# Meta Clustering of Neural Bandits

Yikun Ban*
yikunb2@illinois.edu
University of Illinois at Urbana-Champaign
Champaign, USA

Yunzhe Qi*
yunzheq2@illinois.edu
University of Illinois at Urbana-Champaign
Champaign, USA

Tianxin Wei
twei10@illinois.edu
University of Illinois at Urbana-Champaign
Champaign, USA

Lihui Liu
lihuil2@illinois.edu
University of Illinois at Urbana-Champaign
Champaign, USA

Jingrui He
jingrui@illinois.edu
University of Illinois at Urbana-Champaign
Champaign, USA

## ABSTRACT

The contextual bandit has been identified as a powerful framework to formulate the recommendation process as a sequential decision-making process, where each item is regarded as an arm and the objective is to minimize the regret of $T$ rounds. In this paper, we study a new problem, Clustering of Neural Bandits, by extending previous work to the arbitrary reward function, to strike a balance between user heterogeneity and user correlations in the recommender system. To solve this problem, we propose a novel algorithm called M-CNB, which utilizes a meta-learner to represent and rapidly adapt to dynamic clusters, along with an informative Upper Confidence Bound (UCB)-based exploration strategy. We provide an instance-dependent performance guarantee for the proposed algorithm that withstands the adversarial context, and we further prove the guarantee is at least as good as state-of-the-art (SOTA) approaches under the same assumptions. In extensive experiments conducted in both recommendation and online classification scenarios, M-CNB significantly outperforms SOTA baselines. This shows the effectiveness of the proposed approach in improving online recommendation and online classification performance.

## CCS CONCEPTS

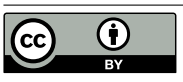
• **Theory of computation** → **Online learning algorithms**; • **Information systems** → **Personalization**.

## KEYWORDS

Contextual Bandits; Recommendation; User Modeling; Meta Learning

**ACM Reference Format:**
Yikun Ban, Yunzhe Qi, Tianxin Wei, Lihui Liu, and Jingrui He. 2024. Meta Clustering of Neural Bandits. In *Proceedings of the 30th ACM SIGKDD Conference on Knowledge Discovery and Data Mining (KDD '24), August 25–29, 2024, Barcelona, Spain.* ACM, New York, NY, USA, 12 pages. https://doi.org/10.1145/3637528.3671691

*Both authors contributed equally to this paper.





*KDD '24, August 25–29, 2024, Barcelona, Spain*

ACM ISBN 979-8-4007-0490-1/24/08.
https://doi.org/10.1145/3637528.3671691

## 1 INTRODUCTION

Recommender systems play an integral role in various online businesses, including e-commerce platforms and online streaming services. They leverage user correlations to assist the perception of user preferences, a field of study spanning several decades. In the past, considerable effort has been directed toward supervised-learning-based collaborative filtering methods within relatively static environments [20, 45]. However, the ideal recommender systems should adapt over time to consistently meet user interests. Consequently, it is natural to formulate the recommendation process as a sequential decision-making process. In this paradigm, the recommender engages with users, observes their online feedback (i.e., rewards), and optimizes the user experience for long-term benefits, rather than fitting a model on the collected static data based on supervised learning [8, 16, 53]. Based on this idea, this paper focuses on the formulation of contextual bandits, where each item is treated as an arm (context) in a recommendation round, and the primary objective is to minimize the cumulative regret over $T$ rounds and tackle the dilemma of exploitation and exploration in the sequential decision-making process [1, 3, 17, 18, 33, 33, 34, 37, 40, 41].

Linear contextual bandits model a user's preference through a linear reward function based on arm contexts [1, 11, 32]. However, given the substantial growth of users in recommender systems, it can be overly ambitious to represent all user preferences with a single reward function, and it may overlook the user correlations if each user is modeled as a single bandit. To address this challenge, a series of methods known as clustering of linear bandits [3, 17, 18, 33, 34] have emerged, which represent each cluster of users as a reward function, achieving a balance between user heterogeneity and user correlations. Note that the cluster information is unknown in this problem setting. In essence, with each user being treated as a linear contextual bandit, these methods adopt graph-based techniques to dynamically cluster users, and leverage user correlations for making arm recommendations. However, it is crucial to acknowledge the limitations of this line of works: they all rely on linear reward functions, and user clusters are represented as linear combinations of individual bandit parameters. The assumptions of linearity in

reward functions and the linear representation of clusters may not hold up well in real-world applications [46, 58].

In relaxation of the assumption on reward mapping functions, inspired by recent advances in the single neural bandit [57, 58] where a neural network is assigned to learn *an* unknown reward function, we study the new problem of Clustering of Neural Bandits (CNB) in this paper. Different from the single neural bandit [57, 58] and clustering of linear bandits [3, 17, 18, 33, 34], CNB introduces the bandit clusters built upon the *arbitrary* reward functions, which can be either linear or non-linear. Meanwhile, we note that the underlying clusters are usually not static over specific arm contexts [33]. For example, in the personalized recommendation task, two users (bandits) may both like "country music", but can have different opinions on "rock music". Therefore, adapting to arm-specific "relative clusters" in a dynamic environment is one of the main challenges in this problem. We propose a novel algorithm, Meta Clustering of Neural Bandits (M-CNB), to solve the CNB problem. Next, we will summarize our key ideas and contributions.

**Methodology**. To address the CNB problem, we must confront three key challenges: (1) Efficiently determining a user's relative group: Our approach involves employing a neural network, named the "user learner," to estimate each user's preferences. By grouping users with similar preferences, we efficiently create clusters with a process taking $O(n)$ time, where $n$ is the number of bandits (users). (2) Effective parametric representation of dynamic clusters: Inspired by advancements in meta-learning [15, 55], we introduce a meta-learner capable of representing and swiftly adapting to evolving clusters. In each round $t$, the meta-learner leverages its perceived knowledge from prior rounds $\{1, \ldots, t-1\}$ to rapidly adapt to new clusters via a few samples. This enables the rapid acquisition of nonlinear cluster representations, marking our first main contribution. (3) Balancing exploitation and exploration with relative bandit clusters: Our second main contribution is proposing an informative UCB-type exploration strategy, which takes into account both user-side and meta-side information for balancing the exploration and exploitation. By addressing these three main challenges, our approach manages to solve the CNB problem effectively and efficiently.

**Theoretical analysis**. To obtain a regret upper bound for the proposed algorithm, we need to tackle the following three challenges: (1) Analyzing neural meta-learner in bandit framework: To finish the analysis, we must build a confidence ellipsoid for the meta-learner approximation, which is one of the main research gaps. To deal with this gap, we bridge the meta-learner and user-learner via the Neural Tangent Kernel (NTK) regression and build the confidence ellipsoid upon the user-learner, which allows us to achieve a more comprehensive understanding of the meta-learner's behavior. (2) Reducing the naive $\widetilde{O}(\sqrt{nT})$ regret upper bound: $\widetilde{O}(\sqrt{T})$ is roughly the regret effort to learn a single neural bandit, and thus $\widetilde{O}(\sqrt{nT})$ are the regret efforts to learn $n$ neural bandits for $n$ users. We reduce the $\widetilde{O}(\sqrt{nT})$ efforts to $\widetilde{O}(\sqrt{qT})$, where $q$ is the expected number of clusters. This also indicates the proposed algorithm can leverage the collaborative effects among users. (3) Adversarial attack on contexts: In most neural bandit works, a common assumption is that the NTK matrix is non-singular, requiring that no two observed contexts (items) are identical or parallel [57, 58]. This vulnerability makes their regret analysis susceptible to adversarial attacks and less practical in real-world scenarios. In face of this challenge, we provide an instance-dependent regret analysis that withstands the context attack, and allows the contexts to be repeatedly observed. Furthermore, under the same assumptions as in existing works, we demonstrate that our regret upper bound is at least as good as SOTA approaches. The above efforts to address the challenges in the theoretical analysis is our third main contribution.

**Evaluations**. We evaluate the proposed algorithm in two scenarios: Online recommendation and Online classification with bandit feedback. For the first scenario, which naturally lends itself to CNB, we assess the algorithm's performance on four recommendation datasets. Since online classification has been widely used to evaluate neural bandits [5, 57, 58], we evaluate the algorithms on eight classification datasets where each class can be considered as a bandit (user), and correlations among classes are expected to be exploited. We compare the proposed algorithm with 8 strong baselines and show the superior performance of the proposed algorithm. Additionally, we offer the empirical analysis of the algorithm's time complexity, and conduct extensive sensitivity studies to investigate the impact of critical hyperparameters. The above empirical evaluation is our fourth main contribution.

Next, detailed discussion regarding related works is placed in Section 2. After introducing the problem definition in Section 3, we present the proposed algorithm, M-CNB, in Section 4 together with theoretical analysis in Section 5. Then, we provide the experimental results in Section 6 and conclude the paper in Section 7.

## 2 RELATED WORK

In this section, we briefly review the related works, including clustering of bandits and neural bandits.

**Clustering of bandits**. CLUB [17] first studies collaborative effects among users in contextual bandits where each user hosts an unknown vector to represent the behavior based on the linear reward function. CLUB formulates user similarity on an evolving graph and selects an arm leveraging the clustered groups. Then, Gentile et al. [18], Li et al. [33] propose to cluster users based on specific contents and select arms leveraging the aggregated information of conditioned groups. Li et al. [34] improves the clustering procedure by allowing groups to split and merge. Ban and He [3] uses seed-based local clustering to find overlapping groups, different from global clustering on graphs. Korda et al. [27], Wu et al. [50], Yang et al. [54] also study clustering of bandits with various settings in recommender systems. However, all these works are based on the linear reward assumption, which may fail in many real-world applications.

**Neural bandits**. Lipton et al. [35], Riquelme et al. [42] adapt the Thompson Sampling (TS) to the last layer of deep neural networks to select an action. However, these approaches do not provide regret analysis. Zhou et al. [58] and Zhang et al. [57] first provide the regret analysis of UCB-based and TS-based neural bandits, where they apply ridge regression on the space of gradients. Ban et al. [4] studies a multi-facet bandit problem with a UCB-based exploration. Jia et al. [24] perturbs the training samples for incorporating both exploitation and exploration. EE-Net [5] proposes to use another neural network for exploration. [52] combines the last-layer neural

network embedding with linear UCB to improve the computation efficiency. Dutta et al. [14] uses an off-the-shelf meta-learning approach to solve the contextual bandit problem in which the expected reward is formulated as Q-function. Santana et al. [43] proposes a Hierarchical Reinforcement Learning framework for recommendation in the dynamic experiments, where a meta-bandit is used for the selected independent recommender system. Kassraie and Krause [25] revisit Neural-UCB type algorithms and shows the $\widetilde{O}(\sqrt{T})$ regret bound without the restrictive assumptions on the context. Hong et al. [21], Maillard and Mannor [36] study the latent bandit problem where the reward distribution of arms are conditioned on some unknown discrete latent state and prove the $\widetilde{O}(\sqrt{T})$ regret bound for their algorithm as well. Federated bandits [10] consider dealing with multiple bandits (agents) while preserving the privacy of each bandit. However, the above works either focus on the different problem settings or overlook the clustering of bandits.

