Variance-Aware Linear UCB with Deep Representation for Neural Contextual Bandits

Ha Manh Bui **Enrique Mallada** **Anqi Liu**
Johns Hopkins University, Baltimore, MD, U.S.A.

Abstract

By leveraging the representation power of deep neural networks, neural upper confidence bound (UCB) algorithms have shown success in contextual bandits. To further balance the exploration and exploitation, we propose Neural-σ^2-LinearUCB, a variance-aware algorithm that utilizes σ_t^2, i.e., an upper bound of the reward noise variance at round t, to enhance the uncertainty quantification quality of the UCB, resulting in a regret performance improvement. We provide an oracle version for our algorithm characterized by an oracle variance upper bound σ_t^2 and a practical version with a novel estimation for this variance bound. Theoretically, we provide rigorous regret analysis for both versions and prove that our oracle algorithm achieves a better regret guarantee than other neural-UCB algorithms in the neural contextual bandits setting. Empirically, our practical method enjoys a similar computational efficiency, while outperforming state-of-the-art techniques by having a better calibration and lower regret across multiple standard settings, including on the synthetic, UCI, MNIST, and CIFAR-10 datasets.

1 Introduction

The stochastic multi-armed contextual bandits is a sequential decision-making problem that is related to various real-world applications, e.g., healthcare, finance, recommendation, etc. Specifically, this setting considers the interaction between an agent and an environment. In each round, the agent receives a context from the environment and then decides based on a finite arm set. After each decision, the agent receives a reward and its goal is to maximize the cumulative reward over rounds (Sutton and Barto, 2018).

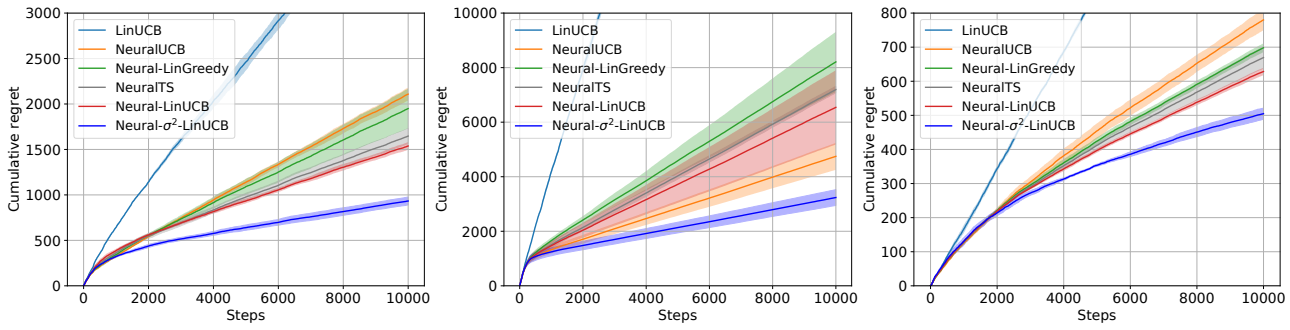
Method	High UCB quality with σ_t^2	Neural-regret analysis	Empirical efficiency
NeuralUCB	✗	✓	✗
Neural-LinUCB	✗	✓	✓
Variance-aware-UCB	✓	✗	✗
Ours	✓	✓	✓

Table 1: Contribution comparison between methods in utilizing variance bound σ_t^2 to improve UCB uncertainty quality, neural-regret analysis, and empirical efficiency.

To balance the exploration and exploitation, several algorithms for this setting have been proposed (Lattimore and Szepesvári, 2020; Bubeck and Cesa-Bianchi, 2012). Among these methods, based on the principle of Optimism in the Face of Uncertainty (OFUL) and the power of Deep Neural Networks (DNN), Neural Upper Confidence Bound (NeuralUCB) (Zhou et al., 2020) and Neural Linear Upper Confidence Bound (Neural-LinUCB) (Xu et al., 2022a) have become the most practical and are the State-of-the-art (SOTA) techniques. Specifically, NeuralUCB is a natural extension of Linear Upper Confidence Bound (LinUCB) (Li et al., 2010; Chu et al., 2011), which uses a DNN-based random feature mapping to approximate the underlying reward function. Yet, it is computationally inefficient since the Upper Confidence Bound (UCB) is performed over the entire DNN parameter space. Neural-LinUCB improves the efficiency by learning a mapping that transforms the raw context input into feature vectors using a DNN, and then performing a UCB exploration over the linear output layer of the network. These methods achieve $\tilde{\mathcal{O}}(Rd\sqrt{T})$ regret upper bound, where R is the upper bound of the absolute value of the reward noise, d is the feature context dimension, and T is the learning time horizon. This is equivalent to the result of LinUCB in the linear contextual bandits setting (Abbasi-yadkori et al., 2011).

The predictive uncertainty of the UCB, especially when derived from modern DNN, however, can be inaccu-

Proceedings of the 28th International Conference on Artificial Intelligence and Statistics (AISTATS) 2025, Mai Khao, Thailand. PMLR: Volume 258.



(a) $h_1(\mathbf{x}) = 10(\mathbf{x}^\top \boldsymbol{\theta})^2$ (b) $h_2(\mathbf{x}) = \mathbf{x}^\top \boldsymbol{\theta}^\top \boldsymbol{\theta} \mathbf{x}$ (c) $h_3(\mathbf{x}) = \cos(3\mathbf{x}^\top \boldsymbol{\theta})$

Figure 1: Cumulative regret results on the synthetic data across 10 runs with different seeds. More baselines for comparison and a zoom-in figure are in Fig. 17. *A short demo is available at this Google Colab link.*

rate and impose a bottleneck on the regret performance (Kuleshov et al., 2018). To tackle this challenge, the idea of improving UCB uncertainty estimation quality to enhance regret performance has shown promising results (Kuleshov and Precup, 2014; Auer et al., 2002). Notably, Malik et al. (2019); Deshpande et al. (2024) have shown that calibrated neural-UCB algorithms can result in a lower cumulative regret. Yet, they require a post-hoc re-calibration step on additional hold-out data for every round, leading to inefficiency in practice. Theoretically, in the linear contextual bandits setting, recent works have shown that variance-aware-UCB algorithms (Zhou et al., 2021; Zhao et al., 2023), i.e., using the reward noise variance to improve uncertainty estimation quality of UCB, can further achieve a tighter regret bound than LinUCB. However, even with this non-neural-network approach, estimating the true variance is non-trivial, and such algorithms are often not practically feasible. As a result, there are usually no experimental results shown in the previous literature for this variance-aware-UCB domain.

Therefore, towards a variance-aware neural-UCB algorithm that is both rigorous and practical, we propose Neural Variance-Aware Linear Upper Confidence Bound (Neural-σ^2-LinUCB). Since estimating the true variance with DNN is challenging, Neural-σ^2-LinUCB leverages σ_t^2, i.e., the upper bound of the reward noise variance at round t, to enhance the uncertainty quantification quality of the UCB, resulting in a regret performance improvement. We propose two versions, including an oracle version that uses a given variance upper bound σ_t^2 and a practical version that estimates this variance bound. We formally provide regret guarantees for both versions and prove our oracle version achieves a tighter regret guarantee with DNN than other neural-UCB bandits. Succinctly, for each round, our practical version calculates the upper bound of the reward noise variance by using the reward range and the estimated reward mean with DNN. Then, we use this variance-bound information to optimize the linear reward model w.r.t. encoded DNN context features by using a weighted ridge regression minimizer.

The key ideas of this approach are: (1) UCB is performed over the feature representation from the last DNN layer. Therefore, it enjoys computational efficiency of Neural-LinUCB; (2) When σ_t^2 is large, our UCB will be more uncertain, and vice versa. This intuitively helps improve uncertainty estimation quality of UCB, resulting in a better regret guarantee.

Our theoretical and practical contributions are summarized in Tab. 1 and are as follows:

- We propose Neural-σ^2-LinUCB, a variance-aware algorithm that utilizes σ_t^2 to enhance the exploration-exploitation quality of UCB. We provide an oracle and a practical version. The oracle algorithm assumes knowledge on σ_t^2. The practical algorithm estimates σ_t^2 from the reward range and the reward mean estimator with DNN.
- We prove the regret of our practical version is at most $\tilde{\mathcal{O}}\left(R\sqrt{dT} + d\sqrt{\sum_{t=1}^{T} \sigma_t^2 + \epsilon}\right)$, where ϵ is the estimation error of σ_t^2. Notably, our oracle version achieves $\tilde{\mathcal{O}}\left(R\sqrt{dT} + d\sqrt{\sum_{t=1}^{T} \sigma_t^2}\right)$ regret bound. Since our setting considers $\sigma_t \leq R$, this is strictly better than $\tilde{\mathcal{O}}\left(Rd\sqrt{T}\right)$ of Neural-LinUCB.
- We empirically show our proposed method enjoys a similar computational efficiency while outperforming SOTA techniques by having a better calibration and lower regret across multiple contextual bandits settings, including on the synthetic, UCI, MNIST, and CIFAR-10 datasets (e.g., Fig. 1).

2 Background

Notation. We denote $[k]$ is a set $\{1, \cdots, k\}$, $k \in \mathbb{N}$. For a semi-definite matrix $\mathbf{A} \in \mathbb{R}^{d \times d}$ and a vector

$\mathbf{x} \in \mathbb{R}^d$, let $\|\mathbf{x}\|_{\mathbf{A}} = \sqrt{\mathbf{x}^\top \mathbf{A}\mathbf{x}}$ be the Mahalanobis norm. For a complexity $\mathcal{O}(T)$, let us use $\tilde{\mathcal{O}}(T)$ to hide the constant and logarithmic dependence of T. We also use $\mathcal{N}(\cdot)$ to denote the Gaussian distribution and $\mathbb{U}(\cdot)$ for the Uniform distribution.

2.1 Problem setting

In the stochastic K-armed contextual bandits (Lattimore and Szepesvári, 2020), at each round $t \in [T]$, the learning agent observes a context consisting of K feature vectors $\{\mathbf{x}_{t,a} \in \mathbb{R}^d \mid a \in [K]\}$ from the environment, then selects an arm $a_t \in [K]$ based on this context, and receives a corresponding reward r_{t,a_t}. The agent aims to maximize its expected total reward over these T rounds, i.e., minimizing the pseudo-regret

$$\text{Regret}(T) = \mathbb{E}\left[\sum_{t=1}^{T}\left(r_{t,a_t^*} - r_{t,a_t}\right)\right], \tag{1}$$

where $a_t^* = \arg\max_{a\in[K]} \mathbb{E}\left[r_{t,a}\right]$. Following Zhou et al. (2020); Xu et al. (2022a), for any round t, we assume the reward generation, defined as follows

$$r_{t,a_t} = h(\mathbf{x}_{t,a_t}) + \xi_t, \tag{2}$$

where h is an unknown function s.t. $0 \le h(\mathbf{x}) \le 1, \forall \mathbf{x}$. In terms of the reward noise, following Zhou et al. (2021); Abbasi-yadkori et al. (2011); Zhao et al. (2023), we assume ξ_t is a random noise variable that satisfies the following conditions

$$\begin{aligned} &p(|\xi_t| \le R) = 1, \quad \mathbb{E}[\xi_t \mid \mathbf{x}_{1:t,a_{1:t}}, \xi_{1:t-1}] = 0, \\ &\mathbb{E}[\xi_t^2 \mid \mathbf{x}_{1:t,a_{1:t}}, \xi_{1:t-1}] \le \sigma_t^2 \le R^2. \end{aligned} \tag{3}$$

2.2 Neural Linear Upper Confidence Bound

To relax the strong linear-reward assumption, we consider a setting that the unknown function h can be non-linear. Our work builds on Neural-LinUCB (Xu et al., 2022a), which seeks to extend LinUCB by leveraging the approximating power of DNN. In particular, for a neural network

$$f(\mathbf{x}; \boldsymbol{\theta}^*, \mathbf{w}) = \sqrt{m}\boldsymbol{\theta}^{*\top} g_L(\mathbf{W}_L g_{L-1}(\mathbf{W}_{L-1} \cdots g_1(\mathbf{W}_1 \mathbf{x}))), \tag{4}$$

where $\mathbf{x} \in \mathbb{R}^d$ is the input data, $\boldsymbol{\theta}^* \in \mathbb{R}^d$ is the weight vector of the output layer, $\mathbf{w} = (Vec(\mathbf{W}_1)^\top, \cdots, Vec(\mathbf{W}_L)^\top)^\top$, $\mathbf{W}_l \in \mathbb{R}^{m_l \times m_{l-1}}$ is the weight matrix of the l-th layer, $l \in [L]$, and $g_l = g$ is the ReLU activation function, i.e., $g(x) = \max\{0, x\}$ for $x \in \mathbb{R}$. By further assuming that $m_1 = \cdots = m_{L-1} = m$, $m_0 = m_L = d$, one can readily show that the dimension p of vector $\mathbf{w}$ satisfies $p = (L-2)m^2 + 2md$ and the output of the L-th hidden layer of neural network f becomes

$$\boldsymbol{\phi}(\mathbf{x}; \mathbf{w}) = \sqrt{m} g(\mathbf{W}_L g(\mathbf{W}_{L-1} \cdots g(\mathbf{W}_1 \mathbf{x}))). \tag{5}$$

Then, at round t, the agent model chooses the action that maximizing the UCB as follows

$$a_t = \arg\max_{k\in[K]} \{\langle \boldsymbol{\phi}(\mathbf{x}_{t,k}; \mathbf{w}_{t-1}), \boldsymbol{\theta}_{t-1}\rangle + \alpha_t \|\boldsymbol{\phi}(\mathbf{x}_{t,k}; \mathbf{w}_{t-1})\|_{\mathbf{A}_{t-1}^{-1}}\}, \tag{6}$$

where the output layer weights $\boldsymbol{\theta}_{t-1}$ is updated by using the same ridge regression as in linear contextual bandits (Abbasi-yadkori et al., 2011), i.e., we consider $\boldsymbol{\theta}_t = \mathbf{A}_t^{-1}\mathbf{b}_t$, with

$$\begin{aligned} \mathbf{A}_t &= \lambda\mathbf{I} + \sum_{i=1}^{t} \boldsymbol{\phi}(\mathbf{x}_{i,a_i}; \mathbf{w}_{i-1})\boldsymbol{\phi}(\mathbf{x}_{i,a_i}; \mathbf{w}_{i-1})^\top, \\ \mathbf{b}_t &= \sum_{i=1}^{t} r_{i,a_i}\boldsymbol{\phi}(\mathbf{x}_{i,a_i}; \mathbf{w}_{i-1}). \end{aligned} \tag{7}$$

Finally, the DNN model weights $\mathbf{w}$ are optimized every H time steps, i.e., at times $t = qH$, with $q = 1, 2, \cdots$, following the Empirical risk minimization algorithm with the Mean Square Error (MSE) loss function

$$\mathcal{L}_q(\mathbf{w}) = \sum_{i=1}^{qH} \left(\boldsymbol{\theta}_t^\top \boldsymbol{\phi}(\mathbf{x}_{i,a_i}; \mathbf{w}) - r_{i,a_i}\right)^2. \tag{8}$$

By using NTK (Jacot et al., 2018), Neural-LinUCB is proven to achieve $\tilde{\mathcal{O}}(Rd\sqrt{T}) + \tilde{\mathcal{O}}\left(m^{-1/6}T\sqrt{(\mathbf{r}-\tilde{\mathbf{r}})^\top \mathbf{H}^{-1}(\mathbf{r}-\tilde{\mathbf{r}})}\right)$ regret (Xu et al., 2022a), where the first term resembles the regret bound of LinUCB (Abbasi-yadkori et al., 2011). Meanwhile, the second term depends on the estimation error of the neural network f for the reward-generating function $\mathbf{r}$, its estimation $\tilde{\mathbf{r}}$, and the NTK matrix $\mathbf{H}$ (we relegate to Sec. 4 for the precise definition of $\mathbf{r}$, $\tilde{\mathbf{r}}$, and $\mathbf{H}$). Following the assumption that $\mathbf{r}^\top \mathbf{H}^{-1}\mathbf{r}$ can be upper bounded by a constant (can be bounded by the RKHS norm of $\mathbf{r}$ if it belongs to the RKHS induced by $\mathbf{H}$), i.e., $\|\mathbf{r} - \tilde{\mathbf{r}}\|_{\mathbf{H}^{-1}} = \mathcal{O}(1)$ (Zhou et al., 2020), and by a selection of $m \ge T^3$ (Xu et al., 2022a), then the final regret of Neural-LinUCB becomes $\tilde{\mathcal{O}}(Rd\sqrt{T})$.

3 Neural Variance-Aware Linear Upper Confidence Bound Algorithm

3.1 Oracle algorithm

To improve the UCB quality and regret guarantee in the non-linear contextual bandits, we propose the oracle Neural-σ^2-LinUCB. The main idea of our method is using the high-quality feature representation $\boldsymbol{\phi}(\mathbf{x}_{t,a_t}; \mathbf{w})$ to estimate the mean reward $\langle \boldsymbol{\theta}, \boldsymbol{\phi}(\mathbf{x}_{t,a_t}; \mathbf{w})\rangle$ and the upper bound of the reward noise variance σ_t^2 per round t. Then, based on the OFUL, we make use of this variance upper bound information to optimize the linear model $\boldsymbol{\theta}$ w.r.t. encoded DNN context feature $\boldsymbol{\phi}(\mathbf{x}_{t,a_t}; \mathbf{w})$ by minimizing to the weighted ridge regression objective function as follows

$$\boldsymbol{\theta}_t = \arg\min_{\boldsymbol{\theta}\in\mathbb{R}^d} \lambda||\boldsymbol{\theta}||_2^2 + \sum_{i=1}^{t} \frac{[\langle \boldsymbol{\theta}, \boldsymbol{\phi}(\mathbf{x}_{i,a_i}; \mathbf{w}_{i-1})\rangle - r_{i,a_i}]^2}{\bar{\sigma}_i^2}, \tag{9}$$

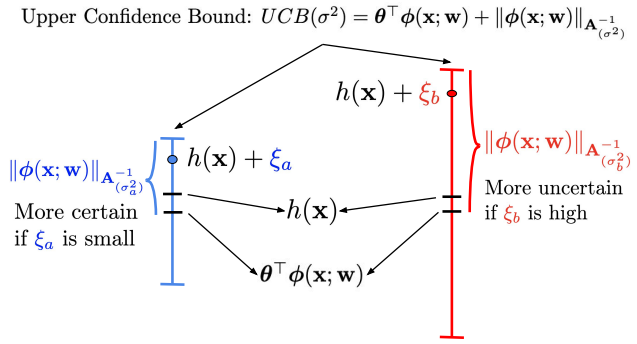


Figure 2: Our Neural-σ^2-LinUCB can be more uncertain (i.e., more exploration) if the reward noise $Var(\xi_t)$ is high, and more certain (i.e., more exploitation) if $Var(\xi_t)$ is small.

where $\bar{\sigma}_i = \max\{R/\sqrt{d}, \sigma_i\}$. Therefore, by computing the optimality conditions of Eq. 9, it follows that $\boldsymbol{\theta}_t = \mathbf{A}_t^{-1}\mathbf{b}_t$, where $\mathbf{A}_t$, which depends on the historical context-arm pairs, and the bias term $\mathbf{b}_t$ are given by

$$\begin{aligned} \mathbf{A}_t &= \lambda\mathbf{I} + \sum_{i=1}^{t} \frac{\boldsymbol{\phi}(\mathbf{x}_{i,a_i};\mathbf{w}_{i-1})\boldsymbol{\phi}(\mathbf{x}_{i,a_i};\mathbf{w}_{i-1})^\top}{\bar{\sigma}_i^2}, \\ \mathbf{b}_t &= \sum_{i=1}^{t} \frac{r_{i,a_i}\boldsymbol{\phi}(\mathbf{x}_{i,a_i};\mathbf{w}_{i-1})}{\bar{\sigma}_i^2}. \end{aligned} \tag{10}$$

The pseudo-code for Neural-σ^2-LinUCB is presented in Alg. 1. From the solution of $\mathbf{A}_t$ and b_t above, we can see that our feature matrix $\mathbf{A}_t$ is weighted by the proxy $\bar{\sigma}_i^2$ of the reward variance upper bound σ_i^2.

Remark 3.1. (Algorithmic comparison between Neural-σ^2-LinUCB and Neural-LinUCB). Consider the confidence set $\mathcal{E}_t = \{\boldsymbol{\theta} \in \mathbb{R}^d : ||\boldsymbol{\theta} - \boldsymbol{\theta}_t||^2_{\mathbf{A}_t}\}$, which is an ellipsoid centred at $\boldsymbol{\theta}_t$ and with principle axis being the eigenvectors of $\mathbf{A}_t$ with corresponding lengths being the reciprocal of the eigenvalues. Compare our $\mathbf{A}_t$ solution in Eq. 10 versus the solution in Eq. 7, we can see that when t grows, the matrix $\mathbf{A}_t$ in Eq. 7 has increasing eigenvalues, which means the volume of the ellipse is also frequently shrinking. Meanwhile, our $\mathbf{A}_t$ solution in Eq. 10 is more flexible by depending on the variance bound σ_t^2. This means that the volume of the ellipse will shrink not too fast if σ_t^2 is high, and not too slow if σ_t^2 is small, suggesting an exploration and exploitation improvement of the UCB.

