
Federated Neural Bandit

Zhongxiang Dai¹, Yao Shu¹, Arun Verma¹, Flint Xiaofeng Fan¹,
 Bryan Kian Hsiang Low¹, Patrick Jaillet²

Dept. of Computer Science, National University of Singapore, Republic of Singapore¹
 Dept. of Electrical Engineering and Computer Science, MIT, USA²

Abstract

Recent works on *neural contextual bandit* have achieved compelling performances thanks to their ability to leverage the strong representation power of neural networks (NNs) for reward prediction. Many applications of contextual bandit involve multiple agents who collaborate without sharing raw observations, giving rise to the setting of *federated contextual bandit*. Existing works on federated contextual bandit rely on linear or kernelized bandit, which may fall short when modeling complicated real-world reward functions. In this regard, we introduce the *federated neural-upper confidence bound* (FN-UCB) algorithm. To better exploit the federated setting, we adopt a weighted combination of two UCBs: UCB^a allows every agent to additionally use the observations from the other agents to accelerate exploration (without sharing raw observations); UCB^b uses an NN with aggregated parameters for reward prediction in a similar way as federated averaging for supervised learning. Notably, the weight between the two UCBs required by our theoretical analysis is amenable to an interesting interpretation, which emphasizes UCB^a initially for *accelerated exploration* and relies more on UCB^b later after enough observations have been collected to train the NNs for accurate reward prediction (i.e., *reliable exploitation*). We prove sub-linear upper bounds on both the cumulative regret and the number of communication rounds of FN-UCB, and use empirical experiments to demonstrate its competitive performances.

1 Introduction

The stochastic multi-armed bandit is a prominent method for sequential decision-making problems due to its principled ability to handle the exploration-exploitation trade-off [4, 7, 25]. In particular, the stochastic contextual bandit problem has received enormous attention thanks to its widespread real-world applications such as recommender systems [27], advertising [32] and healthcare [17]. In every iteration of a stochastic contextual bandit problem, an agent receives a context (i.e., a d -dimensional feature vector) for each one of the K arms, selects one of the K contexts/arms, and observes the corresponding reward. The goal of the agent is to sequentially pull the arms in order to maximize the cumulative reward (or equivalently, minimize the *cumulative regret*) in T iterations.

To minimize the cumulative regret, *linear contextual bandit* algorithms assume that the rewards can be modelled as a linear function of the input contexts [13, 37] and select the arms via classic methods such as upper confidence bound (UCB) [4] or Thompson sampling (TS) [41], hence introducing the Linear UCB [1] and Linear TS [2] algorithms. The potentially restrictive assumption of a linear model was later relaxed by *kernelized contextual bandit* algorithms [9, 43], which assume that the reward function belongs to a reproducing kernel Hilbert space (RKHS) and henceforth model the reward function using kernel ridge regression or Gaussian process (GP) regression. However, this assumption may still be restrictive [48] and the kernelized model may fall short when the reward function is very complicated and difficult to model. To this end, neural networks (NNs), which excel at modelling complicated real-world functions, have been adopted to model the reward function in

contextual bandit, leading to *neural contextual bandit* algorithms [48] such as Neural UCB [48] and Neural TS [47]. Thanks to their ability to use the highly expressive NNs for better reward prediction (i.e., *exploitation*), Neural UCB and Neural TS have been shown to outperform both linear and kernelized contextual bandit methods in practice [47, 48]. Moreover, the cumulative regrets of Neural UCB and Neural TS have been analyzed by leveraging the theory of the *neural tangent kernel* (NTK) [3, 8, 20], hence making these algorithms both provably efficient and practically effective.

The contextual bandit algorithms discussed above are only applicable to problems with a single agent. However, many modern applications of contextual bandit involve multiple agents who (a) collaborate with each other for better performances yet (b) are unwilling to share their raw observations (i.e., the contexts and rewards). For example, companies may collaborate to improve their contextual bandit-based recommendation algorithms, without sharing their sensitive user data [19]; hospitals deploying contextual bandit methods for personalized treatment may collaborate to improve their treatment strategies, without sharing their sensitive patient information [11, 12]. These applications naturally fall under the setting of *federated learning* (FL) [23, 28, 29, 34], which facilitates collaborative learning of supervised learning models (e.g., NNs) without sharing the raw data. In this regard, a number of *federated contextual bandit* algorithms have been developed, which allow bandit agents to collaborate in the federated setting [11, 16, 19, 44]. Specifically, [44] adopted the Linear UCB policy, and developed a mechanism to allow every agent to additionally use the observations from the other agents to **accelerate exploration**, while only requiring the agents to exchange some sufficient statistics instead of their raw observations. However, these previous works have only relied on either linear [16, 19, 44] or kernelized [11, 12] methods, which, as we have discussed above, may lack the expressive power to model complicated real-world reward functions [48]. Therefore, this naturally brings up the need to **use NNs for better exploitation** (i.e., reward prediction) in federated contextual bandit, i.e., the need for a *federated neural contextual bandit* algorithm.

To develop a federated neural contextual bandit algorithm, an important technical challenge is how to leverage the federated setting to simultaneously: (a) **accelerate exploration** by allowing every agent to additionally use the observations from the other agents without requiring the exchange of raw observations (in a similar way as [44]), and (b) **improve exploitation** by further enhancing the quality of the NN for reward prediction through the federated setting, without requiring centralized training using the observations from all agents. In this work, we provide a theoretically grounded solution to this challenge by deploying a weighted combination of two upper confidence bounds (UCBs). The first UCB (denoted as UCB^a) incorporates the neural tangent features (i.e., the random features embedding of the NTK) into the Linear UCB-based mechanism adopted by [44], which achieves the first goal of accelerating *exploration*. The second UCB (denoted as UCB^b) adopts an aggregated NN, whose parameters are the average of the parameters of the NNs trained by all agents using their local observations, for better reward prediction (i.e., better *exploitation* in the second goal). Hence, UCB^b improves the quality of the NN for reward prediction in a similar way as the most classic FL method of federated averaging (FedAvg) for supervised learning [34]. Notably, our choice of the weight between the two UCBs, which naturally arises during our theoretical analysis, has an interesting practical interpretation (Sec. 4): more weights are given to UCB^a in earlier iterations, which allows us to use the observations from the other agents to accelerate the exploration in the early stage; more weights are assigned to UCB^b only in later iterations, after every agent has collected enough local observations to train its NN for accurate reward prediction (i.e., reliable exploitation).

In this work, we introduce the first federated neural contextual bandit algorithm, named *federated neural-UCB* (FN-UCB) (Sec. 4). We derive an upper bound on its total cumulative regret from all N agents: $R_T = \tilde{O}(\tilde{d}\sqrt{TN} + \tilde{d}_{\max}N\sqrt{TN})$,¹ in which \tilde{d} is the effective dimension of the contexts from all agents and \tilde{d}_{\max} represents the maximum among the N individual effective dimensions of the contexts from the N agents (Sec. 3). The communication complexity (i.e., total number of communication rounds in T iterations) of FN-UCB can be upper-bounded by $C_T = \tilde{O}(\tilde{d}\sqrt{N})$. Lastly, we use both synthetic and real-world contextual bandit experiments to explore the interesting insights about our FN-UCB and demonstrate its effective practical performances (Sec. 6).

¹The \tilde{O} ignores all logarithmic factors.

2 Related Works

Federated learning (FL) has received enormous attention in recent years [23, 28, 29, 34]. A number of recent works have extended the classic K -armed bandit (i.e., the arms are not associated with feature vectors) to the federated setting. [30] and [31] focused on incorporating privacy guarantees into federated K -armed bandit in both centralized and decentralized settings. [38] proposed a setting where the goal is to minimize the regret of a global bandit whose reward of an arm is the average of the rewards of the corresponding arm from all agents, which was later extended by adding personalization such that every agent aims to maximize a weighted combination between the global and local rewards [39]. Subsequent works on federated K -armed bandit have focused on other important aspects such as decentralized communication via the gossip algorithm [49], the security aspect via cryptographic techniques [10], uncoordinated exploration [46], and robustness against Byzantine attacks [14]. Regarding federated linear contextual bandit, [44] proposed a distributed linear contextual bandit algorithm which allows every agent to use the observations from the other agents by only exchanging the sufficient statistics to calculate the Linear UCB policy. Subsequently, [16] extended the method from [44] to consider differential privacy and decentralized communication, [19] considered a setting where every agent is associated with a unique context vector, [26] focused on asynchronous communication, and [21] considered the robustness against Byzantine attacks. Federated kernelized/GP bandit (also named federated Bayesian optimization) has been explored by [11, 12], which focused on the practical problem of hyperparameter tuning in the federated setting.

Since the pioneering works of [48] and [47] which, respectively, introduced Neural UCB and Neural TS, a number of recent works have focused on different aspects of neural contextual bandit. [45] reduced the computational cost of Neural UCB by using an NN as a feature extractor and applying Linear UCB only to the last layer of the learned NN, [24] analyzed the maximum information gain of the NTK and hence derived no-regret algorithms, [18] focused on the batch setting in which the policy is only updated at a small number of time steps, [35] aimed to reduce the memory requirement of Neural UCB, [33] performed an empirical investigation of neural bandit algorithms to verify their practical effectiveness, [6] adopted a separate NN for exploration in neural contextual bandit, [5] applied the convolutional NTK, [22] used perturbed rewards to train the NN to remove the need for explicit exploration, and [36] incorporated offline policy learning into neural contextual bandit.

3 Background and Problem Setting

We use $[k]$ to denote the set $\{1, 2, \dots, k\}$ for a positive integer k , use $\mathbf{0}_k$ to represent a k -dimensional vector of 0's and $\mathbf{0}_{k \times k}$ to denote an all-zero matrix with dimension $k \times k$. Our setting involves N agents with the same reward function h defined on a domain $\mathcal{X} \subset \mathbb{R}^d$. We consider centralized and synchronous communication: the communication is coordinated by a central server, and every agent exchanges information with the central server during a *communication round*. In every iteration $t \in [T]$, every agent $i \in [N]$ receives K context vectors $\mathcal{X}_{t,i} = \{x_{t,i}^k\}_{k \in [K]}$ and chooses one of them $x_{t,i} \in \mathcal{X}_{t,i}$. After that, a noisy observation is produced $y_{t,i} = h(x_{t,i}) + \epsilon$ where ϵ is an R -sub-Gaussian noise. We will analyze the total *cumulative regret* from all N agents in T iterations: $R_T \triangleq \sum_{i=1}^N \sum_{t=1}^T r_{t,i}$, in which $r_{t,i} \triangleq h(x_{t,i}^*) - h(x_{t,i})$ and $x_{t,i}^* \triangleq \arg \max_{x \in \mathcal{X}_{t,i}} h(x)$.

We use $f(x; \theta)$ to denote the output of a fully connected NN for input x with parameters θ (of dimension p_0), and use $g(x; \theta)$ to denote the corresponding gradient. We focus on NNs with ReLU activations, and use $L \geq 2$ and m to denote its depth and width (of every layer), respectively. We follow the initialization technique from [47, 48] to initialize the NN parameters $\theta_0 \sim \text{init}(\cdot)$. Of note, as a common ground for collaboration, we let all N agents share the same initial parameters θ_0 when training their NNs and calculating their *neural tangent features*: $g(x; \theta_0)/\sqrt{m}$ (i.e., the random features mapping of the NTK [47]). Furthermore, we use \mathbf{H} to denote the $(TKN) \times (TKN)$ -dimensional NTK matrix on the set of all TKN contexts [47, 48]. Similarly, we also define \mathbf{H}_i as the $(TK) \times (TK)$ -dimensional NTK matrix for agent i , which only makes use of the TK contexts observed by agent i . We defer the details on the definitions of \mathbf{H} and \mathbf{H}_i 's, the NN $f(x; \theta)$, and the initialization scheme $\theta_0 \sim \text{init}(\cdot)$ to Appendix A due to the limited space.

Next, using \mathbf{h} to denote the (TKN) -dimensional vector of $\mathbf{h} = [h(x_{t,i}^k)]_{t \in [T], i \in [N], k \in [K]}$, we define B as an absolute constant such that $\sqrt{2\mathbf{h}^\top \mathbf{H}^{-1} \mathbf{h}} \leq B$. This is related to the commonly adopted assumption in kernelized bandit that h lies in the RKHS \mathcal{H} induced by the NTK [9, 40] (or equivalently, that the RKHS norm of h , $\|h\|_{\mathcal{H}}$, is upper-bounded by a constant), because $\sqrt{\mathbf{h}^\top \mathbf{H}^{-1} \mathbf{h}} \leq \|h\|_{\mathcal{H}}$

Algorithm 1 FN-UCB (Agent i)

- 1: **inputs:** $\lambda = 1 + 2/T$, $D = \mathcal{O}(\frac{T}{Nd})$, $\theta_0 \sim \text{init}(\cdot)$, $W_{\text{sync}} = \mathbf{0}_{p_0 \times p_0}$, $W_{\text{new},i} = \mathbf{0}_{p_0 \times p_0}$, $B_{\text{sync}} = \mathbf{0}$, $B_{\text{new},i} = \mathbf{0}_{p_0}$, $\alpha_t = 0$, $V_{t,i}^{\text{local}} = \lambda I$, $V_{\text{sync,NN}}^{-1} = (1/\lambda)I$, $V_{\text{last}} = \lambda I$, $t_{\text{last}} = 0$, $\theta_{\text{sync,NN}} = \theta_0$.
 - 2: **for** $t = 1, 2, \dots, T$ **do**
 - 3: $\bar{V}_{t,i} = \lambda I + W_{\text{sync}} + W_{\text{new},i}$, $\bar{\theta}_{t,i} = \bar{V}_{t,i}^{-1}(B_{\text{sync}} + B_{\text{new},i})$
 - 4: Calculate $\text{UCB}_{t,i}^a(x) = \langle g(x; \theta_0) / \sqrt{m}, \bar{\theta}_{t,i} \rangle + \nu_{TKN} \sqrt{\lambda} \|g(x; \theta_0) / \sqrt{m}\|_{\bar{V}_{t,i}^{-1}}$
 - 5: If $\alpha_t \neq 0$, calculate $\text{UCB}_{t,i}^b(x) = f(x; \theta_{\text{sync,NN}}) + \nu_{TK} \sqrt{\lambda} \|g(x; \theta_0) / \sqrt{m}\|_{V_{\text{sync,NN}}^{-1}}$
 - 6: Choose $x_{t,i} = \arg \max_{x \in \mathcal{X}_{t,i}} (1 - \alpha_t) \text{UCB}_{t,i}^a(x) + \alpha_t \text{UCB}_{t,i}^b(x)$
 - 7: Query $x_{t,i}$ to observe $y_{t,i}$
 - 8: Update $W_{\text{new},i} = W_{\text{new},i} + g(x_{t,i}; \theta_0) g(x_{t,i}; \theta_0)^\top / m$, $B_{\text{new},i} = B_{\text{new},i} + y_{t,i} g(x_{t,i}; \theta_0) / \sqrt{m}$
 - 9: Update $V_{t,i}^{\text{local}} = V_{t,i}^{\text{local}} + g(x_{t,i}; \theta_0) g(x_{t,i}; \theta_0)^\top / m$
 - 10: **if** $(t - t_{\text{last}}) \log \frac{\det(\lambda I + W_{\text{sync}} + W_{\text{new},i})}{\det V_{\text{last}}} > D$ **then**
 - 11: Send a synchronisation signal to the central server to start a communication round
 - 12: **if** a communication round is started **then**
 - 13: Train an NN with gradient descent using all agent i 's local observations: $\mathcal{D}_{t,i} = \{(x_{\tau,i}, y_{\tau,i})\}_{\tau \in [t]}$, with θ_0 as the initial parameters, η as the learning rate, J as the number of iterations and equation (1) as the loss function, to obtain θ_t^i
 - 14: Calculate $\alpha_{t,i} = \bar{\sigma}_{t,i,\min}^{\text{local}} / \bar{\sigma}_{t,i,\max}^{\text{local}}$
 - 15: **send** $\{W_{\text{new},i}, B_{\text{new},i}, \theta_t^i, (V_{t,i}^{\text{local}})^{-1}, \alpha_{t,i}\}$ to the central server
 - 16: **receive** $\{W_{\text{sync}}, B_{\text{sync}}, \theta_{\text{sync,NN}}, V_{\text{sync,NN}}^{-1}, \alpha_t\}$ from the central server
 - 17: Set $V_{\text{last}} = W_{\text{sync}} + \lambda I$, $t_{\text{last}} = t$, $W_{\text{new},i} = \mathbf{0}_{p_0 \times p_0}$, and $B_{\text{new},i} = \mathbf{0}$
-

Algorithm 2 Central Server

- 1: **if** a synchronization signal is received from *any agent* **then**
 - 2: Send a signal to all agents to start a communication round
 - 3: **receive** $\{W_{\text{new},i}, B_{\text{new},i}, \theta_t^i, (V_{t,i}^{\text{local}})^{-1}, \alpha_{t,i}\}_{i \in [N]}$
 - 4: Calculate $\theta_{\text{sync,NN}} = \frac{1}{N} \sum_{i=1}^N \theta_t^i$, $V_{\text{sync,NN}}^{-1} = \frac{1}{N} \sum_{i=1}^N (V_{t,i}^{\text{local}})^{-1}$, and $\alpha_t = \min_{i \in [N]} \alpha_{t,i}$
 - 5: Calculate $W_{\text{sync}} = W_{\text{sync}} + \sum_{i=1}^N W_{\text{new},i}$, and $B_{\text{sync}} = B_{\text{sync}} + \sum_{i=1}^N B_{\text{new},i}$
 - 6: Broadcast $\{W_{\text{sync}}, B_{\text{sync}}, \theta_{\text{sync,NN}}, V_{\text{sync,NN}}^{-1}, \alpha_t\}$ to all agents
-

[48]. Following the previous works of [47, 48], we define the effective dimension of \mathbf{H} as $\tilde{d} \triangleq \frac{\log \det(I + \mathbf{H}/\lambda)}{\log(1 + TKN/\lambda)}$ where $\lambda > 0$ is a regularization parameter. Similarly, we define the effective dimension for agent i as $\tilde{d}_i \triangleq \frac{\log \det(I + \mathbf{H}_i/\lambda)}{\log(1 + TK/\lambda)}$ and also define $\tilde{d}_{\max} \triangleq \max_{i \in [N]} \tilde{d}_i$. Of note, the effective dimension is related to the *maximum information gain* γ which is a commonly adopted notion in kernelized bandit [47]: $\tilde{d} \leq 2\gamma_{TKN}/\log(1 + TKN/\lambda)$, $\tilde{d}_i \leq 2\gamma_{TK}/\log(1 + TK/\lambda)$, $\forall i \in [N]$. Consistent with previous works on neural contextual bandit [47, 48], our only assumption on the reward function h is its boundedness: $|h(x)| \leq 1, \forall x \in \mathcal{X}$. We also make the following assumptions for our theoretical analysis, all of which are mild and easily achievable as discussed in [47, 48].

