumption that the error of neural networks can be bounded.

6 Experiments

We empirically evaluate the accuracy (cumulative regret) and efficiency (runtime per step) of InlUCB on both high-dimensional synthetic and real-world datasets with non-linear rewards. We adopt eight bandit methods as baselines: **(1) LinUCB** [7], a linear contextual bandit method using UCB for exploration. Its regret bound is $\widetilde{O}(\sqrt{dT})$; **(2) LinTS** [2], a linear contextual bandit method using Thompson sampling (TS) for exploration. It has a regret bound of $\widetilde{O}(d\sqrt{T})$. **(3) CBRAP** [21], a method that uses random projection to do dimension reduction and UCB for exploration. **(4) KernelUCB** [16], a method that ultilises kernel functions for handling non-linear rewards and uses UCB for exploration. Its regret bound is $\widetilde{O}(\sqrt{\tilde{d}T})$, where $\tilde{d}$ is the effective dimension of kernel matrixes. **(5) NeuralUCB** [24], it uses a fully connected neural network for reward prediction, uses UCB for online exploration, and updates the whole neural network at each step. It has a regret bound of $\widetilde{O}(\tilde{d}\sqrt{T})$; **(6) Neural-Linear** [22], a method that extracts latent features using NN and use TS on the top of the last linear for exploration. The regret bound is not given by authors. **(7) EXP3** [4], a representative adversarial bandits algorithm that pulls arms with probabilities and adjusts such probabilities based on received rewards; **(8) ε-Greedy**: a classic exploration method; with high probability $1-\varepsilon$ pulling the arm with highest average reward in history and with small probability ε pulling an arm randomly.

6.1 Experimental Setting

For UCB-based methods, we tune the constant α through a grid search over $\{0.01, 0.1, 1\}$. For TS-based methods, we do grid search over $\{0.01, 0.01, 1\}$ for the hyper-parameter that controls the covariance of the prior and posterior distributions. For KernelUCB, we adopt radial basis function (RBF) kernel and empirically set the parameters with best results. For EXP3 and ε-Greedy, we do grid search for the exploration parameter over $\{0.01, 0.1, 1\}$. For InlUCB, NeuralUCB and Neural-Linear, we use the same neural network structure: a fully connected network of four layers of size d, d, p and 1, respectively. For CBRAP, the dimension after projection is also p. We vary p from 10 to 100 with step size 10, and vary T from 100 to 1000 with step size 100. For all grid-searched parameters, we choose the best setting for comparisons. For all contextual bandit methods we test their efficiency with respect to the context dimension and the number of steps. Results are averaged over 10 independent runs.

Table 1. Real-world Datasets statistics.

Dataset	Feature dimension	Number of classes	Number of samples
MUSIC	518	193	106,574
FONT	409	20	100,000
MNIST	784	10	60,000

Synthetic Datasets. We generate synthetic datasets with contextual dimension $d = 500$ and $K = 200$ arms. Contextual vectors are chosen uniformly at random from the unit ball. We use three artificial non-linear reward functions: $g(\mathbf{x}) = \cos(3\mathbf{x}^\top \mathbf{a})$ (shorten as COS), $g(\mathbf{x}) = 10(\mathbf{x}^\top \mathbf{a})^2$ (SQU), and $g(\mathbf{x}) = \exp(\mathbf{x}^\top \mathbf{a})$ (EXP), where $\mathbf{a}$ is randomly generated from uniform distribution over unit ball. These typical functions cover a wide range of non-linear mappings [24].

Real-World Datasets. We use three real-world classification datasets: MUSIC and FONT from the UCI Machine Learning Repository [8] and the MNIST dataset [12]. Table 1 lists key statistics of the datasets. Following [15], we transform classification tasks into bandit tasks: each step we randomly select one sample; the agent gets reward 1 if it classifies the sample correctly, and 0 otherwise.

6.2 Results

Figure 2 and Table 2 report results of cumulative regrets and runtime per step, respectively. Overall, InLUCB exhibits the lowest regret and superior efficiency in all cases. Specifically, only InLUCB shows convergence in synthetic datasets, which indicates the fast decrease of offline errors with the number of iterations grows. For real-world datasets, although we do not observe convergence in some cases, InLUCB achieves the lowest regret on all tasks.