At a high level, our oracle algorithm can be seen as a combination of Weighted OFUL and Neural-LinUCB. Yet, its challenges include: (1) It is unclear whether this can bring out a tighter regret bound than Neural-LinUCB; (2) It assumes we are given σ_t^2 at round t while σ_t^2 is often unavailable and is an unknown quantity in practice. Hence, we address the challenge (1) in Thm. 4.5 in Sec. 4. Regarding challenge (2), we next propose a novel practical version to estimate σ_t^2.

3.2 Practical algorithm

Since estimating the uncertainty of the true variance $Var(\xi_t)$ can be unreliable, especially when derived from DNN (Kuleshov et al., 2018; Malik et al., 2019), we instead estimate the variance bound σ_t^2. As illustrated in Fig. 2, our Alg. 1 intuitively means when the reward noise $Var(\xi_t)$ is high, σ_t^2 will be high, yielding a high $\mathbf{A}_t^{-1}$, i.e., more uncertainty for UCB, and vice versa. This suggests a better UCB uncertainty quantification, resulting in a better regret performance.

Recall σ_t^2 in Eq. 3 is the upper bound of the reward noise variance and is bounded by the magnitude R^2, i.e., $\mathbb{E}[\xi_t^2 \mid \mathbf{x}_{1:t,a_{1:t}}, \xi_{1:t-1}] \leq \sigma_t^2 \leq R^2$. Therefore, to estimate σ_t^2 to satisfy this condition, firstly, by the definition of the reward function in Eq. 2 and the reward noise in Eq. 3, we can trivially derive to obtain the form of the mean and the variance of the reward by the theorem as follows:

Theorem 3.2. *The reward r.v. r_{t,a_t} in Eq. 2 has the true mean $\mathbb{E}[r_{t,a_t}] = h(\mathbf{x}_{t,a_t})$ and variance $Var(r_{t,a_t}) = \mathbb{E}[\xi_t^2 \mid \mathbf{x}_{1:t,a_{1:t}}, \xi_{1:t-1}]$. The proof is in Apd. A.1.*

By the mean and variance formulation in Thm. 3.2, we can calculate the upper bound of the reward noise variance σ_t^2 at round t by the following theorem:

Theorem 3.3. *If the reward r.v. r_{t,a_t} is restricted to $[a, b]$ and we know the mean $h(\mathbf{x}_{t,a_t})$, then the variance is bounded by*

$$\mathbb{E}[\xi_t^2 \mid \mathbf{x}_{1:t,a_{1:t}}, \xi_{1:t-1}] \leq \sigma_t^2 := (b - h(\mathbf{x}_{t,a_t}))(h(\mathbf{x}_{t,a_t}) - a) \leq R^2.$$

The proof is in Apd. A.2.

From Thm. 3.3, we can see that given a reward range $[a, b]$, at round t, we can achieve a tighter upper bound of the reward noise variance than R^2. Hence, based on the estimation of the mean, i.e., $\boldsymbol{\theta}_{t-1}^\top \boldsymbol{\phi}(\mathbf{x}_{t,a_t})$, we can obtain the variance bound by calculating

$$\hat{\sigma}_t^2 = (b - \boldsymbol{\theta}_{t-1}^\top \boldsymbol{\phi}(\mathbf{x}_{t,a_t};\mathbf{w}_{t-1}))(\boldsymbol{\theta}_{t-1}^\top \boldsymbol{\phi}(\mathbf{x}_{t,a_t};\mathbf{w}_{t-1}) - a). \tag{11}$$

The efficient estimation $\hat{\sigma}_t^2$ in Eq. 11 is based on the estimation of the reward mean $\boldsymbol{\theta}_{t-1}^\top \boldsymbol{\phi}(\mathbf{x}_{t,a_t})$. So, as this estimation quality improves, our estimation quality for σ_t^2 will improve correspondingly. We visualize the quality of our estimation $\hat{\sigma}_t^2$ for σ_t^2 in Fig. 4 (b). Furthermore, as discussed in Rem. 3.1, σ_t^2 intuitively can improve the uncertainty quantification quality of UCB, we therefore also visualize the calibration performance of our UCB with $\hat{\sigma}_t^2$ in Fig. 4 (a).

Eq. 11 requires knowing the reward range $[a, b]$, which can be plausible in practice. For instance, in the real-world datasets in the experiment section, we may already know the range of $[a, b]$ when defining the reward

function. Another real-world example is in a personalized recommendation website system (Li et al., 2010), the decision to optimize is articles to display to users; context is user data (e.g., browsing history); actions are available news articles; and reward is user engagement (click or no click). Hence, we can set the reward as 1 if a user clicks and 0 otherwise, i.e., $[a, b] = [0, 1]$. It is also worth noticing that our Alg. 1 can be extended to handle cases where the reward range $[a, b]$ is dynamic, i.e., $[a, b]$ changes across rounds $t \in [T]$. We additionally show a result of this extension in Fig. 6 (b).

Alg. 1 with Maximum Likelihood Estimation (MLE). Eq. 8 back-propagate the neural network to update parameter $\mathbf{w}$ by using the MSE. This objective function, however, only tries to improve the mean of the estimation from DNN. To enhance the predictive uncertainty quality of the neural network (Chua et al., 2018; Tran et al., 2020), we further propose Eq. 12 to update parameter $\mathbf{w}$ via MLE. Given our current model, the predictive variance of the expected payoff $\boldsymbol{\theta}_t^\top \boldsymbol{\phi}(\mathbf{x}_{i,a_i}; \mathbf{w})$ is evaluated as $\boldsymbol{\phi}(\mathbf{x}_{i,a_i}; \mathbf{w})^\top \mathbf{A}_t^{-1} \boldsymbol{\phi}(\mathbf{x}_{i,a_i}; \mathbf{w})$. Therefore, we can formalize MLE with the normal distribution via the following loss function

$$\mathcal{L}_q(\mathbf{w}) = \sum_{i=1}^{qH} \left[\frac{1}{2} \log \left(2\pi \cdot \boldsymbol{\phi}(\mathbf{x}_{i,a_i}; \mathbf{w})^\top \mathbf{A}_t^{-1} \boldsymbol{\phi}(\mathbf{x}_{i,a_i}; \mathbf{w}) \right) + \frac{\left[r_{i,a_i} - \boldsymbol{\theta}_t^\top \boldsymbol{\phi}(\mathbf{x}_{i,a_i}; \mathbf{w}) \right]^2}{2 \cdot \boldsymbol{\phi}(\mathbf{x}_{i,a_i}; \mathbf{w})^\top \mathbf{A}_t^{-1} \boldsymbol{\phi}(\mathbf{x}_{i,a_i}; \mathbf{w})} \right]. \tag{12}$$

Algorithm 1 Neural-σ^2-LinUCB (code is in Apd. B.1)

Input: $\lambda, T, H, K, d, L, n, m, \{\alpha_t\}_{t\in[T]}$
$\mathbf{A}_0 \leftarrow \lambda \mathbf{I}$, $\mathbf{b}_0 \leftarrow \mathbf{0}$, $\boldsymbol{\theta}_0 \sim \mathcal{N}(0, 1/d)$ $\mathbf{w}_0 \sim \mathbb{P}(\mathbf{w})$, $q \leftarrow 0$
for $t = 1 \to T$ **do**
  Observe K feature $\mathbf{x}_{t,1}, \cdots, \mathbf{x}_{t,K} \in \mathbb{R}^d$
  for $k = 1 \to K$ **do**
    $p_{t,k} = \boldsymbol{\theta}_{t-1}^\top \boldsymbol{\phi}(\mathbf{x}_{t,k}; \mathbf{w}_{t-1}) + \alpha_t \left\| \boldsymbol{\phi}(\mathbf{x}_{t,k}; \mathbf{w}_{t-1}) \right\|_{\mathbf{A}_{t-1}^{-1}}$ ▷Compute UCB in Eq. 6
  end for
  Chose $a_t = \arg\max_a p_{t,a}$ ▷Select action by Eq. 6
  Observe the corresponding reward r_t
  Receive σ_t^2 in the oracle algorithm, or estimate σ_t^2 by Eq. 11 in the practical algorithm
  $\bar{\sigma}_t^2 = \max(\sigma_t^2, R^2/d)$
  $\mathbf{A}_t \leftarrow \mathbf{A}_{t-1} + \frac{\boldsymbol{\phi}(\mathbf{x}_{t,a_t}; \mathbf{w}_{t-1}) \boldsymbol{\phi}(\mathbf{x}_{t,a_t}; \mathbf{w}_{t-1})^\top}{\bar{\sigma}_t^2}$
  $\mathbf{b}_t \leftarrow \mathbf{b}_{t-1} + \frac{\boldsymbol{\phi}(\mathbf{x}_{t,a_t}; \mathbf{w}_{t-1}) r_t}{\bar{\sigma}_t^2}$
  $\boldsymbol{\theta}_t \leftarrow \mathbf{A}_t^{-1} \mathbf{b}_t$ ▷Update linear model by Eq. 10
  if $\mod(t, H) = 0$ **then**
    Initialize $\mathbf{w}_q^{(0)} = \mathbf{w}_t$
    for $s = 1 \to n$ **do**
      $\mathbf{w}_q^{(s)} = \mathbf{w}_q^{(s-1)} - \eta_q \nabla_{\mathbf{w}} \mathcal{L}_q \left(\mathbf{w}_q^{(s-1)} \right)$
    end for
    $\mathbf{w}_t \leftarrow \mathbf{w}_q^{(n)}$ ▷Update neural model
    $q \leftarrow q + 1$ ▷Update epoch scheduler
  else
    $\mathbf{w}_t \leftarrow \mathbf{w}_{t-1}$
  end if
end for

4 Theoretical analysis

To analyze the regret for Neural-σ^2-LinUCB, for convenience in analysis for the context $\mathbf{x}$ (Zou and Gu, 2019), we first apply the following transformation inspired by existing work (Allen-Zhu et al., 2019; Zhou et al., 2020) to ensure arm contexts are of unit length. In particular, without loss of generality:

Remark 4.1. (Arm context normalization). With unprocessed context $\tilde{\mathbf{x}}_{i,k}$, we formulate the corresponding normalized arm context $\mathbf{x}_{i,k}$ by $\mathbf{x}_{i,k} = \left[\frac{\tilde{\mathbf{x}}_{i,k}}{2 \cdot \|\tilde{\mathbf{x}}_{i,k}\|_2}, \frac{1}{2}, \frac{\tilde{\mathbf{x}}_{i,k}}{2 \cdot \|\tilde{\mathbf{x}}_{i,k}\|_2}, \frac{1}{2} \right]$ to achieve $\|\mathbf{x}_{i,k}\|_2 = 1$, for all $i \geq 1$ and $k \in [K]$. Then, for any context $\|\mathbf{x}_{i,k}\|_2 = 1$, we could replace $\mathbf{x}_{i,k}$ by $\mathbf{x}'_{i,k} = [\mathbf{x}_{i,k}^\top, \mathbf{x}_{i,k}^\top]^\top / \sqrt{2}$ to verify its entries satisfy $[\mathbf{x}_{i,k}]_j = [\mathbf{x}_{i,k}]_{j+d/2}$.

Then, we follow two main assumptions from Zhou et al. (2020); Xu et al. (2022a) for the results in this section to hold. Specifically, the first assumption is about the stability condition on the spectral norm of the neural network gradient (Wang et al., 2014; Balakrishnan et al., 2017; Xu et al., 2017):

Assumption 4.2. For a specific weights parameters $\mathbf{w}_0$, $\exists \ell_{Lip} > 0$ s.t. $\left\| \frac{\partial \boldsymbol{\phi}}{\partial \mathbf{w}}(\mathbf{x}, \mathbf{w}_0) - \frac{\partial \boldsymbol{\phi}}{\partial \mathbf{w}}(\mathbf{x}', \mathbf{w}_0) \right\|_2 \leq \ell_{Lip} \|\mathbf{x} - \mathbf{x}'\|_2$, $\forall \mathbf{x}, \mathbf{x}' \in \{\mathbf{x}_{i,k}\}_{i\in[T],k\in[K]}$.

As discussed in Xu et al. (2022a), Asm. 4.2 is widely made in nonconvex optimization. Furthermore, it is also worth noticing that this assumption is only required on the TK training data points and a specific weight parameter $\mathbf{w}_0$. Therefore, the conditions in Asm. 4.2 will hold if the raw feature data lie in a certain subspace of $\mathbb{R}^d$. To describe our last assumption, it is necessary to describe the NTK matrix $\mathbf{H}$.

Definition 4.3. (Jacot et al., 2018) Define $\mathbf{H} \in \mathbb{R}^{TK \times TK}$ be the Neural Tangent Kernel (NTK) matrix, based on all features vectors $\{\mathbf{x}_{t,k}\}_{t\in[T],k\in[K]}$, renumbered as $\{x_i\}_{i\in[TK]}$. Then for all $i, j \in [TK]$, each entry $\mathbf{H}_{ij} := \frac{1}{2} \left(\tilde{\boldsymbol{\Sigma}}^{(L)}(\mathbf{x}_i, \mathbf{x}_j) + \boldsymbol{\Sigma}^{(L)}(\mathbf{x}_i, \mathbf{x}_j) \right)$, where the covariance between two data point $\mathbf{x}, \mathbf{y} \in \mathbb{R}^d$ is given as follows: $\tilde{\boldsymbol{\Sigma}}^{(0)}(\mathbf{x}, \mathbf{y}) = \boldsymbol{\Sigma}^{(0)}(\mathbf{x}, \mathbf{y}) = \mathbf{x}^\top \mathbf{y}$, $\boldsymbol{\Lambda}^{(l)}(\mathbf{x}, \mathbf{y}) = \begin{bmatrix} \boldsymbol{\Sigma}^{l-1}(\mathbf{x}, \mathbf{x}) & \boldsymbol{\Sigma}^{l-1}(\mathbf{x}, \mathbf{y}) \\ \boldsymbol{\Sigma}^{l-1}(\mathbf{y}, \mathbf{x}) & \boldsymbol{\Sigma}^{l-1}(\mathbf{y}, \mathbf{y}) \end{bmatrix}$, $\boldsymbol{\Sigma}^{(l)}(\mathbf{x}, \mathbf{y}) = 2\mathbb{E}_{(u,v)\sim\mathcal{N}\left(\mathbf{0}, \boldsymbol{\Lambda}^{(l-1)}(\mathbf{x}, \mathbf{y})\right)}[g(u)g(v)]$, $\tilde{\boldsymbol{\Sigma}}^{(l)}(\mathbf{x}, \mathbf{y}) = 2\tilde{\boldsymbol{\Sigma}}^{(l-1)}(\mathbf{x}, \mathbf{y}) \mathbb{E}_{u,v}[g'(u)g'(v)] + \boldsymbol{\Sigma}^{(l)}(\mathbf{x}, \mathbf{y})$, with $(u, v) \sim \mathcal{N}\left(\mathbf{0}, \boldsymbol{\Lambda}^{(l-1)}(\mathbf{x}, \mathbf{y})\right)$, and $g'(\cdot)$ being the derivative of the activation function g.

The last assumption essentially requires the NTK matrix $\mathbf{H}$ to be non-singular (Du et al., 2019; Arora et al., 2019a; Cao and Gu, 2019):

Assumption 4.4. The NTK matrix $\mathbf{H}$ is positive definite, i.e., $\lambda_{\min}(\mathbf{H}) \geq \lambda_0$ for some constants $\lambda_0 > 0$.

Asm. 4.4 could be mild since we can derive from Rem. 4.1 with two ReLU layers (Zou and Gu, 2019; Xu et al., 2022a). We use Asm. 4.4 to characterize the properties of DNN to represent the feature vectors. Following these assumptions, we next provide the regret bound for our oracle Neural-σ^2-LinUCB algorithm:

Theorem 4.5. *Suppose Asm 4.2, and 4.4 hold and further assume that* $\|\theta^*\|_2 \leq M$, $\|\mathbf{x}_t\|_2 \leq G$, *and* $\lambda \geq \max\{1, G^2\}$ *for some* $M, G > 0$. *For any* $\delta \in (0,1)$, *let us choose* α_t *as*

$$\alpha_t = 8\sqrt{d\log\left(1 + td(\log HK)/(\bar{\sigma}_t^2 d\lambda)\right)\log(4t^2/\delta)} + 4R/\bar{\sigma}_t \log(4t^2/\delta) + \lambda^{1/2} M,$$

the step size $\eta_q \leq C_0 \left(d^2 mnT^{5.5} L^6 \log(TK/\delta)\right)^{-1}$, *and the neural network width satisfies* $m = poly(L, d, 1/\delta, H, \log(TK/\delta))$, *then, with probability at least* $1-\delta$ *over the randomness of the neural network initialization, the regret of the oracle algorithm satisfies*

$$Regret(T) \leq C_1 \alpha_T \sqrt{\left(TR^2 + d\sum_{t=1}^{T} \sigma_t^2\right)\log(1 + TG^2/(\lambda R^2))} + \frac{C_2 \ell_{Lip} L^3 d^{5/2} T \sqrt{\log m \log(\frac{1}{\delta}) \log(\frac{TK}{\delta})}\, \|\mathbf{r} - \tilde{\mathbf{r}}\|_{\mathbf{H}^{-1}}}{m^{1/6}},$$

where constants $\{C_i\}_{i\in[2]}$ *are independent of the problem,* $\mathbf{r} = (r(\mathbf{x}_1), r(\mathbf{x}_2), \cdots, r(\mathbf{x}_{TK}))^\top \in \mathbb{R}^{TK}$, *and* $\tilde{\mathbf{r}} = (f(\mathbf{x}_1; \theta_0, \mathbf{w}_0), \cdots, f(\mathbf{x}_{TK}; \theta_{T-1}, \mathbf{w}_{T-1}))^\top \in \mathbb{R}^{TK}$.

The proof for Thm. 4.5 adapts the techniques of the Bernstein inequality for vector-valued martingales over the linear output DNN last layer from Zhou et al. (2021) and the NTK for the raw context-feature DNN mapping from Xu et al. (2022a), details are in Apd. A.3.1. From Thm 4.5, we obtain the following conclusion:

Corollary 4.6. *Under the conditions in Thm. 4.5, then, with probability at least* $1-\delta$, *the regret of the oracle algorithm is bounded by*

$$Regret(T) \leq \tilde{\mathcal{O}}\left(R\sqrt{dT} + d\sqrt{\sum_{t=1}^{T} \sigma_t^2}\right) + \tilde{\mathcal{O}}\left(m^{-1/6} T \sqrt{(\mathbf{r} - \tilde{\mathbf{r}})^\top \mathbf{H}^{-1} (\mathbf{r} - \tilde{\mathbf{r}})}\right).$$

Remark 4.7. (Regret comparison between Neural-σ^2-LinUCB and previous methods). Our second regret term resembles the second regret term bound of Neural-LinUCB, which we can assume to have a constant bound for $\|\mathbf{r} - \tilde{\mathbf{r}}\|_{\mathbf{H}^{-1}} = \mathcal{O}(1)$ and the selection of $m \geq T^3$ (Zhou et al., 2020). So the whole bound depends mainly on the first term. Regarding the first term in our regret upper bound, since $\sigma_t \leq R$ by the condition in Eq. 3, it can be seen that the first term of the regret of our oracle Neural-σ^2-LinUCB, i.e., $\tilde{\mathcal{O}}\left(R\sqrt{dT} + d\sqrt{\sum_{t=1}^{T} \sigma_t^2}\right)$ is strictly better than $\tilde{\mathcal{O}}\left(Rd\sqrt{T}\right)$ of Neural-LinUCB.

Finally, we conclude the regret bound for our practical Neural-σ^2-LinUCB algorithm:

Theorem 4.8. *Under the conditions in Thm. 4.5, then, with probability at least* $1-\delta$, *the regret of the practical algorithm in Alg. 1 is bounded by*

$$Regret(T) \leq \tilde{\mathcal{O}}\left(R\sqrt{dT} + d\sqrt{\sum_{t=1}^{T} \sigma_t^2 + \epsilon}\right) + \tilde{\mathcal{O}}\left(m^{-1/6} T \sqrt{(\mathbf{r} - \tilde{\mathbf{r}})^\top \mathbf{H}^{-1} (\mathbf{r} - \tilde{\mathbf{r}})}\right),$$

where the estimation error is bounded by $\epsilon = d^3 \left(T^2 n^9 L^{11} \log^6(m)\right)^{-1}$. *The proof is in Apd. A.5.*

Remark 4.9. (Regret comparison between oracle and practical version). If ϵ is small enough, then the regret bound in Thm. 4.8 becomes close to the oracle Neural-σ^2-LinUCB in Thm. 4.5, i.e., $\tilde{\mathcal{O}}\left(R\sqrt{dT} + d\sqrt{\sum_{t=1}^{T} \sigma_t^2}\right)$. This is practically possible when the time horizon T increases and we can design a neural network with deep layers L, hidden width size m, and enough training iteration n.