Assumption 1. We assume that there exists $\lambda_0 > 0$ such that $\mathbf{H} \succeq \lambda_0 I$ and $\mathbf{H}_i \succeq \lambda_0 I, \forall i \in [N]$. We also assume that all contexts satisfy $\|x\|_2 = 1$ and $[x]_j = [x]_{j+d/2}, \forall x \in \mathcal{X}_{t,i}, \forall t \in [T], i \in [N]$.

4 Federated Neural-Upper Confidence Bound

Our FN-UCB algorithm is described in Algo. 1 (agents' part) and Algo. 2 (central server's part).

Overview of FN-UCB. Before the beginning of the algorithm, we sample the initial parameters θ_0 and share it with all agents (Sec. 3). In every iteration $t \in [T]$, each agent $i \in [N]$ receives a set of K contexts denoted as $\mathcal{X}_{t,i} = \{x_{t,i}^k\}_{k \in [K]}$ and then uses the weighted combination of $\text{UCB}_{t,i}^a$ and $\text{UCB}_{t,i}^b$ to choose an arm $x_{t,i} \in \mathcal{X}_{t,i}$ to pull (lines 3-6 of Algo. 1). Next, each agent i collects a noisy observation $y_{t,i}$ (line 7 of Algo. 1), and then updates its local information (lines 8-9 of Algo. 1). After that, every agent checks if it has collected enough information since the last communication

round (i.e., checks the criterion in line 10 of Algo. 1); if yes, it sends a synchronization signal to the central server (line 11 of Algo. 1), who then tells all agents to start a communication round (line 2 of Algo. 2). During a communication round, every agent i uses its current history of local observations to train an NN (line 13 of Algo. 1) and sends its updated local information to the central server (line 15 of Algo. 1); the central server then aggregates these information from all agents (lines 4-5 of Algo. 2) and broadcasts the aggregated information back to all agents (line 6 of Algo. 2) to start the next iteration. We refer to those iterations between two communication rounds as an *epoch*,² i.e., our FN-UCB algorithm consists of a number of epochs which are separated by communication rounds.

Of note, every agent i only needs to train an NN in every communication round, i.e., only after the change in the (log) determinant of the covariance matrix of any agent exceeds a threshold D (line 10 of Algo. 1). This has the additional benefit of reducing the computational cost resulting from the training of NNs. Interestingly, this is in a similar spirit as the adaptive batch size scheme from [18], which only re-trains the NN in Neural UCB after the change in the determinant of the covariance matrix exceeds a threshold and is shown to only slightly degrade the performance of Neural UCB.

The Two UCBs. We now introduce the details of $\text{UCB}_{t,i}^a$ and $\text{UCB}_{t,i}^b$ in the next two paragraphs.

Firstly, $\text{UCB}_{t,i}^a$ can be interpreted as the Linear UCB policy [1] using the neural tangent features $g(x; \theta_0)/\sqrt{m}$ as the input features. In iteration t , denote the index of the current epoch as p , then the calculation of $\text{UCB}_{t,i}^a$ (line 4 of Algo. 1) makes use of two types of information. The first type of information, which *uses the observations from all N agents before epoch p* , is used in the calculation of $\text{UCB}_{t,i}^a$ via W_{sync} and B_{sync} (line 3 of Algo. 1). Specifically, as can be seen from line 5 of Algo. 2, W_{sync} and B_{sync} are calculated by the central server by summing up the $W_{\text{new},i}$'s and $B_{\text{new},i}$'s from all agents (i.e., by aggregating the information from all agents), in which $W_{\text{new},i}$ and $B_{\text{new},i}$ are calculated using the local observations of agent i (line 8 of Algo. 1). The second type of information used by $\text{UCB}_{t,i}^a$ (via $W_{\text{new},i}$ and $B_{\text{new},i}$, see line 3 of Algo. 1) makes use of the newly collected local observations of agent i in epoch p . As a result, $\text{UCB}_{t,i}^a$ allows us to utilize the observations from all agents via the federated setting for accelerated exploration, without requiring the agents to share their raw observations. The parameter ν_{TKN} used in the calculation of $\text{UCB}_{t,i}^a$ is defined as $\nu_{TKN} \triangleq B + R[2(\log(3/\delta) + 1) + \tilde{d}\log(1 + TKN/\lambda)]^{1/2}$ where $\delta \in (0, 1)$.

Secondly, $\text{UCB}_{t,i}^b$ leverages the federated setting to improve the quality of NN for reward prediction (i.e., to achieve better exploitation) in a similar way as FedAvg, i.e., by averaging the parameters of the NNs trained by all agents using their local observations [34]. Specifically, when a communication round is started, every agent $i \in [N]$ uses its local observations $\mathcal{D}_{t,i} = \{(x_{\tau,i}, y_{\tau,i})\}_{\tau \in [t]}$ to train an NN (line 13 of Algo. 1). It uses θ_0 as the initial parameters (i.e., shared among all agents, Sec. 3), and trains the NN using gradient descent with a learning rate of η for J training iterations (see Theorem 1 for the choices of η and J in our theoretical analysis) to minimize the following loss function:

$$\mathcal{L}_{t,i}(\theta) = \sum_{\tau=1}^t (f(x_{\tau,i}; \theta) - y_{\tau,i})^2 / 2 + m\lambda \|\theta - \theta_0\|_2^2 / 2. \quad (1)$$

The resulting NN parameters θ_t^i 's from all N agent are sent to the central server (line 16 of Algo. 1), who averages them (line 4 of Algo. 2) and broadcasts the aggregated $\theta_{\text{sync,NN}} = \frac{1}{N} \sum_{i=1}^N \theta_t^i$ back to all agents to be used in the next epoch. In addition, to calculate the second term of $\text{UCB}_{t,i}^b$, every agent needs to calculate the matrix $V_{t,i}^{\text{local}}$ using its local inputs (line 9 of Algo. 1) and send its inverse to the central server (line 15 of Algo. 1) during a communication round; after that, the central server averages the received matrices to obtain $V_{\text{sync,NN}}^{-1} = \frac{1}{N} \sum_{i=1}^N (V_{t,i}^{\text{local}})^{-1}$ and broadcasts it back to all agents to be used in the second term of $\text{UCB}_{t,i}^b$ (line 5 of Algo. 1). Refer to Sec. 5.2 for a more detailed explanation on the validity of $\text{UCB}_{t,i}^b$ as a high-probability upper bound on h (up to additive error terms). The parameter ν_{TK} used in $\text{UCB}_{t,i}^b$ is $\nu_{TK} \triangleq B + R[2(\log(3N/\delta) + 1) + \tilde{d}_{\max} \log(1 + TK/\lambda)]^{1/2}$.

For both UCBs, unlike Neural UCB [48] and Neural TS [47] which use θ_t (the parameters of trained NNs) to calculate the exploration term (the second terms of $\text{UCB}_{t,i}^a$ and $\text{UCB}_{t,i}^b$), we instead use θ_0 . This is consistent with [24] who have shown that the use of θ_0 gives accurate uncertainty estimation.

The Weight between the Two UCBs. Of note, our choice of the weight α_t between the two UCBs, which naturally arises during our theoretical analysis (Sec. 5), has an interesting interpretation in

²The first/last epoch is between a communication round and the beginning/end of the algorithm.

terms of the relative strengths of the two UCBs and the exploration-exploitation trade-off. Specifically, define $\tilde{\sigma}_{t,i}^{\text{local}}(x) \triangleq \sqrt{\lambda} \|g(x; \theta_0) / \sqrt{m}\|_{(V_{t,i}^{\text{local}})^{-1}}$, which intuitively represents our *uncertainty* about the reward at x after conditioning on the local observations of agent i up to iteration t [24].³ Next, we also define $\tilde{\sigma}_{t,i,\min}^{\text{local}} \triangleq \min_{x \in \mathcal{X}} \tilde{\sigma}_{t,i}^{\text{local}}(x)$ and $\tilde{\sigma}_{t,i,\max}^{\text{local}} \triangleq \max_{x \in \mathcal{X}} \tilde{\sigma}_{t,i}^{\text{local}}(x)$, which, respectively, represent our smallest and largest uncertainty across the entire domain. Then, α_t is chosen as: $\alpha_t = \min_{i \in [N]} \alpha_{t,i}$ (line 4 of Algo. 2) where $\alpha_{t,i} = \tilde{\sigma}_{t,i,\min}^{\text{local}} / \tilde{\sigma}_{t,i,\max}^{\text{local}}$ (line 14 of Algo. 1). In other words, $\alpha_{t,i}$ is chosen as *the ratio between the smallest and largest uncertainty across the entire domain* for agent i , and α_t is selected as the smallest such ratio $\alpha_{t,i}$ among all agents. Therefore, α_t is expected to be generally *increasing in t* : $\tilde{\sigma}_{t,i,\min}^{\text{local}}$ is already small after the first few iterations since our uncertainty at the observed input locations (contexts) is very small; on the other hand, $\tilde{\sigma}_{t,i,\max}^{\text{local}}$ is expected to be very large in early iterations and become smaller in later iterations only after we have observed a large number of inputs (contexts) to sufficiently reduce the overall uncertainty in the entire domain. As a result, this implies that we give *more weights to $UCB_{t,i}^a$ in earlier iterations* and assign *more weights to $UCB_{t,i}^b$ in later iterations*. This, interestingly, turns out to have an intriguing practical interpretation: relying more on $UCB_{t,i}^a$ in early iterations is reasonable because $UCB_{t,i}^a$ is able to utilize the observations from all agents to accelerate *exploration* in the early stage (see discussions of $UCB_{t,i}^a$ above); it is also sensible to give more emphasis on $UCB_{t,i}^b$ only in later iterations, because the NN trained by every agent is only able to accurately model the reward function (for reliable *exploitation*) after it has collected enough observations to train its NN.

Communication Cost. To summarize, during a communication round, the parameters an agent sends to the central server include $\{W_{\text{new},i}, B_{\text{new},i}, \theta_t^i, (V_{t,i}^{\text{local}})^{-1}, \alpha_{t,i}\}$ (line 15 of Algo. 1), and the parameters the central server broadcasts to the agents include $\{W_{\text{sync}}, B_{\text{sync}}, \theta_{\text{sync,NN}}, V_{\text{sync,NN}}^{-1}, \alpha_t\}$ (line 6 of Algo. 2). So, the total number of parameters in both cases are $p_0^2 + p_0 + p_0 + p_0^2 + 1 = \mathcal{O}(p_0^2)$ where p_0 is the total number of parameters of the NN, which is acceptable only if the NN is small (i.e., p_0 is small). When the NN is overly large such that the communication cost of $\tilde{\mathcal{O}}(p_0^2)$ becomes excessive, we can follow the practice of previous works [47, 48] to diagonalize the $p_0 \times p_0$ matrices (i.e., only keep the diagonal elements of the matrices), which was adopted to reduce the computational cost. Specifically, we can diagonalize $W_{\text{new},i}$ (line 8 of Algo. 1) and $V_{t,i}^{\text{local}}$ (line 9 of Algo. 1), and let the central server aggregate only the diagonal elements of the corresponding matrices to obtain W_{sync} and $V_{\text{sync,NN}}^{-1}$. As a result, the number of exchanged parameters becomes $p_0 \times 4 + 1 = \mathcal{O}(p_0)$, which is *comparable to the number of exchanged parameters in standard FL for supervised learning* (e.g., FedAvg) where the parameters (or gradients) of the NN are communicated [34]. We will also analyze the total number of required communication rounds by our FN-UCB algorithm in Sec. 5.1.

5 Theoretical Analysis

5.1 Theoretical Results

Regret Upper Bound. For simplicity, we analyze the regret of a simpler variant our algorithm, in which we only choose the weight α_t using the method described in Sec. 4 in the first iteration after every communication round, and set $\alpha_t = 0$ in all other iterations. Note that when communication occurs after every iteration (i.e., when D is sufficiently small), this variant coincides with our original FN-UCB algorithm described in Algos. 1 and 2 (Sec. 4). The regret upper bound is given by the following theorem, whose proof is presented in Appendix B.

Theorem 1. *Let $\delta \in (0, 1)$, $\lambda = 1 + 2/T$, $D = \mathcal{O}(T/(N\tilde{d}))$. Suppose the width m of the NN grows polynomially: $m \geq \text{poly}(\lambda, T, K, N, L, \log(1/\delta), 1/\lambda_0)$. For the gradient descent training (line 13 of Algo. 1), let $\eta = C_4(m\lambda + mTL)^{-1}$ for some constant $C_4 > 0$ and $J = \tilde{\mathcal{O}}(TL/(\lambda C_4))$. Then with probability of at least $1 - \delta$, $R_T = \tilde{\mathcal{O}}(\tilde{d}\sqrt{TN} + \tilde{d}_{\max}N\sqrt{TN})$.*

Refer to Appendix B.1 for the detailed conditions on the width m of the NN, as well as the learning rate η and the number of iterations J for the gradient descent training (line 13 of Algo. 1). The first term of $\tilde{d}\sqrt{TN}$ in the regret upper bound arises due to the use of $UCB_{t,i}^a$ and reflects the benefit of the federated setting. In particular, this term matches the regret upper bound of standard Neural UCB

³Formally, $\tilde{\sigma}_{t,i}^{\text{local}}(x)$ is the Gaussian process posterior standard deviation at x conditioned on all local observations of agent i till iteration t , calculated using the kernel of $\tilde{k}(x, x') = g(x; \theta_0)^\top g(x'; \theta_0)/m$.

[48] running for TN iterations, and hence the average regret across all agents, $\tilde{d}\sqrt{T/N}$, decreases with a larger number N of agents. The second term of $\tilde{d}_{\max}N\sqrt{TN}$ results from the use of $\text{UCB}_{t,i}^b$, which involves two major components of our algorithm: the use of NNs for reward prediction and the aggregation of the NN parameters. Although not reflecting the benefit of a larger N in the regret bound, both components are important to our algorithm. Firstly, the use of NNs for reward prediction is a crucial component in neural contextual bandit in order to exploit the strong representation power of NNs. This is in a similar spirit to the previous works on neural contextual bandit [47, 48] in which the use of NNs for reward prediction does not improve the regret upper bound (compared with using the linear prediction given by the first term of $\text{UCB}_{t,i}^a$) yet improves the practical performance. Secondly, the aggregation of the NN parameters is also important for the performance of our FN-UCB since it allows us to exploit the federated setting in a similar way to FL for supervised learning, which has been repeatedly shown to improve the performance [23]. Moreover, we have also empirically verified that both components (i.e., the use of NNs for reward prediction and the aggregation of NN parameters) are important for the practical performance of our FN-UCB algorithm (Sec. 6.1).