**Other related works**. [29, 44] study meta-learning in Thompson sampling and Hong et al. [22], Wan et al. [47] aims to exploit the hierarchical knowledge among hierarchical Bayesian bandits. However, they focus on the Bayesian or non-contextual bandits.

## 3 PROBLEM: CLUSTERING OF NEURAL BANDITS

In this section, we introduce the CNB problem, motivated by learning correlations among bandits with arbitrary reward functions. Next, we will use the scenarios of personalized recommendation to state the problem setting.

Suppose there are $n$ users (bandits), $N = \{1, \ldots, n\}$, to serve on a platform. In the $t^{\text{th}}$ round, the platform receives a user $u_t \in N$ (unique ID for this user) and prepares the corresponding $K$ candidate arms $\mathbf{X}_t = \{\mathbf{x}_{t,1}, \mathbf{x}_{t,2}, \ldots, \mathbf{x}_{t,K}\}$. Each arm is represented by its $d$-dimensional feature vector $\mathbf{x}_{t,i} \in \mathbb{R}^d, i \in [K] = \{1, \ldots, K\}$, which will encode the information from both the user side and the arm side [32]. Then, the learner is expected to select an arm $\mathbf{x}_t \in \mathbf{X}_t$ and recommend it to $u_t$, where $u_t$ refers to the target or served user. In response to this action, $u_t$ will provide the platform with a corresponding reward (feedback) $r_t$. Here, since different users may generate different rewards towards the same arm, we use $r_{t,i}|u_t$ to represent the reward produced by $u_t$ given $\mathbf{x}_{t,i}$. The formal definition of arm reward is below.

Given $u_t \in N$, the reward $r_{t,i}$ for each candidate arm $\mathbf{x}_{t,i} \in \mathbf{X}_t$ is assumed to be governed by an unknown function by

$$r_{t,i}|u_t = h_{u_t}(\mathbf{x}_{t,i}) + \zeta_{t,i}, \tag{1}$$

where $h_{u_t}$ is an unknown reward function associated with $u_t$, and it can be either linear or non-linear. $\zeta_{t,i}$ is a noise term with zero expectation $\mathbb{E}[\zeta_{t,i}] = 0$. We also assume the reward $r_{t,i} \in [0, 1]$ is bounded, as in many existing works [3, 17, 18]. Note that previous works on clustering of linear bandits all assume $h_{u_t}$ is a linear function with respect to arm $\mathbf{x}_{t,i}$ [3, 17, 18, 33, 34].

Meanwhile, users may exhibit clustering behavior. Inspired by [18, 33], we consider the cluster behavior to be item-varying, i.e., the users who have the same preference on a certain item may have different opinions on another item. Therefore, we formulate a set of users with the same opinions on a certain item as a relative cluster, with the following definition.

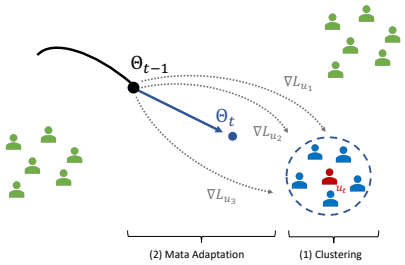



**Figure 1: Clustering and Meta Adaptation: Given $u_t$ and an arm $\mathbf{x}_{t,i}$, (1) M-CNB identifies cluster $\widehat{\mathcal{N}}_{u_t}(\mathbf{x}_{t,i})$, and then (2) meta-learner $\Theta_{t-1}$ rapidly adapt to this cluster, proceeding to (3) the UCB exploration.**

**Definition 3.1** (Relative Cluster)**.** In round $t$, given an arm $\mathbf{x}_{t,i} \in \mathbf{X}_t$, a relative cluster $\mathcal{N}(\mathbf{x}_{t,i}) \subseteq N$ with respect to $\mathbf{x}_{t,i}$ satisfies

(1) $\forall u, u' \in \mathcal{N}(\mathbf{x}_{t,i}), \mathbb{E}[r_{t,i}|u] = \mathbb{E}[r_{t,i}|u']$

(2) $\nexists\, \mathcal{N}' \subseteq N$, s.t. $\mathcal{N}'$ satisfies (1) and $\mathcal{N}(\mathbf{x}_{t,i}) \subset \mathcal{N}'$.

The condition (2) is to guarantee that no other clusters contains $\mathcal{N}(\mathbf{x}_{t,i})$. This cluster definition allows users to agree on certain items while disagree on others, which is consistent with the real-world scenario. Since the users from different clusters are expected to have distinct behavior with respect to $\mathbf{x}_{t,i}$, we provide the following constraint among relative clusters.

**Definition 3.2** ($\gamma$-gap)**.** Given two different cluster $\mathcal{N}(\mathbf{x}_{t,i}), \mathcal{N}'(\mathbf{x}_{t,i})$, there exists a constant $\gamma > 0$, such that

$$\forall u \in \mathcal{N}(\mathbf{x}_{t,i}), u' \in \mathcal{N}'(\mathbf{x}_{t,i}), |\mathbb{E}[r_{t,i}|u] - \mathbb{E}[r_{t,i}|u']| \geq \gamma.$$

For any two clusters in $N$, we assume that they satisfy the $\gamma$-gap constraint. Note that such an assumption is standard in the literature of online clustering of bandit to differentiate clusters [3, 17, 18, 33, 34]. As a result, given an arm $\mathbf{x}_{t,i}$, the bandit pool $N$ can be divided into $q_{t,i}$ *non-overlapping* clusters: $\mathcal{N}_1(\mathbf{x}_{t,i}), \mathcal{N}_2(\mathbf{x}_{t,i}), \ldots, \mathcal{N}_{q_{t,i}}(\mathbf{x}_{t,i})$, where $q_{t,i} \ll n$. Note that the cluster information is *unknown* in the platform.

For the CNB problem, the goal of the learner is to minimize the pseudo regret of $T$ rounds:

$$\mathbf{R}_T = \sum_{t=1}^{T} \mathbb{E}[r_t^* - r_t \mid u_t, \mathbf{X}_t], \tag{2}$$

where $r_t$ is the reward received in round $t$, and $\mathbb{E}[r_t^*|u_t, \mathbf{X}_t] = \max_{\mathbf{x}_{t,i} \in \mathbf{X}_t} h_{u_t}(\mathbf{x}_{t,i})$.

**Notations**. Let $\mathbf{x}_t$ be the arm selected in round $t$, and $r_t$ be the corresponding reward received in round $t$. We use $\|\mathbf{x}_t\|_2$ to represent the Euclidean norm. For each user $u \in N$, let $\mu_t^u$ be the number of rounds that user $u$' learner has been served up to round $t$, and $\mathcal{T}_t^u$ be all of $u$'s historical data up to round $t$. $m$ is the width of neural network and $L$ is depth of neural network in the proposed approach. Given a group $\mathcal{N}$, all its data up to round $t$ can be denoted by $\{\mathcal{T}_t^u\}_{u \in \mathcal{N}} = \{\mathcal{T}_t^u | u \in \mathcal{N}\}$. We use standard $O$ and $\Omega$ notation to hide constants.

## 4 PROPOSED ALGORITHM

In this section, we present our proposed algorithm, denoted as M-CNB, to address the formulated CNB problem. M-CNB leverages the potential correlations among bandits, and aims to rapidly acquire a representation for dynamic relative clusters.

For M-CNB, we utilize a meta-learner, denoted as $\Theta$, to rapidly adapt to clusters, as well as represent the behavior of a cluster. Additionally, there are $n$ user-learners, denoted by $\{\theta^u\}_{u\in N}$, responsible for learning the preference $h_u(\cdot)$ for each user $u \in N$. In terms of the workflow, the primary role of the meta-learner is to determine recommended arms, while the user-learners are primarily utilized for clustering purposes. The meta-learner and user-learners share the same neural network structure, denoted as $f$. And the workflow of M-CNB is divided into three main components: User clustering, Meta adaptation, and UCB-based selection. Then, we proceed to elaborate their details.

**User clustering**. Recall that in Section 3, each user $u \in N$ is governed by an unknown function $h_u$. In this case, we use a neural network $f(\cdot;\theta^u)$, to estimate $h_u$. In round $t \in [T]$, let $u_t$ be the user to serve. Given $u_t$'s past data up to round $t-1$, i.e., $\mathcal{T}_{t-1}^{u_t}$, we can train parameters $\theta^{u_t}$ by minimizing the following loss: $\mathcal{L}(\theta^{u_t}) = \sum_{(\mathbf{x},r)\in\mathcal{T}_{t-1}^{u_t}} (f(\mathbf{x};\theta^{u_t}) - r)^2/2$. Let $\theta_{t-1}^{u_t}$ represent $\theta^{u_t}$ trained on $\mathcal{T}_{t-1}^{u_t}$ in round $t-1$ by stochastic gradient descent (SGD). Therefore, for each $u \in N$, we can obtain the trained parameters $\theta_{t-1}^{u}$. Then, given $u_t$ and an arm $\mathbf{x}_{t,i}$, we return $u_t$'s estimated cluster with respect to arm $\mathbf{x}_{t,i}$ by

$$\widehat{\mathcal{N}}_{u_t}(\mathbf{x}_{t,i}) = \left\{u \in N \;\middle|\; |f(\mathbf{x}_{t,i};\theta_{t-1}^{u}) - f(\mathbf{x}_{t,i};\theta_{t-1}^{u_t})| \le \frac{\nu-1}{\nu}\gamma\right\}. \tag{3}$$

where $\gamma \in (0,1)$ represents the assumed $\gamma$-gap and $\nu > 1$ is a tuning parameter to for the exploration of cluster members.

**Meta adaptation**. We employ one meta-learner $\Theta$ to represent and adapt to the behavior of dynamic clusters. In meta-learning, the meta-learner is trained based on a number of different tasks and can quickly adapt to new tasks with a small amount of new data [15]. Here, we consider a cluster $\mathcal{N}_{u_t}(\mathbf{x}_{t,i})$ as a task and its collected data as the task distribution. As a result, M-CNB has two adaptation phases: meta adaptation, and user adaptation.

*Meta adaptation.* In the $t^{\text{th}}$ round, given a cluster $\widehat{\mathcal{N}}_{u_t}(\mathbf{x}_{t,i})$, we have the available "task distributions" $\{\mathcal{T}_{t-1}^{u}\}_{u\in\widehat{\mathcal{N}}_{u_t}(\mathbf{x}_{t,i})}$. The goal of the meta-learner is to quickly adapt to the bandit cluster. Thus, we randomly draw a few samples from $\{\mathcal{T}_{t-1}^{u}\}_{u\in\widehat{\mathcal{N}}_{u_t}(\mathbf{x}_{t,i})}$ and update $\Theta$ in round $t$ using SGD, denoted by $\Theta_{t,i}$, based on $\Theta_{t-1}$ that is continuously trained on the collected interactions to incorporate the knowledge of past $t-1$ rounds. The workflow is described in Figure 1 and Algorithm 2.

*User adaptation.* In the $t^{\text{th}}$ round, given $u_t$, after receiving the reward $r_t$, we have available data $(\mathbf{x}_t, r_t)$. Then, the user leaner $\theta^{u_t}$ is updated in round $t$ to have a refined clustering capability, denoted by $\theta_t^{u_t}$. As the users in a cluster share the same or similar preferences on a certain item, we update all the user learners in this cluster, described in Algorithm 1 Lines 14-18.

