# CoCoB: Adaptive Collaborative Combinatorial Bandits for Online Recommendation

Cairong Yan[1][0000−0003−0313−8833], Jinyi Han[2] ⋆[0009−0003−8380−2905], Jin Ju[1],
Yanting Zhang[1][0000−0001−6317−1956], Zijian Wang[1][0000−0002−4096−9428], and
Xuan Shao[1][0000−0002−8147−5109]

[1] School of Computer Science and Technology, Donghua University
[2] Shanghai Institute of Artificial Intelligence for Education, East China Normal University

**Abstract.** Clustering bandits have gained significant attention in recommender systems by leveraging collaborative information from neighboring users to better capture target user preferences. However, these methods often lack a clear definition of "similar users" and face challenges when users with unique preferences lack appropriate neighbors. In such cases, relying on divergent preferences of misidentified neighbors can degrade recommendation quality. To address these limitations, this paper proposes an adaptive Collaborative Combinatorial Bandits algorithm (CoCoB). CoCoB employs an innovative two-sided bandit architecture, applying bandit principles to both the user and item sides. The user-bandit employs an enhanced Bayesian model to explore user similarity, identifying neighbors based on a similarity probability threshold. The item-bandit treats items as arms, generating diverse recommendations informed by the user-bandit's output. CoCoB dynamically adapts, leveraging neighbor preferences when available or focusing solely on the target user otherwise. Regret analysis under a linear contextual bandit setting and experiments on three real-world datasets demonstrate CoCoB's effectiveness, achieving an average 2.4% improvement in F1 score over state-of-the-art methods.

**Keywords:** online recommendation · clustering bandits · contextual multi-armed bandits · combinatorial bandits

## 1 Introduction

Personalized content recommendation is vital for online systems like e-commerce and streaming platforms. Traditional methods, which assume fixed user preferences and item sets, struggle in dynamic scenarios where preferences evolve and new users or items appear [1]. This challenge stems from the exploration-exploitation (EE) dilemma: balancing the exploration of new items for long-term

⋆ corresponding author. E-mail:jinyihan099@gmail.com.
This work is supported by the National Natural Science Foundation of China under Grant 62477006 and 62206046.

user satisfaction with exploiting known information for immediate recommendations. Multi-armed bandit (MAB), a reinforcement learning approach, effectively addresses the EE problem by sequentially selecting actions (arms) with unknown reward distributions, using user feedback to refine future choices.

Contextual MAB (CMAB) and combinatorial MAB, as extensions of MAB, are widely studied and applied in real-world recommender systems. CMAB incorporates contextual features for each arm, enabling more personalized and higher-quality decisions [2]. Combinatorial MAB selects multiple arms simultaneously, aligning better with practical needs by recommending multiple items at once. This paper leverages both frameworks to enhance the effectiveness of bandit algorithms in recommender systems.

Most bandit-based recommendation models, including CMAB and combinatorial bandits, take one of two extremes: a global bandit for all users [3], which learns from shared reward information but may overlook individual preferences, or an independent bandit for each user [4], offering fully personalized recommendations but struggling with sparse interaction data. To address this, some methods leverage user collaboration by clustering similar users based on shared preferences and using cluster information for recommendations[5] . This approach allows users within a cluster to collectively estimate arm rewards. However, for users with unique preferences, clustering fails to identify meaningful neighbors, making the process redundant or even detrimental to recommendation performance. This occurs because the preferences of “neighboring users” in such cases deviate significantly from the target user’s preferences.

The clustering method is central to clustering bandits, with two main approaches. The first uses traditional methods like K-Nearest Neighbor (KNN) [6], which are well-established but rely on static user attribute features, making it difficult to adapt to evolving preferences. The second employs graph-based methods, where nodes represent users, edges represent user similarities, and connected components define clusters [7]. While this approach dynamically updates preferences by modifying edges, it fails to re-cluster similar users once separated, even if their preferences align later. Additionally, both approaches lack a clear quantitative definition of “neighboring users”.