Intuitively, the *effective dimension* \tilde{d} measures the actual underlying dimension of the set of all TKN contexts for all agents [47], and $\tilde{d}_{\max} \triangleq \max_{i \in [N]} \tilde{d}_i$ is the maximum among the underlying dimensions of the set of TK contexts for each agent. [47] showed that if all contexts lie in a d' -dimensional subspace of the RKHS induced by the NTK, then the effective dimension of these contexts can be upper-bounded by the constant d' . In this case, our regret upper bound becomes $\tilde{O}(\sqrt{TN}^{3/2})$ which is sub-linear in T . Moreover, without this assumption on the contexts, we can also obtain the worst-case growth rate of our regret upper bound. Specifically, as we discuss in Appendix B.6, by upper-bounding the effective dimensions using the maximum information gains (Sec. 3) and utilizing the growth rate of the maximum information gain of the NTK: $\gamma_T = \tilde{O}(T^{\frac{d-1}{d}})$ [24, 42], we can further re-write our regret upper bound as $R_T = \tilde{O}(\gamma_{TKN}\sqrt{TN} + \gamma_{TKN}N\sqrt{TN}) = \tilde{O}(T^{\frac{3d-2}{2d}}K^{\frac{(d-1)}{d}}N^{3/2})$, which is sub-linear when the input dimension is $d = 1$.

Upper Bound on Communication Complexity. Next, the following theorem gives an upper bound on the total number of communication rounds of our FN-UCB algorithm (proof in Appendix C).

Theorem 2. *If the width m of the NN satisfies: $m \geq \text{poly}(T, K, N, L, \log(1/\delta))$, then with probability of at least $1 - \delta$, the number of communication rounds for FN-UCB satisfies $C_T = \tilde{O}(\tilde{d}\sqrt{N})$.*

The specific condition on m required by Theorem 2 corresponds to condition 1 listed in Appendix B.1 (see Appendix C for details), which is a subset of the conditions required by Theorem 1. Following the same discussion on the effective dimension \tilde{d} presented above, if all contexts lie in a d' -dimensional subspace of the RKHS induced by the NTK, then \tilde{d} can be upper-bounded by the constant d' , leading to a communication complexity of $C_T = \tilde{O}(\sqrt{N})$. Moreover, without this assumption on the contexts, the connection between \tilde{d} and γ_{TKN} allows us to derive a worst-case communication complexity of $C_T = \tilde{O}(\gamma_{TKN}\sqrt{N}) = \tilde{O}(T^{\frac{d-1}{d}}K^{\frac{d-1}{d}}N^{\frac{3d-2}{2d}})$, which is still sub-linear in T .

5.2 Proof Sketch

We give a brief sketch of our regret analysis (detailed proof in Appendix B). To begin with, we need to prove that both $\text{UCB}_{t,i}^a$ and $\text{UCB}_{t,i}^b$ are valid high-probability upper bounds on the reward function h (Appendix B.3), given that the conditions on m , η and J in Appendix B.1 are satisfied.

Since $\text{UCB}_{t,i}^a$ can be viewed as the Linear UCB policy [1] using the neural tangent features $g(x; \theta_0)/\sqrt{m}$ as the input features (Sec. 4), its validity as a high-probability upper bound on h can be proved following similar steps as standard linear and kernelized bandit [1, 9, 47] (see Lemma 3 in Appendix B.3). Next, to prove that $\text{UCB}_{t,i}^b$ is also a high-probability upper bound on h (up to additive error terms), we define $\theta_{t,i}^{\text{local}} \triangleq (V_{t,i}^{\text{local}})^{-1}(\sum_{\tau=1}^t y_{\tau,i}g(x_{\tau,i}; \theta_0)/\sqrt{m})$, which is defined in the same way as $\bar{\theta}_{t,i}$ (line 3 of Algo. 1) except that $\theta_{t,i}^{\text{local}}$ only uses the local observations of agent i . **Firstly**, we show that $f(x; \theta_{\text{sync,NN}})$ (i.e., the prediction of the NN with the aggregated parameters) is close to $(1/N)\sum_{i=1}^N \langle g(x; \theta_0)/\sqrt{m}, \theta_{t,i}^{\text{local}} \rangle$ (i.e., the linear prediction using $\theta_{t,i}^{\text{local}}$, averaged over all agents). This is achieved by showing that the linear approximation of the NN at the initial parameters θ_0 is close to both of these two terms. **Secondly**, we show that the absolute difference between the linear prediction of agent i : $\langle g(x; \theta_0)/\sqrt{m}, \theta_{t,i}^{\text{local}} \rangle$ and the reward function:

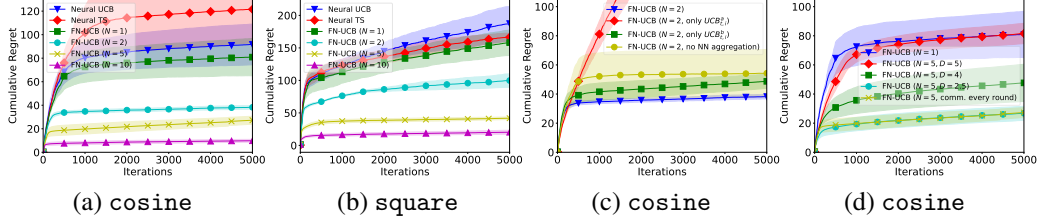


Figure 1: Cumulative regret with varying number of agents for the (a) cosine function and (b) square function. (c) Illustration of the importance of different components of our FN-UCB algorithm (cosine function). (d) Performances with different values of D (cosine function). The average number of rounds of communications are 348.0, 380.0, 456.7 for $D = 5, 4, 2.5$ respectively.

$h(x)$ can be upper-bounded by $\nu_{TK} \sqrt{\lambda} \|g(x; \theta_0) / \sqrt{m}\|_{(V_{t,i}^{\text{local}})^{-1}}$. This can be done following similar steps as the proof for $\text{UCB}_{t,i}^a$ mentioned above. **Thirdly**, using the averaged linear prediction: $(1/N) \sum_{i=1}^N \langle g(x; \theta_0) / \sqrt{m}, \theta_{t,i}^{\text{local}} \rangle$ as an intermediate term, the difference between $f(x; \theta_{\text{sync,NN}})$ and $h(x)$ can be upper-bounded. This implies the validity of $\text{UCB}_{t,i}^b$ as a high-probability upper bound on h (up to additive error terms, which are small given the conditions on m, η and J presented in Appendix B.1), as we have formalized in Lemma 4 in Appendix B.3).

Next, following similar footsteps as the analysis in [44], we separate all epochs into "good" epochs (intuitively, those epochs during which the amount newly collected information from all agents is not too large) and "bad" epochs (details in Appendix B.2), and henceforth separately upper-bound the regrets incurred in these two types of epochs. For good epochs (Appendix B.4), we are able to derive a tight upper bound on the regret $r_{t,i} = h(x_{t,i}^*) - h(x_{t,i})$ in every iteration t by making use of the fact that the change of information in a good epoch is bounded, and hence obtain a tight upper bound on the total regrets in all good epochs. For bad epochs (Appendix B.5), we make use of the result from Appendix B.2 which guarantees that the total number of bad epochs can be upper-bounded. As a result, with the appropriate choice of $D = \mathcal{O}(T/(N\tilde{d}))$, the growth rate of the total regret incurred in bad epochs is smaller than that in good epochs. Lastly, the final regret upper bound follows from adding up the total regrets from good and bad epochs (Appendix B.6).

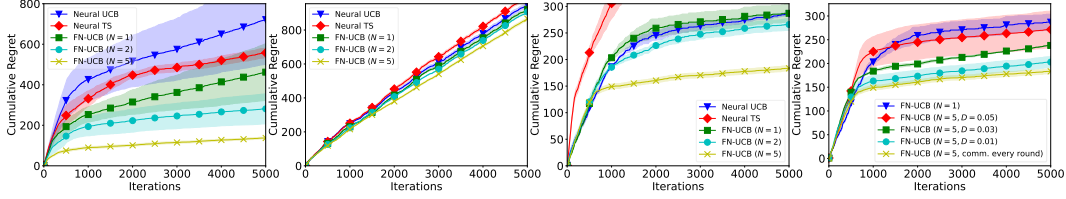
6 Experiments

All figures in this section plot the average cumulative regret across all N agents up to an iteration, which allows us to inspect the benefit the federated setting brings to an agent (on average). In all presented results, unless specified otherwise (by specifying a value of D), a communication round happens after every iteration. All curves stand for the mean and standard error from 3 independent runs. Some experimental details and results are deferred to Appendix D due to the space limitation.

6.1 Synthetic Experiments

We firstly use synthetic experiments to illustrate some interesting insights about our FN-UCB algorithm. Similar to [48], we adopt the synthetic functions of $h(x) = \cos(3\langle a, x \rangle)$ and $h(x) = 10(\langle a, x \rangle)^2$, referred to as the cosine and square functions, respectively. We add a Gaussian observation noise with a standard deviation of 0.01. The parameter a is a 10-dimensional vector randomly sampled from the unit hypersphere. In every iteration, every agent receives $K = 4$ contexts (arms) which are randomly sampled from the unit hypersphere. For fair comparisons, for all methods (including our FN-UCB, Neural UCB and Neural TS), we use the same set of parameters of $\lambda = \nu_{TKN} = \nu_{TK} = 0.1$, and use an NN with one hidden layer and a width of $m = 20$. As suggested by our theoretical analysis (Sec. 4), we choose an increasing sequence of α_t which is linearly increasing (to 1) in the first 700 iterations, and let $\alpha_t = 1$ afterwards.

Fig. 1 presents the results. Fig. 1a and b show that our FN-UCB with $N = 1$ agent performs comparably with Neural UCB and Neural TS, and that the federation of a larger number N of agents consistently improves the performance of our FN-UCB. Of note, the federation of $N = 2$ agents can already provide significant improvements over non-federated algorithms. Fig. 1c gives an illustration of the importance of different components in our FN-UCB. The red curve is obtained by removing $\text{UCB}_{t,i}^b$ (i.e., letting $\alpha_t = 0$), and the green curve corresponds to removing $\text{UCB}_{t,i}^a$. The red curve shows that, despite achieving smaller regrets than the green curve initially thanks to its ability to exploit the observations from the other agents, relying solely on $\text{UCB}_{t,i}^a$ leads to significantly larger



(a) shuttle (b) magic telescope (c) shuttle (diag.) (d) shuttle (diag.)

Figure 2: Results for (a) shuttle and (b) magic ($m = 20$). (c) Results for shuttle with diagonal approximation ($m = 50$). (d) Results for shuttle with different values of D . The average number of communication rounds are 3850.7, 4442.7, 4906.3 for $D = 0.05, 0.03, 0.01$ respectively.

regrets in the long run due to its inability to utilize NNs to model the reward functions. On the other hand, the green curve incurs larger regrets than the red curve initially, however, after more observations are collected (i.e., after the NNs are trained with enough data to accurately model the reward function), it quickly learns to achieve much smaller regrets. These results therefore provide empirical justifications for our discussion on the weight between the two UCBs (Sec. 4), i.e., it is reasonable to use an increasing sequence of α_t such that more weights are given to $UCB_{t,i}^a$ initially yet to $UCB_{t,i}^b$ later. The yellow curve is obtained by removing the step of aggregating (i.e., averaging) the NN parameters (in line 4 of Algo. 2), i.e., when calculating $UCB_{t,i}^b$ (line 5 of Algo. 1), we use θ_t^i and $(V_{t,i}^{\text{local}})^{-1}$ to replace $\theta_{\text{sync,NN}}$ and $V_{\text{sync,NN}}^{-1}$. The results show that the aggregation of the NN parameters significantly improves the performance of FN-UCB (i.e., the blue curve has much smaller regrets than the yellow) and is hence an indispensable part of our FN-UCB algorithm. Lastly, Fig. 1d shows that more frequent communications (i.e., smaller values of D which makes it easier to initiate a communication round, see line 10 of Algo. 1) lead to smaller regrets.

6.2 Real-world Experiments

We adopt the shuttle and magic telescope datasets from the UCI machine learning repository [15], and construct the experiments following a widely used protocol in previous works [27, 47, 48]. A K -class classification problem can be converted into a K -armed contextual bandit problem. In every iteration, an input \mathbf{x} is randomly drawn from the dataset and is then used to construct K context feature vectors: $\mathbf{x}_1 = [\mathbf{x}; \mathbf{0}_d; \dots; \mathbf{0}_d]$, $\mathbf{x}_2 = [\mathbf{0}_d; \mathbf{x}; \dots; \mathbf{0}_d]$, \dots , $\mathbf{x}_K = [\mathbf{0}_d; \dots; \mathbf{0}_d; \mathbf{x}]$, which correspond to the K classes. The reward is 1 if the arm with the correct class is pulled and 0 otherwise. For fair comparisons, we use the same set of parameters of $\lambda = 10$, $\nu_{TKN} = 0.1$ and $\nu_{TK} = 0.01$ for all methods. Fig. 2a and b present the results for the two datasets (one hidden layer, $m = 20$), which show that our FN-UCB with $N = 2$ agents consistently outperform standard Neural UCB and Neural TS, and our performance also improves with the federation of more agents. Fig. 2c shows the results for shuttle when diagonal approximation (Sec. 4) is applied to the NNs (one hidden layer, $m = 50$), in which the results are consistent with those in Fig. 2a.⁴ Moreover, the regrets in Fig. 2c are in general smaller than those in Fig. 2a. This may suggest that in practice, a wider NN with diagonal approximation may be preferable to a narrower NN without diagonal approximation, since they not only improves the performance, but also reduces the computational and communication costs (Sec. 4). Fig. 2d plots the regrets of shuttle (with diagonal approximation) for different values of D , which shows that more frequent communications lead to better performances and are hence consistent with Fig. 1d. For completeness, we also compare with linear and kernelized contextual bandit algorithms (for the experiments in both Secs. 6.1 and 6.2), and the results (Fig. 3, Appendix D) show that they are outperformed by neural contextual bandit algorithms.

7 Conclusion

We introduced FN-UCB, the first federated neural contextual bandit algorithm. We use a weighted combination of two UCBs, and the choice of this weight required by our theoretical analysis has an interesting interpretation which emphasizes accelerated exploration initially and accurate prediction of the aggregated NN later. We derive upper bounds on the regret and communication complexity of FN-UCB, and verify its effectiveness using empirical experiments. Our algorithm is not equipped with privacy guarantees, which may be a potential limitation and will be tackled in future work. A

⁴Since diagonalization increases the scale of the first term in $UCB_{t,i}^a$, we use a heuristic to rescale the values of this term for all contexts such that the max and min values (among all contexts) are 0 and 1 after rescaling.

potential negative societal impact is that our paper may further promote the use of NNs in more applications, which may increase energy consumption and contribute to the greenhouse effect.

References

- [1] Y. Abbasi-Yadkori, D. Pál, and C. Szepesvári. Improved algorithms for linear stochastic bandits. In *Proc. NeurIPS*, volume 24, 2011.
- [2] S. Agrawal and N. Goyal. Thompson sampling for contextual bandits with linear payoffs. In *Proc. ICML*, pages 127–135. PMLR, 2013.
- [3] S. Arora, S. S. Du, W. Hu, Z. Li, R. Salakhutdinov, and R. Wang. On exact computation with an infinitely wide neural net. arXiv:1904.11955, 2019.
- [4] P. Auer. Using confidence bounds for exploitation-exploration trade-offs. *Journal of Machine Learning Research*, 3(Nov):397–422, 2002.
- [5] Y. Ban and J. He. Convolutional neural bandit: Provable algorithm for visual-aware advertising. arXiv:2107.07438, 2021.
- [6] Y. Ban, Y. Yan, A. Banerjee, and J. He. Ee-net: Exploitation-exploration neural networks in contextual bandits. In *International Conference on Learning Representations*, 2022.
- [7] S. Bubeck and N. Cesa-Bianchi. Regret analysis of stochastic and nonstochastic multi-armed bandit problems. *Foundations and Trends® in Machine Learning*, 5(1):1–122, 2012.
- [8] Y. Cao and Q. Gu. Generalization bounds of stochastic gradient descent for wide and deep neural networks. In *Proc. NeurIPS*, volume 32, 2019.
- [9] S. R. Chowdhury and A. Gopalan. On kernelized multi-armed bandits. In *Proc. ICML*, pages 844–853, 2017.
- [10] R. Ciucanu, P. Lafourcade, G. Marcadet, and M. Soare. Samba: A generic framework for secure federated multi-armed bandits. *Journal of Artificial Intelligence Research (JAIR)*, 2022.
- [11] Z. Dai, B. K. H. Low, and P. Jaillet. Federated Bayesian optimization via Thompson sampling. In *Proc. NeurIPS*, 2020.
- [12] Z. Dai, B. K. H. Low, and P. Jaillet. Differentially private federated Bayesian optimization with distributed exploration. In *Proc. NeurIPS*, volume 34, 2021.
- [13] V. Dani, T. P. Hayes, and S. M. Kakade. Stochastic linear optimization under bandit feedback. 2008.
- [14] I. Demirel, Y. Yildirim, and C. Tekin. Federated multi-armed bandits under byzantine attacks. arXiv:2205.04134, 2022.
- [15] D. Dua and C. Graff. UCI machine learning repository, 2017.
- [16] A. Dubey and A. Pentland. Differentially-private federated linear bandits. In *Proc. NeurIPS*, volume 33, pages 6003–6014, 2020.
- [17] K. Greenewald, A. Tewari, S. Murphy, and P. Klasnja. Action centered contextual bandits. *Advances in neural information processing systems*, 30, 2017.
- [18] Q. Gu, A. Karbasi, K. Khosravi, V. Mirrokni, and D. Zhou. Batched neural bandits. arXiv:2102.13028, 2021.
- [19] R. Huang, W. Wu, J. Yang, and C. Shen. Federated linear contextual bandits. In *Proc. NeurIPS*, volume 34, 2021.
- [20] A. Jacot, F. Gabriel, and C. Hongler. Neural tangent kernel: Convergence and generalization in neural networks. In *Proc. NeurIPS*, 2018.
- [21] A. Jadbabaie, H. Li, J. Qian, and Y. Tian. Byzantine-robust federated linear bandits. arXiv:2204.01155, 2022.
- [22] Y. Jia, W. Zhang, D. Zhou, Q. Gu, and H. Wang. Learning neural contextual bandits through perturbed rewards. In *International Conference on Learning Representations*, 2021.
- [23] P. Kairouz, H. B. McMahan, B. Avent, A. Bellet, M. Bennis, A. N. Bhagoji, K. Bonawitz, Z. Charles, G. Cormode, R. Cummings, et al. Advances and open problems in federated learning. arXiv:1912.04977, 2019.