Table 2. Results for runtime per step (in milliseconds).

Algorithm	COS	SQU	EXP	MUSIC	FONT	MNIST
LinUCB	0.9	0.6	0.6	0.8	0.5	0.5
LinTS	6.6	6.5	6.6	6.5	4.7	4.2
CBRAP	0.7	0.9	0.9	0.9	0.1	0.1
KernelUCB	2089.4	2114.8	1850.7	2128.1	2100.6	2228.0
NeuralUCB	1121.2	1075.2	1003.1	1027.3	1193.5	1239.6
Neural-Linear	8.7	6.1	6.0	7.1	1.8	2.3
EXP-3	0.3	0.6	0.6	0.5	0.08	0.08
ε-Greedy	0.01	0.01	0.01	0.009	0.04	0.004
InLUCB (ours)	4.3	5.6	5.7	9.5	3.8	5.0

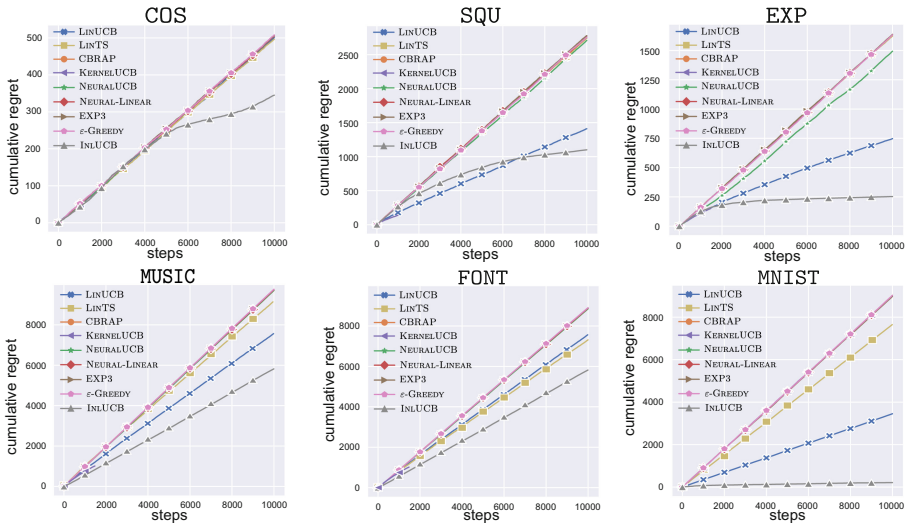

Fig. 2. Results for cumulative regrets.

LinUCB and LinTS show low regrets in some cases but fail to converge, as they are able to take use of complete contextual information but cannot handle non-linear reward functions.

KernelUCB costs more than 10s after 1500 steps since it needs to invert a matrix whose dimension is proportional to the number of steps, which gives the evidence that it is inefficient for practical use. Although Neural-Linear is efficient, it suffers from high regrets since the end-to-end training framework prevents the online exploration from effectively boosting representation learning. CBRAP is relatively efficient but has high regret since empirically we find that it is really sensitive to the initial value of the projection matrix. The classical probability-based exploration techniques EXP3 and ε-Greedy have the highest regret although they are most efficient. The reason is that they lack the capability of modeling environments with contextual information.

Results of sensitivity test (see Appendix C.1) show that NeuralUCB is not applicable to environments with high dimensional contexts. Thus, the result of NeuralUCB in Fig. 2 is reported using a selected subset of features that result in the best performance. Even though NeuralUCB runs on a subset of original contexts, it has extremely high runtime cost. In contrast, InLUCB is three orders of magnitude faster than NeuralUCB for online decision making. We also show that InLUCB has comparable cumulative regrets to NeuralUCB on the same subset of features in Appendix C.2. Thus, we conclude that InLUCB achieves a better balance between the accuracy and online efficiency.

7 Conclusion

We propose InLUCB, the first contextual bandit method that can simultaneously handle high dimensional contexts and non-linear rewards with high online efficiency. InLUCB uses neural networks to model reward functions and creatively adopts an interleaving online/offline update mechanism to combine efficient online exploration and representation learning. We give a general expression of regret bound for InLUCB and present a tighter regret bound under certain conditions. Results of experiments on synthetic and real-world datasets confirm the high accuracy and efficiency of InLUCB.