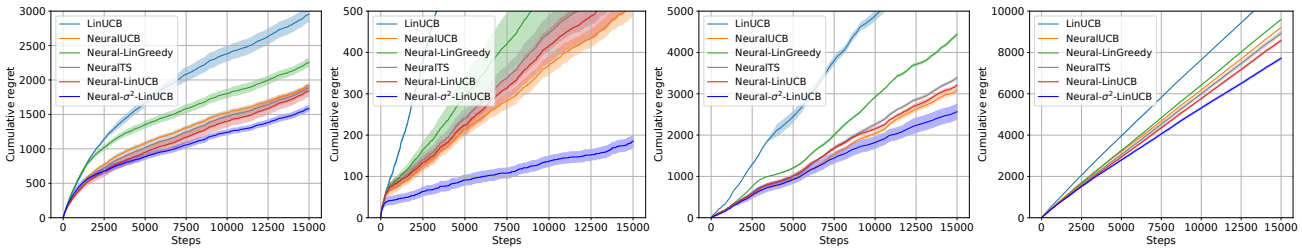
5 Experiments

We empirically compare our practical Neural-σ^2-LinUCB algorithm with five main baselines in the main paper, including LinUCB (Abbasi-yadkori et al., 2011), NeuralUCB (Zhou et al., 2020), NeuralTS (ZHANG et al., 2021), Neural-LinGreedy (Xu et al., 2022a), and Neural-LinUCB (Xu et al., 2022a). More baseline comparisons and experimental details are in Apd. B.

5.1 Synthetic datasets

We follow Zhou et al. (2020) by setting the context dimension $d = 20$, arms number $K = 4$, and time horizon $T = 10000$. We sample the context uniformly at random from the unit ball, i.e., $\mathbf{x}_{t,a} = \left(\mathbf{x}_{t,a}^{(1)}, \mathbf{x}_{t,a}^{(2)}, \cdots, \mathbf{x}_{t,a}^{(d)}\right)$, $\mathbf{x}_{t,a}^{(i)} \sim \mathcal{N}(0,1)/\|\mathbf{x}_{t,a}\|_2$, $\forall i \in [d]$. Then, we define the reward function h with 3 settings: $h_1(\mathbf{x}_{t,a}) = 10(\mathbf{x}_{t,a}^\top \boldsymbol{\theta})^2$, $h_2(\mathbf{x}_{t,a}) = \mathbf{x}_{t,a}^\top \boldsymbol{\theta}^\top \boldsymbol{\theta} \mathbf{x}_{t,a}$, and $h_3(\mathbf{x}_{t,a}) = \cos(3\mathbf{x}_{t,a}^\top \boldsymbol{\theta})$, where $\boldsymbol{\theta}$ is also randomly generated uniformly over the unit ball. For each $h_i(\cdot)$, the reward is generated by $r_{t,a}^{(i)} = h_i(\mathbf{x}_{t,a}) + \xi_t$. We consider a randomly changing variance by setting at each time t, $\xi_t \sim \mathcal{N}(0, \mathbb{E}[\xi_t^2 \mid \mathbf{x}_{1:t,a_{1:t}}, \xi_{1:t-1}])$, where $\mathbb{E}[\xi_t^2 \mid \mathbf{x}_{1:t,a_{1:t}}, \xi_{1:t-1}] \sim \mathbb{U}(0,1)$.

For each algorithm, we run 5 traces with different random seeds per run, and then we summarize their cumu-



(a) MNIST (b) UCI-shuttle (c) UCI-covertype (d) CIFAR-10

Figure 3: Cumulative regret results on the real-world data across 10 runs with different seeds.

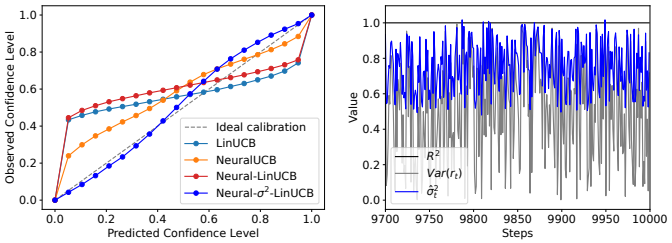
lative regret results in Fig. 1. Firstly, we observe that Neural-σ^2-LinUCB is consistently better than SOTA baselines by having a significantly low cumulative regret as $t \in [T]$ grows. For instance, in the first setting with $h_1(\mathbf{x}_{t,a})$, at the final round $t = T$, our regret is below 1000, which is better than Neural-LinUCB by about 600, and remarkably better than NeuralUCB by about 1200. Secondly, neural-bandit algorithms significantly outperform the non-neural algorithm LinUCB in all settings (more details are in Fig. 17). This continues to confirm the hypothesis that non-linear models can address the limitation of the linear-reward assumption (Riquelme et al., 2018; Zhou et al., 2020).

5.2 Real-world datasets

To validate our model's effectiveness in the real world, we deploy on the MNIST (Lecun et al., 1998), UCI-shuttle (statlog), UCI-covertype (Dua and Graff, 2017), and CIFAR-10 dataset (Alex Krizhevsky, 2009). Following Beygelzimer and Langford (2009), we convert these dataset to K-armed contextual bandits by transforming each labeled data ($\mathbf{x} \in \mathbb{R}^d, \mathbf{y} \in \mathbb{R}^K$) into K context vector $\mathbf{x}^{(1)} = (\mathbf{x}, \mathbf{0}, \cdots, \mathbf{0}), \cdots, \mathbf{x}^{(K)} = (\mathbf{0}, \mathbf{0}, \cdots, \mathbf{x}) \in \mathbb{R}^{dK}$. We define the reward function by 1 if the agent selects the exact arm $i \in [K]$ s.t. $\mathbf{y}_i = 1$, and 0 otherwise.

We compare methods over $T = 15000$ rounds across 5 runs in Fig. 3. In low-dimensional data like UCI-shuttle, our model behaves similarly to the synthetic data with a significantly lower regret. In high-dimensional data like MNIST, UCI-covertype, and CIFAR-10, although all models find it hard to estimate the underlying reward function, Neural-σ^2-LinUCB still consistently outperforms other methods. Furthermore, our results are stable across different running seeds with small variance intervals on 10 runs. In Fig. 14 in Apd. B.4.1, we also show a case when the model capacity (i.e., L and m) increases, we can further achieve a lower cumulative regret on CIFAR-10. To this end, we can see that our method not only has a lower regret than others in the synthetic data but also in the real-world dataset, confirming the tighter regret bound of Thm. 4.5 and 4.8.

5.3 Uncertainty estimation evaluations



(a) (b)

Figure 4: (a) Visualization of calibration error in Eq. 89 with reliability diagram on $h_1(\mathbf{x}_{t,a})$ dataset. (b) Reward variance $Var(r_t)$, our estimation for the variance upper bound σ_t^2, and the upper bound R^2 comparison.

To better understand the uncertainty quality of UCB, Fig. 4 (a) compare the calibration performance across UCB methods. Intuitively, calibration means a p confidence interval contains the reward p of the time (Gneiting et al., 2007) (evaluation metrics details are in Apd. B.3.1). We can see that by leveraging a high-quality estimation for σ_t^2 in Eq. 11, our UCB is more well-calibrated by less over-confidence and under-confidence than other methods. More quantitative results are in Tab. 3 in Apd. B.3.1. We also evaluate calibration by different checkpoints across time steps on a hold-old validation set in Fig. 7, 8, 11, 10 in Apd. B.3.1. Overall, we also observe that our method is more calibrated than other algorithms. These results are consistent with the observation that a calibrated model can further improve the cumulative regret (Malik et al., 2019; Deshpande et al., 2024).

Regarding the estimation quality for σ_t^2 of Eq. 11, we visualize our estimated $\hat{\sigma}_t^2$, the true $Var(r_t)$, and the magnitude R^2 at the last 300 steps in Fig. 4 (b) (details are in Fig. 11 in Apd. B.3.1). We can see that since we set $\mathbb{E}[\xi_t^2 \mid \mathbf{x}_{1:t,a_{1:t}}, \xi_{1:t-1}] \sim \mathbb{U}(0,1)$, so $R^2 = 1$ and in almost all steps, our estimated $\hat{\sigma}_t^2$ has a higher value than $Var(r_t)$ and lower value than R^2, showing a high-quality estimation in our Eq. 11.

5.4 Computational efficiency evaluations

Methods	Arm selection (↓)	DNN update (↓)
NeuralUCB	6.42 ± 0.33	2.28 ± 0.21
NeuralTS	7.27 ± 0.40	2.62 ± 0.29
Neural-LinUCB	**0.43 ± 0.02**	**1.86 ± 0.17**
Neural-σ^2-LinUCB	**0.43 ± 0.02**	**1.86 ± 0.17**

Table 2: Computational cost comparison on MNIST for running 100 rounds: runtime (seconds) on RTX-A5000.

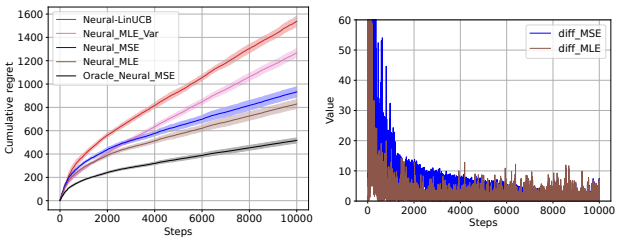
We compare the latency of neural-bandit algorithms in Tab. 2 with $H = 10$ and $t \in [100]$. Overall, similar to Neural-LinUCB, we can see that our method is more efficient than NeuralUCB and NeuralTS in both the arm selection (Lines L-5-8) and DNN update step (L-17-21) in Alg. 1. Especially, since they require to perform UCB/sampling on entire DNN parameters, our method is much faster than by around 6 seconds in the arm selection step by the UCB is performed over the linear mode with the feature from the last DNN layer. A detailed comparison is in Fig. 12 in Apd. B.4. Given better regret performances, Neural-σ^2-LinUCB makes a significant contribution by achieving a balance of computational efficiency, high uncertainty quality, and accurate reward estimation in real-world domains.

5.5 Ablation study for Neural-σ^2-LinUCB

To take a closer look at our Alg. 1, we compare 4 settings, including: (1) using MSE in Eq. 8 with the true variance $Var(r_t)$ from the generating process of the synthetic data (Oracle_Neural_MSE); (2) using MLE in Eq. 12 with the estimated $Var(r_t)$ from $\boldsymbol{\phi}(\mathbf{x}_{t,a_t};\mathbf{w})^\top \mathbf{A}_t^{-1} \boldsymbol{\phi}(\mathbf{x}_{t,a_t};\mathbf{w})$ (Neural_MLE_Var); (3) using the estimated σ_t^2 in Eq. 11 with MSE in Eq. 8 (Neural_MSE, i.e., Neural-σ^2-LinUCB); (4) using the estimated σ_t^2 in Eq. 11 with MLE in Eq. 12 (Neural_MLE). More results are in Apd. B.4.1.

Fig. 5 (a) shows oracle Neural-σ^2-LinUCB, i.e., Oracle_Neural_MSE has the lowest regret on the synthetic data $h_1(\mathbf{x}_{t,a})$. After that is our practical Neural-σ^2-LinUCB versions, including Neural_MSE and Neural_MLE with the estimated σ_t^2 in Eq. 11. Notably, all of them are significantly better than Neural-LinUCB.

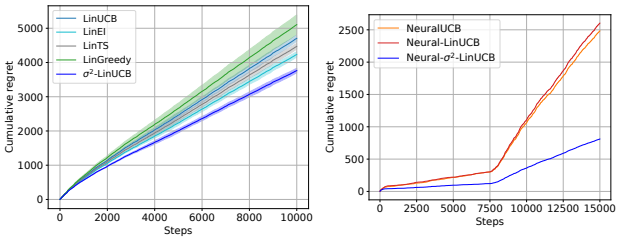
It is also worth noticing that using MLE in Eq. 12 brings out a slightly better performance than MSE in Eq. 8 by a lower regret of Neural_MLE than Neural_MSE in Fig. 5 (a), and a lower variance estimation error in Fig. 5 (b), where the y-axis is the difference between the estimation and the true variance at time t, i.e., $\left|\boldsymbol{\phi}(\mathbf{x}_{i,a_i};\mathbf{w})^\top \mathbf{A}_t^{-1} \boldsymbol{\phi}(\mathbf{x}_{i,a_i};\mathbf{w}) - Var(r_t)\right|$. This confirms the effectiveness of using MLE to improve uncertainty estimation quality of DNN (Chua et al., 2018; Manh Bui and Liu, 2024). That said, estimating the true variance



(a) (b)

Figure 5: (a) Comparison in cumulative regret between different settings. (b) Variance estimation error between MLE and MSE settings of our Neural-σ^2-LinUCB.

$Var(r_t)$ with MLE is still difficult by having a high estimation error. As a result, the regret performance of Neural_MLE_Var is still worse than estimating the upper bound σ_t^2 (i.e., Neural_MLE).



(a) (b)

Figure 6: (a) Cumulative regret results on $h_1(\mathbf{x}) = 10(\mathbf{x}^\top \boldsymbol{\theta})^2$ without DNN setting. (b) Cumulative regret results on UCI-shuttle with dynamic reward range.

We provide an additional result in the linear contextual bandits setting (i.e., without the DNN version). Fig. 6 (a) shows that our results are consistent between linear and neural contextual bandits settings by having a lower regret than other linear baselines.

Finally, we test our model on UCI-shuttle with a dynamic reward range. Specifically, when $t \in [0, T/2]$, we define the reward function by 1 if the agent selects the exact arm $i \in [K]$ s.t. $\mathbf{y}_i = 1$, and 0 otherwise. And, when $t \in [T/2, T]$, we define the reward function by 3 if the agent selects the exact arm $i \in [K]$ s.t. $\mathbf{y}_i = 1$, and 1 otherwise. Therefore, the reward range will be $[a, b] = [0, 1]$ if $t \in [0, T/2]$, and $[a, b] = [1, 3]$ if $t \in [T/2, T]$. Fig. 6 (b) shows that our method achieves a lower cumulative regret than NeuralUCB and Neural-LinUCB in this dynamic reward range setting, confirming our empirical results.

6 Related work

We can categorize methods in the non-linear contextual bandits into three main approaches. First is **using non-parametric modeling**, including percep-

tion (Kakade et al., 2008), random forest (Féraud et al., 2016), Gaussian processes (Srinivas et al., 2010; Krause and Ong, 2011), and kernel space (Valko et al., 2013; Bubeck et al., 2011). The second approach is **reducing to supervised-learning problems**, which optimizes objective function based on fully-labeled data with context-reward pair (Langford and Zhang, 2007; Foster and Rakhlin, 2020; Agarwal et al., 2014).

Our method is relevant to the last approach, which is **considering generalized linear bandits** by decomposing the reward function to a linear and a non-linear link function (Filippi et al., 2010; Li et al., 2017; Jun et al., 2017), e.g., the mixture of linear experts (Beygelzimer and Langford, 2009), or using the non-linear link function by DNN (Riquelme et al., 2018; Kveton et al., 2020; Zhou et al., 2020). That said, our method is sampling-free and more computationally efficient than the sampling-based methods, e.g., the mixture of experts and Neural Thompson Sampling (NeuralTS)-based model (ZHANG et al., 2021; Xu et al., 2022b). Compared to other sampling-free approaches, e.g., NeuralUCB (Zhou et al., 2020; Ban et al., 2022), our algorithm has a lower regret and also is more efficient by NeuralUCB has a $\tilde{\mathcal{O}}(R\tilde{d}\sqrt{T})$ regret bound, where $\tilde{d}$ is the dimensions of NTK matrix which can potentially scale with $\mathcal{O}(TK)$. Compared with the most relevant method, i.e., Neural-LinUCB (Xu et al., 2022a), our method enjoys a similar computational complexity, while having a better regret bound.

Variance-aware-UCB algorithms. In the standard bandits setting, leveraging reward uncertainty of the agent model to enhance UCB has shown promising results. In particular, previous work (Kuleshov and Precup, 2014; Audibert et al., 2009) have shown UCB1-Tuned and UCB1-Normal, models using the estimated variance can obtain a lower regret than the standard UCB method (Auer et al., 2002). Regarding the contextual bandits setting, several theoretical studies (Zhao et al., 2023; Ye et al., 2023; Zhou and Gu, 2022) have shown that variance-dependent regret bounds for linear contextual bandits are lower than the sublinear regret of LinUCB. However, all of them only consider the linear contextual bandits setting while Neural-σ^2-LinUCB considers the non-linear case. Furthermore, these algorithms often require a known variance of the noise and its upper bound (Zhou et al., 2021; Zhou and Gu, 2022). Additionally, they are inefficient in terms of computation or even computationally intractable in practice (Zhang et al., 2021; Kim et al., 2022; Zhao et al., 2023). As a result, none of these works provide empirical evidence with experimental results.

In particular, we can compare with Suplin + Adaptive Variance-aware Exploration (SAVE) (Zhao et al., 2023), which can be seen as a variance-aware version of SupLinUCB (Chu et al., 2011). Beyond the differences between linear and neural contextual bandits, our setting is also different because our Neural-σ^2-LinUCB and our other baselines in Sec. 5 are not given knowledge of the time horizon T, i.e., the knowledge of the optimal regret. Meanwhile, SAVE is a theoretical approach that is not practical since their algorithm requires prior knowledge of T (i.e., α and K in Zhao et al. (2023)) to balance exploration and exploitation.

In summary, compared to previous works, we provide both theoretical evidence for variance-aware neural bandits and empirical evidence with the practical algorithm in real-world contextual settings. We believe we are one of the first works that analyze regret bound for variance-aware-UCB with DNN. Importantly, we also introduce a practical algorithm with extensive empirical results for variance-aware-UCB, opening an empirical direction for this approach.

7 Conclusion

Neural-UCB bandits have shown success in practice and theoretically achieve $\tilde{\mathcal{O}}(Rd\sqrt{T})$ regret bound. To enhance the UCB quality and regret guarantee, we introduce Neural-σ^2-LinUCB, a variance-aware algorithm that leverages σ_t^2, i.e., an upper bound of the reward noise variance at round t. We propose an oracle algorithm with an oracle σ_t^2 and a practical version with a novel estimation for σ_t^2. We analyze regret bounds for both oracle and practical versions. Notably, our oracle algorithm achieves a tighter bound with $\tilde{\mathcal{O}}\left(R\sqrt{dT} + d\sqrt{\sum_{t=1}^{T}\sigma_t^2}\right)$ regret. Given a reward range, our practical algorithm estimates σ_t^2 using the estimated reward mean with DNN. Experimentally, our practical method enjoys a similar computational efficiency while outperforming SOTA techniques by having a lower calibration error and lower cumulative regret across different settings on benchmark datasets. With these promising results, we hope that our work will open a door for the direction of understanding uncertainty estimation to enhance neural-bandit algorithms in both theoretical and practical aspects.

Acknowledgments

We would like to thank the anonymous reviewers for their helpful comments. H.M.Bui is supported by the Discovery Award of Johns Hopkins University and the Challenge grant from the JHU Institute of Assured Autonomy. A.Liu is partially supported by the Amazon Research Award, the Discovery Award of the Johns Hopkins University, and a seed grant from the JHU Institute of Assured Autonomy.

References

Richard S. Sutton and Andrew G. Barto. *Reinforcement Learning: An Introduction*. The MIT Press, second edition, 2018.

Tor Lattimore and Csaba Szepesvári. *Bandit Algorithms*. Cambridge University Press, 2020.

Sébastien Bubeck and Nicolò Cesa-Bianchi. Regret analysis of stochastic and nonstochastic multi-armed bandit problems. *Foundations and Trends® in Machine Learning*, 2012.

Dongruo Zhou, Lihong Li, and Quanquan Gu. Neural contextual bandits with UCB-based exploration. In *Proceedings of the 37th International Conference on Machine Learning*, 2020.

Pan Xu, Zheng Wen, Handong Zhao, and Quanquan Gu. Neural contextual bandits with deep representation and shallow exploration. In *International Conference on Learning Representations*, 2022a.

Lihong Li, Wei Chu, John Langford, and Robert E. Schapire. A contextual-bandit approach to personalized news article recommendation. In *Proceedings of the 19th International Conference on World Wide Web*, 2010.

Wei Chu, Lihong Li, Lev Reyzin, and Robert Schapire. Contextual bandits with linear payoff functions. In *Proceedings of the Fourteenth International Conference on Artificial Intelligence and Statistics*, 2011.

Yasin Abbasi-yadkori, Dávid Pál, and Csaba Szepesvári. Improved algorithms for linear stochastic bandits. In *Advances in Neural Information Processing Systems*, 2011.

Volodymyr Kuleshov, Nathan Fenner, and Stefano Ermon. Accurate uncertainties for deep learning using calibrated regression. In *Proceedings of the 35th International Conference on Machine Learning*, 2018.

Volodymyr Kuleshov and Doina Precup. Algorithms for multi-armed bandit problems, 2014.

Peter Auer, Nicolò Cesa-Bianchi, and Paul Fischer. Finite-time analysis of the multiarmed bandit problem. *Mach. Learn.*, 2002.

Ali Malik, Volodymyr Kuleshov, Jiaming Song, Danny Nemer, Harlan Seymour, and Stefano Ermon. Calibrated model-based deep reinforcement learning. In *Proceedings of the 36th International Conference on Machine Learning*, 2019.

Shachi Deshpande, Charles Marx, and Volodymyr Kuleshov. Online calibrated and conformal prediction improves Bayesian optimization. In *Proceedings of The 27th International Conference on Artificial Intelligence and Statistics*, 2024.