- [24] P. Kassarai and A. Krause. Neural contextual bandits without regret. arXiv:2107.03144, 2021.
- [25] T. Lattimore and C. Szepesvári. *Bandit algorithms*. Cambridge University Press, 2020.
- [26] C. Li and H. Wang. Asynchronous upper confidence bound algorithms for federated linear bandits. In *Proc. AISTATS*, 2022.
- [27] L. Li, W. Chu, J. Langford, and R. 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, 2010.
- [28] Q. Li, Z. Wen, Z. Wu, S. Hu, N. Wang, Y. Li, X. Liu, and B. He. A survey on federated learning systems: vision, hype and reality for data privacy and protection. *IEEE Transactions on Knowledge and Data Engineering*, 2021.
- [29] T. Li, A. K. Sahu, A. Talwalkar, and V. Smith. Federated learning: Challenges, methods, and future directions. arXiv:1908.07873, 2014.
- [30] T. Li and L. Song. Privacy-preserving communication-efficient federated multi-armed bandits. *IEEE Journal on Selected Areas in Communications*, 2022.
- [31] T. Li, L. Song, and C. Fragouli. Federated recommendation system via differential privacy. In *2020 IEEE International Symposium on Information Theory (ISIT)*, pages 2592–2597. IEEE, 2020.
- [32] W. Li, X. Wang, R. Zhang, Y. Cui, J. Mao, and R. Jin. Exploitation and exploration in a performance based contextual advertising system. In *Proceedings of the 16th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 27–36, 2010.
- [33] M. Lisicki, A. Afkanpour, and G. W. Taylor. An empirical study of neural kernel bandits. In *NeurIPS Workshop on Bayesian Deep Learning*, 2021.
- [34] H. B. McMahan, E. Moore, D. Ramage, S. Hampson, et al. Communication-efficient learning of deep networks from decentralized data. In *Proc. AISTATS*, 2017.
- [35] O. Nabati, T. Zahavy, and S. Mannor. Online limited memory neural-linear bandits with likelihood matching. In *Proc. ICML*, 2021.
- [36] T. Nguyen-Tang, S. Gupta, A. T. Nguyen, and S. Venkatesh. Offline neural contextual bandits: Pessimism, optimization and generalization. In *International Conference on Learning Representations*, 2022.
- [37] P. Rusmevichientong and J. N. Tsitsiklis. Linearly parameterized bandits. *Mathematics of Operations Research*, 35(2):395–411, 2010.
- [38] C. Shi and C. Shen. Federated multi-armed bandits. In *Proc. AAAI*, 2021.
- [39] C. Shi, C. Shen, and J. Yang. Federated multi-armed bandits with personalization. In *Proc. AISTATS*, pages 2917–2925. PMLR, 2021.
- [40] N. Srinivas, A. Krause, S. M. Kakade, and M. Seeger. Gaussian process optimization in the bandit setting: No regret and experimental design. In *Proc. ICML*, pages 1015–1022, 2010.
- [41] W. R. Thompson. On the likelihood that one unknown probability exceeds another in view of the evidence of two samples. *Biometrika*, 25(3/4):285–294, 1933.
- [42] S. Vakili, M. Bromberg, J. Garcia, D.-s. Shiu, and A. Bernacchia. Uniform generalization bounds for overparameterized neural networks. arXiv:2109.06099, 2021.
- [43] M. Valko, N. Korda, R. Munos, I. Flaounas, and N. Cristianini. Finite-time analysis of kernelised contextual bandits. In *Proc. UAI*, 2013.
- [44] Y. Wang, J. Hu, X. Chen, and L. Wang. Distributed bandit learning: Near-optimal regret with efficient communication. In *Proc. ICLR*, 2020.
- [45] P. Xu, Z. Wen, H. Zhao, and Q. Gu. Neural contextual bandits with deep representation and shallow exploration. arXiv:2012.01780, 2020.
- [46] Z. Yan, Q. Xiao, T. Chen, and A. Tajer. Federated multi-armed bandit via uncoordinated exploration. In *ICASSP 2022-2022 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 5248–5252. IEEE, 2022.
- [47] W. Zhang, D. Zhou, L. Li, and Q. Gu. Neural Thompson sampling. In *Proc. ICLR*, 2021.

- [48] D. Zhou, L. Li, and Q. Gu. Neural contextual bandits with UCB-based exploration. In *Proc. ICML*, pages 11492–11502. PMLR, 2020.
- [49] Z. Zhu, J. Zhu, J. Liu, and Y. Liu. Federated bandit: A gossiping approach. In *Abstract Proceedings of the 2021 ACM SIGMETRICS/International Conference on Measurement and Modeling of Computer Systems*, pages 3–4, 2021.

A More Background

In this section, we give more details on some of the technical background mentioned in Sec. 3. The details in this section all follow the works of [47, 48], and we present them here for completeness.

Definition of the NN $f(x; \theta)$. Let $\mathbf{W}_1 \in \mathbb{R}^{m \times d}$, $\mathbf{W}_l \in \mathbb{R}^{m \times m}, \forall l = 2, \dots, L-1$, and $\mathbf{W}_L \in \mathbb{R}^{m \times 1}$, then the NN $f(x; \theta)$ is defined as

$$\begin{aligned} f_1 &= \mathbf{W}_1 x, \\ f_l &= \mathbf{W}_l \text{ReLU}(f_{l-1}), \forall l = 2 \dots, L, \\ f(x; \theta) &= \sqrt{m} f_L, \end{aligned}$$

in which $\text{ReLU}(z) = \max(z, 0)$ denotes the rectified linear unit (ReLU) activation function and is applied to each element of f_{l-1} . With this definition of the NN, θ denotes the collection of all parameters of the NN: $\theta = (\text{vec}(\mathbf{W}_1), \dots, \text{vec}(\mathbf{W}_L)) \in \mathbb{R}^{p_0}$.

Details of the Initialization Scheme $\theta_0 \sim \text{init}(\cdot)$. To obtain the initial parameters θ_0 , for each $l = 1, \dots, L-1$, let $\mathbf{W}_l = \begin{pmatrix} \mathbf{W} & \mathbf{0} \\ \mathbf{0} & \mathbf{W} \end{pmatrix}$ where each entry of \mathbf{W} is independently sampled from $\mathcal{N}(0, 4/m)$, and let $\mathbf{W}_L = (\mathbf{w}^\top, -\mathbf{w}^\top)$ where each entry of \mathbf{w} is independently sampled from $\mathcal{N}(0, 2/m)$. This initialization scheme is the same as that used by the works of [47, 48].

Definitions of the NTK Matrices \mathbf{H} and \mathbf{H}_i 's. To simplify the exposition here, we use $\{x^j\}_{j=1, \dots, TKN}$ to denote the set of all contexts from all iterations, all arms and *all agents*: $\{x_{t,i}^k\}_{t \in [T], k \in [K], i \in [N]}$. We can then define

$$\begin{aligned} \tilde{\mathbf{H}}_{i,j}^{(1)} &= \Sigma_{i,j}^{(1)} = \langle x^i, x^j \rangle, \mathbf{A}_{i,j}^{(l)} = \begin{pmatrix} \Sigma_{i,i}^{(l)} & \Sigma_{i,j}^{(l)} \\ \Sigma_{i,j}^{(l)} & \Sigma_{j,j}^{(l)} \end{pmatrix}, \\ \Sigma_{i,j}^{(l+1)} &= 2\mathbb{E}_{(u,v) \sim \mathcal{N}(\mathbf{0}, \mathbf{A}_{i,j}^{(l)})} \max(u, 0) \max(v, 0), \\ \tilde{\mathbf{H}}_{i,j}^{(l+1)} &= 2\tilde{\mathbf{H}}_{i,j}^{(l)} \mathbb{E}_{(u,v) \sim \mathcal{N}(\mathbf{0}, \mathbf{A}_{i,j}^{(l)})} \mathbb{1}(u > 0) \mathbb{1}(v > 0) + \Sigma_{i,j}^{(l+1)}. \end{aligned}$$

With these definitions, the NTK matrix is defined as $\mathbf{H} = (\tilde{\mathbf{H}}^{(L)} + \Sigma^{(L)})/2$. Similarly, \mathbf{H}_i can be obtained in the same way by only using all contexts from agent i in the definitions above, i.e., now we use $\{x^j\}_{j=1, \dots, TK}$ to denote $\{x_{t,i}^k\}_{t \in [T], k \in [K]}$ and plug these TK contexts into the definitions above to obtain $\tilde{\mathbf{H}}_i$.

B Proof of Regret Upper Bound (Theorem 1)

We use p to index different epochs and denote by P the total number of epochs. We use t_p to denote the first iteration of epoch p , and use E_p to represent the length (i.e., number of iterations) of epoch p . Throughout our theoretical analysis, we will denote different error probabilities as $\delta_1, \dots, \delta_6$, which we will combine via a union bound at the end of the proof to ensure that our final regret upper bound holds with probability of at least $1 - \delta$.

B.1 Conditions on the Width m of the Neural Networks

We list here the detailed conditions on the width m of the NN that are needed by our theoretical analysis. These include two types of conditions, some of them (conditions 1-4) are required for our regret upper bound to hold (i.e., they are used during the proof to derive the regret upper bound), whereas the others (conditions 5-6) are used after the final regret upper bound is derived to ensure that the final regret upper bound is small (see Appendix B.6).

When presenting our detailed proofs starting from the next subsection, we will refer to each of these conditions whenever they are used by the corresponding lemmas. Different lemmas may use different leading constants in their required condition (i.e., lower bound) on m , but here we use the same constant $C > 0$ for all lower bounds for simplicity, which can be considered as simply taking the maximum among all these different leading constants for different lemmas.

1. $m \geq CT^6 K^6 N^6 L^6 \log(TKNL/\delta)$,
2. $m \geq CT^4 K^4 N^4 L^6 \log(T^2 K^2 N^2 L/\delta)/\lambda^4$,
3. $m \geq C\sqrt{\lambda}L^{-3/2}[\log(TKNL^2/\delta)]^{3/2}$,
4. $m(\log m)^{-3} \geq CTL^{12}\lambda^{-1} + CT^7\lambda^{-8}L^{18}(\lambda + LT)^6 + CL^{21}T^7\lambda^{-7}(1 + \sqrt{T/\lambda})^6$.
5. $m(\log m)^{-3} \geq CT^{10}N^6\lambda^{-4}L^{18}$,
6. $m(\log m)^{-3} \geq CT^{16}N^6L^{24}\lambda^{-10}(1 + \sqrt{T/\lambda})^6$.

Some of these conditions above can be combined, but we leave them as separate conditions to make it easier to refer to the corresponding place in the proof where a particular condition is needed.

Furthermore, to achieve a small upper bound on the cumulative regret, we also need to place some conditions on the learning rate η and number of iterations J for the gradient descent training (line 13 of Algo. 1). Specifically, we need to choose the learning rate as

$$\eta = C_4(m\lambda + mTL)^{-1}, \quad (2)$$

in which $C_4 > 0$ is an absolute constant such that $C_4 \leq 1 + TL$, and choose

$$J = \frac{1}{C_4} \left(1 + \frac{TL}{\lambda}\right) \log \left(\frac{1}{3C_2N} \sqrt{\frac{\lambda}{T^3L}} \right) = \tilde{O}(TL/(\lambda C_4)). \quad (3)$$

B.2 Definition of Good and Bad Epochs

Denote the matrix V_{last} (see line 17 of Algo. 1) after epoch p as V_p . As a result, the matrix V_P is calculated using all selected inputs from all agents: $V_P = \sum_{t=1}^T \sum_{i=1}^N g(x_{t,i}; \theta_0)g(x_{t,i}; \theta_0)^\top / m + \lambda I$. Define $V_0 \triangleq \lambda I$. Imagine that we have a hypothetical agent which chooses all $T \times N$ queries $\{x_{t,i}\}_{t \in [T], i \in [N]}$ sequentially in a round-robin fashion (i.e., the hypothetical agent chooses $x_{1,1}, x_{1,2}, \dots, x_{2,1}, x_{2,2}, \dots, x_{T,N}$), and denote the corresponding hypothetical covariance matrix as $\tilde{V}_{t,i} = \sum_{\tau=1}^{t-1} \sum_{j=1}^N g(x_{\tau,j}; \theta_0)g(x_{\tau,j}; \theta_0)^\top / m + \sum_{j=1}^i g(x_{t,j}; \theta_0)g(x_{t,j}; \theta_0)^\top / m + \lambda I$. We represent the indices of this hypothetical agent by $t' \in [TN]$ to distinguish it from our original multi-agent setting. Define $J_{TN} \triangleq [g(x_{t'}; \theta_0)]_{t' \in [TN]}$ which is a $p_0 \times (TN)$ matrix, and define $\mathbf{K}_{TN} \triangleq J_{TN}^\top J_{TN} / m$, which is a $(TN) \times (TN)$ matrix. According to these definitions, we have that

Lemma 1 (Lemma B.7 of [47]). *Let $\delta_1 \in (0, 1)$. If $m \geq CT^6 N^6 K^6 L^6 \log(TNKL/\delta_1)$, we have with probability of at least $1 - \delta_1$ that*

$$\log \det(I + \lambda^{-1} \mathbf{K}_{t'}) \leq \log \det(I + \lambda^{-1} \mathbf{H}) + 1, \forall t' \in [TN].$$

The condition on m given in Lemma 1 corresponds to condition 1 listed in Appendix B.1, except that δ_1 is used here instead of δ in Appendix B.1. Lemma 1 allows us to derive the following equation, which we will use (at the end of this section) to justify that the total number of "bad" epochs is not

too large.

$$\begin{aligned}
\sum_{p=0}^{P-1} \log \frac{\det V_{p+1}}{\det V_p} &= \log \frac{\det V_P}{\det V_0} \stackrel{(a)}{=} \log \frac{\det (J_{TN} J_{TN}^\top / m + \lambda I)}{\det V_0} \\
&= \log \frac{\det (\lambda (\lambda^{-1} J_{TN} J_{TN}^\top / m + I))}{\det V_0} \\
&\stackrel{(b)}{=} \log \frac{\lambda^{p_0} \det (\lambda^{-1} J_{TN} J_{TN}^\top / m + I)}{\lambda^{p_0}} \\
&= \log \det (\lambda^{-1} J_{TN} J_{TN}^\top / m + I) \tag{4} \\
&\stackrel{(c)}{=} \log \det (\lambda^{-1} J_{TN}^\top J_{TN} / m + I) \\
&= \log \det (\lambda^{-1} \mathbf{K}_{TN} + I) \\
&\stackrel{(d)}{\leq} \log \det (\lambda^{-1} \mathbf{H} + I) + 1 \\
&\stackrel{(e)}{=} \tilde{d} \log(1 + TKN/\lambda) + 1 \triangleq R'.
\end{aligned}$$

Step (a) is because $V_P = J_{TN} J_{TN}^\top / m + \lambda I$ according to our definition of J_{TN} above. Step (b) follows from our definition of $V_0 = \lambda I$ above, as well as some standard properties of matrix determinant. Step (c) follows because: $\det(\mathbf{A} \mathbf{A}^\top + I) = \det(\mathbf{A}^\top \mathbf{A} + I)$. Step (d) has made use of Lemma 1 above, which suggests that Equation (4) holds with probability of at least $1 - \delta_1$. Step (e) follows from the definition of $\tilde{d} \triangleq \frac{\log \det(I + \mathbf{H}/\lambda)}{\log(1 + TKN/\lambda)}$ (Sec. 3). In the last step, we have defined $R' \triangleq \tilde{d} \log(1 + TKN/\lambda) + 1$. We further define $R \triangleq \lceil R' \rceil$, in which $\lceil \cdot \rceil$ denotes the ceiling operator. Now we define all epochs p 's which satisfy the following condition as "good epochs":

$$1 \leq \frac{\det V_p}{\det V_{p-1}} \leq e, \tag{5}$$

and define all other epochs as "bad epochs". The first inequality trivially holds for all epochs according to the way in which the matrices are constructed. It is easy to verify that the second inequality holds for at least R epochs (with probability of at least $1 - \delta_1$). This is because if the second inequality is violated for more than R epochs (i.e., if $\log \frac{\det V_p}{\det V_{p-1}} > 1$ for more than R epochs), then $\sum_{p=0}^{P-1} \log \frac{\det V_{p+1}}{\det V_p} > R$, which violates equation (4). This suggests that *there are no more than R bad epochs* (with probability of at least $1 - \delta_1$). From here onwards, we will denote the set of good epochs by $\mathcal{E}^{\text{good}}$ and the set of bad epochs by \mathcal{E}^{bad} .