To deal with these issues in current clustering bandits approaches, this paper propose an adaptive Collaborative Combinatorial Bandits (CoCoB) algorithm. Built on a linear contextual bandit framework, CoCoB is a simple yet flexible solution. Our contributions are threefold:

- Introduced a two-sided bandit framework, consisting of a user-bandit to model user similarities and an item-bandit to generate recommendation lists based on these preferences. This design enables the CoCoB algorithm to adaptively combine user collaboration and individual preferences.
- Proposed an enhanced Bayesian bandit-based strategy for selecting neighboring users. This approach quantitatively defines neighboring users by setting a similarity probability threshold, eliminating the need for user attributes. It effectively identifies and determines the presence of neighboring users.

- Provided a thorough analysis of the CoCoB algorithm, including a theoretical regret bound under the linear contextual bandit framework, and experimental results on three real-world datasets showing CoCoB's superiority over state-of-the-art contextual bandit algorithms.

## 2 Problem Formulation

We frame a typical e-commerce recommendation task as an adaptive collaborative combinatorial bandit problem, focusing on dynamically identifying neighboring users and integrating this into the combinatorial bandit framework. **Environment**. Given an item set $I$ and a user set $U$ with $|U| = m$, at each iteration, a user $u$ is randomly selected. At time $t$, the agent identifies $u$'s neighbors, adding them to the neighboring user set $N_{ut}$. If neighbors are found, their preference data is incorporated to help infer $u$'s preferences. Since users' preferences evolve over time, the neighboring user set is updated regularly in each iteration.

**Actions**. At time $t$, the items in the candidate set $E_t \subseteq I$ are modeled as arms, where $|E_t| = J$. Based on the environment and reward experience from previous iterations, the agent makes selections from the top-$K$ performing arms tailored to the user, which can be expressed as $A_t$ ($A_t \subseteq E_t$).

**Context**. Contextual information, such as an item's color, can enhance action quality. At time $t$, the agent receives a set of context vectors $C_t = \{x_1, \cdots, x_J\}$ ($x_j \in R^d, 1 \leq j \leq J$, where $d$ is the feature vector dimension), representing the contextual information of $J$ candidate items for the target user. Since the users change frequently and may be served multiple times, the contextual information the agent encounters is uncertain at each step.

**Policy**. In the contextual bandit setting, an unknown user vector $\theta_u \in R^d$ determines user $u$'s preferences. At time $t$, the expected payoff $\mu_{at}$ of item $a$ is typically expressed as a linear function of $\theta_u$.

$$\mu_{at} = \tilde{\theta}_{N_{ut}}^{\mathrm{T}} x_a + f_{UCB}^{a}, \tag{1}$$

where $\tilde{\theta}_{N_{ut}}$ represents the context vector that determines the preferences of $N_{ut}$. T is the transpose symbol, and $f_{UCB}^{a}$ is a function that is used to represent the upper confidence bound of arm $a$. If there are no neighboring users, $N_{ut} = \{u\}$. At time $t$, the $K$ arms with the highest estimated rewards are selected, and the responding items are added to the recommendation list $A_t$.

**Reward**. The reward reflects the target user's satisfaction with the recommended items, typically modeled as a Bernoulli distribution: a reward of 1 for satisfaction and 0 for dissatisfaction. In recommender systems, feedback can be explicit, such as ratings, or implicit, like clicks or add-to-carts. While explicit feedback is costly and requires user involvement, implicit feedback is more common and still indicates user preferences. At time $t$, the agent recommends $A_t$ and calculates the reward $\bar{r}_t$ based on user feedback for the $K$ items in $A_t$.

$$\bar{r}_t = \frac{1}{K} \sum_{a \in A_t} r_a, \tag{2}$$

where $r_a$ is the reward of arm $a$. The goal of the bandit algorithm is to maximize the total reward $\sum_{t=1}^{T} \bar{r}_t$ over $T$ steps. This paper primarily focuses on bounding the cumulative regret of the bandit algorithm. At time $t$,

$$Reg\,(T) = \sum_{t=1}^{T} Reg(t) = \sum_{t=1}^{T} r_t^* - r_t, \tag{3}$$

where $Reg\,(T)$ is the cumulative regret over $T$ steps, and $r_t^*$ represents the reward of the optimal $K$ arms at time $t$.

The recommendation process proceeds as follows. At time $t$, the system observes the environment to check for neighboring users. If present, their preferences are incorporated into the context to aid in the recommendation. Based on this contextual information and prior knowledge, the system recommends a list of items to the target user. The system then collects feedback on the recommendations and calculates the reward to inform future recommendations.