Dongruo Zhou, Quanquan Gu, and Csaba Szepesvari. Nearly minimax optimal reinforcement learning for linear mixture markov decision processes. In *Proceedings of Thirty Fourth Conference on Learning Theory*, 2021.

Heyang Zhao, Jiafan He, Dongruo Zhou, Tong Zhang, and Quanquan Gu. Variance-dependent regret bounds for linear bandits and reinforcement learning: Adaptivity and computational efficiency. In *Proceedings of Thirty Sixth Conference on Learning Theory*, 2023.

Arthur Jacot, Franck Gabriel, and Clement Hongler. Neural tangent kernel: Convergence and generalization in neural networks. In *Advances in Neural Information Processing Systems*, 2018.

Kurtland Chua, Roberto Calandra, Rowan McAllister, and Sergey Levine. Deep reinforcement learning in a handful of trials using probabilistic dynamics models. In *Advances in Neural Information Processing Systems*, 2018.

Kevin Tran, Willie Neiswanger, Junwoong Yoon, Qingyang Zhang, Eric Xing, and Zachary W Ulissi. Methods for comparing uncertainty quantifications for material property predictions. *Machine Learning: Science and Technology*, 2020.

Difan Zou and Quanquan Gu. An improved analysis of training over-parameterized deep neural networks. In *Advances in Neural Information Processing Systems*, 2019.

Zeyuan Allen-Zhu, Yuanzhi Li, and Zhao Song. A convergence theory for deep learning via over-parameterization. In *Proceedings of the 36th International Conference on Machine Learning*, 2019.

Zhaoran Wang, Han Liu, and Tong Zhang. Optimal computational and statistical rates of convergence for sparse nonconvex learning problems. *The Annals of Statistics*, 2014.

Sivaraman Balakrishnan, Martin J. Wainwright, and Bin Yu. Statistical guarantees for the EM algorithm: From population to sample-based analysis. *The Annals of Statistics*, 2017.

Pan Xu, Jian Ma, and Quanquan Gu. Speeding up latent variable gaussian graphical model estimation via nonconvex optimization. In *Advances in Neural Information Processing Systems*, 2017.

Simon Du, Jason Lee, Haochuan Li, Liwei Wang, and Xiyu Zhai. Gradient descent finds global minima of deep neural networks. In *Proceedings of the 36th International Conference on Machine Learning*, 2019.

Sanjeev Arora, Simon S Du, Wei Hu, Zhiyuan Li, Russ R Salakhutdinov, and Ruosong Wang. On exact computation with an infinitely wide neural

net. In *Advances in Neural Information Processing Systems*, 2019a.

Yuan Cao and Quanquan Gu. Generalization bounds of stochastic gradient descent for wide and deep neural networks. In *Advances in Neural Information Processing Systems*, 2019.

Weitong ZHANG, Dongruo Zhou, Lihong Li, and Quanquan Gu. Neural thompson sampling. In *International Conference on Learning Representations*, 2021.

Carlos Riquelme, George Tucker, and Jasper Snoek. Deep bayesian bandits showdown: An empirical comparison of bayesian deep networks for thompson sampling. In *International Conference on Learning Representations*, 2018.

Y. Lecun, L. Bottou, Y. Bengio, and P. Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 1998.

Dheeru Dua and Casey Graff. Uci machine learning repository, 2017.

Geoffrey Hinton Alex Krizhevsky. Learning multiple layers of features from tiny images, 2009.

Alina Beygelzimer and John Langford. The offset tree for learning with partial labels. In *Proceedings of the 15th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2009.

Tilmann Gneiting, Fadoua Balabdaoui, and Adrian E. Raftery. Probabilistic Forecasts, Calibration and Sharpness. *Journal of the Royal Statistical Society Series B: Statistical Methodology*, 2007.

Ha Manh Bui and Anqi Liu. Density-regression: Efficient and distance-aware deep regressor for uncertainty estimation under distribution shifts. In *Proceedings of The 27th International Conference on Artificial Intelligence and Statistics*, 2024.

Sham M. Kakade, Shai Shalev-Shwartz, and Ambuj Tewari. Efficient bandit algorithms for online multiclass prediction. *ICML '08: Proceedings of the 25th international conference on Machine learning*, 2008.

Raphaël Féraud, Robin Allesiardo, Tanguy Urvoy, and Fabrice Clérot. Random forest for the contextual bandit problem. In *Proceedings of the 19th International Conference on Artificial Intelligence and Statistics*, 2016.

Niranjan Srinivas, Andreas Krause, Sham Kakade, and Matthias Seeger. Gaussian process optimization in the bandit setting: no regret and experimental design. In *Proceedings of the 27th International Conference on International Conference on Machine Learning*, 2010.

Andreas Krause and Cheng Ong. Contextual gaussian process bandit optimization. In *Advances in Neural Information Processing Systems*, 2011.

Michal Valko, Nathan Korda, Rémi Munos, Ilias Flaounas, and Nello Cristianini. Finite-time analysis of kernelised contextual bandits. In *Proceedings of the Twenty-Ninth Conference on Uncertainty in Artificial Intelligence*, 2013.

Sébastien Bubeck, Rémi Munos, Gilles Stoltz, and Csaba Szepesvári. x-armed bandits. *Journal of Machine Learning Research*, 2011.

John Langford and Tong Zhang. The epoch-greedy algorithm for multi-armed bandits with side information. In *Advances in Neural Information Processing Systems*, 2007.

Dylan Foster and Alexander Rakhlin. Beyond UCB: Optimal and efficient contextual bandits with regression oracles. In *Proceedings of the 37th International Conference on Machine Learning*, 2020.

Alekh Agarwal, Daniel Hsu, Satyen Kale, John Langford, Lihong Li, and Robert Schapire. Taming the monster: A fast and simple algorithm for contextual bandits. In *Proceedings of the 31st International Conference on Machine Learning*, 2014.

Sarah Filippi, Olivier Cappe, Aurélien Garivier, and Csaba Szepesvári. Parametric bandits: The generalized linear case. In *Advances in Neural Information Processing Systems*, 2010.

Lihong Li, Yu Lu, and Dengyong Zhou. Provably optimal algorithms for generalized linear contextual bandits. In *Proceedings of the 34th International Conference on Machine Learning*, 2017.

Kwang-Sung Jun, Aniruddha Bhargava, Robert Nowak, and Rebecca Willett. Scalable generalized linear bandits: Online computation and hashing. In *Advances in Neural Information Processing Systems*, 2017.

Branislav Kveton, Manzil Zaheer, Csaba Szepesvari, Lihong Li, Mohammad Ghavamzadeh, and Craig Boutilier. Randomized exploration in generalized linear bandits. In *Proceedings of the Twenty Third International Conference on Artificial Intelligence and Statistics*, 2020.

Pan Xu, Hongkai Zheng, Eric V Mazumdar, Kamyar Azizzadenesheli, and Animashree Anandkumar. Langevin Monte Carlo for contextual bandits. In *Proceedings of the 39th International Conference on Machine Learning*, 2022b.

Yikun Ban, Yuchen Yan, Arindam Banerjee, and Jingrui He. EE-net: Exploitation-exploration neural networks in contextual bandits. In *International Conference on Learning Representations*, 2022.

Jean-Yves Audibert, Rémi Munos, and Csaba Szepesvári. Exploration-exploitation tradeoff using variance estimates in multi-armed bandits. *Theor. Comput. Sci.*, 2009.

Chenlu Ye, Wei Xiong, Quanquan Gu, and Tong Zhang. Corruption-robust algorithms with uncertainty weighting for nonlinear contextual bandits and Markov decision processes. In *Proceedings of the 40th International Conference on Machine Learning*, 2023.

Dongruo Zhou and Quanquan Gu. Computationally efficient horizon-free reinforcement learning for linear mixture MDPs. In *Advances in Neural Information Processing Systems*, 2022.

Zihan Zhang, Jiaqi Yang, Xiangyang Ji, and Simon Shaolei Du. Improved variance-aware confidence sets for linear bandits and linear mixture MDP. In *Advances in Neural Information Processing Systems*, 2021.

Yeoneung Kim, Insoon Yang, and Kwang-Sung Jun. Improved regret analysis for variance-adaptive linear bandits and horizon-free linear mixture MDPs. In *Advances in Neural Information Processing Systems*, 2022.

Sanjeev Arora, Simon Du, Wei Hu, Zhiyuan Li, and Ruosong Wang. Fine-grained analysis of optimization and generalization for overparameterized two-layer neural networks. In *Proceedings of the 36th International Conference on Machine Learning*, 2019b.

Jaehoon Lee, Lechao Xiao, Samuel Schoenholz, Yasaman Bahri, Roman Novak, Jascha Sohl-Dickstein, and Jeffrey Pennington. Wide neural networks of any depth evolve as linear models under gradient descent. In *Advances in Neural Information Processing Systems*, 2019.

Houssam Zenati, Alberto Bietti, Eustache Diemert, Julien Mairal, Matthieu Martin, and Pierre Gaillard. Efficient kernelized ucb for contextual bandits. In *Proceedings of The 25th International Conference on Artificial Intelligence and Statistics*, 2022.

Ha Manh Bui and Anqi Liu. Density-softmax: Efficient test-time model for uncertainty estimation and robustness under distribution shifts. In *Proceedings of the 41st International Conference on Machine Learning*, 2024.

Sudeep Salgia. Provably and practically efficient neural contextual bandits. In *Proceedings of the 40th International Conference on Machine Learning*, 2023.

Checklist

1. For all models and algorithms presented, check if you include:
   (a) A clear description of the mathematical setting, assumptions, algorithm, and/or model. [Yes]
   (b) An analysis of the properties and complexity (time, space, sample size) of any algorithm. [Yes]
   (c) (Optional) Anonymized source code, with specification of all dependencies, including external libraries. [Yes]
2. For any theoretical claim, check if you include:
   (a) Statements of the full set of assumptions of all theoretical results. [Yes]
   (b) Complete proofs of all theoretical results. [Yes]
   (c) Clear explanations of any assumptions. [Yes]
3. For all figures and tables that present empirical results, check if you include:
   (a) The code, data, and instructions needed to reproduce the main experimental results (either in the supplemental material or as a URL). [Yes]
   (b) All the training details (e.g., data splits, hyperparameters, how they were chosen). [Yes]
   (c) A clear definition of the specific measure or statistics and error bars (e.g., with respect to the random seed after running experiments multiple times). [Yes]
   (d) A description of the computing infrastructure used. (e.g., type of GPUs, internal cluster, or cloud provider). [Yes]
4. If you are using existing assets (e.g., code, data, models) or curating/releasing new assets, check if you include:
   (a) Citations of the creator If your work uses existing assets. [Yes]
   (b) The license information of the assets, if applicable. [Not Applicable]
   (c) New assets either in the supplemental material or as a URL, if applicable. [Not Applicable]
   (d) Information about consent from data providers/curators. [Not Applicable]
   (e) Discussion of sensible content if applicable, e.g., personally identifiable information or offensive content. [Not Applicable]
5. If you used crowdsourcing or conducted research with human subjects, check if you include:
   (a) The full text of instructions given to participants and screenshots. [Not Applicable]
   (b) Descriptions of potential participant risks, with links to Institutional Review Board (IRB) approvals if applicable. [Not Applicable]
   (c) The estimated hourly wage paid to participants and the total amount spent on participant compensation. [Not Applicable]

Variance-Aware Linear UCB with Deep Representation for Neural Contextual Bandits (Supplementary Material)

Broader impacts. Contextual bandits involve several artificial intelligence applications, e.g., personal healthcare, finance, recommendation systems, etc. There has been growing interest in using DNN to improve bandits algorithms. Our Neural-σ^2-LinUCB improves the quality of such models by having a lower calibration error and a lower regret. This could particularly benefit the aforementioned high-stake applications.

Limitations:

1. **The gap between oracle and practical algorithm.** Although showing a better regret guarantee in theory and experiments than other related methods, there is still a gap between our oracle and practical algorithm as we need to estimate the upper bound variance with a known reward range $[a, b]$ and the magnitude R.
2. **The gap between theory and experiments**. Similar to the literature on neural bandits, although α_t has a specific form in Theorem 4.5, in experiments, we set it to be constant for a fair comparison with other baselines. Specifically, the exploration rate α_t in Xu et al. (2022a) (γ_t in Zhou et al. (2020)) is also set to be a constant instead of the true value in the theorems. We also compare with the true value α_t in the theory in Figure 16.

Remediation. Given the aforementioned limitations, we encourage people who extend our work to proactively confront the model design and parameters to desired behaviors in real-world use cases.

Future work. We plan to tackle Neural-σ^2-LinUCB limitation, reduce assumptions in theory, and add more estimation techniques for σ_t^2 to enhance the quality of the practical algorithm.

Reproducibility. The source code to reproduce our results is available at https://github.com/Angie-Lab-JHU/neuralVarLinUCB. We provide all proofs in Appendix A, experimental settings, and detailed results in Appendix B.

A Proofs

A.1 Proof of Theorem 3.2

Proof. By the definition of the noise random variable, i.e., $\mathbb{E}[\xi_t \mid \mathbf{x}_{1:t,a_{1:t}}, \xi_{1:t-1}] = 0$ in Equation 3, we have

$$Var(\xi_t \mid \mathbf{x}_{1:t,a_{1:t}}, \xi_{1:t-1}) = \mathbb{E}[\xi_t^2 \mid \mathbf{x}_{1:t,a_{1:t}}, \xi_{1:t-1}] - \{\mathbb{E}[\xi_t \mid \mathbf{x}_{1:t,a_{1:t}}, \xi_{1:t-1}]\}^2 \tag{13}$$

$$= \mathbb{E}[\xi_t^2 \mid \mathbf{x}_{1:t,a_{1:t}}, \xi_{1:t-1}]. \tag{14}$$

Since $\mathbb{E}[\xi_t \mid \mathbf{x}_{1:t,a_{1:t}}, \xi_{1:t-1}] = 0$ by Definition in Equation 3, applying the Law of total variance, we get

$$Var(\xi_t) = \mathbb{E}\left[Var(\xi_t \mid \mathbf{x}_{1:t,a_{1:t}}, \xi_{1:t-1})\right] + Var(\mathbb{E}\left[\xi_t \mid \mathbf{x}_{1:t,a_{1:t}}, \xi_{1:t-1}\right]) \tag{15}$$

$$= \mathbb{E}\left[\mathbb{E}[\xi_t^2 \mid \mathbf{x}_{1:t,a_{1:t}}, \xi_{1:t-1}]\right] + Var(0) = \mathbb{E}[\xi_t^2 \mid \mathbf{x}_{1:t,a_{1:t}}, \xi_{1:t-1}]. \tag{16}$$

On the other hand, by the Law of Expectation, we have

$$\mathbb{E}[\xi_t] = \mathbb{E}\left[\mathbb{E}[\xi_t \mid \mathbf{x}_{1:t,a_{1:t}}, \xi_{1:t-1}]\right] = \mathbb{E}[0] = 0. \tag{17}$$

Using definition in Equation 2, i.e., $r_{t,a_t} = h(\mathbf{x}_{t,a_t}) + \xi_t$, since $\mathbb{E}[h(\mathbf{x}_{t,a_t})] = h(\mathbf{x}_{t,a_t})$ and $Var(h(\mathbf{x}_{t,a_t})) = 0$, by the linearity of expectation, we obtain

$$\mathbb{E}[r_{t,a_t}] = \mathbb{E}[h(\mathbf{x}_{t,a_t})] + \mathbb{E}[\xi_t] = h(\mathbf{x}_{t,a_t}), \tag{18}$$

and by the sum of the variance of independent variables, yielding

$$Var(r_{t,a_t}) = Var(h(\mathbf{x}_{t,a_t})) + Var(\xi_t) + 2 \cdot Cov(h(\mathbf{x}_{t,a_t}), \xi_t) \tag{19}$$

$$= \mathbb{E}[\xi_t^2 \mid \mathbf{x}_{1:t,a_{1:t}}, \xi_{1:t-1}] \tag{20}$$

of Theorem 3.2. □

A.2 Proof of Theorem 3.3

Proof. Let us first consider a random variable X is restricted in the interval $[0, 1]$. Note that for all $x \in [0, 1]$, we always have $x^2 \leq x$, yielding $\mathbb{E}[X^2] \leq \mathbb{E}[X]$. Therefore, we have

$$Var(X) = \mathbb{E}[X^2] - \mathbb{E}[X]^2 \leq \mathbb{E}[X] - \mathbb{E}[X]^2 = \mathbb{E}[X](1 - \mathbb{E}[X]). \tag{21}$$

To generalize to intervals $[a, b]$ with $b > a$, consider r_{t,a_t} in Theorem 3.2 restricted to $[a, b]$. Let us define

$$X = \frac{r_{t,a_t} - a}{b - a}, \tag{22}$$

which is restricted in $[0, 1]$. Equivalently, $r_{t,a_t} = (b - a)X + a$, thus, we get

$$Var(r_{t,a_t}) = \mathbb{E}[\xi_t^2 \mid \mathbf{x}_{1:t,a_{1:t}}, \xi_{1:t-1}] = (b - a)^2 \cdot Var(X) \leq (b - a)^2 \cdot \mathbb{E}[X](1 - \mathbb{E}[X]), \tag{23}$$

where the inequality is based on the first result. Hence, by substituting

$$\mathbb{E}[X] = \frac{h(\mathbf{x}_{t,a_t}) - a}{b - a}, \tag{24}$$

we obtain the first bound

$$Var(r_{t,a_t}) = \mathbb{E}[\xi_t^2 \mid \mathbf{x}_{1:t,a_{1:t}}, \xi_{1:t-1}] \leq (b - a)^2 \cdot \frac{h(\mathbf{x}_{t,a_t}) - a}{b - a} \left(1 - \frac{h(\mathbf{x}_{t,a_t}) - a}{b - a}\right) \tag{25}$$

$$= (b - a)^2 \cdot \frac{h(\mathbf{x}_{t,a_t}) - a}{b - a} \cdot \frac{b - h(\mathbf{x}_{t,a_t})}{b - a} \tag{26}$$

$$= (h(\mathbf{x}_{t,a_t}) - a)(b - h(\mathbf{x}_{t,a_t})) \tag{27}$$

of Theorem 3.3, To show the second bound, let us consider the function

$$\mathcal{F}(h(\mathbf{x}_{t,a_t})) = (h(\mathbf{x}_{t,a_t}) - a)(b - h(\mathbf{x}_{t,a_t})) = -h(\mathbf{x}_{t,a_t})^2 + (a + b) \cdot h(\mathbf{x}_{t,a_t}) - ab. \tag{28}$$

Since $\mathcal{F}(h(\mathbf{x}_{t,a_t}))$ is a quadratic function, we know that $\mathcal{F}(h(\mathbf{x}_{t,a_t}))$ is maximized at $h(\mathbf{x}_{t,a_t}) = \frac{a+b}{2}$, yielding

$$\mathcal{F}(h(\mathbf{x}_{t,a_t})) \leq \frac{(b - a)^2}{4}. \tag{29}$$

On the other hand, by the definition in Equation 3, we have $-R \leq \xi_t \leq R$, by the reward definition in Equation 2, we obtain the second bound

$$\mathcal{F}(h(\mathbf{x}_{t,a_t})) \leq \frac{(b - a)^2}{4} = \frac{\left[(h(\mathbf{x}_{t,a_t}) + R) - (h(\mathbf{x}_{t,a_t}) - R)\right]^2}{4} = R^2 \tag{30}$$

of Theorem 3.3. □

A.3 Proof of Theorem 4.5 and Corollary 4.6

This proof is based on the following provable Lemma:

Lemma A.1. *(The elliptical potential lemma (Abbasi-yadkori et al., 2011)). Let $\{\mathbf{x}_t\}_{t=1}^{\infty}$ be a sequence in $\mathbb{R}^d$ and $\lambda > 0$. Suppose $||\mathbf{x}_t||_2 \leq G$ and $\lambda \geq \max\{1, G^2\}$ for some $G > 0$. Let $\mathbf{A}_t = \lambda\mathbf{I} + \sum_{s=1}^{t} \mathbf{x}_t\mathbf{x}_t^\top$. Then, we have*

$$\det(\mathbf{A}_t) \leq (\lambda + tG^2/d)^d, \text{ and, } \sum_{t=1}^{T} ||\mathbf{x}_t||^2_{\mathbf{A}_{t=1}^{-1}} \leq 2\log\frac{\det(\mathbf{A}_T)}{\det(\lambda\mathbf{I})} \leq 2d\log(1 + TG^2/(\lambda d)).$$

Lemma A.2. *(Approximate reward by a linear function around initial point (Xu et al., 2022a)). Suppose Assumption 4.4 holds, for matrix $\nabla_{\mathbf{w}}\boldsymbol{\phi}(\mathbf{x}; \mathbf{w}) \in \mathbb{R}^{d\times p}$, there exists $\mathbf{w}^* \in \mathbb{R}^p$ s.t. $||\mathbf{w}^* - \mathbf{w}^{(0)}||_2 \leq (1/\sqrt{m})\sqrt{(\mathbf{r} - \tilde{\mathbf{r}})^\top\mathbf{H}^{-1}(\mathbf{r} - \tilde{\mathbf{r}})}$ and*

$$r(\mathbf{x}_{t,k}) = \boldsymbol{\theta}^{*\top}\boldsymbol{\phi}(\mathbf{x}_{t,k}; \mathbf{w}_{t-1}) + \boldsymbol{\theta}_0^\top\nabla_{\mathbf{w}^{(0)}}\boldsymbol{\phi}(\mathbf{x}_{t,k}; \mathbf{w}^{(0)})(\mathbf{w}^* - \mathbf{w}^{(0)}),$$

for all $k \in [K]$ and $t \in [T]$.