B.3 Validity of the Upper Confidence Bound

In this section, we prove that the upper confidence bound used in our algorithm, $(1 - \alpha_t) \text{UCB}_{t,i}^a(x) + \alpha_t \text{UCB}_{t,i}^b(x)$ (used in line 6 of Algo. 1), is a valid high-probability upper bound on the reward function h . We will achieve this by separately proving that $\text{UCB}_{t,i}^a$ and $\text{UCB}_{t,i}^b$ are valid high-probability upper bounds on h in the next two sections.

B.3.1 Validity of $\text{UCB}_{t,i}^a$ as A High-Probability Upper Bound on h :

To begin with, we will need the following lemma from [47].

Lemma 2 (Lemma B.3 of [47]). *Let $\delta_2 \in (0, 1)$. There exists a constant $C > 0$ such that if*

$$m \geq CT^4 K^4 N^4 L^6 \log(T^2 K^2 N^2 L / \delta_2) / \lambda_0^4,$$

then with probability of $\geq 1 - \delta_2$ over random initializations of θ_0 , there exists a $\theta^ \in \mathbb{R}^{p_0}$ such that*

$$h(x) = \langle g(x; \theta_0), \theta^* - \theta_0 \rangle, \quad \sqrt{m} \|\theta^* - \theta_0\|_2 \leq \sqrt{2 \mathbf{h}^\top \mathbf{H}^{-1} \mathbf{h}} \leq B, \quad \forall x \in \mathcal{X}_{t,i}, t \in [T], i \in [N]. \tag{6}$$

The condition on m required by Lemma 2 corresponds to condition 2 listed in Appendix B.1, except that δ_2 is used here instead of δ as in Appendix B.1. The following lemma formally guarantees the validity of $\text{UCB}_{t,i}^a$ as a high-probability upper-bound on h .

Lemma 3. *Let $\delta_3 \in (0, 1)$ and $\nu_{TKN} = B + R\sqrt{2(\log(1/\delta_3) + 1) + \tilde{d}\log(1 + TKN/\lambda)}$. We have with probability of at least $1 - \delta_1 - \delta_2 - \delta_3$ for all $t \in [T], i \in [N]$, that*

$$|h(x) - \langle g(x; \theta_0) / \sqrt{m}, \bar{\theta}_{t,i} \rangle| \leq \nu_{TKN} \sqrt{\lambda} \|g(x; \theta_0) / \sqrt{m}\|_{V_{t,i}^{-1}}, \forall x \in \mathcal{X}_{t,i}$$

Proof. Lemma 3 can be proved by following similar steps as the proof of Lemma 4.3 in the work of [47]. Specifically, the proof of Lemma 3 requires Lemmas B.3, B.6 and B.7 of [47] (after being adapted for our setting). The adapted versions of Lemmas B.3 and B.7 of [47] have been presented in our Lemma 2 and Lemma 1, respectively. Of note, Lemma 1 and Lemma 2 require some conditions on the width m of the NN, which have been listed as conditions 1 and 2 in Appendix B.1. Lastly, Lemma B.6 of [47], which makes use of Theorem 1 of [9], can be directly applied in our setting and introduces an error probability of δ_3 (which appears in the expression of ν_{TKN}). As a result, Lemma 3 holds with probability of at least $1 - \delta_1 - \delta_2 - \delta_3$, in which the error probabilities come from Lemma 1 (δ_1), Lemma 2 (δ_2) and the application of Lemma B.6 of [47] (δ_3). \square

B.3.2 Validity of $\text{UCB}_{t,i}^b$ as A High-Probability Upper Bound on h :

Note that $\text{UCB}_{t,i}^b$ is updated only in every communication round. We denote the set of iterations after which $\text{UCB}_{t,i}^b$ is updated (i.e., the last iteration in every epoch) as $\mathcal{T}_{-1} \triangleq \{t_p - 1\}_{p=2, \dots, P-1}$, which immediately implies that $\mathcal{T}_{-1} \subset [T]$ and hence $|\mathcal{T}_{-1}| \leq T$.

Lemma 4. *Let $\delta_4, \delta_5 \in (0, 1)$, and $\nu_{TK} = B + R\sqrt{2(\log(N/\delta_4) + 1) + \tilde{d}_{\max} \log(1 + TK/\lambda)}$. Suppose the width m of the NN satisfies $m \geq C\sqrt{\lambda}L^{-3/2}[\log(TKNL^2/\delta_5)]^{3/2}$ for some constant $C > 0$, as well as condition 4 in Appendix B.1. Suppose the learning rate η and number of iterations J of the gradient descent training satisfy the conditions in (2) and (3) (Appendix B.1), respectively. We have with probability of at least $1 - \delta_4 - \delta_5$ for all $t \in \mathcal{T}_{-1}, i \in [N]$, that*

$$|h(x) - f(x; \theta_{\text{sync,NN}})| \leq \nu_{TK} \sqrt{\lambda} \|g(x; \theta_0) / \sqrt{m}\|_{V_{\text{sync,NN}}^{-1}} + \varepsilon_{\text{linear}}(m, T), \forall x \in \mathcal{X}_{t,i}$$

Proof. Note that the condition on m listed in the lemma, $m \geq C\sqrt{\lambda}L^{-3/2}[\log(TKNL^2/\delta_5)]^{3/2}$, corresponds to condition 3 listed in Appendix B.1 except that δ_5 is used here instead of δ . Therefore, the validity of Lemma 4 requires conditions 3 and 4 on m (Appendix B.1) to be satisfied. For ease of exposition, we separate our proof into three steps.

Step 1: NN Output $f(x; \theta_{\text{sync,NN}})$ Is Close to (Averaged) Linear Prediction

Based on Lemma C.2 of [47], if the conditions on m listed in Lemma 4 is satisfied, then for any $\tilde{\theta}$ such that $\|\tilde{\theta} - \theta_0\|_2 \leq 2\sqrt{t/(m\lambda)}$, there exists a constant $C_1 > 0$ such that we have with probability of at least $1 - \delta_5$ over random initializations θ_0 that

$$\begin{aligned} |f(x; \tilde{\theta}) - \langle g(x; \theta_0), \tilde{\theta} - \theta_0 \rangle| &\leq C_1 t^{2/3} m^{-1/6} \lambda^{-2/3} L^3 \sqrt{\log m} \\ &\leq C_1 T^{2/3} m^{-1/6} \lambda^{-2/3} L^3 \sqrt{\log m} \\ &\triangleq \varepsilon_{\text{linear},1}(m, T), \end{aligned} \quad (7)$$

which holds $\forall x \in \mathcal{X}_{t,i}, t \in [T], i \in [N]$.

Also note that according to Lemma C.1 of [47], if conditions 3 and 4 on m listed in Appendix B.1, as well as the condition on η (2), are satisfied, then we have with probability of at least $1 - \delta_5$ over random initializations θ_0 that $\|\theta_t^i - \theta_0\|_2 \leq 2\sqrt{t/m\lambda}, \forall i \in [N]$. An immediate implication is that the aggregated NN parameters $\theta_{\text{sync,NN}} = \frac{1}{N} \sum_{i=1}^N \theta_t^i$ also satisfies:

$$\|\theta_{\text{sync,NN}} - \theta_0\|_2 = \left\| \frac{1}{N} \sum_{i=1}^N \theta_t^i - \theta_0 \right\|_2 \leq \frac{1}{N} \sum_{i=1}^N \|\theta_t^i - \theta_0\|_2 \leq 2\sqrt{t/m\lambda}.$$

This implies that equation (7) holds for $\theta_{\text{sync,NN}}$ with probability of at least $1 - 2\delta_5$:

$$|f(x; \theta_{\text{sync,NN}}) - \langle g(x; \theta_0), \theta_{\text{sync,NN}} - \theta_0 \rangle| \leq \varepsilon_{\text{linear},1}(m, T). \quad (8)$$

Next, note that the θ_t^i is obtained by training only using agent i 's local observations (line 13 of Algo. 1). Define $\theta_{t,i}^{\text{local}} = (V_{t,i}^{\text{local}})^{-1}(\sum_{\tau=1}^t y_{\tau,i} g(x_{\tau,i}; \theta_0)/\sqrt{m})$. Note that $\theta_{t,i}^{\text{local}}$ is calculated in the same way as $\bar{\theta}_{t,i}$ (line 3 of Algo. 1), except that its calculation only involves agent i 's local observations. Next, making use of Lemmas C.1 and C.4 of [47], we can follow similar steps as equation C.3 of [47] (in Appendix C.2 of [47]) to show that there exists constants $C_2 > 0$ and $C_3 > 0$ such that we have $\forall x \in \mathcal{X}_{t,i}, t \in [T], i \in [N]$ that

$$\begin{aligned} & |\langle g(x; \theta_0), \theta_t^i - \theta_0 \rangle - \langle g(x; \theta_0)/\sqrt{m}, \theta_{t,i}^{\text{local}} \rangle| \\ & \leq C_2(1 - \eta m \lambda)^J \sqrt{tL/\lambda} + C_3 m^{-1/6} \sqrt{\log mL^4 t^{5/3} \lambda^{-5/3}} (1 + \sqrt{t/\lambda}) \\ & \leq C_2(1 - \eta m \lambda)^J \sqrt{TL/\lambda} + C_3 m^{-1/6} \sqrt{\log mL^4 T^{5/3} \lambda^{-5/3}} (1 + \sqrt{T/\lambda}) \\ & \triangleq \varepsilon_{\eta,J} + \varepsilon_{\text{linear},2}(m, T). \end{aligned} \quad (9)$$

We refer to $\langle g(x; \theta_0)/\sqrt{m}, \theta_{t,i}^{\text{local}} \rangle$ as the **linear prediction** because it is the prediction of the linear model with the neural tangent features $g(x; \theta_0)/\sqrt{m}$ as the input features, conditioned on the local observations of agent i . Note that similar to (8) which also relies on Lemma C.1 of [47], (9) also requires conditions 3 and 4 on m , as well as the condition on η , in Appendix B.1 to be satisfied. (9) holds with probability of at least $1 - 2\delta_5$, where the error probabilities come from the use of Lemmas C.1 and C.4 of [47].

Next, we can bound the difference between $f(x; \theta_{\text{sync,NN}})$ (i.e., the prediction of the NN with the aggregated parameters) and the averaged linear predictions of all agents calculated using their local observations:

$$\begin{aligned} & |f(x; \theta_{\text{sync,NN}}) - \frac{1}{N} \sum_{i=1}^N \langle g(x; \theta_0)/\sqrt{m}, \theta_{t,i}^{\text{local}} \rangle| \leq |f(x; \theta_{\text{sync,NN}}) - \frac{1}{N} \sum_{i=1}^N \langle g(x; \theta_0), \theta_t^i - \theta_0 \rangle| \\ & \quad + |\frac{1}{N} \sum_{i=1}^N \langle g(x; \theta_0), \theta_t^i - \theta_0 \rangle - \frac{1}{N} \sum_{i=1}^N \langle g(x; \theta_0)/\sqrt{m}, \theta_{t,i}^{\text{local}} \rangle| \\ & \leq |f(x; \theta_{\text{sync,NN}}) - \langle g(x; \theta_0), \theta_{\text{sync,NN}} - \theta_0 \rangle| \\ & \quad + \frac{1}{N} \sum_{i=1}^N |\langle g(x; \theta_0), \theta_t^i - \theta_0 \rangle - \langle g(x; \theta_0)/\sqrt{m}, \theta_{t,i}^{\text{local}} \rangle| \\ & \leq \varepsilon_{\text{linear},1}(m, T) + \frac{1}{N} \sum_{i=1}^N (\varepsilon_{\eta,J} + \varepsilon_{\text{linear},2}(m, T)) \\ & \leq \varepsilon_{\text{linear},1}(m, T) + \varepsilon_{\eta,J} + \varepsilon_{\text{linear},2}(m, T) \\ & \triangleq \varepsilon_{\text{linear}}(m, T). \end{aligned} \quad (10)$$

In the second inequality, we plugged in the definition of $\theta_{\text{sync,NN}} = \frac{1}{N} \sum_{i=1}^N \theta_t^i$. In the third inequality, we have made use of (8) and (9). Equation (10) holds with probability of at least $1 - 4\delta_5$, where the error probabilities come from (8) ($2\delta_5$) and (9) ($2\delta_5$), respectively. Now we replace δ_5 by $\delta_5/4$, which ensures that (10) holds with probability of at least $1 - \delta_5$. This will only introduce a factor of 4 within the log of condition 3 on m (Appendix B.1), which is ignored since it can be absorbed by the constant C .

Step 2: Linear Prediction Is Close to the Reward Function $h(x)$

In the proof in this section, we will also need a "local" variant of the confidence bound of Lemma 3, i.e., the confidence bound of Lemma 3 calculated only using the local observations of an agent i :

Lemma 5 ([47]). *We have with probability of at least $1 - \delta_4$ for all $t \in \mathcal{T}_{-1} \subset [T], i \in [N]$, that*

$$|h(x) - \langle g(x; \theta_0)/\sqrt{m}, \theta_{t,i}^{\text{local}} \rangle| \leq \nu_{TK} \sqrt{\lambda} \|g(x; \theta_0)/\sqrt{m}\|_{(V_{t,i}^{\text{local}})^{-1}}, \forall x \in \mathcal{X}_{t,i}.$$

Proof. Similar to the proof of Lemma 3 (Appendix B.3.1), the proof of Lemma 5 also requires of Lemmas B.3, B.6 and B.7 from [47], which, in this case, can be directly applied to our setting (except that we need an additional union bound over all N agents). The implication of the additional union bound on the error probabilities is taken care of by the additional term of N within the log in the expression of ν_{TK} (Lemma 4), and in conditions 1 and 2 on m (see Appendix B.1, and also Lemmas 2 and 1). The required lower bounds on m by the local variants of Lemmas B.3 and B.7 (required in the proof here) are smaller than those given in Lemmas 2 and 1 and hence do not need to appear in the conditions in Appendix B.1. By letting the sum of the three error probabilities (resulting from the applications of Lemmas B.3, B.6 and B.7 of [47]) be δ_4 , we can ensure that Lemma 5 holds with probability of at least $1 - \delta_4$. For simplicity, we let the error probability for Lemma B.6 be δ_4 , which leads to the cleaner expression of ν_{TK} in Lemma 4. This means that the error probabilities for Lemmas B.3 and B.7 are very small, which can be accounted for by simply increasing the value of the absolute constant C in conditions 1 and 2 on m (Appendix B.1) and hence does not affect our main theoretical analysis. \square

Step 3: Combining Results from Step 1 and Step 2

Next, we are ready to prove the validity of $\text{UCB}_{t,i}^b$ by using the averaged linear prediction $\frac{1}{N} \sum_{i=1}^N \langle g(x; \theta_0) / \sqrt{m}, \theta_{t,i}^{\text{local}} \rangle$ as an intermediate term:

$$\begin{aligned}
& |f(x; \theta_{\text{sync,NN}}) - h(x)| \\
& \leq |f(x; \theta_{\text{sync,NN}}) - \frac{1}{N} \sum_{i=1}^N \langle g(x; \theta_0) / \sqrt{m}, \theta_{t,i}^{\text{local}} \rangle + \frac{1}{N} \sum_{i=1}^N \langle g(x; \theta_0) / \sqrt{m}, \theta_{t,i}^{\text{local}} \rangle - h(x)| \\
& \leq \frac{1}{N} \sum_{i=1}^N |\langle g(x; \theta_0) / \sqrt{m}, \theta_{t,i}^{\text{local}} \rangle - h(x)| + \varepsilon_{\text{linear}}(m, T) \\
& \leq \frac{1}{N} \sum_{i=1}^N \nu_{TK} \sqrt{\lambda} \|g(x; \theta_0) / \sqrt{m}\|_{(V_{t,i}^{\text{local}})^{-1}} + \varepsilon_{\text{linear}}(m, T) \\
& = \nu_{TK} \frac{1}{N} \sum_{i=1}^N \sqrt{\lambda g(x; \theta_0)^\top (V_{t,i}^{\text{local}})^{-1} g(x; \theta_0) / m} + \varepsilon_{\text{linear}}(m, T) \\
& \leq \nu_{TK} \sqrt{\frac{1}{N} \sum_{i=1}^N \lambda g(x; \theta_0)^\top (V_{t,i}^{\text{local}})^{-1} g(x; \theta_0) / m} + \varepsilon_{\text{linear}}(m, T) \\
& = \nu_{TK} \sqrt{\lambda g(x; \theta_0)^\top \left(\frac{1}{N} \sum_{i=1}^N (V_{t,i}^{\text{local}})^{-1} \right) g(x; \theta_0) / m} + \varepsilon_{\text{linear}}(m, T) \\
& = \nu_{TK} \sqrt{\lambda g(x; \theta_0)^\top (V_{\text{sync,NN}}^{-1}) g(x; \theta_0) / m} + \varepsilon_{\text{linear}}(m, T) \\
& = \nu_{TK} \sqrt{\lambda} \|g(x; \theta_0) / \sqrt{m}\|_{V_{\text{sync,NN}}^{-1}} + \varepsilon_{\text{linear}}(m, T).
\end{aligned} \tag{11}$$

The second inequality has made use of (10), the third inequality follows from Lemma 5, the fourth inequality results from the concavity of the square root function. In the second last equality, we have plugged in the definition of $V_{\text{sync,NN}}^{-1} = \frac{1}{N} \sum_{i=1}^N (V_{t,i}^{\text{local}})^{-1}$. As a result, (11) holds with probability of at least $1 - \delta_4 - \delta_5$, in which the error probabilities come from Equation (10) (δ_5) and Lemma 5 (δ_4). In other words, Lemma 4 (i.e., the validity of $\text{UCB}_{t,i}^b$) holds with probability of at least $1 - \delta_4 - \delta_5$. \square

B.4 Regret Upper Bound for Good Epochs

In this section, we derive an upper bound on the total regrets incurred in all good epochs $\mathcal{E}^{\text{good}}$ (Appendix B.2).