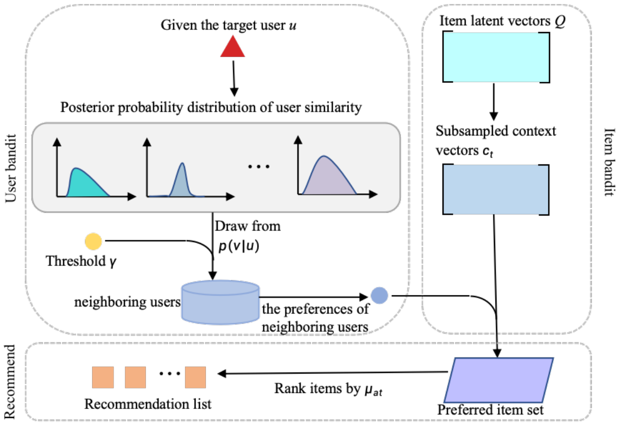

Fig. 1: The general framework of the proposed CoCoB recommendation.

# 3 Methodology

## 3.1 CoCoB Algorithm

The CoCoB algorithm adaptively exploits the collaborative effect between users to extract user preferences. The general framework of the proposed CoCoB recommendation is illustrated in Fig. 1. The pseudocode is shown in Algorithm 1. Three parts of CoCoB algorithm will be introduced below.

**User-bandit part**. User-bandit implements the function of user collaboration by finding the neighboring users. As an improved TS bandit, it adaptively exploits user collaboration to discover user preferences. According to the TS principle, users are modeled as arms to explore similarities among users. There are available $m$ arms that can be pulled each time, representing that no more than $m$ users in the system can be neighboring users for the target user. At time $t$, for a target user $u$, user-bandit derives an estimate of $u$'s potential neighboring user $v$ from the Beta posterior with parameters $\alpha\left(v \mid u\right)$ and $\beta\left(v \mid u\right)$. $\alpha\left(v \mid u\right)$ represents the times $u$ is satisfied with the item selected by $v$ and $\beta\left(v \mid u\right)$ means the opposite. Here, $\alpha\left(v \mid u\right) = \alpha\left(u \mid v\right)$ and $\beta\left(v \mid u\right) = \beta\left(u \mid v\right)$. The sampled value $p_t\left(v \mid u\right)$ of each user $v$ given by the target user $u$ is defined as:

$$p_t\left(v \mid u\right) = Beta\left(\alpha\left(v \mid u\right), \beta\left(v \mid u\right)\right). \tag{4}$$

Suppose $\gamma$ is a threshold parameter that controls the similarity probability of users. In this paper, if $p_t\left(v \mid u\right) \geq \gamma$, $v$ is defined as $u$'s neighboring user at time $t$. According to Bayesian theory, the larger $\gamma$ is, the more reliable the neighboring users are. $N_{ut}$ is the neighboring user set of $u$ at time $t$. If $N_{ut}$ is empty, it means that there are no neighboring users of $u$ under the constraint of the posterior probability threshold $\gamma$. To facilitate unified operation, user $u$ is added to $N_{ut}$ when $N_{ut}$ is empty. It represents that only $u$'s individual preference information is used to assist the recommendation. Moreover, the user preferences change dynamically. At different times, the same user may have different neighboring users or even no neighboring users. Therefore, the similarities between the target user and other users in the system need to be calculated each time. A detailed step-wise description of user-bandit is shown in lines 5∼14 in Algorithm 1.

**Item-bandit part**. Item-bandit is responsible for providing users with a list of recommendations for each round like other combinatorial bandits. First, each user profile is initialized at the beginning of Algorithm 1. Vector $w_u$ serves as a proxy to the unknown preferences of user $u$, which is defined as $w_u = M_u^{-1} b_u$. At time $t$, CoCoB servers user $u$ by providing an item list $A_t$ from a set of items $E_t$ represented as $C_t$. According to equation 2, the upper confidence estimation $\mu_{at}$ of each arm $a$ at time $t$ is decided by aggregated confidence bounds and aggregated proxy vectors, defined as

$$\mu_{at} = \hat{w}_{N_{ut}}^{\mathrm{T}} x_a + f\left(x_a \mid N_{ut}\right) = \hat{w}_{N_{ut}}^{\mathrm{T}} x_a + e\sqrt{x_a^{\mathrm{T}} \hat{M}_{N_{ut}}^{-1} x_a \log\left(1+t\right)}, \tag{5}$$

where $e$ is the exploration probability, and $\hat{w}_{N_{ut}}$ and $\hat{M}_{N_{ut}}$ are the parameters of $N_{ut}$, defined as line 15∼17 in Algorithm 1. CoCoB selects a set of items with the top $K$ largest upper confidence estimation according to equation 5. A detailed stepwise description of the item-bandit approach is shown in lines 18∼20 in Algorithm 1.