Note that for the clustering of linear bandits works [3, 17, 18, 33, 34], they represent the cluster behavior $\Theta$ by the linear combination of bandit-learners, e.g., $\Theta = \frac{1}{|\widehat{\mathcal{N}}_{u_t}(\mathbf{x}_{t,i})|}\sum_{u\in\widehat{\mathcal{N}}_{u_t}(\mathbf{x}_{t,i})}\theta_t^u$. This can lead to limited representation power of the cluster learner, and their linear reward assumptions may not necessarily hold for real world settings [58]. Instead, we use the meta adaptation to update the meta-learner $\Theta_{t-1}$ according to $\widehat{\mathcal{N}}_{u_t}(\mathbf{x}_{t,i})$, which can represent non-linear combinations of user-learners [15, 48].

**UCB-based Exploration**. To balance the trade-off between the exploitation of the currently available information and the exploration of new matches, we introduce the following UCB-based selection criterion. Based on Lemma A.12, the cumulative error induced by meta-learner is controlled by

**Algorithm 1** M-CNB

1: **Input:** $T$ (number of rounds), $\gamma$, $\nu$ (cluster exploration parameter), $S$ (norm parameter), $\delta$ (confidence level) , $\eta_1, \eta_2$ (learning rate), $m$(width of neural network).
2: Initialize $\Theta_0$; $\theta_0^u = \Theta_0$, $\mu_0^u = 0$, $\mathcal{T}_0^u = \emptyset, \forall u \in N$
3: Observe one data for each $u \in N$
4: **for** $t = 1, 2, \ldots, T$ **do**
5: Receive a target user $u_t \in N$ and observe $k$ arms $\mathbf{X}_t = \{\mathbf{x}_{t,1}, \ldots, \mathbf{x}_{t,k}\}$
6: **for** $i \in [k]$ **do**
7: Determine $\widehat{\mathcal{N}}_{u_t}(\mathbf{x}_{t,i}) = \{u \in N \mid |f(\mathbf{x}_{t,i};\theta_{t-1}^u) - f(\mathbf{x}_{t,i};\theta_{t-1}^{u_t})| \le \frac{\nu-1}{\nu}\gamma\}$.
8: $\Theta_{t,i} = \text{SGD\_Meta}\left(\widehat{\mathcal{N}}_{u_t}(\mathbf{x}_{t,i}), \Theta_{t-1}\right)$
9: $\mathbf{U}_{t,i} = f(\mathbf{x}_{t,i};\Theta_{t,i}) + \frac{\|\nabla_\Theta f(\mathbf{x}_{t,i};\Theta_{t,i}) - \nabla_\theta f(\mathbf{x}_{t,i};\theta_0^{u_t})\|_2}{m^{1/4}} + \sqrt{\frac{S+1}{2\mu_t^u}} + \sqrt{\frac{2\log(1/\delta)}{\mu_t^u}}$
10: **end for**
11: $\widehat{i} = \arg_{i\in[k]}\max \mathbf{U}_{t,i}$
12: Play $\mathbf{x}_{t,\widehat{i}}$ and observe reward $r_{t,\widehat{i}}$
13: $\mathbf{x}_t = \mathbf{x}_{t,\widehat{i}}$, $r_t = r_{t,\widehat{i}}$, $\Theta_t = \Theta_{t,\widehat{i}}$
14: **for** $u \in \widehat{\mathcal{N}}_{u_t}(\mathbf{x}_t)$ **do**
15: $\mathcal{L}_t(\theta_t^u) = (f(\mathbf{x}_t;\theta_t^u) - r_t)^2/2$
16: $\theta_t^u = \theta_t^u - \eta_1 \nabla_{\theta_t^u}\mathcal{L}_t(\theta_t^u)$ # User Adaptation
17: $\mu_t^u = \mu_{t-1}^u + 1$ , $\mathcal{T}_t^u = \mathcal{T}_{t-1}^u \cup \{(\mathbf{x}_t, r_t)\}$
18: **end for**
19: **for** $u \notin \widehat{\mathcal{N}}_{u_t}(\mathbf{x}_t)$ **do**
20: $\theta_t^u = \theta_{t-1}^u$, $\mu_t^u = \mu_{t-1}^u$ , $\mathcal{T}_t^u = \mathcal{T}_{t-1}^u$
21: **end for**
22: **end for**

**Algorithm 2** SGD_Meta $(\widehat{\mathcal{N}}_{u_t}(\mathbf{x}_{t,i}), \Theta_{t-1})$

$\widehat{\mathcal{N}} = \widehat{\mathcal{N}}_{u_t}(\mathbf{x}_{t,i})$
**for** $u \in \widehat{\mathcal{N}}$ **do**
Randomly draw $(\mathbf{x}^u, r^u)$ from $\mathcal{T}_{t-1}^u$
$\mathcal{L}_u(\Theta_{t-1}) = (f(\mathbf{x}^u;\Theta_{t-1}) - r^u)^2/2$
**end for**
$\mathcal{L}_{t-1}(\widehat{\mathcal{N}}) = \frac{1}{|\widehat{\mathcal{N}}|}\sum_{u\in\widehat{\mathcal{N}}}\mathcal{L}_u(\Theta_{t-1})$
$\Theta_{t,i} = \Theta_{t-1} - \eta_2\nabla_{\Theta_{t-1}}\mathcal{L}_{t-1}(\widehat{\mathcal{N}})$ # Meta Adaptation
**Return:** $\Theta_{t,i}$

$$\sum_{t=1}^{T} \mathbb{E}_{r_t|\mathbf{x}_t}\left[|f(\mathbf{x}_t;\Theta_t) - r_t| \,\middle|\, u_t\right] \leq \sum_{t=1}^{T} \underbrace{\frac{O(\|\nabla_\Theta f(\mathbf{x}_t;\Theta_t) - \nabla_\theta f(\mathbf{x}_t;\theta_0^{u_t})\|_2)}{m^{1/4}}}_{\text{Meta-side info}} + \sum_{u\in N} \mu_T^u \Bigg[\underbrace{O\left(\sqrt{\frac{S+1}{2\mu_T^u}}\right) + \sqrt{\frac{2\log(1/\delta)}{\mu_T^u}}}_{\text{User-side info}}\Bigg],$$

where $\nabla_\Theta f(\mathbf{x}_t;\Theta_t)$ incorporates the discriminative information of meta-learner acquired from the correlations within the relative cluster $\widehat{\mathcal{N}}_{u_t}(\mathbf{x}_t)$ and $O(\frac{1}{\sqrt{\mu_T^u}})$ shows the shrinking confidence interval of user-learner to a specific user $u$. Then, we select an arm according to: $\mathbf{x}_t = \arg_{\mathbf{x}_{t,i}\in\mathbf{X}_t} \max \mathbf{U}_{t,i}$ ( where $\mathbf{U}_{t,i}$ is calculated in Line 9).

In summary, Algorithm 1 depicts the workflow of M-CNB. In each round $t$, given a target user and a pool of candidate arms, we compute the meta-learner and its bound for each relative cluster (Line 6-10). Then, we choose the arm according to the UCB-type strategy (Line 11). After receiving the reward, we update the user-learners. Note that the meta-learner has been updated in Line 8.

Then, we discuss the time complexity of Algorithm 1. Here, with $n$ being the number of users, M-CNB will take $O(n)$ to find the cluster for the served user. Given the detected cluster $\widehat{\mathcal{N}}$, it takes $O(|\widehat{\mathcal{N}}|)$ to update the meta-learner by SGD. Suppose $\mathbb{E}[|\widehat{\mathcal{N}}|] = n/\hat{q}$ and $n/\hat{q} \ll n$. Therefore, the overall test time complexity of Algorithm 1 is $O(K(n + n/\hat{q}))$. To scale M-CNB for deployment in large recommender systems, we can rely on the assistance of pre-processing tools: Pre-clustering of users and Pre-selection of items. On the one hand, we can perform pre-clustering of users based on the user features or other information. Then, let a pre-cluster (instead of a single user) hold a neural network, which will significantly reduce $n$. On the other hand, we can conduct the pre-selection of items based on item and user features, to reduce $K$ substantially. For instance, we only consider the restaurants that are near the serving user for the restaurant recommendation task. Furthermore, we can also control the magnitude of $n/\hat{q}$ by tuning the hyperparameter $\nu$ based on the actual application scenario. Consequently, M-CNB can effectively serve as a core component of large-scale recommender systems.

## 5 REGRET ANALYSIS

In this section, we provide the performance guarantee of M-CNB, which is built in the over-parameterized neural networks regime.

As the standard setting in contextual bandits, all arms are normalized to the unit length. Given an arm $\mathbf{x}_{t,i} \in \mathbb{R}^d$ with $\|\mathbf{x}_{t,i}\|_2 = 1$, $t \in [T], i \in [K]$, without loss of generality, we define $f$ as a fully-connected network with depth $L \geq 2$ and width $m$:

$$f(\mathbf{x}_{t,i};\theta \text{ or } \Theta) = \mathbf{W}_L\sigma(\mathbf{W}_{L-1}\sigma(\mathbf{W}_{L-2}\dots\sigma(\mathbf{W}_1\mathbf{x}_{t,i}))) \quad (4)$$

where $\sigma$ is the ReLU activation function, $\mathbf{W}_1 \in \mathbb{R}^{m\times d}$, $\mathbf{W}_l \in \mathbb{R}^{m\times m}$, for $2 \leq l \leq L-1$, $\mathbf{W}^L \in \mathbb{R}^{1\times m}$, and

$$\theta, \Theta = [\text{vec}(\mathbf{W}_1)^\top, \text{vec}(\mathbf{W}_2)^\top, \dots, \text{vec}(\mathbf{W}_L)^\top]^\top \in \mathbb{R}^p.$$

Note that our analysis results can also be readily generalized to other neural architectures such as CNNs and ResNet [2, 12]. Then, we employ the following initialization [7] for $\theta$ and $\Theta$ : For $l \in [L-1]$, each entry of $\mathbf{W}_l$ is drawn from the normal distribution $\mathcal{N}(0, 2/m)$; Each entry of $\mathbf{W}_L$ is drawn from the normal distribution $\mathcal{N}(0, 1/m)$. Here, given $R > 0$, we define the following function class:

$$B(\theta_0, R) = \{\theta \in \mathbb{R}^p : \|\theta - \theta_0\|_2 \leq R/m^{1/4}\}. \quad (5)$$

The term $B(\theta_0, R)$ defines a function class ball centered at the random initialization point $\theta_0$ and with a radius of $R$. This definition was originally introduced in the context of analyzing over-parameterized neural networks, and it can be found in the works of [7] and [2]. Recall that $q_{t,i}$ represents the number of clusters given $\mathbf{x}_{t,i}$. For the simplicity of analysis, we assume $\mathbb{E}[q_{t,i}] = q, t \in [T], i \in [K]$. Let $\{(\mathbf{x}_t, r_t)\}_{t=1}^{TK}$ represent all the data in $T$ rounds and define the squared loss $\mathcal{L}_t(\theta) = (f(\mathbf{x}_t;\theta) - r_t)^2/2$. Then, we provide the instance-dependent regret upper bound for M-CNB with the following theorem.