Lemma A.3. *(Upper bounds of the neural network's output and its gradient (Xu et al., 2022a)). Suppose Assumptions 4.4 holds, then for any round index $t \in [T]$, suppose it is in the q-th epoch, i.e., $t = (q-1)/H + i$ for some $i \in [H]$. If the step size η_q satisfies*

$$\eta \leq \frac{C_0}{d^2 m n T^{5.5} L^6 \log(TK/\delta)},$$

and the width of the neural network satisfies

$$m \geq \max\{L \log(TK/\delta), dL^2 \log(m/\delta), \delta^{-6} H^{18} L^{16} \log^3(TK)\},$$

then, with probability at least $1-\delta$ we have

$$||\mathbf{w}_t - \mathbf{w}^{(0)}||_2 \leq \frac{\delta^{3/2}}{m^{1/2} T n^{9/2} L^6 \log^3(m)},$$
$$||\nabla_{\mathbf{w}^{(0)}} \boldsymbol{\phi}(\mathbf{x}_{t,a_k}; \mathbf{w}^{(0)})||_F \leq C_1 \sqrt{dLm},$$
$$||\boldsymbol{\phi}(\mathbf{x}; \mathbf{w}_t)||_2 \leq \sqrt{d \log(n) \log(TK/\delta)},$$

for all $t \in [T]$, $k \in [K]$.

Lemma A.4. *(The confidence bound of our estimation). Suppose Assumption 4.4 holds and assume $||\boldsymbol{\theta}^*||_2 \leq M$, for some positive constant $M > 0$, then for any $\delta \in (0,1)$, with probability at least $1-\delta$, the distance between the estimate weights vector $\boldsymbol{\theta}_t$ and $\boldsymbol{\theta}^*$ can be bounded as follows*

$$\left\| \boldsymbol{\theta}_t - \boldsymbol{\theta}^* - \mathbf{A}_t^{-1} \sum_{s=1}^{t} \frac{\phi(\mathbf{x}_{s,a_s}; \mathbf{w}_{s-1})}{\bar{\sigma}_s} \boldsymbol{\theta}_0^\top \nabla_{\mathbf{w}^{(0)}} \frac{\boldsymbol{\phi}(\mathbf{x}_{s,a_s}; \mathbf{w}^{(0)})}{\bar{\sigma}_s} (\mathbf{w}^* - \mathbf{w}^{(0)}) \right\|_{\mathbf{A}_t}$$
$$\leq 8\sqrt{d \log\left(1 + td(\log HK)/(\bar{\sigma}_t^2 d\lambda)\right) \log(4t^2/\delta)} + 4R/\bar{\sigma}_t \log(4t^2/\delta) + \lambda^{1/2} M,$$

for any $t \in [T]$. The proof is in Appendix A.4.

Lemma A.5. *(Small neighborhood of the initialization point (Cao and Gu, 2019)). Let $\mathbf{w}, \mathbf{w}'$ be in the neighborhood of $\mathbf{w}_0$, i.e., $\mathbf{w}, \mathbf{w}' \in \mathbb{B}(\mathbf{w}_0, \omega)$ for some $\omega > 0$. Consider the neural network defined in Equation 5, if the width m and the radius ω of the neighborhood satisfy*

$$m \geq C_0 \max\{dL^2 \log(m/\delta), \omega^{-4/3} L^{8/3} \log(TK) \log(m/(\omega\delta))\}, \text{ and, } \omega \leq C_1 L^{-5} (\log m)^{-3/2},$$

then for all $\mathbf{x} \in \{\mathbf{x}_{t,k}\}_{t \in [T], k \in [K]}$, with probability at least $1-\delta$ it holds that

$$|\phi_j(\mathbf{x}; \mathbf{w}) - \hat{\phi}_j \mathbf{x}; \mathbf{w})| \leq C_2 \omega^{4/3} L^3 d^{-1/2} \sqrt{m \log m},$$

where $\hat{\phi}_j(\mathbf{x}; \mathbf{w})$ is the linearization of $\phi_j(\mathbf{x}; \mathbf{w})$ at $\mathbf{w}'$ defined as follow:

$$\hat{\phi}_j(\mathbf{x}; \mathbf{w}) = \phi_j(\mathbf{x}; \mathbf{w}') + \langle \nabla_{\mathbf{w}} \phi_j(\mathbf{x}; \mathbf{w}'), \mathbf{w} - \mathbf{w}' \rangle .$$

Lemma A.6. *(Extra term of confidence bound (Xu et al., 2022a)). Assume that $\mathbf{A}_t = \lambda \mathbf{I} + \sum_{s=1}^{t} \boldsymbol{\phi}_s \boldsymbol{\phi}_s^\top$, where $\boldsymbol{\phi}_t \in \mathbb{R}^d$ and $||\boldsymbol{\phi}_t|| \leq G$ for all $t \geq 1$ and some constants $\lambda, G > 0$. Let $\{\zeta_t\}_{t=1,\cdots}$ be a real-value sequence s.t. $|\zeta_t| \leq U$ for some constant $U > 0$. Then we have*

$$\left\| \mathbf{A}_t^{-1} \sum_{s=1}^{t} \boldsymbol{\phi}_s \zeta_s \right\|_2 \leq 2Ud, \quad \forall t = 1, 2, \cdots .$$

A.3.1 Proof of Theorem 4.5

Proof. For a time horizon T, without loss of generality, assume $T = QH$ for epoch number Q, episode length H to backpropagte f, then we have the regret as follows

$$\text{Regret}(T) = \mathbb{E}\left[\sum_{t=1}^{T} \left(r_{t,a_t^*} - r_{t,a_t}\right)\right] = \mathbb{E}\left[\sum_{q=1}^{Q} \sum_{i=1}^{H} \left(r_{qH_{i+1}, a^*_{qH_{i+1}}} - r_{qH_{i+1}, a_{qH_{i+1}}}\right)\right], \tag{31}$$

i.e., we rewrite the time index $t = qH + 1$ as the i-th iteration in the q-th epoch. By Lemma A.2, there exists vector $\mathbf{w}^* \in \mathbb{R}^p$ s.t. we can write the expectation of the reward generating function as a linear function

$$\begin{aligned} h(\mathbf{x}_{t,a_t^*}) - h(\mathbf{x}_{t,a_t}) &= \boldsymbol{\theta}_0^\top \left[\nabla_{\mathbf{w}^{(0)}} \boldsymbol{\phi}(\mathbf{x}_{t,a_t^*}; \mathbf{w}^{(0)}) - \nabla_{\mathbf{w}^{(0)}} \boldsymbol{\phi}(\mathbf{x}_{t,a_t}; \mathbf{w}^{(0)})\right](\mathbf{w}^* - \mathbf{w}^{(0)}) \\ &\quad + \boldsymbol{\theta}^{*\top}\left[\boldsymbol{\phi}(\mathbf{x}_{t,a_t^*}; \mathbf{w}_{t-1}) - \boldsymbol{\phi}(\mathbf{x}_{t,a_t}; \mathbf{w}_{t-1})\right] \qquad (32) \\ &= \boldsymbol{\theta}_0^\top \left[\nabla_{\mathbf{w}^{(0)}} \boldsymbol{\phi}(\mathbf{x}_{t,a_t^*}; \mathbf{w}^{(0)}) - \nabla_{\mathbf{w}^{(0)}} \boldsymbol{\phi}(\mathbf{x}_{t,a_t}; \mathbf{w}^{(0)})\right](\mathbf{w}^* - \mathbf{w}^{(0)}) \\ &\quad + \boldsymbol{\theta}_{t-1}^\top\left[\boldsymbol{\phi}(\mathbf{x}_{t,a_t^*}; \mathbf{w}_{t-1}) - \boldsymbol{\phi}(\mathbf{x}_{t,a_t}; \mathbf{w}_{t-1})\right] - (\boldsymbol{\theta}_{t-1} - \boldsymbol{\theta}^*)^\top\left[\boldsymbol{\phi}(\mathbf{x}_{t,a_t^*}; \mathbf{w}_{t-1}) - \boldsymbol{\phi}(\mathbf{x}_{t,a_t}; \mathbf{w}_{t-1})\right]. \qquad (33) \end{aligned}$$

For the first term, we can bound as follows

$$\begin{aligned} &\boldsymbol{\theta}_0^\top \left[\nabla_{\mathbf{w}^{(0)}} \boldsymbol{\phi}(\mathbf{x}_{t,a_t^*}; \mathbf{w}^{(0)}) - \nabla_{\mathbf{w}^{(0)}} \boldsymbol{\phi}(\mathbf{x}_{t,a_t}; \mathbf{w}^{(0)})\right](\mathbf{w}^* - \mathbf{w}^{(0)}) \\ &\leq ||\boldsymbol{\theta}_0||_2 \left\|\nabla_{\mathbf{w}^{(0)}} \boldsymbol{\phi}(\mathbf{x}_{t,a_t^*}; \mathbf{w}^{(0)}) - \nabla_{\mathbf{w}^{(0)}} \boldsymbol{\phi}(\mathbf{x}_{t,a_t}; \mathbf{w}^{(0)})\right\|_2 ||\mathbf{w}^* - \mathbf{w}^{(0)}||_2 \qquad (34) \\ &\leq \ell_{Lip}||\boldsymbol{\theta}_0||_2||\mathbf{x}_{t,a_t^*} - \mathbf{x}_{t,a_t}||_2||\mathbf{w}^* - \mathbf{w}^{(0)}||_2 \text{ (by Assumption 4.2).} \qquad (35) \end{aligned}$$

For the second term, we can bound as follows, by the UCB algorithm, we have

$$\boldsymbol{\theta}_{t-1}^\top \boldsymbol{\phi}(\mathbf{x}_{t,a_t^*}; \mathbf{w}_{t-1}) + \alpha_t||\boldsymbol{\phi}(\mathbf{x}_{t,a_t^*}; \mathbf{w}_{t-1})||_{\mathbf{A}_{t-1}^{-1}} \leq \boldsymbol{\theta}_{t-1}^\top \boldsymbol{\phi}(\mathbf{x}_{t,a_t}; \mathbf{w}_{t-1}) + \alpha_t||\boldsymbol{\phi}(\mathbf{x}_{t,a_t}; \mathbf{w}_{t-1})||_{\mathbf{A}_{t-1}^{-1}}, \qquad (36)$$

so, we get

$$\boldsymbol{\theta}_{t-1}^\top\left[\boldsymbol{\phi}(\mathbf{x}_{t,a_t^*}; \mathbf{w}_{t-1}) - \boldsymbol{\phi}(\mathbf{x}_{t,a_t}; \mathbf{w}_{t-1})\right] \leq \alpha_t||\boldsymbol{\phi}(\mathbf{x}_{t,a_t}; \mathbf{w}_{t-1})||_{\mathbf{A}_{t-1}^{-1}} - \alpha_t||\boldsymbol{\phi}(\mathbf{x}_{t,a_t^*}; \mathbf{w}_{t-1})||_{\mathbf{A}_{t-1}^{-1}}. \qquad (37)$$

For the last term, we need to prove that the estimate of weights parameter $\boldsymbol{\theta}_{t-1}$ lies in a confidence ball centered at $\boldsymbol{\theta}^*$. For the ease of notation, we define

$$\mathbf{M}_t = \mathbf{A}_t^{-1}\sum_{s=1}^{t} \frac{\boldsymbol{\phi}(\mathbf{x}_{s,a_s}; \mathbf{w}_{s-1})}{\bar{\sigma}_s}\boldsymbol{\theta}_0^\top \nabla_{\mathbf{w}^{(0)}} \frac{\boldsymbol{\phi}(\mathbf{x}_{s,a_s}; \mathbf{w}^{(0)})}{\bar{\sigma}_s})(\mathbf{w}^* - \mathbf{w}^{(0)}). \qquad (38)$$

Then the last term can be bounded as follows

$$\begin{aligned} &-(\boldsymbol{\theta}_{t-1} - \boldsymbol{\theta}^*)^\top\left[\boldsymbol{\phi}(\mathbf{x}_{t,a_t^*}; \mathbf{w}_{t-1}) - \boldsymbol{\phi}(\mathbf{x}_{t,a_t}; \mathbf{w}_{t-1})\right] \\ &= -(\boldsymbol{\theta}_{t-1} - \boldsymbol{\theta}^* - \mathbf{M}_{t-1})^\top \boldsymbol{\phi}(\mathbf{x}_{t,a_t^*}; \mathbf{w}_{t-1}) + (\boldsymbol{\theta}_{t-1} - \boldsymbol{\theta}^* - \mathbf{M}_{t-1})^\top \boldsymbol{\phi}(\mathbf{x}_{t,a_t}; \mathbf{w}_{t-1}) \\ &\quad - \mathbf{M}_{t-1}^\top\left[\boldsymbol{\phi}(\mathbf{x}_{t,a_t^*}; \mathbf{w}_{t-1}) - \boldsymbol{\phi}(\mathbf{x}_{t,a_t}; \mathbf{w}_{t-1})\right] \qquad (39) \\ &\leq ||\boldsymbol{\theta}_{t-1} - \boldsymbol{\theta}^* - \mathbf{M}_{t-1}||_{\mathbf{A}_{t-1}}||\boldsymbol{\phi}(\mathbf{x}_{t,a_t^*}; \mathbf{w}_{t-1})||_{\mathbf{A}_{t-1}^{-1}} + ||\boldsymbol{\theta}_{t-1} - \boldsymbol{\theta}^* - \mathbf{M}_{t-1}||_{\mathbf{A}_{t-1}}||\boldsymbol{\phi}(\mathbf{x}_{t,a_t}; \mathbf{w}_{t-1})||_{\mathbf{A}_{t-1}^{-1}} \\ &\quad + \left\|\mathbf{M}_{t-1}^\top\left[\boldsymbol{\phi}(\mathbf{x}_{t,a_t^*}; \mathbf{w}_{t-1} - \mathbf{x}_{t,a_t}; \mathbf{w}_{t-1})\right]\right\| \qquad (40) \\ &\leq \alpha_t||\boldsymbol{\phi}(\mathbf{x}_{t,a_t^*}; \mathbf{w}_{t-1})||_{\mathbf{A}_{t-1}^{-1}} + \alpha_t||\boldsymbol{\phi}(\mathbf{x}_{t,a_t}; \mathbf{w}_{t-1})||_{\mathbf{A}_{t-1}^{-1}} + ||\mathbf{M}_{t-1}||_2||\boldsymbol{\phi}(\mathbf{x}_{t,a_t^*}; \mathbf{w}_{t-1}) - \boldsymbol{\phi}(\mathbf{x}_{t,a_t}; \mathbf{w}_{t-1})||_2 \\ &\quad \text{(by Lemma A.4 and the choice of } \alpha_t). \qquad (41) \end{aligned}$$

Hence, we get

$$\begin{aligned} h(\mathbf{x}_{t,a_t^*}) - h(\mathbf{x}_{t,a_t}) &\leq \ell_{Lip}||\boldsymbol{\theta}_0||_2||\mathbf{x}_{t,a_t^*} - \mathbf{x}_{t,a_t}||_2||\mathbf{w}^* - \mathbf{w}^{(0)}||_2 + 2\alpha_t||\boldsymbol{\phi}(\mathbf{x}_{t,a_t}; \mathbf{w}_{t-1})||_{\mathbf{A}_{t-1}^{-1}} \\ &\quad + ||\mathbf{M}_{t-1}||_2||\boldsymbol{\phi}(\mathbf{x}_{t,a_t^*}; \mathbf{w}_{t-1}) - \boldsymbol{\phi}(\mathbf{x}_{t,a_t}; \mathbf{w}_{t-1})||_2. \qquad (42) \end{aligned}$$

Recall the linearization of ϕ_j in Lemma A.5, we have

$$\hat{\boldsymbol{\phi}}(\mathbf{x}, \mathbf{w}_{t-1}) = \boldsymbol{\phi}(\mathbf{x}, \mathbf{w}_0) + \nabla_{\mathbf{w}_0}\boldsymbol{\phi}(\mathbf{x}; \mathbf{w}_0)(\mathbf{w}_{t-1} - \mathbf{w}_0). \qquad (43)$$

Note that by the initialization, we have $\boldsymbol{\phi}(\mathbf{x}; \mathbf{w}_0) = \mathbf{0}$ for any $\mathbf{x} \in \mathbb{R}^d$. Thus, it holds that

$$\begin{aligned} \boldsymbol{\phi}(\mathbf{x}_{t,a_t^*}; \mathbf{w}_{t-1}) - \boldsymbol{\phi}(\mathbf{x}_{t,a_t}; \mathbf{w}_{t-1}) &= \boldsymbol{\phi}(\mathbf{x}_{t,a_t^*}; \mathbf{w}_{t-1}) + \boldsymbol{\phi}(\mathbf{x}_{t,a_t^*}; \mathbf{w}_0) + \boldsymbol{\phi}(\mathbf{x}_{t,a_t}; \mathbf{w}_0) - \boldsymbol{\phi}(\mathbf{x}_{t,a_t}; \mathbf{w}_{t-1}) \qquad (44) \\ &= \boldsymbol{\phi}(\mathbf{x}_{t,a_t^*}; \mathbf{w}_{t-1}) - \hat{\boldsymbol{\phi}}(\mathbf{x}_{t,a_t^*}; \mathbf{w}_{t-1}) + \nabla_{\mathbf{w}_0}\boldsymbol{\phi}(\mathbf{x}_{t,a_t^*}; \mathbf{w}_0)(\mathbf{w}_{t-1} - \mathbf{w}_0) \\ &\quad + \boldsymbol{\phi}(\mathbf{x}_{t,a_t}; \mathbf{w}_{t-1}) - \hat{\boldsymbol{\phi}}(\mathbf{x}_{t,a_t}; \mathbf{w}_{t-1}) - \nabla_{\mathbf{w}_0}\boldsymbol{\phi}(\mathbf{x}_{t,a_t}; \mathbf{w}_0)(\mathbf{w}_{t-1} - \mathbf{w}_0), \qquad (45) \end{aligned}$$

yielding

$$
\begin{aligned}
&||\phi(\mathbf{x}_{t,a_t^*};\mathbf{w}_{t-1}) - \phi(\mathbf{x}_{t,a_t};\mathbf{w}_{t-1})||_2 \\
&\leq ||\phi(\mathbf{x}_{t,a_t^*};\mathbf{w}_{t-1}) - \hat{\phi}(\mathbf{x}_{t,a_t^*};\mathbf{w}_{t-1})||_2 + ||\phi(\mathbf{x}_{t,a_t};\mathbf{w}_{t-1}) - \hat{\phi}(\mathbf{x}_{t,a_t};\mathbf{w}_{t-1})||_2 \\
&\quad + ||\left(\nabla_{\mathbf{w}_0}\boldsymbol{\phi}(\mathbf{x}_{t,a_t^*};\mathbf{w}_0) - \nabla_{\mathbf{w}_0}\boldsymbol{\phi}(\mathbf{x}_{t,a_t};\mathbf{w}_0)\right)(\mathbf{w}_{t-1} - \mathbf{w}_0)||_2 \qquad (46)\\
&\leq C_0\omega^{4/3}L^3d^{1/2}\sqrt{m\log m} + \ell_{\text{Lip}}||\mathbf{x}_{t,a_t^*} - \mathbf{x}_{t,a_t}||_2||\mathbf{w}_{t-1} - \mathbf{w}^{(0)}||_2 \text{ (by Lemma A.5 and Assumption 4.2).} \qquad (47)
\end{aligned}
$$

Plugging into Equation 42, we get

$$
\begin{aligned}
h(\mathbf{x}_{t,a_t^*}) - h(\mathbf{x}_{t,a_t}) &\leq \ell_{Lip}||\boldsymbol{\theta}_0||_2||\mathbf{x}_{t,a_t^*} - \mathbf{x}_{t,a_t}||_2||\mathbf{w}^* - \mathbf{w}^{(0)}||_2 + 2\alpha_t||\boldsymbol{\phi}(\mathbf{x}_{t,a_t};\mathbf{w}_{t-1})||_{\mathbf{A}_{t-1}^{-1}} \\
&\quad + ||\mathbf{M}_{t-1}||_2||\boldsymbol{\phi}(\mathbf{x}_{t,a_t^*};\mathbf{w}_{t-1}) - \boldsymbol{\phi}(\mathbf{x}_{t,a_t};\mathbf{w}_{t-1})||_2 \qquad (48)\\
&\leq \ell_{Lip}||\boldsymbol{\theta}_0||_2||\mathbf{x}_{t,a_t^*} - \mathbf{x}_{t,a_t}||_2||\mathbf{w}^* - \mathbf{w}^{(0)}||_2 + 2\alpha_t||\boldsymbol{\phi}(\mathbf{x}_{t,a_t};\mathbf{w}_{t-1})||_{\mathbf{A}_{t-1}^{-1}} \\
&\quad + ||\mathbf{M}_{t-1}||_2\left(C_0\omega^{4/3}L^3d^{1/2}\sqrt{m\log m} + \ell_{\text{Lip}}||\mathbf{x}_{t,a_t^*} - \mathbf{x}_{t,a_t}||_2||\mathbf{w}_{t-1} - \mathbf{w}^{(0)}||_2\right). \qquad (49)
\end{aligned}
$$