**Update part**. CoCoB observes the target user's response to the recommendation list $A_t$, calculates the reward obtained by each item, and finally takes the average reward $\bar{r}_t$ of all items in $A_t$ as the recommendation reward and the average contextual vector $\tilde{x}_t$ of all items in $A_t$ as the feature presentation of the recommendation list. Then, CoCoB uses $\bar{r}_t$ and $\tilde{x}_t$ to update the user parameter $w_u$ by solving a regularized least square problem as shown. Besides, it also updates the similarity probability distribution between target user and other users according to the recommendation reward $\bar{r}_t$. If $\bar{r}_t$ is greater than 0, it indicates that neighboring users in $N_{ut}$ provide positive help for this recommendation, then for each user $v$ in $N_{ut}$, update parameter $\alpha(v \mid u)$. Otherwise, update $\beta(v \mid u)$.

**Algorithm 1:** CoCoB

```
Input: Parameters $\gamma$, $\alpha$, $\beta$.
Output: A recommendation list $A_t$ at time $t$.
Initialize each $u$ by $b_u \leftarrow 0 \in R^d$ and $M_u \leftarrow I \in R^{d \times d}$;
for $t \leftarrow 1$ to $T$ do
    Initialize recommendation list $A_t \leftarrow \emptyset$;
    Receive a user $u$ to be served, $w_u \leftarrow M_u^{-1} b_u$;
    Find $u$'s neighborhood $N_{ut} \leftarrow \emptyset$;
    for all $v \in U$ do
        $p_t(v \mid u) \leftarrow Beta(\alpha(v \mid u), \beta(v \mid u))$;
        if $p_t(v \mid u) \geq \gamma$ then
            $N_{ut} \leftarrow N_{ut} \cup \{v\}$;
        end
    end
    if $N_{ut} = \emptyset$ then
        $N_{ut} \leftarrow N_{ut} \cup \{u\}$;
    end
    Set $\hat{M}_{N_{ut}} \leftarrow \frac{1}{|N_{ut}|} \sum_{v \in N_{ut}} M_v$,
    $\hat{b}_{N_{ut}} \leftarrow \frac{1}{|N_{ut}|} \sum_{v \in N_{ut}} b_v$,
    $\hat{w}_{N_{ut}} \leftarrow \hat{M}_{N_{ut}}^{-1} \hat{b}_{N_{ut}}$;
    Estimate $\mu_{at}$, $a \in E_t$;
    Sort $\mu_{at}$ and select the top $K$ value with the order $(a_1^t, \cdots, a_K^t)$;
    Recommend $A_t \leftarrow A_t \cup (a_1^t, \cdots, a_K^t)$ to user $u$ and observe the
      reward $\{r_a\}_{a \in A_t}$;
    Set $\tilde{x}_t = \frac{1}{K} \sum_{a \in A_t} x_a$,
    $\bar{r}_t = \frac{1}{K} \sum_{a \in A_t} r_a$;
    Run Update;
end
```

### 3.2 Theoretical Analysis

**Algorithm complexity analysis**. Let $m$ be the number of users, $J$ the number of candidate items, and $d$ the feature dimension. At each step, finding neighboring users takes $O(m)$. Each recommendation has a complexity of $O\left(Jd^2\right)$, with matrix inverse updated using the Sherman–Morrison formula. Updating user parameters requires $O\left(d^2\right)$, and updating the posterior probability distribution takes $O\left(\bar{N}\right)$, where $\bar{N}$ is the average number of neighboring users. Thus, the computational time complexity for $T$ rounds of CoCoB is $O\left(T\left(m + d^2(J+1) + \bar{N}\right)\right)$. In cases with no neighboring users, $\bar{N}$ is 1.