B.4.1 Auxiliary Inequalities

We firstly derive two auxiliary results which will be used in the proofs later.

To begin with, for agent i and iteration t in a good epoch $p \in \mathcal{E}^{\text{good}}$, we have that

$$\begin{aligned}
\sqrt{\lambda} \|g(x; \theta_0) / \sqrt{m}\|_{\bar{V}_{t,i}^{-1}} &= \sqrt{\lambda g(x; \theta_0)^\top \bar{V}_{t,i}^{-1} g(x; \theta_0) / m} \\
&\leq \sqrt{\lambda g(x; \theta_0)^\top \tilde{V}_{t,i}^{-1} g(x; \theta_0) / m \frac{\det \tilde{V}_{t,i}}{\det \bar{V}_{t,i}}} \\
&\leq \sqrt{\lambda g(x; \theta_0)^\top \tilde{V}_{t,i}^{-1} g(x; \theta_0) / m \frac{\det V_p}{\det V_{p-1}}} \tag{12} \\
&\leq \sqrt{e \lambda g(x; \theta_0)^\top \tilde{V}_{t,i}^{-1} g(x; \theta_0) / m} \\
&= \sqrt{e \lambda} \|g(x; \theta_0) / \sqrt{m}\|_{\tilde{V}_{t,i}^{-1}}.
\end{aligned}$$

Recall that $\bar{V}_{t,i}$ (line 3 of Algo. 1) is used by agent i in iteration t to select $x_{t,i}$ (via $\text{UCB}_{t,i}^a$), and that the matrix $\tilde{V}_{t,i}$ is defined for the hypothetical agent which sequentially chooses all TN queries $\{x_{t,i}\}_{t \in [T], i \in [N]}$ in a round-robin fashion (Appendix B.2). The first inequality in (12) above follows from Lemma 12 of [1]. The second inequality is because V_p contains more information than $\tilde{V}_{t,i}$ (since V_p is calculated using all the inputs selected *after* epoch p), and V_{p-1} contains less information than $\bar{V}_{t,i}$ (because compared with V_{p-1} , $\bar{V}_{t,i}$ additionally contains the local inputs selected by agent i in the current epoch p). In the last inequality, we have made use of the definition of good epochs, i.e., $(\det V_p) / (\det V_{p-1}) \leq e$ (Appendix B.2).

Next, we also need the following auxiliary result for agent i and iteration t in a good epoch $p \in \mathcal{E}^{\text{good}}$:

$$\begin{aligned}
\sqrt{\lambda} \|g(x_{t,i}; \theta_0) / \sqrt{m}\|_{V_{\text{sync,NN}}^{-1}} &= \sqrt{\lambda g(x_{t,i}; \theta_0)^\top V_{\text{sync,NN}}^{-1} g(x_{t,i}; \theta_0) / m} \\
&= \sqrt{\lambda g(x_{t,i}; \theta_0)^\top \left(\frac{1}{N} \sum_{j=1}^N (V_{t_p,j}^{\text{local}})^{-1} \right) g(x_{t,i}; \theta_0) / m} \\
&= \sqrt{\frac{1}{N} \sum_{j=1}^N \lambda g(x_{t,i}; \theta_0)^\top (V_{t_p,j}^{\text{local}})^{-1} g(x_{t,i}; \theta_0) / m} \tag{13} \\
&\leq \frac{1}{\sqrt{N}} \sum_{j=1}^N \sqrt{\lambda g(x_{t,i}; \theta_0)^\top (V_{t_p,j}^{\text{local}})^{-1} g(x_{t,i}; \theta_0) / m} \\
&\leq \frac{1}{\sqrt{N}} \sum_{j=1}^N \sqrt{\lambda} \|g(x_{t,i}; \theta_0) / \sqrt{m}\|_{(V_{t_p,j}^{\text{local}})^{-1}}.
\end{aligned}$$

The first inequality is because $\sqrt{a+b} \leq \sqrt{a} + \sqrt{b}$. Note that in the equation above, $V_{t_p,j}^{\text{local}}$ is indexed by t_p because in epoch p , every $V_{t_p,j}^{\text{local}}$ used in the aggregation to obtain $V_{\text{sync,NN}}^{-1}$ is calculated using agent j 's local observations before iteration t_p , i.e., before the first iteration of epoch p .

B.4.2 Upper Bound on the Instantaneous Regret $r_{t,i}$

Here we assume that both $\text{UCB}_{t,i}^a$ and $\text{UCB}_{t,i}^b$ hold (hence we ignore the error probabilities here), which we have proved in Appendix B.3. We now derive an upper bound on the instantaneous regret

$r_{t,i} = h(x_{t,i}^*) - h(x_{t,i})$ for agent i and iteration t in a good epoch $p \in \mathcal{E}^{\text{good}}$:

$$\begin{aligned}
r_{t,i} &= h(x_{t,i}^*) - h(x_{t,i}) \\
&= \alpha_t h(x_{t,i}^*) + (1 - \alpha_t) h(x_{t,i}) - h(x_{t,i}) \\
&\leq \alpha_t \text{UCB}_{t,i}^b(x_{t,i}^*) + \alpha_t \varepsilon_{\text{linear}}(m, T) + (1 - \alpha_t) \text{UCB}_{t,i}^a(x_{t,i}^*) - h(x_{t,i}) \\
&\leq \alpha_t \text{UCB}_{t,i}^b(x_{t,i}) + (1 - \alpha_t) \text{UCB}_{t,i}^a(x_{t,i}) + \alpha_t \varepsilon_{\text{linear}}(m, T) - h(x_{t,i}) \\
&= \alpha_t \left(\text{UCB}_{t,i}^b(x_{t,i}) - h(x_{t,i}) \right) + (1 - \alpha_t) \left(\text{UCB}_{t,i}^a(x_{t,i}) - h(x_{t,i}) \right) + \alpha_t \varepsilon_{\text{linear}}(m, T) \\
&\leq \alpha_t \left(2\nu_{TK} \sqrt{\lambda} \|g(x_{t,i}; \theta_0) / \sqrt{m}\|_{V_{\text{sync,NN}}^{-1}} + \varepsilon_{\text{linear}}(m, T) \right) + \\
&\quad (1 - \alpha_t) \left(2\nu_{TKN} \sqrt{\lambda} \|g(x_{t,i}; \theta_0) / \sqrt{m}\|_{\tilde{V}_{t,i}^{-1}} \right) + \alpha_t \varepsilon_{\text{linear}}(m, T) \\
&\leq \alpha_t \left(2\nu_{TK} \frac{1}{\sqrt{N}} \sum_{j=1}^N \sqrt{\lambda} \|g(x_{t,i}; \theta_0) / \sqrt{m}\|_{(V_{t_p,j}^{\text{local}})^{-1}} + \varepsilon_{\text{linear}}(m, T) \right) + \\
&\quad (1 - \alpha_t) \left(2\nu_{TKN} \sqrt{e\lambda} \|g(x_{t,i}; \theta_0) / \sqrt{m}\|_{\tilde{V}_{t,i}^{-1}} \right) + \alpha_t \varepsilon_{\text{linear}}(m, T) \\
&= \alpha_t 2\nu_{TK} \frac{1}{\sqrt{N}} \sum_{j=1}^N \sqrt{\lambda} \|g(x_{t,i}; \theta_0) / \sqrt{m}\|_{(V_{t_p,j}^{\text{local}})^{-1}} + \\
&\quad (1 - \alpha_t) 2\nu_{TKN} \sqrt{e\lambda} \|g(x_{t,i}; \theta_0) / \sqrt{m}\|_{\tilde{V}_{t,i}^{-1}} + 2\alpha_t \varepsilon_{\text{linear}}(m, T) \\
&\triangleq (1 - \alpha_t) 2\nu_{TKN} \sqrt{e\lambda} \tilde{\sigma}_{t,i}(x_{t,i}) + \alpha_t 2\nu_{TK} \frac{1}{\sqrt{N}} \sum_{j=1}^N \tilde{\sigma}_{t_p,j}^{\text{local}}(x_{t,i}) + 2\alpha_t \varepsilon_{\text{linear}}(m, T).
\end{aligned} \tag{14}$$

The first inequality makes use of Lemma 3 (i.e., the validity of $\text{UCB}_{t,i}^a$) and Lemma 4 (i.e., the validity of $\text{UCB}_{t,i}^b$). The second inequality follows from the way in which $x_{t,i}$ is selected (line 6 of Algo. 1): $x_{t,i} = \arg \max_{x \in \mathcal{X}_{t,i}} (1 - \alpha_t) \text{UCB}_{t,i}^a(x) + \alpha_t \text{UCB}_{t,i}^b(x)$. The third inequality again makes use of Lemma 3 and Lemma 4, as well as the expressions of $\text{UCB}_{t,i}^a$ and $\text{UCB}_{t,i}^b$. In the fourth inequality, we have made use of the auxiliary inequalities of (12) and (13) we derived in the last section. In the last step, we have defined $\tilde{\sigma}_{t_p,j}^{\text{local}}(x_{t,i}) \triangleq \sqrt{\lambda} \|g(x_{t,i}; \theta_0) / \sqrt{m}\|_{(V_{t_p,j}^{\text{local}})^{-1}}$ which represents the GP posterior standard deviation (using the kernel of $\tilde{k}(x, x') = g(x; \theta_0)^\top g(x'; \theta_0) / m$) conditioned on all agent j 's local observations before iteration t_p . Note that $\tilde{\sigma}_{t_p,j}^{\text{local}}(x_{t,i})$ is the same as the one defined in Sec. 4 of the main text, in the paragraph where we explain the weight between the two UCBs. Similarly, we have also defined $\tilde{\sigma}_{t,i}(x_{t,i}) \triangleq \sqrt{\lambda} \|g(x_{t,i}; \theta_0) / \sqrt{m}\|_{\tilde{V}_{t,i}^{-1}}$, which represents the GP posterior standard deviation conditioned on the observations of the hypothetical agent before $x_{t,i}$ is selected (Appendix B.2).

Next, we will separately derive upper bounds on the summation (across all good epochs and all agents) of the first and second terms of the upper bound from equation (14).

B.4.3 Upper Bound on the Sum of the First term of (14)

Here, similar to [24], we denote as κ_0 an upper bound on the value of the NTK function for any input: $\langle g(x; \theta_0) / \sqrt{m}, g(x; \theta_0) / \sqrt{m} \rangle \leq \kappa_0, \forall x \in \mathcal{X}_{t,i}, t \in [T], i \in [N]$. As a result, we can use it to show that both $\tilde{\sigma}_{t,i}(x)$ and $\tilde{\sigma}_{t_p,j}^{\text{local}}(x)$ can be upper-bounded: $\tilde{\sigma}_{t,i}(x) \leq \sqrt{\kappa_0}$ and $\tilde{\sigma}_{t_p,j}^{\text{local}}(x) \leq \sqrt{\kappa_0}$. To show this, following the notations of Appendix B.2, we denote $\tilde{V}_{t,i} = J_{t,i} J_{t,i}^\top + \lambda I$ where $J_{t,i} = \left[[g(x_{\tau,j}; \theta_0)]_{\tau \in [t-1], j \in [N]}, [g(x_{t,j}; \theta_0)]_{j \in [i]} \right]$ which is a $p_0 \times [(t-1)N + i]$ matrix. Then

we have

$$\begin{aligned}
\tilde{\sigma}_{t,i}^2(x) &= \lambda \|g(x; \theta_0)/\sqrt{m}\|_{\tilde{\mathcal{V}}_{t,i}^{-1}}^2 \\
&= \lambda g(x; \theta_0)^\top (J_{t,i} J_{t,i}^\top + \lambda I)^{-1} g(x; \theta_0)/m \\
&= \lambda g(x; \theta_0)^\top \left(\frac{1}{\lambda} I - \frac{1}{\lambda} J_{t,i} (I + J_{t,i}^\top \frac{1}{\lambda} J_{t,i})^{-1} J_{t,i}^\top \frac{1}{\lambda} \right) g(x; \theta_0)/m \\
&= g(x; \theta_0)^\top g(x; \theta_0)/m - (g(x; \theta_0)^\top/\sqrt{m}) J_{t,i} (\lambda I + J_{t,i}^\top J_{t,i})^{-1} J_{t,i}^\top (g(x; \theta_0)/\sqrt{m}) \\
&= g(x; \theta_0)^\top g(x; \theta_0)/m - \left\| (g(x; \theta_0)^\top/\sqrt{m}) J_{t,i} \right\|_{(\lambda I + J_{t,i}^\top J_{t,i})^{-1}}^2 \\
&\leq g(x; \theta_0)^\top g(x; \theta_0)/m \leq \kappa_0
\end{aligned} \tag{15}$$

where we used the matrix inversion lemma in the third equality. Using similar derivations also allows us to show that $(\tilde{\sigma}_{t_p,j}^{\text{local}}(x))^2 \leq \kappa_0$. Therefore, we have that $\tilde{\sigma}_{t,i}(x) \leq \sqrt{\kappa_0}$ and $\tilde{\sigma}_{t_p,j}^{\text{local}}(x) \leq \sqrt{\kappa_0}$.

Denoting the set of iterations from all good epochs as $\mathcal{T}^{\text{good}}$, we can derive an upper bound the first term of (14), summed across all agents $i \in [N]$ and all iteration in good epochs $\mathcal{T}^{\text{good}}$:

$$\begin{aligned}
\sum_{i=1}^N \sum_{t \in \mathcal{T}^{\text{good}}} (1 - \alpha_t) 2\nu_{TKN} \sqrt{e} \tilde{\sigma}_{t,i}(x_{t,i}) &\stackrel{(a)}{\leq} 2\sqrt{e}\nu_{TKN} \sum_{i=1}^N \sum_{t=1}^T \tilde{\sigma}_{t,i}(x_{t,i}) \\
&\stackrel{(b)}{=} 2\sqrt{e}\nu_{TKN} \sum_{i=1}^N \sum_{t=1}^T \min\{\tilde{\sigma}_{t,i}(x_{t,i}), \sqrt{\kappa_0}\} \\
&\stackrel{(c)}{\leq} 2\sqrt{e}\nu_{TKN} \sum_{i=1}^N \sum_{t=1}^T \min\{\sqrt{\kappa_0} \tilde{\sigma}_{t,i}(x_{t,i}), \sqrt{\kappa_0}\} \\
&\leq 2\sqrt{e}\nu_{TKN} \sqrt{\kappa_0} \sum_{i=1}^N \sum_{t=1}^T \min\{\tilde{\sigma}_{t,i}(x_{t,i}), 1\} \\
&\stackrel{(d)}{\leq} 2\sqrt{e}\nu_{TKN} \sqrt{\kappa_0} \sqrt{TN \sum_{i=1}^N \sum_{t=1}^T \min\{\tilde{\sigma}_{t,i}^2(x_{t,i}), 1\}} \\
&\stackrel{(e)}{\leq} 2\sqrt{e}\nu_{TKN} \sqrt{\kappa_0} \sqrt{TN [2\lambda \log \det(\lambda^{-1} \mathbf{K}_{TN} + I)]} \\
&\stackrel{(f)}{\leq} 2\sqrt{2e}\nu_{TKN} \sqrt{\kappa_0} \sqrt{TN \lambda [\log \det(\lambda^{-1} \mathbf{H} + I) + 1]} \\
&= 2\sqrt{e}\nu_{TKN} \sqrt{\kappa_0} \sqrt{TN \lambda [\tilde{d} \log(1 + TNK/\lambda) + 1]}
\end{aligned} \tag{16}$$

Step (a) follows from $\alpha_t \leq 1, \forall t \geq 1$ and summing across all iterations $[T]$ instead of only those iterations $\mathcal{T}^{\text{good}}$ in good epochs. Step (b) follows because $\tilde{\sigma}_{t,i}(x) \leq \sqrt{\kappa_0}$ as discussed above. In step (c), we have assumed that $\kappa_0 \geq 1$; however, if $\kappa_0 < 1$, the proof still goes through since we can directly upper-bound $\min\{\tilde{\sigma}_{t,i}(x_{t,i}), \sqrt{\kappa_0}\}$ by $\min\{\tilde{\sigma}_{t,i}(x_{t,i}), 1\}$, after which the only modification we need to make to the equation above is to remove the dependency on multiplicative term of $\sqrt{\kappa_0}$. Step (d) results from the Cauchy–Schwarz inequality. Step (e) can be derived following the proof of Lemma 4.8 of [47] (in Appendix B.7 of [47]). Step (f) follows from Lemma 1 and hence holds with probability of at least $1 - \delta_1$. The last equality simply plugs in the definition of the effective dimension \tilde{d} (Sec. 3).