By Remark 4.1, we have $||\mathbf{x}_{t,a_t^*} - \mathbf{x}_{t,a_t}||_2 \leq 2$. By Lemma A.2 and Lemma A.3, we have

$$
||\mathbf{w}^* - \mathbf{w}^{(0)}||_2 \leq \sqrt{1/m(\mathbf{r}-\tilde{\mathbf{r}})^\top\mathbf{H}^{-1}(\mathbf{r}-\tilde{\mathbf{r}})} \quad \text{and} \quad ||\mathbf{w}_t - \mathbf{w}^{(0)}||_2 \leq \frac{\delta^{3/2}}{m^{1/2}Tn^{9/2}L^6\log^3(m)}. \qquad (50)
$$

Additionally, since the entries of $\boldsymbol{\theta}_0$ are i.i.d. generated from $\mathcal{N}(0, 1/d)$, we have $||\boldsymbol{\theta}_0||_2 \leq 2(2 + \sqrt{d^{-1}\log(1/\delta)})$ with probability at least $1-\delta$ for any $\delta > 0$. By Lemma A.3, we have $||\nabla_{\mathbf{w}^{(0)}}\boldsymbol{\phi}(\mathbf{x}_{t,a_t};\mathbf{w}^{(0)})||_F \leq C_1\sqrt{dm}$. Therefore,

$$
|\boldsymbol{\theta}_0^\top\nabla_{\mathbf{w}^{(0)}}\boldsymbol{\phi}(\mathbf{x}_{s,a_s};\mathbf{w}^{(0)})(\mathbf{w}^* - \mathbf{w}^{(0)})| \leq C_2 d\sqrt{\log(1/\delta)(\mathbf{r}-\tilde{\mathbf{r}})^\top\mathbf{H}(\mathbf{r}-\tilde{\mathbf{r}})}. \qquad (51)
$$

Then, by the definition of $\mathbf{M}_t$ and Lemma A.6, we have

$$
||\mathbf{M}_{t-1}||_2 \leq C_3 d^2\sqrt{\log(1/\delta)(\mathbf{r}-\tilde{\mathbf{r}})^\top\mathbf{H}^{-1}(\mathbf{r}-\tilde{\mathbf{r}})}. \qquad (52)
$$

Continue plugging into Equation 48, we get

$$
\begin{aligned}
h(\mathbf{x}_{t,a_t^*}) - h(\mathbf{x}_{t,a_t}) &\leq C_4\ell_{Lip}m^{-1/2}\sqrt{\log(1/\delta)(\mathbf{r}-\tilde{\mathbf{r}})^\top\mathbf{H}^{-1}(\mathbf{r}-\tilde{\mathbf{r}})} + 2\alpha_t||\boldsymbol{\phi}(\mathbf{x}_{t,a_t};\mathbf{w}_{t-1})||_{\mathbf{A}_{t-1}^{-1}} \\
&\quad + \left(C_0\omega^{4/3}L^3d^{1/2}\sqrt{m\log m} + \frac{2\ell_{\text{Lip}}\delta^{3/2}}{m^{1/2}Tn^{9/2}L^6\log^3(m)}\right)C_3d^2\sqrt{\log(1/\delta)(\mathbf{r}-\tilde{\mathbf{r}})^\top\mathbf{H}^{-1}(\mathbf{r}-\tilde{\mathbf{r}})}. \qquad (53)
\end{aligned}
$$

Combining with the fact that $\omega = \mathcal{O}\left(m^{-1/2}||\mathbf{r}-\tilde{\mathbf{r}}||_{\mathbf{H}^{-1}}\right)$ by Lemma A.2, using Cauchy's inequality, we obtain

$$
\begin{aligned}
\text{Regret}(T) &\leq \sqrt{\sum_{t=1}^{T}4\alpha_t^2\bar{\sigma}_t^2||\boldsymbol{\phi}(\mathbf{x}_{t,a_t};\mathbf{w}_{t-1})/\bar{\sigma}_t||^2_{\mathbf{A}_{t-1}^{-1}}} + C_4\ell_{\text{Lip}}m^{-1/2}T\sqrt{\log(1/\delta)}||\mathbf{r}-\tilde{\mathbf{r}}||_{\mathbf{H}^{-1}} \\
&\quad + \left(\frac{C_0TL^3d^{1/2}\sqrt{\log m}||\mathbf{r}-\tilde{\mathbf{r}}||^{4/3}_{\mathbf{H}^{-1}}}{m^{1/6}} + \frac{2\ell_{\text{Lip}}\delta^{3/2}}{m^{1/2}n^{9/2}L^6\log^3(m)}\right)C_3d^2\sqrt{\log(1/\delta)}||\mathbf{r}-\tilde{\mathbf{r}}||_{\mathbf{H}^{-1}}. \qquad (54)
\end{aligned}
$$

Let $\sigma_{T_{\min}} = \min_{t\in[T]}\sigma_t$, since $\bar{\sigma}_t = \max\{R/\sqrt{d}, \sigma_t\}$, yielding $\bar{\sigma}_{T_{\min}} \geq R/\sqrt{d}$, therefore, we have

$$
\begin{aligned}
\alpha_t &= 8\sqrt{d\log\left(1 + td(\log HK)/(\bar{\sigma}_t^2 d\lambda)\right)\log(4t^2/\delta)} + 4R/\bar{\sigma}_t\log(4t^2/\delta) + \lambda^{1/2}M \qquad (55)\\
&\leq 8\sqrt{d\log\left(1 + Td(\log HK)/(\lambda R^2)\right)\log(4T^2/\delta)} + 4R/\sigma_{T_{\min}}\log(4T^2/\delta) + \lambda^{1/2}M. \qquad (56)
\end{aligned}
$$

Since $\bar{\sigma}_t^2 = \max\{R^2/d, \sigma_t^2\} \leq R^2/d + \sigma_t^2$, $\sqrt{|x|+|y|} \leq \sqrt{|x|} + \sqrt{|y|}$, using the upper bound of α_t in Lemma A.4 and Lemma A.1, we finally obtain

$$\begin{aligned}\text{Regret}(T) \leq C_5 \alpha_T \sqrt{\left(TR^2 + d\sum_{t=1}^{T}\sigma_t^2\right)\log(1+TG^2/(\lambda R^2))} \\ + C_6 \ell_{Lip} L^3 d^{5/2} m^{-1/6} T\sqrt{\log m \log(1/\delta)\log(TK/\delta)}||\mathbf{r}-\tilde{\mathbf{r}}||_{\mathbf{H}^{-1}}\end{aligned} \tag{57}$$

of Theorem 4.5. □

A.3.2 Proof of Corollary 4.6

Proof. It directly follows the result in Theorem 4.5 by using the Big $\mathcal{O}$ notation. □

A.4 Proof of Lemma A.4

This proof is based on the following provable Lemma of Zhou et al. (2021):

Lemma A.7. *(Bernstein inequality for vector-valued martingales (Zhou et al., 2021)). Let $\{\mathcal{G}_t\}_{t=1}^{\infty}$ be a filtration, $\{x_t, \xi_t\}_{t\geq 1}$ a stochastic process so that $x_t \in \mathbb{R}^d$ is $\mathcal{G}_t$-measurable and $\xi_t \in \mathbb{R}$ is $\mathcal{G}_{t+1}$-measurable. Fix $R, G, \sigma, \lambda > 0$, $\boldsymbol{\theta}^* \in \mathbb{R}^d$. For $t \geq 1$ let $r_t = \langle \boldsymbol{\theta}^*, x_t\rangle + \xi_t$ and suppose that ξ_t, x_t also satisfy*

$$|\xi_t| \leq R, \quad \mathbb{E}[\xi_t|\mathcal{G}_t] = 0, \quad \mathbb{E}[\xi_t^2|\mathcal{G}_t] \leq \sigma^2, \quad ||x_t||_2 \leq G.$$

Then, for any $0 \leq \delta \leq 1$, with prob. at least $1-\delta$, we have

$$\forall t > 0, \quad \left\|\sum_{i=1}^{t} x_i \xi_i\right\|_{V_t^{-1}} \leq \beta_t, \quad ||\boldsymbol{\theta}_t - \boldsymbol{\theta}^*||_{V_t} \leq \beta_t + \sqrt{\lambda}||\boldsymbol{\theta}^*||_2,$$

where for $t \geq 1$, $\boldsymbol{\theta}_t = V_t^{-1} b_t$, $V_t = \lambda\mathbf{I} + \sum_{i=1}^{t} x_i x_i^\top$, $b_t = \sum_{i=1}^{t} r_i x_i$ and

$$\beta_t = 8\sigma\sqrt{d\log(1+tG^2/(d\lambda))\log(4t^2/\delta)} + 4R\log(4t^2/\delta).$$

Now, we provide our proof of Lemma A.4 as follows:

Proof. Let $\boldsymbol{\Phi}_t = [\boldsymbol{\phi}(\mathbf{x}_{1,a_1}; \mathbf{w}_0), \cdots, \boldsymbol{\phi}(\mathbf{x}_{1,a_t}; \mathbf{w}_{t-1})] \in \mathbb{R}^{d\times t}$ be the collection of feature vectors of the chosen arms up to time t and $\mathbf{r}_t = (r_{1,a_1}, \cdots, r_{t,a_t})^\top$ be the concatenation of all received rewards. According to Algorithm 1, we have $\mathbf{A}_t = \lambda\mathbf{I} + \frac{\boldsymbol{\Phi}_t\boldsymbol{\Phi}_t^\top}{\bar{\sigma}_t^2}$ and thus

$$\boldsymbol{\theta}_t = \mathbf{A}_t^{-1}\mathbf{b}_t = \left(\lambda\mathbf{I} + \frac{\boldsymbol{\Phi}_t\boldsymbol{\Phi}_t^\top}{\bar{\sigma}_t^2}\right)^{-1}\frac{\boldsymbol{\Phi}_t\mathbf{r}_t}{\bar{\sigma}_t^2}. \tag{58}$$

By Lemma A.2, we can rewrite the reward as

$$r_{t,a_t} = \langle \boldsymbol{\theta}^*, \boldsymbol{\phi}(\mathbf{x}_{t,a_t}; \mathbf{w}_{t-1})\rangle + \boldsymbol{\theta}_0^\top \nabla_{\mathbf{w}^{(0)}}\boldsymbol{\phi}(\mathbf{x}_{t,a_t}; \mathbf{w}^{(0)})(\mathbf{w}^* - \mathbf{w}^{(0)}). \tag{59}$$

Therefore, it holds that

$$\boldsymbol{\theta}_t = \mathbf{A}_t^{-1}\frac{\boldsymbol{\Phi}_t\boldsymbol{\Phi}_t^\top}{\bar{\sigma}_t^2}\boldsymbol{\theta}^* + \mathbf{A}_t^{-1}\sum_{s=1}^{t}\frac{\boldsymbol{\phi}(\mathbf{x}_{s,a_s}; \mathbf{w}_{s-1})}{\bar{\sigma}_s}\left(\boldsymbol{\theta}_0^\top\nabla_{\mathbf{w}^{(0)}}\frac{\boldsymbol{\phi}(\mathbf{x}_{s,a_s}; \mathbf{w}^{(0)})}{\bar{\sigma}_s}(\mathbf{w}^* - \mathbf{w}^{(0)}) + \frac{\xi_s}{\bar{\sigma}_s}\right) \tag{60}$$

$$= \boldsymbol{\theta}^* - \lambda\mathbf{A}_t^{-1}\boldsymbol{\theta}^* + \mathbf{A}_t^{-1}\sum_{s=1}^{t}\frac{\boldsymbol{\phi}(\mathbf{x}_{s,a_s}; \mathbf{w}_{s-1})}{\bar{\sigma}_s}\left(\boldsymbol{\theta}_0^\top\nabla_{\mathbf{w}^{(0)}}\frac{\boldsymbol{\phi}(\mathbf{x}_{s,a_s}; \mathbf{w}^{(0)})}{\bar{\sigma}_s}(\mathbf{w}^* - \mathbf{w}^{(0)}) + \frac{\xi_s}{\bar{\sigma}_s}\right). \tag{61}$$

Then for any $\delta \in (0,1)$, by triangle inequality, we have

$$\left\|\boldsymbol{\theta}_t - \boldsymbol{\theta}^* - \mathbf{A}_t^{-1}\frac{\boldsymbol{\Phi}_t}{\bar{\sigma}_t}\boldsymbol{\Theta}_t\nabla_{\mathbf{w}^{(0)}}\frac{\boldsymbol{\Phi}_t}{\bar{\sigma}_t}(\mathbf{w}^* - \mathbf{w}^{(0)})\right\|_{\mathbf{A}_t} \leq \lambda||\boldsymbol{\theta}^*||_{\boldsymbol{A}_t^{-1}} + \left\|\frac{\boldsymbol{\Phi}_t}{\bar{\sigma}_t}\frac{\boldsymbol{\xi}_t}{\bar{\sigma}_t}\right\|_{\mathbf{A}_t^{-1}}. \tag{62}$$

Applying Lemma A.7 to

$$|\xi_t/\bar{\sigma}_t| \leq R/\bar{\sigma}_t, \quad \mathbb{E}[(\xi_t/\bar{\sigma}_t)|\mathcal{G}_t] = 0, \quad \mathbb{E}[(\xi_t/\bar{\sigma}_t)^2|\mathcal{G}_t] \leq 1, \quad ||\boldsymbol{\phi}(\mathbf{x};\mathbf{w})/\bar{\sigma}_t||_2 \leq G/\bar{\sigma}_t, \tag{63}$$

and the fact that $||\boldsymbol{\phi}(\mathbf{x};\mathbf{w})||_2 \leq C\sqrt{d\log HK}$, we get

$$\left\|\frac{\boldsymbol{\Phi}_t}{\bar{\sigma}_t}\frac{\boldsymbol{\xi}_t}{\bar{\sigma}_t}\right\|_{\mathbf{A}_t^{-1}} \leq 8\sqrt{d\log\left(1+tC^2d(\log HK)/(\bar{\sigma}_t^2 d\lambda)\right)\log(4t^2/\delta)} + 4R/\bar{\sigma}_t\log(4t^2/\delta). \tag{64}$$

Combining with the fact that $||\boldsymbol{\theta}^*||_{\mathbf{A}_t^{-1}} \leq \lambda^{-1/2}||\boldsymbol{\theta}^*||_2 \leq \lambda^{-1/2}M$ by Lemma A.2 and the assumption that $||\boldsymbol{\theta}^*||_2 \leq M$, we obtain

$$\begin{aligned}
&\left\|\boldsymbol{\theta}_t - \boldsymbol{\theta}^* - \mathbf{A}_t^{-1}\sum_{s=1}^{t}\frac{\boldsymbol{\phi}(\mathbf{x}_{s,a_s};\mathbf{w}_{s-1})}{\bar{\sigma}_s}\boldsymbol{\theta}_0^\top\nabla_{\mathbf{w}^{(0)}}\frac{\boldsymbol{\phi}(\mathbf{x}_{s,a_s};\mathbf{w}^{(0)})}{\bar{\sigma}_s}(\mathbf{w}^* - \mathbf{w}^{(0)})\right\|_{\mathbf{A}_t} \\
&\leq 8\sqrt{d\log\left(1+tC^2d(\log HK)/(\bar{\sigma}_t^2 d\lambda)\right)\log(4t^2/\delta)} + 4R/\bar{\sigma}_t\log(4t^2/\delta) + \lambda^{1/2}M
\end{aligned} \tag{65}$$

of Lemma A.4. □

A.5 Proof of Theorem 4.8

Proof. Firstly, we can bound the estimation error of the upper bound of the reward noise variance at round t as follows

$$\begin{aligned}
|\sigma_t^2 - \hat{\sigma}_t^2| &= \left|\left(b - \boldsymbol{\theta}^{*\top}\boldsymbol{\phi}(\mathbf{x}_{t,a_t};\mathbf{w}^*)\right)\left(\boldsymbol{\theta}^{*\top}\boldsymbol{\phi}(\mathbf{x}_{t,a_t};\mathbf{w}^*) - a\right) - \left(b - \boldsymbol{\theta}_t^\top\boldsymbol{\phi}(\mathbf{x}_{t,a_t};\mathbf{w})\right)\left(\boldsymbol{\theta}_t^\top\boldsymbol{\phi}(\mathbf{x}_{t,a_t};\mathbf{w}) - a\right)\right| && (66)\\
&= \Big|\Big(b\cdot\boldsymbol{\theta}^{*\top}\boldsymbol{\phi}(\mathbf{x}_{t,a_t};\mathbf{w}^*) - ba - \boldsymbol{\theta}^{*\top}\boldsymbol{\phi}(\mathbf{x}_{t,a_t};\mathbf{w}^*)\boldsymbol{\theta}^{*\top}\boldsymbol{\phi}(\mathbf{x}_{t,a_t};\mathbf{w}^*) + \boldsymbol{\theta}^{*\top}\boldsymbol{\phi}(\mathbf{x}_{t,a_t};\mathbf{w}^*)\cdot a\Big) \\
&\quad - \left(b\cdot\boldsymbol{\theta}_t^\top\boldsymbol{\phi}(\mathbf{x}_{t,a_t};\mathbf{w}) - ba - \boldsymbol{\theta}_t^\top\boldsymbol{\phi}(\mathbf{x}_{t,a_t};\mathbf{w})\boldsymbol{\theta}_t^\top\boldsymbol{\phi}(\mathbf{x}_{t,a_t};\mathbf{w}) + \boldsymbol{\theta}_t^\top\boldsymbol{\phi}(\mathbf{x}_{t,a_t};\mathbf{w})\cdot a\right)\Big| && (67)\\
&= \Big|(b+a)\left(\boldsymbol{\theta}^{*\top}\boldsymbol{\phi}(\mathbf{x}_{t,a_t};\mathbf{w}^*) - \boldsymbol{\theta}_t^\top\boldsymbol{\phi}(\mathbf{x}_{t,a_t};\mathbf{w})\right) \\
&\quad - \left[\boldsymbol{\theta}^{*\top}\boldsymbol{\phi}(\mathbf{x}_{t,a_t};\mathbf{w}^*)\boldsymbol{\theta}^{*\top}\boldsymbol{\phi}(\mathbf{x}_{t,a_t};\mathbf{w}^*) - \boldsymbol{\theta}_t^\top\boldsymbol{\phi}(\mathbf{x}_{t,a_t};\mathbf{w})\boldsymbol{\theta}_t^\top\boldsymbol{\phi}(\mathbf{x}_{t,a_t};\mathbf{w})\right]\Big| && (68)\\
&= \Big|(b+a)\left(\boldsymbol{\theta}^{*\top}\boldsymbol{\phi}(\mathbf{x}_{t,a_t};\mathbf{w}^*) - \boldsymbol{\theta}_t^\top\boldsymbol{\phi}(\mathbf{x}_{t,a_t};\mathbf{w})\right) \\
&\quad - \left[\left(\boldsymbol{\theta}^{*\top}\boldsymbol{\phi}(\mathbf{x}_{t,a_t};\mathbf{w}^*) - \boldsymbol{\theta}_t^\top\boldsymbol{\phi}(\mathbf{x}_{t,a_t};\mathbf{w})\right)\left(\boldsymbol{\theta}^{*\top}\boldsymbol{\phi}(\mathbf{x}_{t,a_t};\mathbf{w}^*) + \boldsymbol{\theta}_t^\top\boldsymbol{\phi}(\mathbf{x}_{t,a_t};\mathbf{w})\right)\right]\Big| && (69)\\
&= \left|\left(\boldsymbol{\theta}^{*\top}\boldsymbol{\phi}(\mathbf{x}_{t,a_t};\mathbf{w}^*) - \boldsymbol{\theta}_t^\top\boldsymbol{\phi}(\mathbf{x}_{t,a_t};\mathbf{w})\right)\left[b+a - \left(\boldsymbol{\theta}^{*\top}\boldsymbol{\phi}(\mathbf{x}_{t,a_t};\mathbf{w}^*) + \boldsymbol{\theta}_t^\top\boldsymbol{\phi}(\mathbf{x}_{t,a_t};\mathbf{w})\right)\right]\right| && (70)\\
&= \left|\boldsymbol{\theta}^{*\top}\boldsymbol{\phi}(\mathbf{x}_{t,a_t};\mathbf{w}^*) - \boldsymbol{\theta}_t^\top\boldsymbol{\phi}(\mathbf{x}_{t,a_t};\mathbf{w})\right|\left|b+a - \left(\boldsymbol{\theta}^{*\top}\boldsymbol{\phi}(\mathbf{x}_{t,a_t};\mathbf{w}^*) + \boldsymbol{\theta}_t^\top\boldsymbol{\phi}(\mathbf{x}_{t,a_t};\mathbf{w})\right)\right|. && (71)
\end{aligned}$$