**Regret analysis**. The design is primarily based on the proof idea of CAB [8]. The cumulative regret $Reg(T)$ of CoCoB consists of two terms, defined as:

$$Reg(T) = Reg(T)_{\text{item-bandit}} + Reg(T)_{\text{user-bandit}} \; . \tag{6}$$

The first term is the regret analysis of item-bandit, which follows the typical $\sqrt{T}$-style term seen in linear bandit regret analyses [9, 10]. For independent users, the first term takes the form $\sqrt{d \log T \sum_{t=1}^{T} m}$. However, in CoCoB, the dependence on the total number of users $m$ is replaced by a much smaller quantity, $\frac{m}{|N_{ut}|}$, which reflects the expected number of context-dependent user clusters. The regret bound for the item-bandit, $Reg(T)_{item-bandit}$, is defined as:

$$\text{Reg}(T)_{\text{item-bandit}} \le 9e \sqrt{d \log T \sum_{t=1}^{T} \frac{m}{|N_{ut}|}}. \tag{7}$$

The second term is the regret analysis of user-bandit, a variant of the Multi-play MAB problem. In this case, the optimal arms are the top $B$ arms (i.e., arms $[B]$), while the suboptimal arms are the remaining ones (i.e., arms $N_{ut} \setminus [B]$). Let the selected suboptimal arm and the excluded optimal arm be $i$ and $j$, respectively. The regret is defined as

$$\text{Reg}(T)_{\text{user-bandit}} = \sum_{t=1}^{T} \left( \sum_{j \in [B]} \mu_j - \sum_{i \in N_{ut}} \mu_i \right). \tag{8}$$

It was proven that for any strongly consistent algorithm and suboptimal arm $i$, the number of arm $i$ draws $Z(i, T+1)$ is lower-bounded as

$$E[Z(\{i, T+1\})] \le \frac{\log T}{(1-\varepsilon)\, \text{dist}(\mu_i, \mu_B)}, \tag{9}$$

where $dist\,(p, q) = p \log{(pq)} + (1-p) \log{(1-p)(1-q)}$ is the Kullback-Leibler $(KL)$ divergence between two Bernoulli distributions with the expectation $p$ and $q$ and $\varepsilon$ $(0 < \varepsilon < 1)$ is a constant. For user-bandit, the highest similarity of target users is the users themselves, which means that $\max \mu_{u \in U} = 1$. Then the loss in the expected regret at each round is given by

$$\sum_{j \in [B]} \mu_j - \sum_{i \in N_{ut}} \mu_i = \sum_{j \in [B] \setminus [N_{ut}]} \mu_j - \sum_{i \in N_{ut} \setminus [B]} \mu_i \le \sum_{i \in N_{ut} \setminus [B]} (1 - \mu_i). \tag{10}$$

The regret is expressed as

$$\text{Reg}(T)_{\text{user -bandit}} \le \sum_{t=1}^{T} \sum_{i \in N_{ut} \setminus [B]} (1 - \mu_i) = \sum_{i \in N_{ut} \setminus [B]} (1 - \mu_i)\, Z(i, T+1). \tag{11}$$

Combining the Equation 9, it is calculated that

$$\text{Reg}(T)_{\text{user-bandit}} \le \frac{\log T}{(1-\varepsilon) dist(\mu_i, 1)} \sum_{i \in N_{ut} \setminus [B]} (1 - \mu_i). \tag{12}$$

Therefore, according to Equation 7 and Equation 12, we can get the following Theorem 1.

**Theorem 1.** *Suppose CoCoB is executed under the linear contextual bandit setting as described in Section 3, and the collaborative effect of users satisfy the assumptions stated in Section 4. Let* $e = \mathcal{O}\left(\sqrt{logT}\right)$. *Then the cumulative regret* $\sum_{t=1}^{T} Reg\left(t\right)$ *of CoCoB can be upper bounded as*

$$\mathrm{Reg}(T) \leq 9e\sqrt{d\log T\sum_{t=1}^{T}\frac{m}{|N_{ut}|}} + \frac{\log T}{(1-\varepsilon)dist(\mu_i,1)}\sum_{i\in N_{ut}\setminus[B]}(1-\mu_i)\,. \qquad (13)$$