**Theorem 5.1.** *Given the number of rounds $T$ and $\gamma$, for any $\delta \in (0,1)$, $R > 0$, suppose $m \geq \widetilde{\Omega}(poly(T, L, R) \cdot Kn\log(1/\delta))$, $\eta_1 = \eta_2 = \frac{R^2}{\sqrt{m}}$, and $\mathbb{E}[|\mathcal{N}_{u_t}(\mathbf{x}_t)|] = \frac{n}{q}, t \in [T]$. Then, with probability at least $1-\delta$ over the initialization, Algorithm 1 achieves the following regret upper bound:*

$$\mathbf{R}_T \leq \sqrt{qT \cdot S_{TK}^* + O(1)} + O(\sqrt{2qT\log(O(1)/\delta)}).$$

*where $S_{TK}^* = \inf_{\theta\in B(\theta_0,R)} \sum_{t=1}^{TK} \mathcal{L}_t(\theta)$.*

Theorem 5.1 provides a regret bound for M-CNB, which consists of two main terms. The first term is instance-dependent and relates to the squared error achieved by the function class $B(\theta_0, R)$ on the data. The second term is a standard large-deviation error term.

There are some noteworthy properties regarding Theorem 5.1. One important aspect is that it depends on the parameter $q$, which represents the expected number of clusters, rather than the number of users $n$. Specifically, $\widetilde{O}(\sqrt{T})$ corresponds to the regret effort for learning a single bandit, and thus $\widetilde{O}(\sqrt{nT})$ is an estimate of the regret effort for learning $n$ bandits. However, Theorem 5.1 refines this naive bound to $\widetilde{O}(\sqrt{qT})$, linking the regret effort to the actual underlying clusters among users.

Another advantage of Theorem 5.1 is that it makes no assumptions about the contexts $\{\mathbf{x}_t\}_{t=1}^{TK}$ used in the problem. This makes Theorem 5.1 robust against adversarial attacks on the contexts and allows the observed contexts to contain repeated items. In contrast, existing neural bandit algorithms like [25, 57, 58] rely on Assumption 5.1 for the contexts, and their regret upper bounds can be disrupted by straightforward adversarial attacks, e.g., creating two identical contexts with different rewards.

The term $S_{TK}^*$ reflects the "regression difficulty" of fitting all the data using a given function class, while the radius $R$ controls the richness or complexity of that function class. It's important to note that the choice of $R$ is flexible, although it's not without constraints: specifically, the value of $m$ must be larger than a polynomial of $R$. When $R$ is set to a larger value, it expands the function class $B(\theta_0, R)$, which means it can potentially fit a wider range of data. Consequently, this tends to make $S_{TK}^*$ smaller. Recent advances in

the convergence of neural networks, as demonstrated by [2] and [12], have shown that there is an optimal region around the initialization point in over-parameterized neural networks. This suggests that, with the proper choice of $R$, term $S^*_{TK}$ can be constrained to a small constant value.

Next, we show the common assumption made on existing neural bandits, and prove that Theorem 5.1 is no worse than their regret bounds under the same assumption. The analysis is associated with the Neural Tangent Kernel (NTK) matrix as follows:

**Definition 5.2** (NTK [23, 49])**.** Let $\mathcal{N}$ denote the normal distribution. Given the data instances $\{\mathbf{x}_t\}_{t=1}^T$, for all $i, j \in [T]$, define

$$\mathbf{H}^0_{i,j} = \Sigma^0_{i,j} = \langle \mathbf{x}_i, \mathbf{x}_j \rangle, \quad \mathbf{A}^l_{i,j} = \begin{pmatrix} \Sigma^l_{i,i} & \Sigma^l_{i,j} \\ \Sigma^l_{j,i} & \Sigma^l_{j,j} \end{pmatrix}$$
$$\Sigma^l_{i,j} = 2\mathbb{E}_{a,b\sim\mathcal{N}(\mathbf{0},\mathbf{A}^{l-1}_{i,j})}[\sigma(a)\sigma(b)],$$
$$\mathbf{H}^l_{i,j} = 2\mathbf{H}^{l-1}_{i,j}\mathbb{E}_{a,b\sim\mathcal{N}(\mathbf{0},\mathbf{A}^{l-1}_{i,j})}[\sigma'(a)\sigma'(b)] + \Sigma^l_{i,j}.$$

Then, the NTK matrix is defined as $\mathbf{H} = (\mathbf{H}^L + \Sigma^L)/2$.

**Assumption 5.1.** *There exists $\lambda_0 > 0$, such that $\mathbf{H} \geq \lambda_0 \mathbf{I}$*

The assumption 5.1 is generally held in the literature of neural bandits [4, 5, 10, 24, 52, 57, 58] to ensure the existence of a solution for NTK regression. This assumption holds true when any two contexts in $\{\mathbf{x}_t\}_{t=1}^{TK}$ are not linearly dependent or parallel. Then, the SOTA regret upper bound for a *single* neural bandit ($n = 1$) [4, 10, 57, 58] is as follows:

$$\widetilde{O}(\sqrt{\widetilde{d}T}(S + \sqrt{\widetilde{d}})). \tag{6}$$

There are two complexity terms in the regret bounds [5, 58]. The first complexity term is $S = \sqrt{\mathbf{h}^\top \mathbf{H}^{-1}\mathbf{h}}$, where

$$\mathbf{h} = [h_{u_1}(\mathbf{x}_1), h_{u_1}(\mathbf{x}_2), \ldots, h_{u_T}(\mathbf{x}_{TK})]^\top \in \mathbb{R}^{TK}.$$

The purpose of the term $S$ is to provide an upper bound on the optimal parameters in the context of NTK regression. However, it's important to note that the value of $S$ becomes unbounded (i.e., $\infty$) when the matrix $\mathbf{H}$ becomes singular. This singularity can be induced by an adversary who creates two identical or parallel contexts, causing problems in their analysis.

The second complexity term is the effective dimension $\tilde{d}$, defined as $\widetilde{d} = \frac{\log\det(\mathbf{I}+\mathbf{H})}{\log(1+TK)}$, which describes the actual underlying dimension in the RKHS space spanned by NTK. The following lemma is to show an upper bound of $S^*_{TK}$ under the same assumption.

**Lemma 5.3.** *Suppose Assumption 5.1 and conditions in Theorem 5.1 holds where $m \geq \widetilde{\Omega}(poly(T, L) \cdot Kn\lambda_0^{-1}\log(1/\delta))$. With probability at least $1-\delta$ over the initialization, there exists $\theta' \in B(\theta_0, \widetilde{\Omega}(T^{3/2}))$, such that*

$$\mathbb{E}[S^*_{TK}] \leq \mathbb{E}[\sum_{t=1}^{TK} \mathcal{L}_t(\theta')] \leq \widetilde{O}\left(\sqrt{\widetilde{d}} + S\right)^2 \cdot \widetilde{d}.$$

Lemma 5.3 provides an upper bound for $S^*_{TK}$ by setting $R = \widetilde{\Omega}(T^{3/2})$. Subsequently, by applying the Hoeffding-Azuma inequality over $S^*_{TK}$ and replacing $S^*_{TK}$ with this upper bound, Theorem 5.1 can be reformulated as $\widetilde{O}(\sqrt{\widetilde{d}T}(S + \sqrt{\widetilde{d}}))$ for a single neural bandit or $\widetilde{O}(\sqrt{q\widetilde{d}T}(S + \sqrt{\widetilde{d}}))$ for $n$ users (CNB problem). This transformation implies that Theorem 5.1 is at least as good as the SOTA upper bounds represented by Eq. (6).

## 6 EXPERIMENTS

In this section, we evaluate M-CNB's empirical performance on both online recommendation and classification scenarios. Our source code are anonymously available at *https://anonymous.4open.science/r/Mn-C35C/*.

**Recommendation datasets**. We use four public datasets, Amazon [39], Facebook [31], Movielens [19], and Yelp [1], to evaluate M-CNB's ability in discovering and exploiting user clusters to improve the recommendation performance. Amazon is an E-commerce recommendation dataset consisting of 883636 review ratings. Facebook is a social recommendation dataset with 88234 links. MovieLens is a movie recommendation dataset consisting of 25 million reviews between $1.6 \times 10^5$ users and $6 \times 10^4$ movies. Yelp is a shop recommendation dataset released in the Yelp dataset challenge, composed of 4.7 million review entries made by 1.18 million users towards $1.57 \times 10^5$ merchants. For these four datasets, we extract ratings in the reviews and build the rating matrix by selecting the top 10000 users and top 10000 items (friends, movies, shops) with the most rating records. Then, we use the singular-value decomposition (SVD) to extract a normalized 10-dimensional feature vector for each user and item. The goal of this problem is to select the item with good ratings. Given an item and a specific user, we generate the reward by using the user's rating stars for this item. If the user's rating is more than 4 stars (5 stars total), its reward is 1; Otherwise, its reward is 0. Here, we use pre-clustering (K-means) to form the user pool with 50 users (pre-clusters). Then, in each round, a user $u_t$ is randomly drawn from the user pool. For the arm pool, we randomly choose one restaurant (movie) rated by $u_t$ with reward 1 and randomly pick the other 9 restaurants (movies) rated by $u_t$ with 0 reward. With each restaurant or movie corresponding to an arm, the goal for the learner is to pick the arm with the highest reward.

**Classification datasets**. In our online classification with bandit feedback experiments, we utilized a range of well-known classification datasets, including Mnist [30], Notmnist [6], Cifar10 [28], Emnist (Letter) [9], Fashion [51], as well as the Shuttle, Mushroom, and MagicTelescope (MT) datasets [13]. Here, we provide some preliminaries for this setup. In the round $t \in [T]$, given an instance $\mathbf{x}_t \in \mathbb{R}^d$ drawn from some distribution, we aim to classify $\mathbf{x}_t$ among $K$ classes. $\mathbf{x}_t$ is first transformed into $K$ long vectors: $\mathbf{x}_{t,1} = (\mathbf{x}^\top, 0, \ldots, 0)^\top, \mathbf{x}_{t,2} = (0, \mathbf{x}^\top, \ldots, 0)^\top, \ldots, \mathbf{x}_{t,K} = (0, 0, \ldots, \mathbf{x}^\top)^\top \in \mathbb{R}^{dK}$, matching $K$ classes respectively. The index of the arm that the learner selects is the class predicted by the learner. Then, the reward is defined as 1 if $\mathbf{x}_t$ belongs to this class; otherwise, the reward is 0. In other words, each arm represents a specific class. For example, $\mathbf{x}_{t,1}$ is only presented to Class 1; $\mathbf{x}_{t,2}$ is only presented to Class 2. This problem has been studied in almost all the neural bandit works [5, 25, 57, 58]. Compared to these works, we aim to learn the correlations among classes to improve performance. Thus, we formulate one class as a user (bandit) (i.e.,