B.4.4 Upper Bound on the Sum of the Second term of (14)

In this subsection, we derive an upper bound on the sum of the second term in equation (14) across all good epochs and all agents.

For the proof here, we need a "local" version of Lemma 1, i.e., a version of Lemma 1 which only makes use of the contexts of an agent i . Define $\mathbf{K}_{t,i}$ as the local counterpart to $\mathbf{K}_{t,\nu}$ (from Lemma 1), i.e., $\mathbf{K}_{t,i}$ is the $t \times t$ matrix calculated using only agent j 's local contexts up to (and including)

iteration t . Specifically, define $J_{t,i} \triangleq [g(x_{\tau,i}; \theta_0)]_{\tau \in [t]}$ which is a $p_0 \times t$ matrix, then $\mathbf{K}_{t,i}$ is defined as $\mathbf{K}_{t,i} \triangleq J_{t,i}^\top J_{t,i} / m$, which is a $t \times t$ matrix. Also recall that in the main text, we have defined \mathbf{H}_i as the local counterpart of \mathbf{H} for agent i (Sec. 3). The next lemma gives our desired local version of Lemma 1.

Lemma 6 (Lemma B.7 of [47]). *If $m \geq CT^6 K^6 L^6 \log(TNKL/\delta_6)$, we have with probability of at least $1 - \delta_6$ that*

$$\log \det(I + \lambda^{-1} \mathbf{K}_{t,i}) \leq \log \det(I + \lambda^{-1} \mathbf{H}_i) + 1,$$

for all $t \in [T], i \in [N]$.

We needed to take a union bound over all N agents, which explains the factor of N within the log in the lower bound on m given in Lemma 6. Note that the required lower bound on m by Lemma 6 is smaller than that of Lemma 1 (by a factor of N^6), therefore, the condition on m in Lemma 6 is ignored in the conditions listed in Appendix B.1.

Of note, throughout the entire epoch p , $\tilde{\sigma}_{t_p,j}^{\text{local}}(x_{t,i})$ is calculated *conditioned on all the local observations of agent j before iteration t_p* . Denote by $\mathcal{T}^{(p)}$ the iteration indices in epoch p : $\mathcal{T}^{(p)} = \{t_p, \dots, t_p + E_p - 1\}$. In the proof in this section, as we have discussed in the first paragraph of Sec. 5.1, we analyze a simpler variant of our algorithm where we only set $\alpha_t > 0$ in the first iteration after a communication round, i.e., $\alpha_t > 0, \forall t \in \{t_p\}_{p \in [P]}$ and $\alpha_t = 0, \forall t \in [T] \setminus \{t_p\}_{p \in [P]}$. Now we are ready to derive an upper bound on the second term in equation (14), summed over all agents and all good epochs:

$$\begin{aligned} \sum_{i=1}^N \sum_{p \in \mathcal{E}^{\text{good}}} \sum_{t \in \mathcal{T}^{(p)}} \alpha_t 2\nu_{TK} \frac{1}{\sqrt{N}} \sum_{j=1}^N \tilde{\sigma}_{t_p,j}^{\text{local}}(x_{t,i}) &\stackrel{(a)}{\leq} 2\nu_{TK} \frac{1}{\sqrt{N}} \sum_{i=1}^N \sum_{p \in [P]} \sum_{t \in \mathcal{T}^{(p)}} \alpha_t \sum_{j=1}^N \tilde{\sigma}_{t_p,j}^{\text{local}}(x_{t,i}) \\ &\stackrel{(b)}{\leq} 2\nu_{TK} \frac{1}{\sqrt{N}} \sum_{i=1}^N \sum_{j=1}^N \sum_{p \in [P]} \alpha_{t_p} \tilde{\sigma}_{t_p,j}^{\text{local}}(x_{t_p,i}) \\ &\stackrel{(c)}{\leq} 2\nu_{TK} \frac{1}{\sqrt{N}} \sum_{i=1}^N \sum_{j=1}^N \sum_{p \in [P]} \tilde{\sigma}_{t_p,j}^{\text{local}}(x_{t_p,j}) \\ &\stackrel{(d)}{\leq} 2\nu_{TK} \frac{1}{\sqrt{N}} \sum_{i=1}^N \sum_{j=1}^N \sum_{t=1}^T \tilde{\sigma}_{t-1,j}^{\text{local}}(x_{t,j}) \end{aligned} \tag{17}$$

The inequality in step (a) results from summing across all epoch $p \in [P]$ instead of only good epochs $p \in \mathcal{E}^{\text{good}}$. Step (b) follows since $\alpha_t = 0, \forall t \in [T] \setminus \{t_p\}_{p \in [P]}$ as we discussed above, therefore, for every epoch p , we only need to keep the first term of $t = t_p$ in the summation of $t \in \mathcal{T}^{(p)}$. To understand step (c), recall that in the main text (Sec. 4, the paragraph "The Weight between the Two UCBs"), we have defined: $\tilde{\sigma}_{t,i,\min}^{\text{local}} \triangleq \min_{x \in \mathcal{X}} \tilde{\sigma}_{t,i}^{\text{local}}(x)$ and $\tilde{\sigma}_{t,i,\max}^{\text{local}} \triangleq \max_{x \in \mathcal{X}} \tilde{\sigma}_{t,i}^{\text{local}}(x), \forall i \in [N]$. Next, note that our algorithm selects α_t by: $\alpha_t = \min_{i \in [N]} \alpha_{t,i}$ (line 4 of Algo. 2) where $\alpha_{t,i} = \tilde{\sigma}_{t,i,\min}^{\text{local}} / \tilde{\sigma}_{t,i,\max}^{\text{local}}$ (line 14 of Algo. 1). As a result, we have that

$$\alpha_{t_p} = \min_{i \in [N]} \alpha_{t_p,i} = \min_{i \in [N]} \frac{\tilde{\sigma}_{t_p,i,\min}^{\text{local}}}{\tilde{\sigma}_{t_p,i,\max}^{\text{local}}} \leq \frac{\tilde{\sigma}_{t_p,j,\min}^{\text{local}}}{\tilde{\sigma}_{t_p,j,\max}^{\text{local}}} \leq \frac{\tilde{\sigma}_{t_p,j}^{\text{local}}(x_{t_p,j})}{\tilde{\sigma}_{t_p,j}^{\text{local}}(x_{t_p,i})},$$

which tells us that $\alpha_t \tilde{\sigma}_{t_p,j}^{\text{local}}(x_{t_p,i}) \leq \tilde{\sigma}_{t_p,j}^{\text{local}}(x_{t_p,j})$ and hence leads to step (c). Step (d) results from summing across all iterations $[T]$ instead of only the first iteration of every epoch: $t \in \{t_p\}_{p \in [P]}$.

Next, we can derive an upper bound on the inner summation over $t = 1, \dots, T$ from (17):

$$\begin{aligned}
\sum_{t=1}^T \tilde{\sigma}_{t-1,j}^{\text{local}}(x_{t,j}) &\stackrel{(a)}{\leq} \sqrt{\kappa_0} \sum_{t=1}^T \min\{\tilde{\sigma}_{t-1,j}^{\text{local}}(x_{t,j}), 1\} \\
&\stackrel{(b)}{\leq} \sqrt{\kappa_0} \sqrt{T \sum_{t=1}^T \min\{(\tilde{\sigma}_{t-1,j}^{\text{local}}(x_{t,j}))^2, 1\}} \\
&\stackrel{(c)}{\leq} \sqrt{\kappa_0} \sqrt{T[2\lambda \log \det(\lambda^{-1} \mathbf{K}_{T,j} + I)]} \\
&\stackrel{(d)}{\leq} \sqrt{2} \sqrt{\kappa_0} \sqrt{T\lambda [\log \det(\lambda^{-1} \mathbf{H}_j + I) + 1]} \\
&= \sqrt{2} \sqrt{\kappa_0} \sqrt{T\lambda [\tilde{d}_j \log(1 + TK/\lambda) + 1]}.
\end{aligned} \tag{18}$$

Step (a) is obtained in the same way as steps (b) and (c) in (16) (Appendix B.4.3), i.e., we have made use of $\tilde{\sigma}_{t_p-1,j}^{\text{local}}(x) \leq \sqrt{\kappa_0}$ and assumed that $\kappa_0 \geq 1$. Again note that if $\kappa_0 < 1$, then the proof still goes through since $\tilde{\sigma}_{t-1,j}^{\text{local}}(x_{t,j}) \leq \min\{\tilde{\sigma}_{t-1,j}^{\text{local}}(x_{t,j}), \sqrt{\kappa_0}\} \leq \min\{\tilde{\sigma}_{t-1,j}^{\text{local}}(x_{t,j}), 1\}$, after which the only modification we need to make to the equation above is to remove the dependency on multiplicative term of $\sqrt{\kappa_0}$. Step (b) makes use of the Cauchy–Schwarz inequality. Step (c), similar to step (e) of (16), is derived following the proof of Lemma 4.8 of [47] (in Appendix B.7 of [47]). Step (d) follows from Lemma 6 and hence holds with probability of at least $1 - \delta_6$. In the last equality, we have simply plugged in the definition of \tilde{d}_j (Sec. 3).

Now we can plug (18) into (17) to obtain

$$\begin{aligned}
\sum_{i=1}^N \sum_{p \in \mathcal{E}^{\text{good}}} \sum_{t \in \mathcal{T}^{(p)}} \alpha_t 2\nu_{TK} \frac{1}{\sqrt{N}} \sum_{j=1}^N \tilde{\sigma}_{t_p,j}^{\text{local}}(x_{t,i}) \\
\leq 2\nu_{TK} \frac{1}{\sqrt{N}} \sum_{i=1}^N \sum_{j=1}^N \sqrt{2} \sqrt{\kappa_0} \sqrt{T\lambda [\tilde{d}_j \log(1 + TK/\lambda) + 1]} \\
= 2\sqrt{2} \nu_{TK} \sqrt{\kappa_0} \sqrt{N} \sum_{j=1}^N \sqrt{T\lambda [\tilde{d}_j \log(1 + TK/\lambda) + 1]}.
\end{aligned} \tag{19}$$

B.4.5 Putting Things Together

Finally, recall that our derived upper bound on $r_{t,i}$ in (14) contains three terms (the third term is simply an error term), and now we can make use of our derived upper bound on the first term (Appendix B.4.3) and the second term (Appendix B.4.4), summed over all agents and all good epochs, to obtain an upper bound on the total regrets incurred in all good epochs:

$$\begin{aligned}
R_T^{\text{good}} &= \sum_{i=1}^N \sum_{t \in \mathcal{T}^{\text{good}}} r_{t,i} \\
&\leq 2\sqrt{e} \nu_{TKN} \sqrt{\kappa_0} \sqrt{TN\lambda [\tilde{d} \log(1 + TNK/\lambda) + 1]} + \\
&\quad 2\sqrt{2} \nu_{TK} \sqrt{\kappa_0} \sqrt{N} \sum_{j=1}^N \sqrt{T\lambda [\tilde{d}_j \log(1 + TK/\lambda) + 1]} + TN\varepsilon_{\text{linear}}(m, T) \\
&= \tilde{O}\left(\sqrt{\tilde{d}} \sqrt{TN\tilde{d}} + \sqrt{\tilde{d}_{\max}} \sqrt{NN} \sqrt{T\tilde{d}_{\max}} + TN\varepsilon_{\text{linear}}(m, T)\right) \\
&= \tilde{O}\left(\tilde{d} \sqrt{TN} + \tilde{d}_{\max} N^{3/2} \sqrt{T} + TN\varepsilon_{\text{linear}}(m, T)\right).
\end{aligned} \tag{20}$$

In the second last equality, we have used $\nu_{TKN} = \tilde{O}(\sqrt{\tilde{d}})$ and $\nu_{TK} = \tilde{O}(\sqrt{\tilde{d}_{\max}})$.

B.5 Regret Upper Bound for Bad Epochs

In this section, we derive an upper bound on the total regrets from all bad epochs. To begin with, we firstly derive an upper bound on the total regrets of any bad epoch p denoted as $R^{[p]}$:

$$\begin{aligned}
R^{[p]} &= \sum_{i=1}^N \sum_{t=t_p}^{t_p+E_p-1} r_{t,i} \stackrel{(a)}{\leq} \sum_{i=1}^N \left(2 + 2 + \sum_{t=t_p+1}^{t_p+E_p-2} r_{t,i} \right) \\
&\stackrel{(b)}{\leq} \sum_{i=1}^N \left[4 + \sum_{t=t_p+1}^{t_p+E_p-2} (\text{UCB}_{t,i}^a(x_{t,i}^*) - h(x_{t,i})) \right] \\
&\stackrel{(c)}{\leq} \sum_{i=1}^N \left[4 + \sum_{t=t_p+1}^{t_p+E_p-2} (\text{UCB}_{t,i}^a(x_{t,i}) - h(x_{t,i})) \right] \\
&\stackrel{(d)}{\leq} \sum_{i=1}^N \left[4 + \sum_{t=t_p+1}^{t_p+E_p-2} 2\nu_{TKN} \sqrt{\lambda} \|g(x_{t,i}; \theta_0) / \sqrt{m}\|_{\bar{V}_{t,i}^{-1}} \right] \\
&\stackrel{(e)}{\leq} \sum_{i=1}^N \left(4 + 2\nu_{TKN} \sqrt{\kappa_0} \sum_{t=t_p}^{t_p+E_p-2} \min\{\sqrt{\lambda} \|g(x_{t,i}; \theta_0) / \sqrt{m}\|_{\bar{V}_{t,i}^{-1}}, 1\} \right) \\
&\stackrel{(f)}{\leq} \sum_{i=1}^N \left(4 + 2\nu_{TKN} \sqrt{\kappa_0 \lambda} \sum_{t=t_p}^{t_p+E_p-2} \min\{\|g(x_{t,i}; \theta_0) / \sqrt{m}\|_{\bar{V}_{t,i}^{-1}}, 1\} \right)
\end{aligned} \tag{21}$$

Step (a) follows from simply upper-bounding the regrets of the first and last iteration within this epoch by 2. Step (b) makes use of the validity of $\text{UCB}_{t,i}^a$ (Lemma 3). Step (c) follows because $\alpha_t = 0, \forall t \in [T] \setminus \{t_p\}_{p \in [P]}$ (i.e., we set $\alpha_t = 0$ except for the first iteration of all epochs), which implies that after the first iteration of an epoch, $x_{t,i}$ is selected by only maximizing $\text{UCB}_{t,i}^a$ (line 6 of Algo. 1). Step (d) again uses Lemma 3, as well as the expression of $\text{UCB}_{t,i}^a$. Step (e) is obtained in the same way as steps (b) and (c) in (16) (Appendix B.4.3). Specifically, since $\langle g(x; \theta_0), g(x; \theta_0) \rangle \leq \kappa_0$ (Appendix B.4.3), therefore, $\sqrt{\lambda} \|g(x_{t,i}; \theta_0) / \sqrt{m}\|_{\bar{V}_{t,i}^{-1}} \leq \sqrt{\kappa_0}$, which can be proved by following the same steps as (15). As a result, if we assume that $\kappa \geq 1$, then $\sqrt{\lambda} \|g(x_{t,i}; \theta_0) / \sqrt{m}\|_{\bar{V}_{t,i}^{-1}} = \min\{\sqrt{\lambda} \|g(x_{t,i}; \theta_0) / \sqrt{m}\|_{\bar{V}_{t,i}^{-1}}, \sqrt{\kappa_0}\} \leq \sqrt{\kappa_0} \min\{\sqrt{\lambda} \|g(x_{t,i}; \theta_0) / \sqrt{m}\|_{\bar{V}_{t,i}^{-1}}, 1\}$; in the other case where $\kappa_0 < 1$, then $\sqrt{\lambda} \|g(x_{t,i}; \theta_0) / \sqrt{m}\|_{\bar{V}_{t,i}^{-1}} = \min\{\sqrt{\lambda} \|g(x_{t,i}; \theta_0) / \sqrt{m}\|_{\bar{V}_{t,i}^{-1}}, \sqrt{\kappa_0}\} \leq \min\{\sqrt{\lambda} \|g(x_{t,i}; \theta_0) / \sqrt{m}\|_{\bar{V}_{t,i}^{-1}}, 1\}$. Here we have assumed $\kappa_0 \geq 1$ for simplicity, since when $\kappa_0 < 1$, the equation above still holds except that we can remove the dependency on $\sqrt{\kappa_0}$. Step (f) follows because $\lambda = 1 + 2/T > 1$.