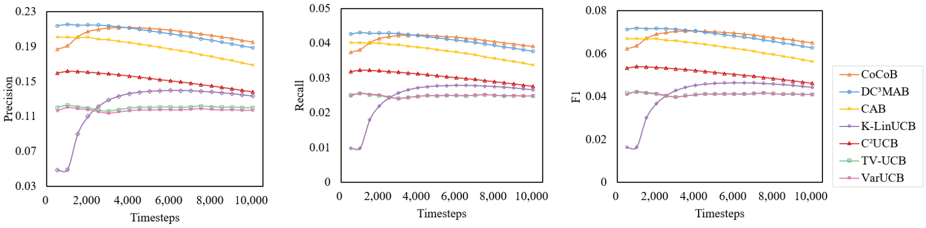

Fig. 2: Recommendation performance of different algorithms on IJCAI-15.

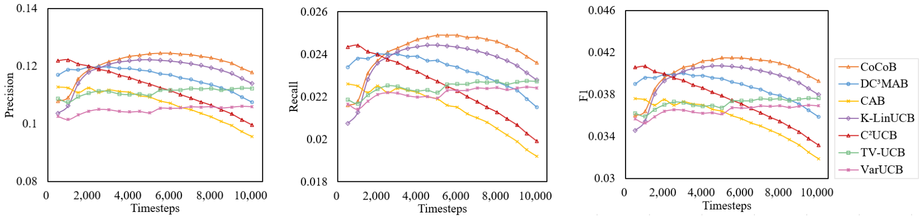

Fig. 3: Recommendation performance of different algorithms on Retailrocket.

# 4 Experimental Result and Analysis

## 4.1 Experimental Settings

Experiments were conducted on three real-world datasets, IJCAI-15, Retailrocket, and Yoochoose. Following previous work [11], user preferences for interacting items were assigned ratings: 1 for clicks, 2 for favorites, 3 for add-to-cart, and 4 for purchases. Matrix factorization was used to generate the users' contextual feature vectors. Four metrics were used to measure the quality and

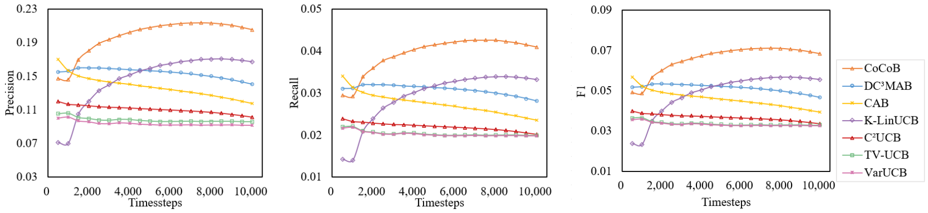

Fig. 4: Recommendation performance of different algorithms on Yoochoose.

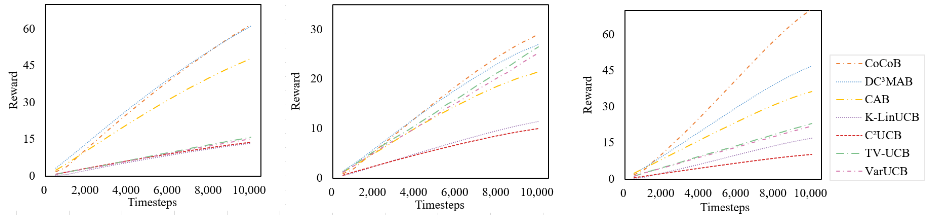

Fig. 5: The performance of different algorithms on the IJCAI-15, Retailrocket, and Yoochoose datasets as measured by cumulative reward. The reward values correspond to the experimental results divided by 1,000.

effectiveness of the algorithms: cumulative reward (CR), precision, recall, and F1 score. We compare CoCoB with several algorithms, including $C^2$UCB [12], $DC^3MAB$ [13], CAB [8], $K$-LinUCB [10], VarUCB [14], and TV-UCB [15].