[1] https://www.yelp.com/dataset

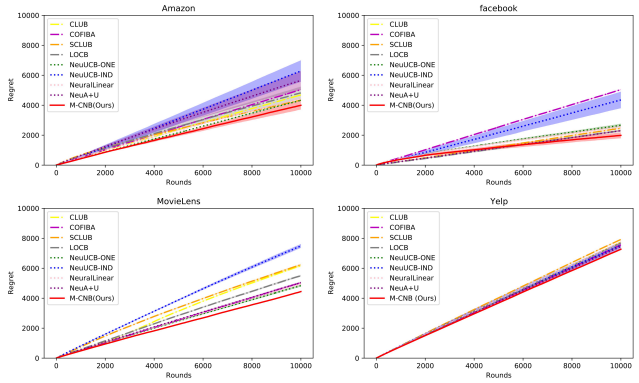



**Figure 2: Regret comparison on recommendation datasets.**

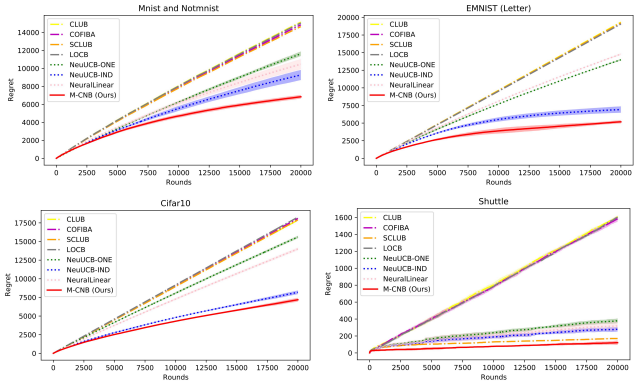



**Figure 3: Regret comparison on Mnist and Notmnist, Cifar10, EMNIST(Letter), and Shuttle.**

a user in the recommendation scenario) and all the samples belonging to this class are deemed as the data of this user. This set of experiments aims to evaluate M-CNB's ability to learn various non-linear reward functions, as well as the ability of discovering and exploiting the correlations among classes. Additionally, we extended the evaluation by combining the Mnist and Notmnist datasets to simulate a more challenging application scenario, given that both datasets involve 10-class classification problems.

**Baselines**. We compare M-CNB with SOTA baselines as follows: (1) CLUB [17] clusters users based on the connected components in the user graph and refines the groups incrementally; (2) COFIBA [33] clusters on both the user and arm sides based on the evolving graph, and chooses arms using a UCB-based exploration strategy; (3) SCLUB [34] improves the algorithm CLUB by allowing groups to merge and split, to enhance the group representation; (4) LOCB [3] uses the seed-based clustering and allows groups to be overlapped. Then, it chooses the best group candidates for arm selection; (5) NeuUCB-ONE [58] uses one neural network to formulate all users, and selects arms via a UCB-based recommendation; (6) NeuUCB-IND [58] uses one neural network to formulate one user separately (totally $n$ networks) and applies the same strategy to choose arms. (7) NeuA+U: we concatenate the arm features and user features together and treat them as the input for the neural network. Note that the user features are only available on Movielens and Yelp datasets. Thus, we only report the results on these two datasets for NeuA+U. (8) NeuralLinear: following the existing work [38, 56]. A shared neural network is built for all users to get an embedding for each arm. which is fed into the linear bandit with the clustering procedure. Since LinUCB [32] and KernalUCB [46] are outperformed by the above baselines, we will not include them for comparison.

**Configurations**. We run all experiments on a server with the NVIDIA Tesla V100 SXM2 GPU. For all the baselines, they all have two parameters: $\lambda$ that is to tune the regularization at initialization and $\alpha$ which is to adjust the UCB value. To find their best performance, we conduct the grid search for $\lambda$ and $\alpha$ over $(0.01, 0.1, 1)$ and $(0.0001, 0.001, 0.01, 0.1)$ respectively. For LOCB, the number of random seeds is set as 20 following their default setting. For M-CNB, we set $\nu$ as 5 and $\gamma$ as 0.4 to tune the cluster, and $S$ is set to 1. To ensure fair comparison, for all neural methods, we use the same simple neural network with 2 fully-connected layers, and the width $m$ is set as 100. To save the running time, we train the neural networks every 10 rounds in the first 1000 rounds and train the neural networks every 100 rounds afterwards. In our implementation, we use Adam [26] for SGD. In the end, we choose the best results for the comparison and report the mean and standard deviation (shadows in figures) of 10 runs for all methods.

**Results**. Figure 2-4 reports the average regrets of all the methods on the recommendation and classification datasets. Figure 2 displays the regret curves for all the methods evaluated on the MovieLens and Yelp datasets. In these experiments, M-CNB consistently outperforms all the baseline methods, showcasing its effectiveness. Specifically, M-CNB improves performance by 5.8% on Amazon, 7.7 % on Facebook, 8.1 % on MovieLens, and 2.0 % on Yelp, compared to the best-performing baseline. These superior results can be attributed to two specific advantages that M-CNB offers over the two types of baseline methods. In contrast to conventional linear clustering of bandits (CLUB, COFIBA, SCLUB, LOCB), M-CNB has the capability to learn non-linear reward functions. This flexibility allows M-CNB to excel in scenarios where user preferences exhibit non-linearity in terms of arm contexts. In comparison to neural bandits (NeuUCB-ONE, NeuUCB-IND, NeuA+U, NeuralLinear), M-CNB takes advantage of user clustering and leverages the correlations within these clusters, as captured by the meta-learner. This exploitation of inter-user correlations enables M-CNB to enhance recommendation performance. By combining these advantages, M-CNB achieves substantial improvements over the MovieLens and Yelp datasets, demonstrating its prowess in addressing collaborative neural bandit problems and enhancing recommendation systems. Note M-CNB's regret rate decreases on these four datasets, even though the "linear-like" behavior in Figure 2.

Figures 3 and 4 show the regret comparison on ML datasets, where M-CNB outperforms all the baselines. Here, each class can be thought of as a user in these datasets. The ML datasets exhibit non-linear reward functions concerning the arms, making them challenging for conventional clustering of linear bandits (CLUB, COFIBA, SCLUB, LOCB). These methods may struggle to capture the non-linearity of the reward functions, resulting in sub-optimal performance. Among the neural baselines, NeuUCB-ONE benefits from the representation power of neural networks. However, it treats all users (classes) as a single cluster, overlooking the variations and

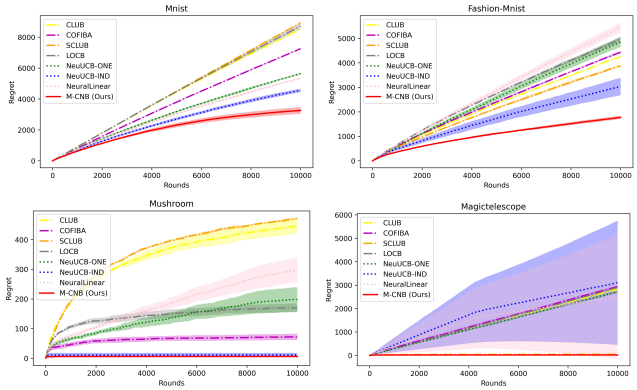



**Figure 4: Regret comparison on Mnist, Fashion-Mnist, Mushroom, and MagicTelescope.**

correlations among them. On the other hand, NeuUCB-IND deals with users individually, neglecting the potential benefits of leveraging collaborative knowledge among users. NeuralLinear uses one shared embedding (neural network) for all users, which may not be the optimal solution given the user heterogeneity. M-CNB's advantage lies in its ability to exploit shared knowledge within clusters of classes that exhibit strong correlations. It leverages this common knowledge to improve its performances across different tasks, as it can efficiently adapt its meta-learner based on past clusters. More discussions with baselines are placed in Appendix B.1.

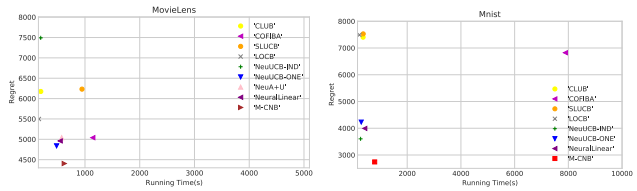



**Figure 5: Running time vs. Performance for all methods.**

**Running time analysis**. Figure 5 demonstrates the trade-off between running time and cumulative regret on both the Movielens and Mnist datasets, where the unit of the x-axis is seconds. As M-CNB is under the framework of neural bandits, we use NeuUCB-ONE as the baseline (1.0). The results indicate that M-CNB takes comparable computation costs (1.6× on Movielens and 2.9× on Mnist) to NeuUCB-ONE while substantially improving performance. This suggests that M-CNB can be deployed to significantly enhance performance when the user correlation is a crucial factor (e.g., recommendation tasks), with only a moderate increase in computational overhead.

Now, let us delve into the analysis of the running time for M-CNB. Specifically, we can break down the computational cost of M-CNB into three main components: (1) *Clustering*: to form the user cluster (Line 7 in Algorithm 1); (2) *Meta adaptation*: to train a meta-model (Algorithm 2); (3) *User-learner training*: to train the user-learners (Lines 14-18 in Algorithm 1).

Table 1 provides the breakdown of the time cost for the three main components of M-CNB. Clustering: This part's time cost grows linearly with the number of users $n$ because it has a time

**Table 1: Breakdown time cost for M-CNB in a round (seconds) with different number of users on MovieLens.**

| | n =500 | n = 5000 | n = 10000 | n = 20000 |
|---|---|---|---|---|
| Clustering | 0.006 | 0.057 | 0.113 | 0.228 |
| Meta adaptation | 0.003 | 0.002 | 0.003 | 0.003 |
| User-learner training | 0.067 | 0.068 | 0.096 | 0.078 |

complexity of $O(n)$ for clustering. As discussed previously, leveraging pre-clustering techniques can significantly reduce this cost. It is also important to note that all clustering methods inherently have this time cost, and it is challenging to further reduce it. Meta adaptation: Due to the benefits of meta-learning, this part requires only a few steps of gradient descent to train a model with good performances. Consequently, the time cost for meta-adaptation is relatively trivial. User-learner training: While this part may require more SGD steps to converge, it is important to recognize that it is primarily used for clustering purposes. Therefore, the frequency of training user-learners can be reduced to decrease the cost. In summary, M-CNB aims to achieve the clustering of neural bandits and can manage to strike a good balance between the computational cost and the model performance.

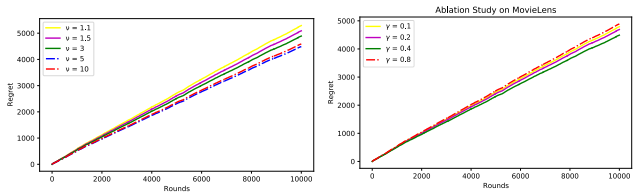



**Figure 6: Sensitivity study for $\nu$ and $\gamma$ on MovieLens Dataset.**

**Study for $\nu$ and $\gamma$.** Figure 6 illustrates the performance variation of M-CNB concerning the parameters $\nu$ and $\gamma$. For the sake of discussion, we will focus on $\nu$ but note that $\gamma$ plays a similar role in terms of controlling clustering. When $\nu$ is set to a value like 1.1, the exploration range of clusters becomes very narrow. In this case, the inferred cluster size in each round, $|\widehat{\mathcal{N}}_{u_t}(\mathbf{x}_{t,i})|$, tends to be small. This means that the inferred cluster $\widehat{\mathcal{N}}_{u_t}(\mathbf{x}_{t,i})$ is more likely to consist of true members of $u_t$'s relative cluster. However, there is a drawback regarding this narrow exploration range: it might result in missing out on potential cluster members in the initial phases of learning. On the other hand, setting $\nu$ to a larger value, like $\nu = 5$, widens the exploration range of clusters. This means that there are more opportunities to include a larger number of members in the inferred cluster. However, continuously increasing $\nu$ does not necessarily lead to improved performances, because excessively large values of $\nu$ might result in inferred clusters that include non-collaborative users and clustering noise. Therefore, in practice, we recommend to set $\nu$ to a relatively large number (e.g., $\nu = 5$) that strikes a balance between the exploration and exploitation.