Next, we derive an upper bound on the inner summation in (21).

$$\begin{aligned}
&\sum_{t=t_p}^{t_p+E_p-2} \min\{\|g(x_{t,i}; \theta_0) / \sqrt{m}\|_{\bar{V}_{t,i}^{-1}}, 1\} \\
&\stackrel{(a)}{\leq} \sqrt{(E_p - 1) \sum_{t=t_p}^{t_p+E_p-2} \min\{\|g(x_{t,i}; \theta_0) / \sqrt{m}\|_{\bar{V}_{t,i}^{-1}}^2, 1\}} \\
&\stackrel{(b)}{\leq} \sqrt{(E_p - 1) 2 \log \frac{\det \bar{V}_{t_p+E_p-2,i}}{\det \bar{V}_{t_p,i}}} \\
&\stackrel{(c)}{\leq} \sqrt{2((t_p + E_p - 2) - t_{\text{last}}) \log \frac{\det V_{t_p+E_p-2,i}}{\det V_{\text{last}}}} \\
&\stackrel{(d)}{\leq} \sqrt{2D}.
\end{aligned} \tag{22}$$

Step (a) follows from the Cauchy–Schwarz inequality. Step (b) makes use of Lemma 11 of [1]. In step (c), we used the notations of $t_{\text{last}} = t_p - 1$, $\bar{V}_{t_p, i} = V_{\text{last}}$ (this is because in the first iteration t_p of an epoch, $W_{\text{new}, i} = \mathbf{0}_{p_0 \times p_0}$ and hence $\bar{V}_{t_p, i} = V_{\text{last}} = W_{\text{sync}} + \lambda I$), and $V_{t_p + E_p - 2, i} = \bar{V}_{t_p + E_p - 2, i} + g(x_{t, i}; \theta_0)g(x_{t, i}; \theta_0)^\top / m$, and also used $\det \bar{V}_{t_p + E_p - 2, i} \leq \det V_{t_p + E_p - 2, i}$. To understand step (d), note that the term in step (c): $((t_p + E_p - 2) - t_{\text{last}}) \log \frac{\det V_{t_p + E_p - 2, i}}{\det V_{\text{last}}}$ is exactly the criterion we use to check whether to start a communication round in iteration $t = t_p + E_p - 2$ (line 10 of Algo. 1). Since $t = t_p + E_p - 2$ is not the last iteration in this epoch (i.e., we did not start a communication round after checking this criterion in iteration $t = t_p + E_p - 2$), therefore, this criterion is not satisfied in iteration $t = t_p + E_p - 2$, i.e., $((t_p + E_p - 2) - t_{\text{last}}) \log \frac{\det V_{t_p + E_p - 2, i}}{\det V_{\text{last}}} \leq D$, which explains step (d).

Next, we can plug (22) into (21) to obtain:

$$R^{[p]} = \sum_{i=1}^N \sum_{t=t_p}^{t_p + E_p - 1} r_{t, i} \leq \sum_{i=1}^N \left(4 + 2\nu_{TKN} \sqrt{\kappa_0 \lambda} \sqrt{2D}\right) = \left(4 + 2\nu_{TKN} \sqrt{2\kappa_0 \lambda D}\right) N, \quad (23)$$

which gives an upper bound on the total regret from *any* bad epoch. Now recall that as we have discussed in Sec. B.2, there are no more than R bad epochs (with probability of at least $1 - \delta_1$). Therefore, the total regret of *all* bad epochs can be upper-bounded by:

$$\begin{aligned} R_T^{\text{bad}} &\leq R \left(4 + 2\nu_{TKN} \sqrt{2\kappa_0 \lambda D}\right) N \\ &\leq \left(\tilde{d} \log(1 + TKN/\lambda) + 1\right) \left(4 + 2\nu_{TKN} \sqrt{2\kappa_0 \lambda D}\right) N \\ &= \tilde{O}\left(\tilde{d} \sqrt{\tilde{d}} \sqrt{DN}\right) \\ &= \tilde{O}\left((\tilde{d})^{3/2} \sqrt{DN}\right). \end{aligned} \quad (24)$$

In the second last equality, we have used $\nu_{TKN} = \tilde{O}(\sqrt{\tilde{d}})$. By choosing $D = \mathcal{O}\left(\frac{T}{N\tilde{d}}\right)$ (line 1 of Algo. 1), we can further express the above upper bound on the total regrets from all bad epochs as:

$$\begin{aligned} R_T^{\text{bad}} &= \mathcal{O}\left(\sqrt{\frac{T}{N\tilde{d}}} (\tilde{d})^{3/2} N\right) \\ &= \mathcal{O}\left(\tilde{d} \sqrt{TN}\right). \end{aligned} \quad (25)$$

B.6 Final Regret Upper Bound

Here we derive an upper bound on the total cumulative regret by adding up the regrets resulting from all good epochs (Appendix B.4) and all bad epochs (Appendix B.5):

$$\begin{aligned} R_T &= R_T^{\text{good}} + R_T^{\text{bad}} \\ &= \tilde{O}\left(\tilde{d} \sqrt{TN} + \tilde{d}_{\max} N^{3/2} \sqrt{T} + TN \varepsilon_{\text{linear}}(m, T) + \tilde{d} \sqrt{TN}\right) \\ &= \tilde{O}\left(\tilde{d} \sqrt{TN} + \tilde{d}_{\max} N^{3/2} \sqrt{T} + TN \varepsilon_{\text{linear}}(m, T)\right). \end{aligned} \quad (26)$$

This regret upper bound holds with probability of at least $1 - \delta_1 - \delta_2 - \delta_3 - \delta_4 - \delta_5 - \delta_6$. We let $\delta_3 = \delta_4 = \delta/3$, which leads to the expressions of ν_{TKN} and ν_{TK} given in the main paper (Sec. 4). We let $\delta_1 = \delta_2 = \delta_5 = \delta_6 = \delta/12$, and this will only introduce an additional factor of $\log 12$ in the first three conditions on m in Appendix B.1 which can be absorbed by the constant C .

Next, the last term from the upper bound in (26) can be further written as:

$$\begin{aligned} TN \varepsilon_{\text{linear}}(m, T) &= TN \left(\varepsilon_{\text{linear}, 1}(m, T) + \varepsilon_{\text{linear}, 2}(m, T) + \varepsilon_{\eta, J}\right) \\ &= TNC_1 T^{2/3} m^{-1/6} \lambda^{-2/3} L^3 \sqrt{\log m} + TNC_3 m^{-1/6} \sqrt{\log m} L^4 T^{5/3} \lambda^{-5/3} (1 + \sqrt{T/\lambda}) \\ &\quad + TNC_2 (1 - \eta m \lambda)^J \sqrt{TL/\lambda} \end{aligned} \quad (27)$$

It can be easily verified that as long as $m(\log m)^{-3} \geq 3^6 C_1^6 T^{10} N^6 \lambda^{-4} L^{18}$ and $m(\log m)^{-3} \geq 3^6 C_3^6 T^{16} N^6 L^{24} \lambda^{-10} (1 + \sqrt{T/\lambda})^6$ (which are ensured by conditions 5 and 6 on m in Appendix B.1), then the first and second terms in (27) can both be upper-bounded by $1/3$. Moreover, if the conditions on η and J presented in Appendix B.1 are satisfied, i.e., if we choose the learning rate as $\eta = C_4(m\lambda + mTL)^{-1}$ in which $C_4 > 0$ is an absolute constant such that $C_4 \leq 1 + TL$, and choose $J = \frac{1}{C_4} \left(1 + \frac{TL}{\lambda}\right) \log\left(\frac{1}{3C_2N} \sqrt{\frac{\lambda}{T^3L}}\right) = \tilde{O}(TL/(\lambda C_4))$, then the third term in (27) can also be upper-bounded by $1/3$.

As a result, the last term from the upper bound in (26) can be upper-bounded by 1, and hence the regret upper bound becomes:

$$R_T = \tilde{O}\left(\tilde{d}\sqrt{TN} + \tilde{d}_{\max}N^{3/2}\sqrt{T}\right). \quad (28)$$

Regret Upper Bound in Terms of the Maximum Information Gain γ . Next, we perform some further analysis of the final regret upper bound derived above, which allows us to inspect the order of growth of our regret upper bound in the worst-case scenario. We have defined in Sec. 3 that $\tilde{d} \leq 2\gamma_{TKN}/\log(1 + TKN/\lambda)$, $\tilde{d}_i \leq 2\gamma_{TK}/\log(1 + TK/\lambda)$, $\forall i \in [N]$ and $\tilde{d}_{\max} = \max_{i \in [N]} \tilde{d}$. As a result, in our derivations in (20) and (24), we can replace $\tilde{d}\log(1 + TKN/\lambda)$ by $2\gamma_{TKN}$ and replace $\tilde{d}_j \log(1 + TK/\lambda)$ by $2\gamma_{TK}$, after which the regret upper bound becomes

$$R_T = \tilde{O}\left(\gamma_{TKN}\sqrt{TN} + \gamma_{TK}N^{3/2}\sqrt{T}\right). \quad (29)$$

The growth rate of the maximum information gain of NTK has been characterized by previous works: $\gamma_T = \tilde{O}(T^{\frac{d-1}{d}})$ [24, 42]. This implies that our regret upper bound can be further expressed as

$$R_T = \tilde{O}\left(K^{\frac{(d-1)}{d}}(TN)^{\frac{3d-2}{2d}} + K^{\frac{(d-1)}{d}}T^{\frac{3d-2}{2d}}N^{3/2}\right) = \tilde{O}\left(K^{\frac{(d-1)}{d}}T^{\frac{3d-2}{2d}}N^{3/2}\right).$$

When the input dimension is $d = 1$, this regret upper bound is sub-linear: $R_T = \tilde{O}(T^{1/2}N^{3/2})$.

C Proof of Upper Bound on Communication Complexity (Theorem 2)

In this section, we derive an upper bound on the communication complexity (i.e., the total number of communication rounds) of our FN-UCB algorithm. Define $\zeta \triangleq \sqrt{DT}/R$. An immediate implication is that there can be at most $\lceil T/\zeta \rceil$ epochs whose length is larger than ζ . Next, we try to derive an upper bound on the number of epochs whose length is smaller than ζ .

Note that if an epoch p contains less than ζ iterations, then because of our criterion to start a communication round (line 10 of Algo. 1), we have that $\log \frac{\det V_p}{\det V_{p-1}} > \frac{D}{\zeta}$. Also recall that equation (4) (Appendix B.2) tells us that:

$$\sum_{p=0}^{P-1} \log \frac{\det V_{p+1}}{\det V_p} \leq R' \leq R, \quad (30)$$

with probability of at least $1 - \delta_1 \geq 1 - \delta$. Therefore, there can be at most $\lceil \frac{R\zeta}{D} \rceil = \lceil \frac{R\zeta}{D} \rceil$ such epochs whose length is smaller than ζ . As a result, the total number of epochs can be upper-bounded by:

$$\lceil T/\zeta \rceil + \lceil \frac{R\zeta}{D} \rceil = \mathcal{O}\left(\sqrt{\frac{TR}{D}}\right). \quad (31)$$

Recall that $R = \tilde{O}(\tilde{d})$ (Appendix B.2). Therefore, with probability of at least $1 - \delta_1 \geq 1 - \delta$, the total number of epochs can be upper-bounded by $\tilde{O}\left(\sqrt{\frac{T\tilde{d}}{D}}\right)$.

Since we have chosen $D = \tilde{O}\left(\frac{T}{N\tilde{d}}\right)$ (line 1 of Algo. 1), therefore, the total number of epochs can be upper-bounded by $\tilde{O}\left(\sqrt{\frac{T\tilde{d}}{\frac{T}{N\tilde{d}}}}\right) = \tilde{O}(\tilde{d}\sqrt{N}) = \tilde{O}\left(\gamma_{TKN}\sqrt{N}\right)$, which is sub-linear in T since $\gamma_{TKN} = \tilde{O}\left((TKN)^{\frac{d-1}{d}}\right)$. The proof here, and hence Theorem 2, makes use of Lemma 1. Therefore, we only need condition 1 on m listed in Appendix B.1 to hold, and do not require any condition on η and J .

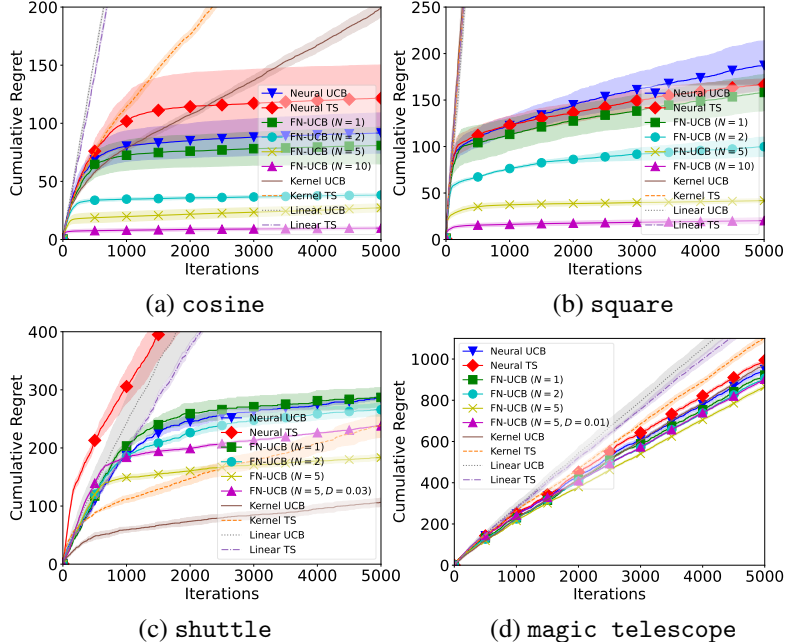


Figure 3: Cumulative regrets for the (a) cosine, (b) square, (c) shuttle (with diagonalization), and (d) magic telescope experiments, with additional comparisons with Linear UCB, Linear TS, Kernel UCB and Kernel TS.

D More Experimental Details

Some of the experimental details (e.g., the number of layers and the width m of the NN used in every experiment) are already described in the main text (Sec. 6). Following the works of [47, 48], when training the NN (line 13 of Algo. 1) for agent i , we use the NN parameters resulting from the last gradient descent training of agent i (instead of θ_0) as the initial parameters, in order to accelerate the training procedure. Every time we train an NN, we use stochastic gradient descent to train the NN for 30 iterations with a learning rate of 0.01. To save computational cost, we stop training the NNs after 2000 iterations, i.e., after 2000 iterations, all NN parameters are no longer updated. Also to reduce the computational cost, when checking the criterion in line 10 of Algo. 1, we diagonalize (i.e., only keep the diagonal elements of) the two matrices for which we need to calculate the determinant. Our experiments are run on a server with 96 CPUs, an NVIDIA A100 GPU with a memory of 40GB, a RAM of 256GB, running the Ubuntu system.

The shuttle dataset is publicly available at [https://archive.ics.uci.edu/ml/datasets/Statlog+\(Shuttle\)](https://archive.ics.uci.edu/ml/datasets/Statlog+(Shuttle)) and contains no personally identifiable information or offensive content. It includes 58000 instances, has an input dimension of $d = 9$ and contains $K = 7$ classes/arms. As a result, according to the way in which the contexts are constructed (Sec. 6.2), every context feature vector has a dimension of $9 \times 7 = 63$. The magic telescope dataset is publicly available at <https://archive.ics.uci.edu/ml/datasets/magic+gamma+telescope> and contains no personally identifiable information or offensive content. The dataset contains 19020 instances, has an input dimension of $d = 10$ and $K = 2$ classes/arms. As a result, every context feature vector has a dimension of $10 \times 2 = 20$.

When comparing with Linear-UCB, Linear TS, Kernelized UCB and Kernelized TS, we follow the work of [47] to set $\lambda = 1$ and perform a grid search within $\nu \in \{1, 0.1, 0.01\}$. The results showing comparisons with these algorithms, for both the synthetic experiments (Sec. 6.1) and real-world experiments (Sec. 6.2), are presented in Fig. 3. The figures show that both linear and kernelized contextual bandit algorithms are outperformed by neural contextual bandit algorithms, which is consistent with the observations from [47, 48].