### 4.2 Experiments Results and Analysis

**Main Results.** The results are shown in Fig. 2, Fig. 3, Fig. 4 and Fig. 5. All results were averaged over five runs. From the results, we observe that when the number of experiment rounds exceeds 2,000, CoCoB outperforms other algorithms on all metrics. However, when the Timesteps value is small, CoCoB performs poorly compared to other baselines, such as $DC^3MAB$. This is because, for new users, the system lacks information about their preferences and assumes that similar users do not exist in the system. In such cases, CoCoB relies solely on the target user's information to make recommendations. In contrast, other baselines consistently utilize information from other users to assist with recommendations. Moreover, a common pattern emerges in the performance of these algorithms, as shown in the three figures: an initial dip in recommendation accuracy is followed by a subsequent increase that eventually stabilizes. This phenomenon can be attributed to the exploratory phase of bandit algorithms.

## 5 Conclusion

This paper proposes CoCoB, a novel bandit algorithm that enhances online recommendation performance by leveraging user collaboration. CoCoB adopts a dual mechanism with user-bandit and item-bandit components. The user-bandit, an improved Thompson sampling model, dynamically identifies neighboring users based on a predefined similarity threshold, ensuring alignment with the target user's preferences. The item-bandit, a combinatorial bandit, generates a recommendation list in each iteration. Empirical results demonstrate its superior performance and efficiency.

## References

1. Xiao Xu, Fang Dong, Yanghua Li, Shaojian He, and Xin Li. Contextual-bandit based personalized recommendation with time-varying user interests. In *Proceedings of AAAI*, pages 6518–6525, 2020.
2. Zhao Li, Junshuai Song, Zehong Hu, Zhen Wang, and Jun Gao. Constrained dual-level bandit for personalized impression regulation in online ranking systems. *ACM Transactions on Knowledge Discovery from Data*, 16:1–23, 2021.
3. Shuai Li, Baoxiang Wang, Shengyu Zhang, and Wei Chen. Contextual combinatorial cascading bandits. In *Proceedings of ICML*, pages 1245–1253, 2016.
4. Anisio Lacerda. Multi-objective ranked bandits for recommender systems. *Neurocomputing*, 246:12–24, 2017.
5. Qing Wang, Chunqiu Zeng, Wubai Zhou, Tao Li, S Sitharama Iyengar, Larisa Shwartz, and Genady Ya Grabarnik. Online interactive collaborative filtering using multi-armed bandit with dependent arms. *IEEE Transactions on Knowledge and Data Engineering*, 31(8):1569–1580, 2018.
6. Trong T Nguyen and Hady W Lauw. Dynamic clustering of contextual multi-armed bandits. In *Proceedings of CIKM*, pages 1959–1962, 2014.
7. Claudio Gentile, Shuai Li, and Giovanni Zappella. Online clustering of bandits. In *Proceedings of ICML*, pages 757–765, 2014.
8. Claudio Gentile, Shuai Li, Purushottam Kar, Alexandros Karatzoglou, Giovanni Zappella, and Evans Etrue. On context-dependent clustering of bandits. In *Proceedings of ICML*, pages 1253–1262, 2017.
9. Peter Auer, Nicolo Cesa-Bianchi, and Paul Fischer. Finite-time analysis of the multiarmed bandit problem. *Machine learning*, 47:235–256, 2002.
10. Lihong Li, Wei Chu, John Langford, and Robert E Schapire. A contextual-bandit approach to personalized news article recommendation. In *Proceedings of the WWW*, pages 661–670, 2010.
11. Cairong Yan, Junli Xian, Yongquan Wan, and Pengwei Wang. Modeling implicit feedback based on bandit learning for recommendation. *Neurocomputing*, 447:244–256, 2021.
12. Lijing Qin, Shouyuan Chen, and Xiaoyan Zhu. Contextual combinatorial bandit and its application on diversified online recommendation. In *Proceedings of SDM*, pages 461–469, 2014.
13. Cairong Yan, Haixia Han, Yanting Zhang, Dandan Zhu, and Yongquan Wan. Dynamic clustering based contextual combinatorial multi-armed bandit for online recommendation. *Knowledge-Based Systems*, 257:109927, 2022.
14. Lei Xu, Chunxiao Jiang, Yi Qian, Youjian Zhao, Jianhua Li, and Yong Ren. Dynamic privacy pricing: A multi-armed bandit approach with time-variant rewards. *IEEE Transactions on Information Forensics and Security*, 12:271–285, 2016.
15. Cairong Yan, Hualu Xu, Haixia Han, Yanting Zhang, and Zijian Wang. Thompson sampling with timevarying reward for contextual bandits. In *Proceedings of DASFAA*, pages 54–63, 2023.