By the triangle inequality and Hölder's inequality

$$\begin{aligned}
|\sigma_t^2 - \hat{\sigma}_t^2| &\leq \left(\left|\boldsymbol{\theta}^{*\top}\boldsymbol{\phi}(\mathbf{x}_{t,a_t};\mathbf{w}^*)\right| + \left|\boldsymbol{\theta}_t^\top\boldsymbol{\phi}(\mathbf{x}_{t,a_t};\mathbf{w})\right|\right)\left(|b+a| + \left|\boldsymbol{\theta}^{*\top}\boldsymbol{\phi}(\mathbf{x}_{t,a_t};\mathbf{w}^*) + \boldsymbol{\theta}_t^\top\boldsymbol{\phi}(\mathbf{x}_{t,a_t};\mathbf{w}\right|\right) && (72)\\
&\leq \left(\left|\boldsymbol{\theta}^{*\top}\boldsymbol{\phi}(\mathbf{x}_{t,a_t};\mathbf{w}^*)\right| + \left|\boldsymbol{\theta}_t^\top\boldsymbol{\phi}(\mathbf{x}_{t,a_t};\mathbf{w})\right|\right)\left(|b+a| + \left|\boldsymbol{\theta}^{*\top}\boldsymbol{\phi}(\mathbf{x}_{t,a_t};\mathbf{w}^*)\right| + \left|\boldsymbol{\theta}_t^\top\boldsymbol{\phi}(\mathbf{x}_{t,a_t};\mathbf{w}\right|\right) && (73)\\
&\leq \left(\left|\boldsymbol{\theta}^{*\top}\boldsymbol{\phi}(\mathbf{x}_{t,a_t};\mathbf{w}^*)\right| + \|\boldsymbol{\theta}_t\|_2\|\boldsymbol{\phi}(\mathbf{x}_{t,a_t};\mathbf{w})\|\right)\left(|b+a| + \left|\boldsymbol{\theta}^{*\top}\boldsymbol{\phi}(\mathbf{x}_{t,a_t};\mathbf{w}^*)\right| + \|\boldsymbol{\theta}_t\|_2\|\boldsymbol{\phi}(\mathbf{x}_{t,a_t};\mathbf{w}\|\right). && (74)
\end{aligned}$$

Since $0 \leq \boldsymbol{\theta}^{*\top}\boldsymbol{\phi}(\mathbf{x}_{t,a_t};\mathbf{w}^*) = h(\mathbf{x}_{t,a_t}) \leq 1$, we can further bound

$$\left|\sigma_t^2 - \hat{\sigma}_t^2\right| \leq \left(1 + \|\boldsymbol{\theta}_t\|_2\|\boldsymbol{\phi}(\mathbf{x}_{t,a_t};\mathbf{w})\|\right)\left(|b+a| + 1 + \|\boldsymbol{\theta}_t\|_2\|\boldsymbol{\phi}(\mathbf{x}_{t,a_t};\mathbf{w})\|\right). \tag{75}$$

Using the result from proof of Lemma A.3, we get

$$||\boldsymbol{\theta}_t||_2 = \left\|\left(\lambda\mathbf{I} + \sum_{i=1}^{t}\boldsymbol{\phi}(\mathbf{x}_{i,a_i};\mathbf{w}_{i-1})\boldsymbol{\phi}(\mathbf{x}_{i,a_i};\mathbf{w}_{i-1})^\top\right)^{-1}\sum_{i=1}^{t}\boldsymbol{\phi}(\mathbf{x}_{i,a_i};\mathbf{w}_{i-1})\hat{\mathbf{r}}\right\|_2 \leq 2d. \tag{76}$$

On the other hand, by $\boldsymbol{\phi}(\mathbf{x}_{t,a_t};\mathbf{w}_0)=\mathbf{0}$, using Lemma A.5, we have

$$\begin{aligned}
\|\boldsymbol{\phi}(\mathbf{x}_{t,a_t};\mathbf{w})\| &= \|\boldsymbol{\phi}(\mathbf{x}_{t,a_t};\mathbf{w})-\boldsymbol{\phi}(\mathbf{x}_{t,a_t};\mathbf{w}_0)\| && (77)\\
&= \left\|\boldsymbol{\phi}(\mathbf{x}_{t,a_t};\mathbf{w})-\hat{\boldsymbol{\phi}}(\mathbf{x}_{t,a_t};\mathbf{w})+\left\langle\nabla_{\mathbf{w}^{(0)}}\boldsymbol{\phi}(\mathbf{x}_{t,a_k};\mathbf{w}^{(0)}),\left(\mathbf{w}_t-\mathbf{w}^{(0)}\right)\right\rangle\right\| && (78)\\
&\le \left\|\boldsymbol{\phi}(\mathbf{x}_{t,a_t};\mathbf{w})-\hat{\boldsymbol{\phi}}(\mathbf{x}_{t,a_t};\mathbf{w})\right\|+\left\|\left\langle\nabla_{\mathbf{w}^{(0)}}\boldsymbol{\phi}(\mathbf{x}_{t,a_k};\mathbf{w}^{(0)}),\left(\mathbf{w}_t-\mathbf{w}^{(0)}\right)\right\rangle\right\| \text{ (by triangle inequality)} && (79)\\
&\le \left\|\boldsymbol{\phi}(\mathbf{x}_{t,a_t};\mathbf{w})-\hat{\boldsymbol{\phi}}(\mathbf{x}_{t,a_t};\mathbf{w})\right\|+\left\|\nabla_{\mathbf{w}^{(0)}}\boldsymbol{\phi}(\mathbf{x}_{t,a_k};\mathbf{w}^{(0)})\right\|_{\mathcal{F}}\left\|\mathbf{w}_t-\mathbf{w}^{(0)}\right\|_2 \text{ (by Hölder's inequality)} && (80)\\
&\le C_0\omega^{4/3}L^3d^{-1/2}\sqrt{m\log m}+\frac{\delta^{3/2}\sqrt{d}}{Tn^{9/2}L^{11/2}\log^3(m)} \text{ (by Lemma A.5 and A.3).} && (81)
\end{aligned}$$

Therefore, combining with the fact that $\omega=\mathcal{O}\left(m^{-1/2}\left\|\mathbf{r}-\tilde{\mathbf{r}}\right\|_{\mathbf{H}^{-1}}\right)$ by Lemma A.2, we get

$$\|\boldsymbol{\theta}_t\|_2\left\|\boldsymbol{\phi}(\mathbf{x}_{t,a_t};\mathbf{w})\right\|\le\frac{C_1L^3\sqrt{d\log m}\left\|\mathbf{r}-\tilde{\mathbf{r}}\right\|_{\mathbf{H}^{-1}}^{4/3}}{m^{1/6}}+\frac{2\delta^{3/2}d^{3/2}}{Tn^{9/2}L^{11/2}\log^3(m)}. \tag{82}$$

Using the result that $\mathbf{r}^\top\mathbf{H}^{-1}\mathbf{r}$ can be bounded by the RKHS norm of $\mathbf{r}$ if it belongs to the RKHS induced by the NTK (Xu et al., 2022a; Zhou et al., 2020; Arora et al., 2019b,a; Lee et al., 2019), we obtain

$$\begin{aligned}
\left|\sigma_t^2-\hat{\sigma}_t^2\right| &\le \left(1+\frac{C_1L^3\sqrt{d\log m}\left\|\mathbf{r}-\tilde{\mathbf{r}}\right\|_{\mathbf{H}^{-1}}^{4/3}}{m^{1/6}}+\frac{2\delta^{3/2}d^{3/2}}{Tn^{9/2}L^{11/2}\log^3(m)}\right)\\
&\left(|b+a|+1+\frac{C_1L^3\sqrt{d\log m}\left\|\mathbf{r}-\tilde{\mathbf{r}}\right\|_{\mathbf{H}^{-1}}^{4/3}}{m^{1/6}}+\frac{2\delta^{3/2}d^{3/2}}{Tn^{9/2}L^{11/2}\log^3(m)}\right)=\tilde{\mathcal{O}}\left(\frac{d^3}{T^2n^9L^{11}\log^6(m)}\right). \qquad (83)
\end{aligned}$$

Using the regret in Equation 54 from the proof of Theorem A.3.1, i.e.,

$$\begin{aligned}
\text{Regret}(T) &\le \sqrt{\sum_{t=1}^{T}4\alpha_t^2\bar{\sigma}_t^2||\boldsymbol{\phi}(\mathbf{x}_{t,a_t};\mathbf{w}_{t-1})/\bar{\sigma}_t||^2_{\mathbf{A}_{t-1}^{-1}}}+C_4\ell_{\text{Lip}}m^{-1/2}T\sqrt{\log(1/\delta)}||\mathbf{r}-\tilde{\mathbf{r}}||_{\mathbf{H}^{-1}}\\
&+\left(\frac{C_0TL^3d^{1/2}\sqrt{\log m}||\mathbf{r}-\tilde{\mathbf{r}}||_{\mathbf{H}^{-1}}^{4/3}}{m^{1/6}}+\frac{2\ell_{\text{Lip}}\delta^{3/2}}{m^{1/2}n^{9/2}L^6\log^3(m)}\right)C_3d^2\sqrt{\log(1/\delta)}||\mathbf{r}-\tilde{\mathbf{r}}||_{\mathbf{H}^{-1}}. \qquad (84)
\end{aligned}$$

Let $\hat{\sigma}_{T_{\min}}=\min_{t\in[T]}\hat{\sigma}_t$, since $\bar{\sigma}_t=\max\{R/\sqrt{d},\hat{\sigma}_t\}$, yielding $\bar{\sigma}_{T_{\min}}\ge R/\sqrt{d}$, therefore, we have

$$\begin{aligned}
\alpha_t &= 8\sqrt{d\log\left(1+td(\log HK)/(\bar{\sigma}_t^2d\lambda)\right)\log(4t^2/\delta)}+4R/\bar{\sigma}_t\log(4t^2/\delta)+\lambda^{1/2}M && (85)\\
&\le 8\sqrt{d\log\left(1+Td(\log HK)/(\lambda R^2)\right)\log(4T^2/\delta)}+4R/\hat{\sigma}_{T_{\min}}\log(4T^2/\delta)+\lambda^{1/2}M=\tilde{\mathcal{O}}\left(\sqrt{d}\right). && (86)
\end{aligned}$$

Since $\bar{\sigma}_t^2=\max\{R^2/d,\hat{\sigma}_t^2\}\le R^2/d+\hat{\sigma}_t^2\le R^2/d+\sigma_t^2+\tilde{\mathcal{O}}\left(\frac{d^3}{T^2n^9L^{11}\log^6(m)}\right)$ (by Equation 83), $\sqrt{|x|+|y|}\le\sqrt{|x|}+\sqrt{|y|}$, using the upper bound of α_t in Lemma A.4 and Lemma A.1, we finally obtain

$$\begin{aligned}
\text{Regret}(T) &\le C_5\alpha_T\sqrt{\left[TR^2+d\sum_{t=1}^{T}\left(\sigma_t^2+\frac{d^3}{T^2n^9L^{11}\log^6(m)}\right)\right]\log(1+TG^2/(\lambda R^2))}\\
&+C_6\ell_{Lip}L^3d^{5/2}m^{-1/6}T\sqrt{\log m\log(1/\delta)\log(TK/\delta)}||\mathbf{r}-\tilde{\mathbf{r}}||_{\mathbf{H}^{-1}} && (87)\\
&\le\tilde{\mathcal{O}}\left(R\sqrt{dT}+d\sqrt{\sum_{t=1}^{T}\sigma_t^2+\frac{d^3}{T^2n^9L^{11}\log^6(m)}}\right)+\tilde{\mathcal{O}}\left(m^{-1/6}T\sqrt{(\mathbf{r}-\tilde{\mathbf{r}})^\top\mathbf{H}^{-1}(\mathbf{r}-\tilde{\mathbf{r}})}\right) && (88)
\end{aligned}$$

of Theorem 4.8. □

B Experimental Details

B.1 Demo notebook code for Algorithm 1

```
import torch

class NeuralVarLinearUCB:
 def __init__(self, dim, n_arm=4, lamdba=1, nu_R=1, hidden=100):
  #Initialize model parameters
  self.func = Network(dim, hidden_size=hidden).cuda()
  self.context_list, self.arm_list, self.reward  = [], [], []
  self.theta = np.random.uniform(-1, 1, (self.n_arm, dim))
  self.b = np.zeros((self.n_arm, dim))
  self.A_inv = np.array([np.eye(dim) for _ in range(self.n_arm)])
  self.sigma = self.nu_R/dim

 def select(self, context):
  #Select action by UCB
  features = self.func(context).cpu().detach().numpy()
  ucb = [np.sqrt(np.dot(features[a,:], np.dot(self.A_inv[a], features[a,:].T))) for a in range(self.n_arm)]
  mu = [np.dot(features[a,:], self.theta[a]) for a in range(self.n_arm)]
  arm = np.argmax(mu + ucb)
  return arm, mu[arm]

 def train(self, context, arm_select, reward):
  #Update neural network model parameters
  self.context_list.append(torch.from_numpy(context[arm_select].reshape(1, -1)).float())
  self.arm_list.append(arm_select)
  self.reward.append(reward)
  optimizer = optim.SGD(self.func.parameters(), lr=1e-2, weight_decay=self.lamdba)
  train_set = []
  for idx in range(len(self.context_list)):
   train_set.append((self.context_list[idx], self.arm_list[idx], self.reward[idx]))
  train_loader = DataLoader(train_set, batch_size = 64, shuffle = True)
  for batch_idx, (samples, arms, labels) in enumerate(train_loader):
   optimizer.zero_grad()
   features = self.func(samples.cuda())
   mu = (features * torch.from_numpy(self.theta[arms]).float().cuda()).sum()
   A_inv = torch.from_numpy(self.A_inv[arms]).float().cuda()
   sigma = (features * torch.squeeze(torch.bmm(A_inv, torch.unsqueeze(features, 2)))).sum()
   loss = torch.mean(1/2 * torch.log(2*np.pi*sigma) + (labels-mu)**2/(2*sigma))
   loss.backward()
   optimizer.step()

 def update_model(self, context, arm_select, reward, mu, a_low, b_up):
  #Update linear model parameters
  context = self.func(context)
  self.theta = np.array([np.matmul(self.A_inv[a], self.b[a]) for a in range(self.n_arm)])
  self.sigma = (b_up - mu) * (mu - a_low)
  self.sigma = max(self.sigma, self.nu_R/dim)
  self.b[arm_select] += (context[arm_select] * reward[arm_select])/self.sigma
  self.A_inv[arm_select] = inv_sherm_morri(context[arm_select,:]/np.sqrt(self.sigma), self.A_inv[arm_select])

if __name__ == "__main__"
 agent = NeuralVarLinearUCB(dim = 20)
 list_regrets = []
 for t in range(T):
  context, rwd, psd_rwd = contexts[t], rewards[t], psd_rewards[t]
  arm_select, mu = agent.select(context)
  regret = np.max(psd_rwd) - psd_rwd[arm_select]
  list_regrets.append(regret)
  agent.update_model(context, arm_select, rwd, mu, a_low, b_up)
  if t%100 == 0:
   agent.train(context, arm_select, rwd[arm_select])
 plot(list_regrets)
```

B.2 Experimental settings

Dataset and hyper-parameters details, We deploy the models on five datasets in the main paper. Regarding real-world data, we use the MNIST dataset which contains 70000, $d = 28 \times 28$ digit handwriting images with $K = 10$ classes (Lecun et al., 1998); the UCI-shuttle (statlog) dataset related to physics and chemistry area, containing 58000 features with $d = 9$ numerical attributes and $K = 7$ classes (Dua and Graff, 2017); the UCI-covertype includes 581012 biology instances with $d = 54$ forest cover types attributes and $K = 7$ classes;

the CIFAR-10 dataset consists 60000, $d = 32 \times 32 \times 3$ color images in $K = 10$ classes. Regarding the model architecture and hyper-parameters settings, we mainly follow Xu et al. (2022a); Zhou et al. (2020). In particular, we use DNN with ReLU activation, $L = 2$ layers, $m = 100$ dimension for the encoder weights matrices, and the last output feature dimension $m_L = d = 20$ for the synthetic and $d = 64$ for real-world datasets. We set $\lambda = 1$, the exploration rate $\alpha_t = 0.02$, and the number of iterations to update the neural network $n = 1000$. We also set $H = 100$ (10 on MNIST, UCI-covertype, and CIFAR-10) rounds starting from round 2000 (10000 on MNIST, UCI-covertype, and CIFAR-10)) to update DNN weights $\mathbf{w}$ following Neural-LinUCB setting.

Baseline details. Regarding the baseline comparison, there also exists GLMUCB (Filippi et al., 2010; Zenati et al., 2022), KernelUCB (Valko et al., 2013), and BootstrappedNN (Riquelme et al., 2018). That said, since Xu et al. (2022a); Zhou et al. (2020) has shown NeuralUCB and Neural-LinUCB better than them in such setting, while our results have lower regret than NeuralUCB and Neural-LinUCB, this implies that our Neural-σ^2-LinUCB is also better than these aforementioned related baselines in terms of cumulative regret. We additionally show this comparison in Figure 17.

Source code and computing systems. Our source code includes the notebook demo, dataset scripts, setup for the environment, and our provided code (detail in README.md). We run our code on a single GPU: NVIDIA RTX A5000-24564MiB with 8-CPUs: AMD Ryzen Threadripper 3960X 24-Core with 8GB RAM per each and require 8GB available disk space for storage.

B.3 Additional results

B.3.1 Uncertainty estimation evaluations

We evaluate the uncertainty quality by using calibration and sharpness of models across time horizon $t \in [T]$ (Manh Bui and Liu, 2024; Bui and Liu, 2024). Regarding calibration, this intuitively means that a p confidence interval contains the target reward r p of the time. Hence, given a forecast from UCB at time t, let $F_t : \mathbb{R} \to [0, 1]$ to denote the CDF of this forecast at $\mathbf{x}_t$, then the calibration error for this forecast is

$$cal_1(\{F_t, r_t\}_{t=1}^T) := \sum_{j=1}^{m} \left(p_j - \frac{|\{r_t | F_t(r_t) \leq p_j, t = 1, \cdots, T\}|}{T} \right)^2, \tag{89}$$

for each threshold p_j from the chosen of m confidence level $0 \leq p_1 < p_2 < \cdots < p_m \leq 1$.

Regarding sharpness, this means that the confidence intervals should be as tight as possible, i.e., $Var(F_t)$ of the random variable whose CDF is F_t to be small (Kuleshov et al., 2018). Formally, the sharpness score follows

$$sha(F_1, \cdots, F_T) := \sqrt{\frac{1}{T} \sum_{t=1}^{T} var(F_t)}. \tag{90}$$

We show the quantitative results for calibration in Equation 89 and sharpness in Equation 90 in Table 3 and qualitatively visualize on Figure 4 (a).

Methods	Cumulative reward (↑)	Calibration Error (↓)	Sharpness (↓)
LinUCB	7459.0812 ± 32.9722	0.7425 ± 0.0301	0.2095 ± 0.0191
NeuralUCB	10658.3046 ± 60.5330	0.2634 ± 0.0146	1.0733 ± 0.0110
Neural-LinUCB	10929.2430 ± 58.8243	0.8991 ± 0.1840	0.2042 ± 0.0213
Neural-σ^2-LinUCB	**11326.6471 ± 50.1880**	**0.1492 ± 0.0659**	**0.8242 ± 0.2802**

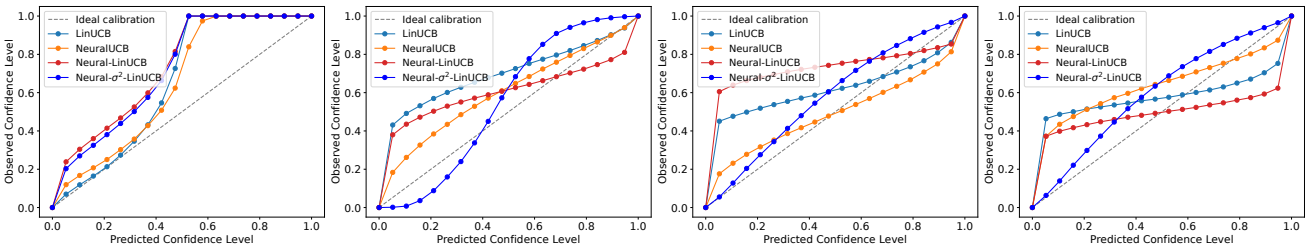
Table 3: Cumulative reward and uncertainty quality performance (i.e., calibration (Malik et al., 2019) and sharpness (Gneiting et al., 2007)) on $h_1(\mathbf{x}) = 10(\mathbf{x}^\top \boldsymbol{\theta})^2$ dataset, averaged over 5 trials. **Our Neural-σ^2-LinUCB is more well-calibrated and still having a sharp UCB, resulting in a higher cumulative rewards than other methods.**

To further understand the improvement of uncertainty estimation across the learning time horizon, we additionally evaluate calibration on hold-out validation data over different checkpoints across time steps, $t = \{0, 2000, 5000, 7500\}$. Since we use validation data to evaluate, then the calibration in this setting is as follows: given a forecast from UCB at time t, let $F_i^t : \mathbb{R} \to [0, 1]$ to denote the CDF of this forecast at $\mathbf{x}_i$, then the calibration error for this forecast is

$$cal_2(\{F_i^t, r_i\}_{i=1}^n) := \sum_{j=1}^{m} \left(p_j - \frac{\left|\{r_i | F_i^t(r_i) \leq p_j, i = 1, \cdots, n\}\right|}{n} \right)^2, \tag{91}$$

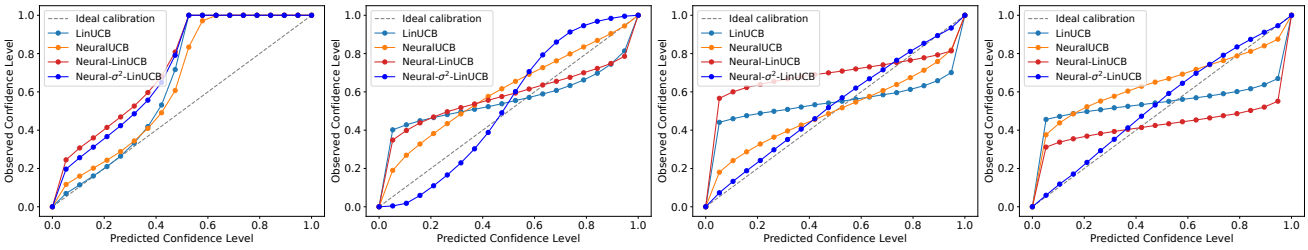
where n is the number of samples in the validation set.