**Study for $S$.** Figure 7 provides insight into the sensitivity of M-CNB concerning the parameter $S$ in Algorithm 1. It is evident that M-CNB exhibits robust performance across a range of values for $S$. This robustness can be attributed to the strong discriminability of the meta-learner and the derived upper bound. Even with varying $S$ values, the relative order of arms ranked by M-CNB experiences only slightly changes. This consistency in arm rankings

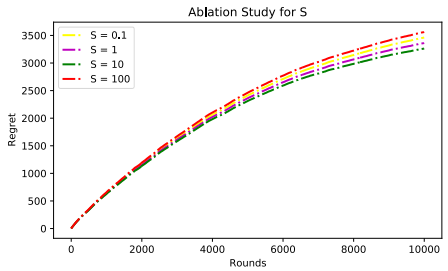



**Figure 7: Sensitivity study for $S$ on Mnist Dataset.**

demonstrates that M-CNB is capable of maintaining the robust performance, which in turn reduces the need for extensive hyper-parameter tuning.

## 7 CONCLUSION

In this paper, we study the Cluster of Neural Bandits problem to incorporate correlation in bandits with generic reward assumptions. Then, we propose a novel algorithm, M-CNB, to solve this problem, where a meta-learner is assigned to represent and rapidly adapt to dynamic clusters, along with an informative UCB-type exploration strategy. Moreover, we provide the instance-dependent regret analysis for M-CNB. In the end, to demonstrate the effectiveness of M-CNB, we conduct extensive experiments to evaluate its empirical performance against strong baselines on recommendation and classification datasets.

## ACKNOWLEDGEMENT

This work is supported by National Science Foundation under Award No. IIS-2002540, and Agriculture and Food Research Initiative (AFRI) grant no. 2020-67021-32799/project accession no.1024178 from the USDA National Institute of Food and Agriculture. The views and conclusions are those of the authors and should not be interpreted as representing the official policies of the funding agencies or the government.

## REFERENCES

[1] Y. Abbasi-Yadkori, D. Pál, and C. Szepesvári. Improved algorithms for linear stochastic bandits. In *Advances in Neural Information Processing Systems*, pages 2312–2320, 2011.
[2] Z. Allen-Zhu, Y. Li, and Z. Song. A convergence theory for deep learning via over-parameterization. In *International Conference on Machine Learning*, pages 242–252. PMLR, 2019.
[3] Y. Ban and J. He. Local clustering in contextual multi-armed bandits. In *Proceedings of the Web Conference 2021*, pages 2335–2346, 2021.
[4] Y. Ban, J. He, and C. B. Cook. Multi-facet contextual bandits: A neural network perspective. In *The 27th ACM SIGKDD Conference on Knowledge Discovery and Data Mining, Virtual Event, Singapore, August 14-18, 2021*, pages 35–45, 2021.
[5] Y. Ban, Y. Yan, A. Banerjee, and J. He. Ee-net: Exploitation-exploration neural networks in contextual bandits. *arXiv preprint arXiv:2110.03177*, 2021.
[6] Y. Bulatov. Notmnist dataset. *Google (Books/OCR), Tech. Rep.[Online]. Available: http://yaroslavvb. blogspot. it/2011/09/notmnist-dataset. html*, 2, 2011.
[7] Y. Cao and Q. Gu. Generalization bounds of stochastic gradient descent for wide and deep neural networks. *Advances in Neural Information Processing Systems*, 32:10836–10846, 2019.
[8] M. Chen, C. Xu, V. Gatto, D. Jain, A. Kumar, and E. Chi. Off-policy actor-critic for recommender systems. In *Proceedings of the 16th ACM Conference on Recommender Systems*, pages 338–349, 2022.
[9] G. Cohen, S. Afshar, J. Tapson, and A. Van Schaik. Emnist: Extending mnist to handwritten letters. In *2017 international joint conference on neural networks (IJCNN)*, pages 2921–2926. IEEE, 2017.
[10] Z. Dai, Y. Shu, A. Verma, F. X. Fan, B. K. H. Low, and P. Jaillet. Federated neural bandit. *arXiv preprint arXiv:2205.14309*, 2022.
[11] V. Dani, T. P. Hayes, and S. M. Kakade. Stochastic linear optimization under bandit feedback. 2008.
[12] S. Du, J. Lee, H. Li, L. Wang, and X. Zhai. Gradient descent finds global minima of deep neural networks. In *International Conference on Machine Learning*, pages 1675–1685. PMLR, 2019.
[13] D. Dua and C. Graff. UCI machine learning repository, 2017. URL http://archive.ics.uci.edu/ml.
[14] P. Dutta, M. Kit, J. S. Kim, M. Mascaro, et al. Automl for contextual bandits. *arXiv preprint arXiv:1909.03212*, 2019.
[15] C. Finn, P. Abbeel, and S. Levine. Model-agnostic meta-learning for fast adaptation of deep networks. In *International Conference on Machine Learning*, pages 1126–1135. PMLR, 2017.
[16] C. Gao, K. Huang, J. Chen, Y. Zhang, B. Li, P. Jiang, S. Wang, Z. Zhang, and X. He. Alleviating matthew effect of offline reinforcement learning in interactive recommendation. *arXiv preprint arXiv:2307.04571*, 2023.
[17] C. Gentile, S. Li, and G. Zappella. Online clustering of bandits. In *International Conference on Machine Learning*, pages 757–765, 2014.
[18] C. Gentile, S. Li, P. Kar, A. Karatzoglou, G. Zappella, and E. Etrue. On context-dependent clustering of bandits. In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, pages 1253–1262. JMLR. org, 2017.
[19] F. M. Harper and J. A. Konstan. The movielens datasets: History and context. *Acm transactions on interactive intelligent systems (tiis)*, 5(4):1–19, 2015.
[20] X. He, L. Liao, H. Zhang, L. Nie, X. Hu, and T.-S. Chua. Neural collaborative filtering. In *Proceedings of the 26th international conference on world wide web*, pages 173–182, 2017.
[21] J. Hong, B. Kveton, M. Zaheer, Y. Chow, A. Ahmed, and C. Boutilier. Latent bandits revisited. *Advances in Neural Information Processing Systems*, 33:13423–13433, 2020.
[22] J. Hong, B. Kveton, M. Zaheer, and M. Ghavamzadeh. Hierarchical bayesian bandits. In *International Conference on Artificial Intelligence and Statistics*, pages 7724–7741. PMLR, 2022.
[23] A. Jacot, F. Gabriel, and C. Hongler. Neural tangent kernel: Convergence and generalization in neural networks. In *Advances in neural information processing systems*, pages 8571–8580, 2018.
[24] Y. Jia, W. Zhang, D. Zhou, Q. Gu, and H. Wang. Learning neural contextual bandits through perturbed rewards. In *International Conference on Learning Representations*, 2022.
[25] P. Kassraie and A. Krause. Neural contextual bandits without regret. In *International Conference on Artificial Intelligence and Statistics*, pages 240–278. PMLR, 2022.
[26] D. P. Kingma and J. Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.
[27] N. Korda, B. Szörényi, and L. Shuai. Distributed clustering of linear bandits in peer to peer networks. In *Journal of machine learning research workshop and conference proceedings*, volume 48, pages 1301–1309. International Machine Learning Societ, 2016.
[28] A. Krizhevsky, G. Hinton, et al. Learning multiple layers of features from tiny images. 2009.
[29] B. Kveton, M. Konobeev, M. Zaheer, C.-w. Hsu, M. Mladenov, C. Boutilier, and C. Szepesvari. Meta-thompson sampling. In *International Conference on Machine Learning*, pages 5884–5893. PMLR, 2021.
[30] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.
[31] J. Leskovec and J. Mcauley. Learning to discover social circles in ego networks. *Advances in neural information processing systems*, 25, 2012.
[32] L. Li, W. Chu, J. Langford, and R. E. Schapire. A contextual-bandit approach to personalized news article recommendation. In *Proceedings of the 19th international conference on World wide web*, pages 661–670, 2010.
[33] S. Li, A. Karatzoglou, and C. Gentile. Collaborative filtering bandits. In *Proceedings of the 39th International ACM SIGIR conference on Research and Development in Information Retrieval*, pages 539–548, 2016.
[34] S. Li, W. Chen, S. Li, and K.-S. Leung. Improved algorithm on online clustering of bandits. In *Proceedings of the 28th International Joint Conference on Artificial Intelligence*, pages 2923–2929. AAAI Press, 2019.
[35] Z. Lipton, X. Li, J. Gao, L. Li, F. Ahmed, and L. Deng. Bbq-networks: Efficient exploration in deep reinforcement learning for task-oriented dialogue systems. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 32, 2018.
[36] O.-A. Maillard and S. Mannor. Latent bandits. In *International Conference on Machine Learning*, pages 136–144. PMLR, 2014.
[37] T. M. McDonald, L. Maystre, M. Lalmas, D. Russo, and K. Ciosek. Impatient bandits: Optimizing recommendations for the long-term without delay. In *Proceedings of the 29th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, pages 1687–1697, 2023.
[38] O. Nabati, T. Zahavy, and S. Mannor. Online limited memory neural-linear bandits with likelihood matching. In *International Conference on Machine Learning*, pages 7905–7915. PMLR, 2021.
[39] J. Ni, J. Li, and J. McAuley. Justifying recommendations using distantly-labeled reviews and fine-grained aspects. In *Proceedings of the 2019 conference on empirical*