References

1. Agrawal, S., Goyal, N.: Analysis of thompson sampling for the multi-armed bandit problem. J. Mach. Learn. Res. **23**(4), 357–364 (2011)
2. Agrawal, S., Goyal, N.: Thompson sampling for contextual bandits with linear payoffs. In: ICML, pp. 127–135 (2013)
3. Auer, P.: Using confidence bounds for exploitation-exploration trade-offs. J. Mach. Learn. Res. **3**(Nov), 397–422 (2002)
4. Auer, P., Cesa-Bianchi, N., Fischer, P.: Finite-time analysis of the multiarmed bandit problem. Mach. Learn. **47**(2–3), 235–256 (2002)
5. Barron, A.R.: Universal approximation bounds for superpositions of a sigmoidal function. IEEE Trans. Inf. Theory **39**(3), 930–945 (1993)
6. Bastani, H., Bayati, M.: Online decision making with high-dimensional covariates. Oper. Res. **68**(1), 276–294 (2020)
7. Chu, W., Li, L., Reyzin, L., Schapire, R.: Contextual bandits with linear payoff functions. In: AISTATS, pp. 208–214 (2011)
8. Dua, D., Graff, C.: UCI machine learning repository (2017). http://archive.ics.uci.edu/ml
9. Ghosh, A., Chowdhury, S.R., Gopalan, A.: Misspecified linear bandits. In: AAAI, pp. 3761–3767 (2017)
10. Jacot, A., Gabriel, F., Hongler, C.: Neural tangent kernel: convergence and generalization in neural networks. In: NIPS, pp. 8580–8589 (2018)
11. Kim, G.S., Paik, M.C.: Doubly-robust lasso bandit. In: NIPS, pp. 5869–5879 (2019)
12. LeCun, Y., Bottou, L., Bengio, Y., Haffner, P.: Gradient-based learning applied to document recognition. Proc. IEEE **86**(11), 2278–2324 (1998)
13. Li, L., Chu, W., Langford, J., Schapire, R.E.: A contextual-bandit approach to personalized news article recommendation. In: WWW, pp. 661–670 (2010)
14. Li, L., Lu, Y., Zhou, D.: Provably optimal algorithms for generalized linear contextual bandits. In: ICML, pp. 2071–2080 (2017)
15. Riquelme, C., Tucker, G., Snoek, J.: Deep Bayesian bandits showdown. In: ICLR (2018)
16. Valko, M., Korda, N., Munos, R., Flaounas, I., Cristianini, N.: Finite-time analysis of kernelised contextual bandits. In: UAI, pp. 654–663 (2013)
17. Vaswani, S., Kveton, B., Wen, Z., Ghavamzadeh, M., Lakshmanan, L.V., Schmidt, M.: Model-independent online learning for influence maximization. In: ICML, pp. 3530–3539 (2017)
18. Wang, X., Wei, M., Yao, T.: Minimax concave penalized multi-armed bandit model with high-dimensional covariates. In: ICML, pp. 5200–5208 (2018)

19. Weng, J., Hwang, W.S.: Online image classification using IHDR. Int. J. Doc. Anal. Recogn. **5**(2–3), 118–125 (2003)
20. Xie, M., Yin, W., Xu, H.: AutoBandit: a meta bandit online learning system. In: Proceedings of the Thirtieth International Joint Conference on Artificial Intelligence, IJCAI-21, pp. 5028–5031 (2021)
21. Yu, X., Lyu, M.R., King, I.: CBRAP: contextual bandits with random projection. In: AAAI (2017)
22. Zahavy, T., Mannor, S.: Deep neural linear bandits: overcoming catastrophic forgetting through likelihood matching. arXiv preprint arXiv:1901.08612 (2019)
23. Zeng, Z., Li, X., Ma, X., Ji, Q.: Adaptive context recognition based on audio signal. In: 2008 19th International Conference on Pattern Recognition, pp. 1–4. IEEE (2008)
24. Zhou, D., Li, L., Gu, Q.: Neural contextual bandits with UCB-based exploration. In: ICML, pp. 11492–11502 (2020)