We visualize the calibration error in Equation 91 by the reliability diagram across correspond to different arms, $a = \{0, 1, 2, 3\}$ in Figure 7, 8, 9, and 10 correspondingly. Overall, we can see that when $t = 0$, all of the models are uncalibrated because of no learning data. But when t grows, Neural-σ^2-LinUCB are almost always more calibrated than other UCB algorithms. This once again confirms the hypothesis that our Neural-σ^2-LinUCB algorithm can improve the uncertainty quantification quality of UCB.



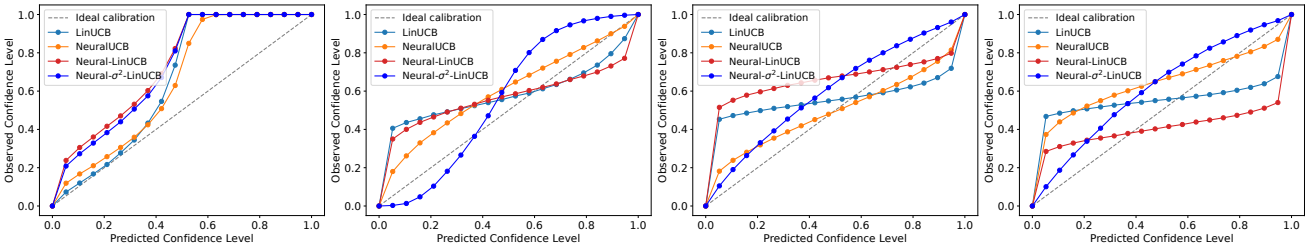
Round $t = 0$ Round $t = 2000$ Round $t = 5000$ Round $t = 7500$

Figure 7: Visualization of calibration error in Equation 91 with reliability diagram on $h_1(\mathbf{x}_{t,a})$ dataset (arm: 0).



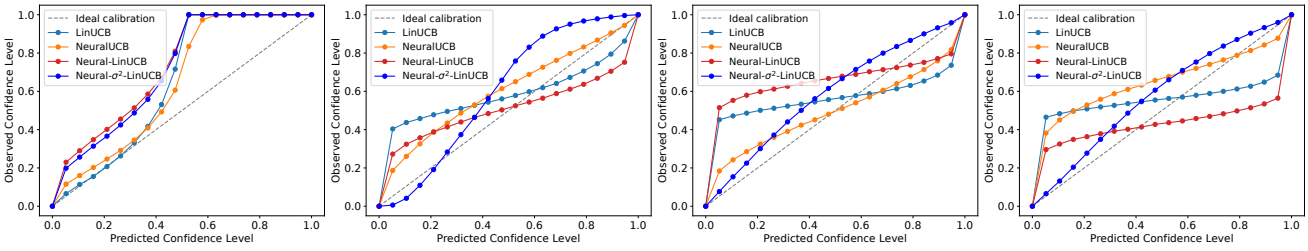
Round $t = 0$ Round $t = 2000$ Round $t = 5000$ Round $t = 7500$

Figure 8: Visualization of calibration error in Equation 91 with reliability diagram on $h_1(\mathbf{x}_{t,a})$ dataset (arm: 1).



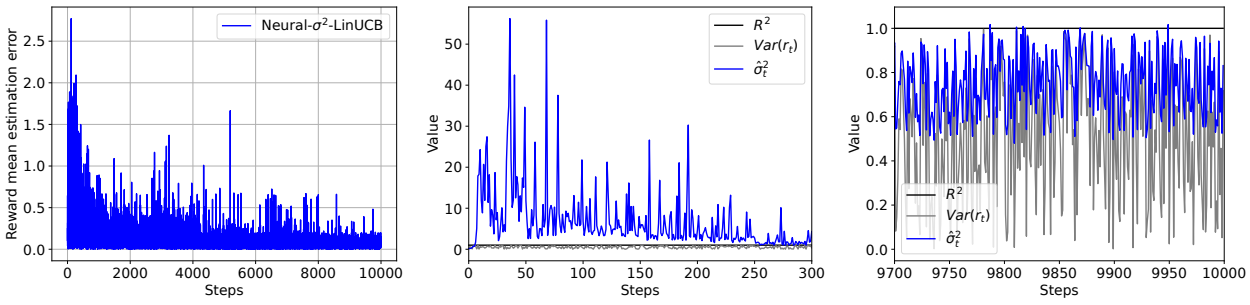
Round $t = 0$ Round $t = 2000$ Round $t = 5000$ Round $t = 7500$

Figure 9: Visualization of calibration error in Equation 91 with reliability diagram on $h_1(\mathbf{x}_{t,a})$ dataset (arm: 2).



Round $t = 0$ Round $t = 2000$ Round $t = 5000$ Round $t = 7500$

Figure 10: Visualization of calibration error in Equation 91 with reliability diagram on $h_1(\mathbf{x}_{t,a})$ dataset (arm: 3).

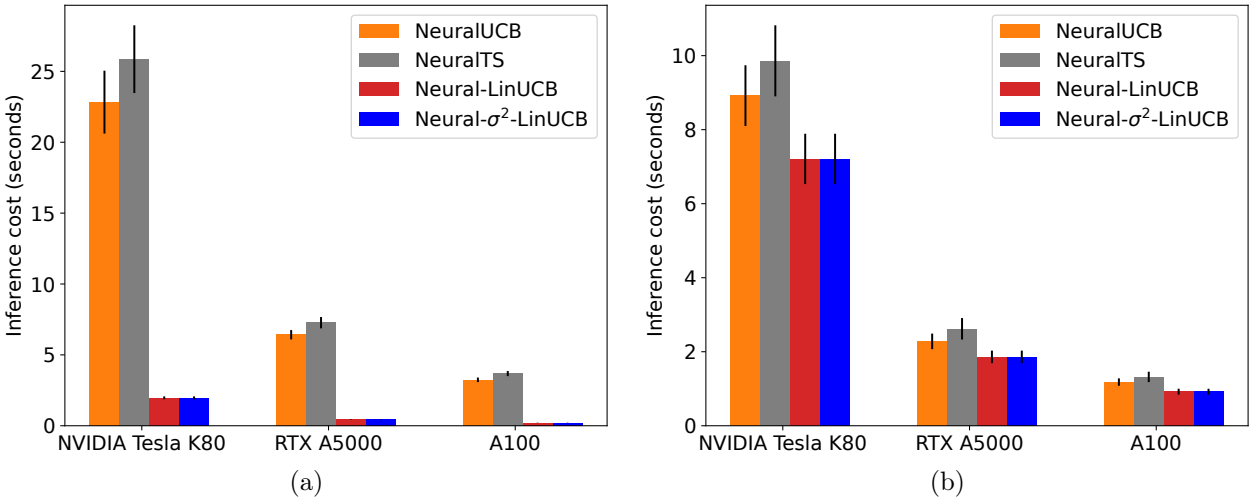


(a) (b) (c)

Figure 11: (a) Reward mean estimation error $\left|h(\mathbf{x}_{t,a_t}) - \theta_{t,a_t}^\top \phi(\mathbf{x}_{t,a_t}; \mathbf{w})\right|$ across $t \in [T]$. (b) Reward variance $Var(r_t)$, our estimation for the variance upper bound σ_t^2, and the upper bound R^2 comparison at the first 300 episodes; (c) Fig 4 (b) in the main paper at the last 300 episodes. **When the reward mean estimation quality improves in Figure 11 (a), the quality of our estimation for σ_t^2 increase and more accurate in Figure 11 (c) when compared to Figure 11 (b).**

To further validate the estimation quality of $\hat{\sigma}_t^2$ in Equation 11. Firstly, recall that Theorem 3.3 implies that the accurate estimation for the variance upper bound σ_t^2 is a necessary condition for good estimation quality for the reward mean $h(\mathbf{x}_{t,a_t})$. Figure 4 (b) in the main paper confirms when we have a good reward estimation in the last 300 episodes, then we can obtain an accurate estimation for the variance upper bound (by $\geq Var(r_t)$ and $\leq R^2$). We add Figure 11 (b) to compare with Figure 4 (b) (i.e., Figure 11 (c)) in the first 300 episodes, we can see that when the reward mean estimation has high estimation errors (see Figure 11 (a)), the estimation for the variance upper bound σ_t^2 is inaccurate.

B.4 Computational efficiency evaluations



(a) (b)

Figure 12: Computational cost comparison of neural contextual bandits algorithms on MNIST for running 100 rounds across three modern GPU architectures, including in the arm selection step (a) and the DNN update step (b).

We extensively evaluate our model on three different settings, including: (1) a single GPU: NVIDIA Tesla K80 accelerator-12GB GDDR5 VRAM with 8-CPUs: Intel(R) Xeon(R) Gold 6248R CPU @ 3.00GHz with 8GB RAM per each; (2) a single GPU: NVIDIA RTX A5000-24564MiB with 8-CPUs: AMD Ryzen Threadripper 3960X 24-Core with 8GB RAM per each; and (3) a single GPU: NVIDIA A100-PCIE-40GB with 8 CPUs: Intel(R) Xeon(R) Gold 6248R CPU @ 3.00GHz with 8GB RAM per each. Figure 12 summarizes these results with the number of rounds to frequently update DNN $H = 10$ and the number of rounds $t = 100$.

Figure 12 (a) shows the latency of the arm selection step in Line 5 to Line 8 of Algorithm 1. We can see that by

computing the UCB value from a linear model on the last feature representation of DNN, our Neural-σ^2-LinUCB and Neural-LinUCB are much more efficient than other baselines. Our better results are consistent across different CPU/GPU architectural settings. For instance, in the lower resource hardware like with NVIDIA Tesla K80, our results are faster than around 20 seconds. Regarding powerful hardware like NVIDIA A100, we are still faster than around 4 seconds. These results are consistent with the result of Xu et al. (2022a) and could be explained by the fact that NeuralUCB and NeuralTS need to perform UCB and Thompson-sampling exploration on all the parameters of DNN. As a result, the lower the computational hardware, the less computationally efficient than our algorithms.

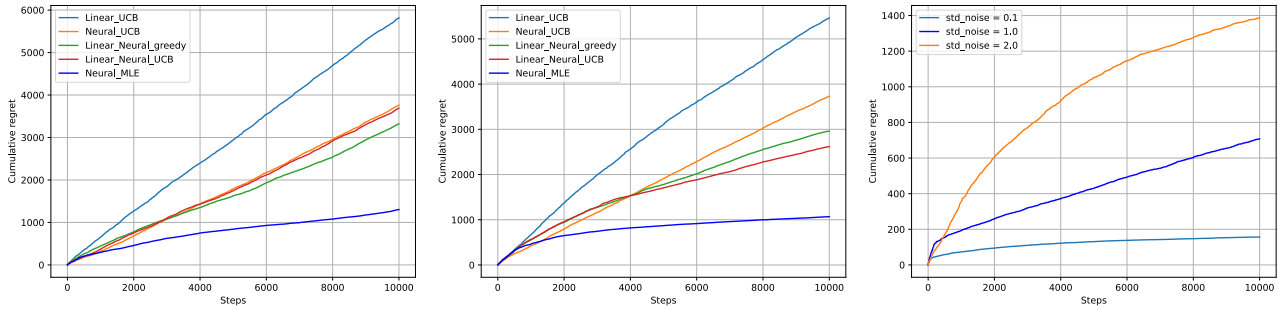
Figure 12 (b) shows the latency of the DNN update step from Line 17 to Line 21 of Algorithm 1. Similarly, we observe that the more powerful the hardware, the less time it takes to optimize the DNN models. Regarding comparison with other baselines, by using the same technique to save computational cost in DNN training from Neural-LinUCB (Xu et al., 2022a), Neural-σ^2-LinUCB also enjoys a more computationally efficient than other neural contextual bandits baselines.

B.4.1 Regret performance evaluations

We additionally show our model behaviors across different types of stochasticity regarding the reward noise ξ_t on $h_1(\mathbf{x}) = 10(\mathbf{x}^\top \boldsymbol{\theta})^2$ dataset in Figure 13. Specifically, from $t = 0$ to $t = T = 10000$, we set ξ_t increase monotonically from 1 to 10 in Figure 13 (a), and decreases monotonically from from 2 to 0 in Figure 13 (b). We observe that Neural-σ^2-LinUCB's results are robust by always having a significantly lower cumulative regret than other baselines. Furthermore, when the noise decreases and reaches very small values at the final steps, our cumulative regret becomes almost constant.

Similarly to the setting of Zhou et al. (2020), we also consider the non-stochasticity for the reward noise variance at round t, i.e., $\xi_t \sim \mathcal{N}(0, \text{std\_noise}^2)$ in Figure 13 (c), where std_noise $= \{0.1, 1.0, 2.0\}$. It can be seen from this figure that when the std_noise decreases, our cumulative regret also decreases respectively. And for the std_noise $= 0.1$, at the final steps, Neural-σ^2-LinUCB's cumulative regret also becomes almost constant.

Regarding the effectiveness of highly noisy cases for the rewards to our model performance, we also observe that when the reward noise level is high (e.g., std_noise $= 2.0$), our model often has a higher cumulative regret than the cases of lower reward noise levels (e.g., std_noise $= \{0.1, 1.0\}$). In addition, when the noise value increases across rounds, all methods have increased cumulative regret (e.g., Figure 13 (a)), but our method is more robust by having a lower cumulative regret than other baselines.



(a) (b) (c)

Figure 13: $h_1(\mathbf{x}) = 10(\mathbf{x}^\top \boldsymbol{\theta})^2$. (a) The noise ξ_t monotonically increases from 1 to 10. (b) The noise ξ_t monotonically decreases from 2 to 0. (c) Comparison across different std_noise.

In Figure 14, we show an ablation study by increasing the model size and a longer time horizon on CIFAR-10. We can see that when the model capacity increases, we can achieve a lower cumulative regret, confirming our claim in Remark 4.9.

In Figure 15, we compare with a heuristic selection for σ_t^2 in practice. We can see that since this is a heuristic selection, σ_t^2 may not satisfy conditions in the Equation 3, leading to worse performances than our proposed estimation in Equation 11.

To explore the effect of the actual value used for exploration rate α_t, we provide a result for setting the true value

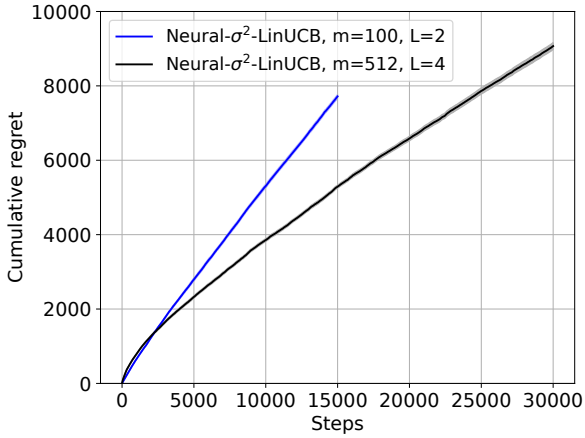


Figure 14: Blue line is the performance of model size $m = 100$, $L = 2$, $\mathbf{w}$ is updated every $H = 10$ rounds starting from $t = 10000$, and $T = 15000$ of our method on CIFAR-10 in Figure 3 (d). The black line is with $m = 512$, $L = 4$, $\mathbf{w}$ is updated every $H = 10$ rounds from $t = 25000$, and $T = 30000$. **This shows when the model capacity and the learning process increase, we can achieve a lower cumulative regret.**

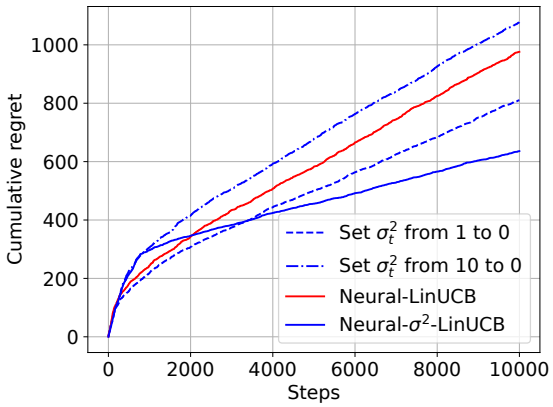


Figure 15: Simple baseline by setting σ_t^2 heuristically on $h_1(\mathbf{x}) = 10(\mathbf{x}^\top \boldsymbol{\theta})^2$ with $R = 1$, including setting (1) σ_t^2 decrease from 1 to 0 and (2) σ_t^2 decrease from 10 to 0 across time horizon T. **Since this is a heuristic, so sometimes better (e.g., if $\sigma_t^2 \leq R^2$), sometimes worse (e.g., if $\sigma_t^2 > R^2$) than Neural Linear Upper Confidence Bound (Neural-LinUCB). But it is always worse than Neural-σ^2-LinUCB that tries to estimate σ_t^2 because our estimation $\hat{\sigma}_t^2$ aims to satisfy Thm 3.3.**

α_t in Theorem 4.5 and comparing with Neural-LinUCB in Figure 16 (we can not show Neural Upper Confidence Bound (NeuralUCB) results because computing γ_t in Zhou et al. (2020) is very computationally expensive as the determinant of the gradient of the neural-net covariance matrix). Figure 16 shows our algorithm is better than Neural-LinUCB, once again confirming our theoretical and experimental results in the main paper.

Finally, we summarize our cumulative regret comparison with all other baselines in Figure 17. It is also worth noticing that we also compare our results with an extension of SAVE in neural bandits, i.e., SAVE-SupNeural-LinUCB by considering the context in their Algorithm 1 (Zhao et al., 2023) as the feature representation from DNN. From Figure 17, we can see that even assuming given knowledge of T, in neural contextual bandits, this approach still performs poorly in experiments because of the over-exploration (Salgia, 2023).

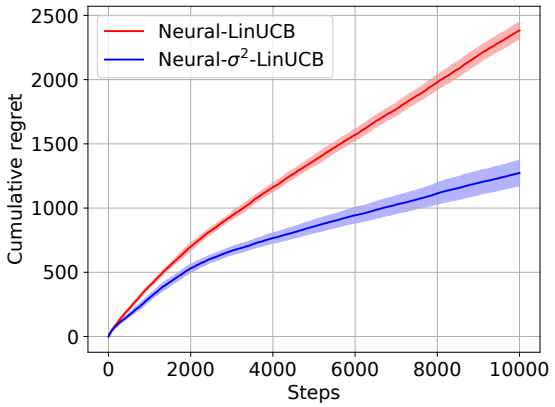


Figure 16: Cumulative regret results on the synthetic data $h_1(\mathbf{x}) = 10(\mathbf{x}^\top \boldsymbol{\theta})^2$ with the true α_t in our Theorem 4.5 with $d = 20$, $H = 100$, $K = 4$, $\lambda = 1$, $M = 0.1$, and $\delta = 0.1$. **This demonstrates that using the α_t suggested by the theory, our method is also better than Neural-LinUCB, confirming our tighter regret bound in Theorem 4.5.**

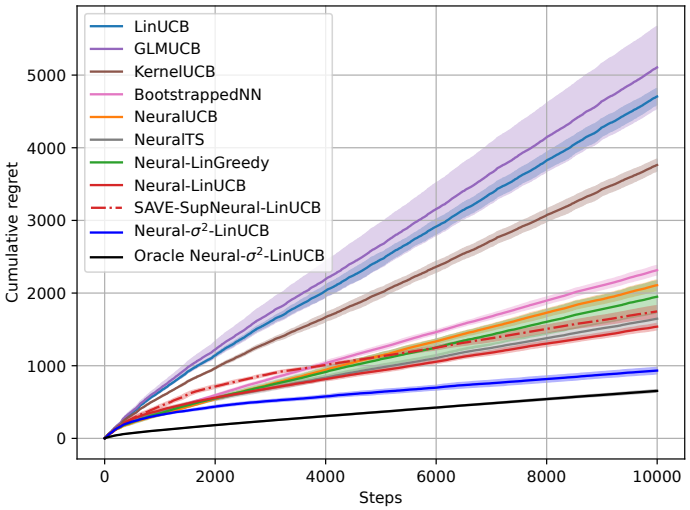


Figure 17: Support figure for Figure 1: Full cumulative regret results on $h_1(\mathbf{x}) = 10(\mathbf{x}^\top \boldsymbol{\theta})^2$ across 10 runs with different seeds.