*methods in natural language processing and the 9th international joint conference on natural language processing (EMNLP-IJCNLP)*, pages 188–197, 2019.
[40] Y. Qi, Y. Ban, and J. He. Neural bandit with arm group graph. In *Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, pages 1379–1389, 2022.
[41] Y. Qi, Y. Ban, and J. He. Graph neural bandits. In *Proceedings of the 29th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, pages 1920–1931, 2023.
[42] C. Riquelme, G. Tucker, and J. Snoek. Deep bayesian bandits showdown: An empirical comparison of bayesian deep networks for thompson sampling. *arXiv preprint arXiv:1802.09127*, 2018.
[43] M. R. Santana, L. C. Melo, F. H. Camargo, B. Brandão, A. Soares, R. M. Oliveira, and S. Caetano. Contextual meta-bandit for recommender systems selection. In *Fourteenth ACM Conference on Recommender Systems*, pages 444–449, 2020.
[44] M. Simchowitz, C. Tosh, A. Krishnamurthy, D. J. Hsu, T. Lykouris, M. Dudik, and R. E. Schapire. Bayesian decision-making under misspecified priors with applications to meta-learning. *Advances in Neural Information Processing Systems*, 34:26382–26394, 2021.
[45] X. Su and T. M. Khoshgoftaar. A survey of collaborative filtering techniques. *Advances in artificial intelligence*, 2009, 2009.
[46] M. Valko, N. Korda, R. Munos, I. Flaounas, and N. Cristianini. Finite-time analysis of kernelised contextual bandits. *arXiv preprint arXiv:1309.6869*, 2013.
[47] R. Wan, L. Ge, and R. Song. Metadata-based multi-task bandits with bayesian hierarchical models. *Advances in Neural Information Processing Systems*, 34: 29655–29668, 2021.
[48] L. Wang, Q. Cai, Z. Yang, and Z. Wang. On the global optimality of model-agnostic meta-learning. In *International Conference on Machine Learning*, pages 9837–9846. PMLR, 2020.
[49] Z. Wang, P. Awasthi, C. Dann, A. Sekhari, and C. Gentile. Neural active learning with performance guarantees. *Advances in Neural Information Processing Systems*, 34:7510–7521, 2021.
[50] J. Wu, C. Zhao, T. Yu, J. Li, and S. Li. Clustering of conversational bandits for user preference learning and elicitation. In *Proceedings of the 30th ACM International Conference on Information & Knowledge Management*, pages 2129–2139, 2021.
[51] H. Xiao, K. Rasul, and R. Vollgraf. Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms. *arXiv preprint arXiv:1708.07747*, 2017.
[52] P. Xu, Z. Wen, H. Zhao, and Q. Gu. Neural contextual bandits with deep representation and shallow exploration. *arXiv preprint arXiv:2012.01780*, 2020.
[53] W. Xue, Q. Cai, R. Zhan, D. Zheng, P. Jiang, and B. An. Resact: Reinforcing long-term engagement in sequential recommendation with residual actor. *arXiv preprint arXiv:2206.02620*, 2022.
[54] L. Yang, B. Liu, L. Lin, F. Xia, K. Chen, and Q. Yang. Exploring clustering of bandits for online recommendation system. In *Fourteenth ACM Conference on Recommender Systems*, pages 120–129, 2020.
[55] H. Yao, Y. Wei, J. Huang, and Z. Li. Hierarchically structured meta-learning. In *International Conference on Machine Learning*, pages 7045–7054. PMLR, 2019.
[56] T. Zahavy and S. Mannor. Deep neural linear bandits: Overcoming catastrophic forgetting through likelihood matching. *arXiv preprint arXiv:1901.08612*, 2019.
[57] W. Zhang, D. Zhou, L. Li, and Q. Gu. Neural thompson sampling. In *International Conference on Learning Representations*, 2021.
[58] D. Zhou, L. Li, and Q. Gu. Neural contextual bandits with ucb-based exploration. In *International Conference on Machine Learning*, pages 11492–11502. PMLR, 2020.

# A PROOF DETAILS OF THEOREM 5.1

Our proof technique is different from related works. [3, 17, 18, 33, 34] are built on the classic linear bandit framework and [25, 57, 58] utilize the kernel-based analysis in the NTK regime. In contrast, we use the generalization bound of user-learner to bound the error incurred in each round and bridge meta-learner with user-learner by bounding their distance, which leads to our final regret bound. Specifically, we decompose the regret of $T$ rounds into three key terms, where the first term is the error induced by user learner $\theta^u$, the second term is the distance between user learner and meta learner, and the third term is the error induced by the meta learner $\Theta$. Then, Lemma A.10 provides an upper bound for the first term. Lemma A.10 is an extension of Lemma A.7, which is the key to removing the input dimension. Lemma A.7 has two terms with the complexity $O(\sqrt{T})$, where the first term is the training error induced by a class of functions around initialization, the second term is the deviation induced by concentration inequality for $f(\cdot;\theta^u)$. Lemma A.11 bounds the distance between user-learner and meta-learner. Lemma A.12 bounds the error induced by the meta learner using triangle inequality bridged by the user learner. Bounding these three terms completes the proof.

## A.1 Analysis for user-learner

Following [2, 7], given an instance $\mathbf{x} \in \mathbb{B}^d$ with $\|\mathbf{x}\|_2 = 1$, we define the outputs of hidden layers of the neural network (Eq. (4)):

$$\mathbf{h}_0 = \mathbf{x}, \mathbf{h}_l = \sigma(\mathbf{W}_l \mathbf{h}_{l-1}), l \in [L-1].$$

Then, we define the binary diagonal matrix functioning as ReLU:

$$\mathbf{D}_l = \mathrm{diag}(\mathbb{1}\{(\mathbf{W}_l \mathbf{h}_{l-1})_1\}, \ldots, \mathbb{1}\{(\mathbf{W}_l \mathbf{h}_{l-1})_m\}), l \in [L-1].$$

Accordingly, given an input $\mathbf{x}$, the neural network (Eq. (4)) is represented by

$$f(\mathbf{x}_t; \theta \text{ or } \Theta) = \mathbf{W}_L(\prod_{l=1}^{L-1} \mathbf{D}_l \mathbf{W}_l)\mathbf{x},$$

and

$$\nabla_{\mathbf{W}_l} f = \begin{cases} [\mathbf{h}_{l-1}\mathbf{W}_L(\prod_{\tau=l+1}^{L-1} \mathbf{D}_\tau \mathbf{W}_\tau)]^\top, l \in [L-1] \\ \mathbf{h}_{L-1}^\top, l = L. \end{cases}$$

Given a reward $r \in [0,1]$, define $\mathcal{L}(\theta) = (f(\mathbf{x};\theta) - r)^2/2$. Then, we have the following auxiliary lemmas.

**Lemma A.1.** *Suppose $m, \eta_1, \eta_2$ satisfy the conditions in Theorem 5.1. With probability at least $1 - O(TKL) \cdot \exp(-\Omega(m\omega^{2/3}L))$ over the random initialization, for all $t \in [T], i \in [k]$, $\theta$ (or $\Theta$) satisfying $\|\theta - \theta_0\|_2 \le \omega$ with $\omega \le O(L^{-9/2}[\log m]^{-3})$, it holds uniformly that*

$$(1), |f(\mathbf{x};\theta)| \le O(1).$$
$$(2), \|\nabla_\theta f(\mathbf{x};\theta)\|_2 \le O(\sqrt{L}).$$
$$(3), \|\nabla_\theta \mathcal{L}(\theta)\|_2 \le O(\sqrt{L})$$

**Lemma A.2.** *Suppose $m, \eta_1, \eta_2$ satisfy the conditions in Theorem 5.1. With probability at least $1 - O(TKL) \cdot \exp(-\Omega(m\omega^{2/3}L))$, for all $t \in [T], i \in [k]$, $\theta, \theta'$ (or $\Theta, \Theta'$ ) satisfying $\|\theta - \theta_0\|_2, \|\theta' - \theta_0\|_2 \le \omega$ with $\omega \le O(L^{-9/2}[\log m]^{-3})$, it holds uniformly that*

$$|f(\mathbf{x};\theta) - f(\mathbf{x};\theta') - \langle \nabla_{\theta'} f(\mathbf{x};\theta'), \theta - \theta' \rangle| \le O(\omega^{1/3}L^2\sqrt{\log m})\|\theta - \theta'\|_2.$$

**Lemma A.3.** *Suppose $m, \eta_1, \eta_2$ satisfy the conditions in Theorem 5.1. With probability at least $1 - O(TKL) \cdot \exp(-\Omega(m\omega^{2/3}L))$, for all $t \in [T], i \in [k]$, $\theta, \theta'$ satisfying $\|\theta - \theta_0\|_2, \|\theta' - \theta_0\|_2 \le \omega$ with $\omega \le O(L^{-9/2}[\log m]^{-3})$, it holds uniformly that*

$$|f(\mathbf{x};\theta) - f(\mathbf{x};\theta')| \le O(\omega\sqrt{L}) + O(\omega^{4/3}L^2\sqrt{\log m}) \quad (1) \tag{7}$$

**Lemma A.4** (Almost Convexity). *Let $\mathcal{L}_t(\theta) = (f(\mathbf{x}_t;\theta) - r_t)^2/2$. Suppose $m, \eta_1, \eta_2$ satisfy the conditions in Theorem 5.1. For any $\epsilon > 0$, with probability at least $1 - O(TKL^2)\exp[-\Omega(m\omega^{2/3}L)]$ over randomness of $\theta_1$, for all $t \in [T]$, and $\theta, \theta'$ satisfying $\|\theta - \theta_0\|_2 \le \omega$ and $\|\theta' - \theta_0\|_2 \le \omega$ with $\omega \le O(L^{-6}[\log m]^{-3/2}\epsilon^{3/4})$, it holds uniformly that*

$$\mathcal{L}_t(\theta') \ge \mathcal{L}_t(\theta) + \langle \nabla_\theta \mathcal{L}_t(\theta), \theta' - \theta \rangle - \epsilon.$$

**Lemma A.5** (User Trajectory Ball). *Suppose $m, \eta_1, \eta_2$ satisfy the conditions in Theorem 5.1. With probability at least $1-O(TKL^2)\exp[-\Omega(m\omega^{2/3}L)]$ over randomness of $\theta_0$, for any $R > 0$, it holds uniformly that*

$$\|\theta_t - \theta_0\|_2 \leq O(R/m^{1/4}) \leq O(L^{-6}[\log m]^{-3/2}T^{-3/4}), t \in [T].$$

**Lemma A.6** (Instance-dependent Loss Bound). *Let $\mathcal{L}_t(\theta) = (f(\mathbf{x}_t;\theta) - r_t)^2/2$. Suppose $m, \eta_1, \eta_2$ satisfy the conditions in Theorem 5.1. With probability at least $1-O(TKL^2)\exp[-\Omega(m\omega^{2/3}L)]$ over randomness of $\theta_0$, given any $R > 0$ it holds that*

$$\sum_{t=1}^{T} \mathcal{L}_t(\theta_t) \leq \sum_{t=1}^{T} \mathcal{L}_t(\theta^*) + O(1) + \frac{TLR^2}{\sqrt{m}}. \tag{8}$$

*where $\theta^* = \arg\inf_{\theta' \in B(\theta_0,R)} \sum_{t=1}^{T} \mathcal{L}_t(\theta')$.*

**Lemma A.7.** *For any $\delta \in (0,1), R > 0$, and $m, \eta_1, \eta_2$ satisfy the conditions in Theorem 5.1. In a round $\tau$ where $u \in N$ is serving user, let $x_\tau$ be the selected arm and $r_\tau$ is the corresponding received reward. Then, with probability at least $1-\delta$ over the randomness of initialization, the cumulative regret induced by $u$ up to round $T$ is upper bounded by:*

$$\frac{1}{\mu_T^u} \sum_{(\mathbf{x}_\tau, r_\tau) \in \mathcal{T}_T^u} \mathop{\mathbb{E}}_{r_\tau|\mathbf{x}_\tau} [|f(\mathbf{x}_\tau;\theta_{\tau-1}^u) - r_\tau|]$$
$$\leq \sqrt{\frac{S_T^*(u) + O(1)}{\mu_T^u}} + O\left(\sqrt{\frac{2\log(O(1)/\delta)}{\mu_T^u}}\right).$$

*where $S_T^*(u) = \inf_{\theta \in B(\theta_0,R)} \sum_{(\mathbf{x}_\tau, r_\tau) \in \mathcal{T}_T^u} L_\tau(\theta)$.*

**Lemma A.8.** *For any $\delta \in (0,1), R > 0$, and $m, \eta_1, \eta_2$ satisfy the conditions in Theorem 5.1. Suppose $\widehat{\mathcal{N}}_{u_t}(\mathbf{x}_t) = \mathcal{N}_{u_t}(\mathbf{x}_t), \forall t \in [T]$. After $T$ rounds, with probability $1-\delta$ over the random initialization, the cumulative error induced by the bandit-learners is upper bounded by*

$$\sum_{t=1}^{T} \frac{1}{|\mathcal{N}_{u_t}(\mathbf{x}_t)|} \sum_{u_{t,i} \in \mathcal{N}_{u_t}(\mathbf{x}_t)} \mathop{\mathbb{E}}_{r_t|\mathbf{x}_t} [|f(\mathbf{x}_t;\theta_{t-1}^{u_{t,i}}) - r_t|]$$
$$\leq \sqrt{qT \cdot S_{Tk}^* \log(O(\delta^{-1})) + O(1)} + O(\sqrt{2qT\log(O(\delta^{-1}))}),$$

*where $S_{Tk}^* = \inf_{\theta \in B(\theta_0,R)} \sum_{t=1}^{Tk} \mathcal{L}_t(\theta)$ .*

**Corollary A.9.** *For any $\delta \in (0,1), R > 0$, and $m, \eta_1, \eta_2$ satisfy the conditions in Theorem 5.1. In a round $\tau$ where $u \in N$ is the serving user, let $x_\tau^*$ be the arm selected according to Bayes-optimal policy $\pi^*$:*

$$x_\tau^* = \arg\max_{\mathbf{x}_{\tau,i}, i \in [k]} h_u(\mathbf{x}_{\tau,i}),$$

*and $r_\tau^*$ is the corresponding reward. Then, with probability at least $1-\delta$ over the randomness of initialization, after $t \in [T]$ rounds, the cumulative regret induced by $u$ with policy $\pi^*$ is upper bounded by:*

$$\frac{1}{\mu_t^u} \sum_{(\mathbf{x}_\tau^*, r_\tau^*) \in \mathcal{T}_t^{u,*}} \mathop{\mathbb{E}}_{r_\tau^*|\mathbf{x}_\tau^*} [|f(\mathbf{x}_\tau^*;\theta_{\tau-1}^{u,*}) - r_\tau^*| \mid \pi^*]$$
$$\leq \sqrt{\frac{S_T^*(u) + O(1)}{\mu_t^u}} + O\left(\sqrt{\frac{2\log(O(1)/\delta)}{\mu_t^u}}\right).$$

*where $S_T^*(u) = \inf_{\theta \in B(\theta_0,R)} \sum_{(\mathbf{x}_\tau^*, r_\tau^*) \in \mathcal{T}_t^{u,*}} L_\tau(\theta)$, and $\mathcal{T}_t^{u,*}$ are stored Bayes-optimal pairs up to round $t$ for $u$, and $\theta_{\tau-1}^{u,*}$ are the parameters trained on $\mathcal{T}_{\tau-1}^{u,*}$ according to SGD_User in round $\tau - 1$.*

**Corollary A.10.** *For any $\delta \in (0,1), R > 0$, and $m, \eta_1, \eta_2$ satisfy the conditions in Theorem 5.1. In round $t \in [T]$, given $u \in N$, let*

$$x_t^* = \arg\max_{\mathbf{x}_{t,i}, i \in [k]} h_u(\mathbf{x}_{t,i})$$

*the Bayes-optimal arm for $u$ and $r_t^*$ is the corresponding reward. Then, with probability at least $1-\delta$ over the random initialization, after $T$ rounds, with probability $1-\delta$ over the random initialization, the cumulative error induced by the bandit-learners is upper bounded by:*

$$\sum_{t=1}^{T} \mathop{\mathbb{E}}_{r_t^*|\mathbf{x}_t^*} [|f(\mathbf{x}_t^*;\theta_{t-1}^{u_t,*}) - r_t^*|]$$
$$\leq \sqrt{qT \cdot S_{Tk}^* + O(1)} + O(\sqrt{2qT\log(O(1)/\delta)}).$$

*where the expectation is taken over $r_\tau^*$ conditioned on $\mathbf{x}_\tau^*$ and $\theta_{t-1}^{u_t,*}$ are the parameters trained on $\mathcal{T}_{t-1}^{u_t,*}$ according to SGD in round $t-1$.*

## A.2 Bridge Meta-learner and User-learner

For brevity, we use $g(\mathbf{x}_t;\theta_{t-1}^u)$ to represent the gradient $\nabla_\theta f(\mathbf{x}_t;\theta_{t-1}^u)$.

**Lemma A.11.** *Suppose $m, \eta_1, \eta_2$ satisfy the conditions in Theorem 5.1. With probability at least $1-O(nTKL^2)\exp[-\Omega(m\omega^{2/3}L)]$ over randomness of $\Theta_0$, for all $t \in [T]$, any $u \in N$ and $\Theta_t$ returned by Algorithm 2, for any $\|\mathbf{x}_t\|_2 = 1$, it holds uniformly for Algorithms 1-2 that*

$$|f(\mathbf{x}_t;\theta_{t-1}^u) - f(\mathbf{x}_t;\Theta_t)| \leq \frac{R\|\nabla_\Theta f(\mathbf{x}_t;\Theta_t) - \nabla_\theta f(\mathbf{x}_t;\theta_0^u)\|_2}{m^{1/4}} + Z, \tag{9}$$

*where*

$$Z = \frac{O(RL^2\sqrt{\log m})}{m^{1/3}} + \frac{O(L^{7/3}R^2\sqrt{\log m})}{m^{1/2}} + \frac{O(2R\sqrt{L})}{m^{1/4}}.$$

**Lemma A.12.** *Suppose $m, \eta_1, \eta_2$ satisfy the conditions in Theorem 5.1. Then, with probability at least $1-\delta$ over the random initialization, for any $\delta \in (0,1), R > 0$, after $t$ rounds, the error induced by meta-learner is upper bounded by:*

$$\sum_{t=1}^{T} \mathop{\mathbb{E}}_{r_t|\mathbf{x}_t} [|f(\mathbf{x}_t;\Theta_t) - r_t| \mid u_t]$$
$$\leq \sum_{t=1}^{T} \frac{R\|g(\mathbf{x}_t;\Theta_t) - g(\mathbf{x}_t;\theta_0^{u_t})\|_2}{m^{1/4}} + \sum_{u \in N} \mu_T^u \left[ O\left(\frac{\sqrt{S_{Tk}^* + O(1)}}{\sqrt{2\mu_T^u}}\right) + \sqrt{\frac{2\log(O(1)/\delta)}{\mu_T^u}} \right]. \tag{10}$$

*where the expectation is taken over $r_t$ conditioned on $x_t$.*

| | NeuUCB-ONE | NeuUCB-IND | M-CNB |
|---|---|---|---|
| 2 layers | 4839.6 | 7491.0 | **4447.1** |
| 4 layers | 5017.2 | 7503.8 | **4498.3** |
| 8 layers | 5033.5 | 7764.5 | **4696.1** |
| 10 layers | 4808.3 | 7697.4 | **4624.7** |

**Table 3: The cumulative regret of 10000 rounds on MovieLens with the different number of layers.**

| | NeuUCB-ONE | NeuUCB-IND | M-CNB |
|---|---|---|---|
| 2 layers | 6964.1 | 6911.7 | **6773.4** |
| 4 layers | 6944.7 | 6942.5 | **6803.2** |
| 8 layers | 7012.8 | 6932.2 | **6878.3** |
| 10 layers | 6992.4 | 6987.4 | **6854.9** |

**Table 4: The cumulative regret of 10000 rounds on Yelp with the different number of layers.**

# B CONNECTIONS WITH NEURAL TANGENT KERNEL

**Lemma B.1** (Lemma 5.3 Restated)**.** *Suppose $m$ satisfies the conditions in Theorem 5.1. With probability at least $1-\delta$ over the initialization, there exists $\theta' \in B(\theta_0, \widetilde{\Omega}(T^{3/2}))$, such that*

$$\mathbb{E}[S_{Tk}^*] \le \sum_{t=1}^{TK} \mathbb{E}[(r_t - f(\mathbf{x}_t;\theta'))^2/2] \le O\left(\sqrt{\widetilde{d}\log(1+TK) - 2\log\delta} + S + 1\right)^2 \cdot \widetilde{d}\log(1+TK).$$

**Lemma B.2.** *Suppose $m$ satisfies the conditions in Theorem 5.1. With probability at least $1-\delta$ over the initialization, there exists $\theta' \in B(\theta_0, \widetilde{\Omega}(T^{3/2}))$ for all $t \in [T]$, such that*

$$\begin{aligned}&|h(\mathbf{x}_t) - f(\mathbf{x}_t;\theta')| \\ \le& O\left(\sqrt{\log\left(\frac{\det(\mathbf{A}_t)}{\det(\mathbf{I})}\right) - 2\log\delta} + S + 1\right)\|g(\mathbf{x}_t;\theta_0)\|_{\mathbf{A}_t^{-1}} \\ &+ O\left(\frac{T^2L^3\sqrt{\log m}}{m^{1/3}}\right)\end{aligned}$$

## B.1 Additional Results

*Convergence rate on MovieLens and Yelp.* In the figures illustrating the average cumulative regrets of M-CNB for different time step intervals of 2000 rounds on the MovieLens and Yelp datasets, it's noticeable that the average regret per round of M-CNB decreases as more time steps are considered. One plausible explanation for the "linear-like" curves in the cumulative regrets is that both the MovieLens and Yelp datasets used for recommendation tasks contain inherent noise. This noise can make it challenging for algorithms to accurately learn the underlying reward mapping function. Consequently, achieving substantial experimental improvements on these datasets can be quite challenging. In essence, the presence of noise in the data can lead to fluctuations in the regret curves, making it appear as if progress is linear rather than exponential or logarithmic. Despite these challenges, the algorithm, M-CNB, continues to make progress in minimizing regret over time.

| Rounds | 2000 | 4000 | 6000 | 8000 | 10000 |
|---|---|---|---|---|---|
| MovieLens | 0.4717 | 0.4555 | 0.4475 | 0.4452 | 0.4442 |
| Yelp | 0.7532 | 0.7395 | 0.7358 | 0.7306 | 0.7269 |

**Table 2: Convergence rate of M-CNB on MovieLens and Yelp**

*Ablation Study for Network Layers.* We run the experiments on MovieLens and Yelp datasets with the different number of layers of neural networks and report the results in Table 3 and 4. M-CNB achieves the best performance in most cases. In this paper, we try to propose a generic framework to combine meta-learning and bandits with the neural network approximation. Since the UCB in M-CNB only depends on the gradient, the neural network can be easily replaced by other different structures.