Improved Algorithm on Online Clustering of Bandits

Shuai Li[1] , **Wei Chen**[2] , **Shuai Li**[3] and **Kwong-Sak Leung**[1]
[1]The Chinese University of Hong Kong
[2]Microsoft
[3]University of Cambridge
{shuaili, ksleung}@cse.cuhk.edu.hk, weic@microsoft.com, shuaili.sli@gmail.com

Abstract

We generalize the setting of online clustering of bandits by allowing non-uniform distribution over user frequencies. A more efficient algorithm is proposed with simple set structures to represent clusters. We prove a regret bound for the new algorithm which is free of the minimal frequency over users. The experiments on both synthetic and real datasets consistently show the advantage of the new algorithm over existing methods.

1 Introduction

The problem of stochastic multi-armed bandit (MAB) [Bubeck *et al.*, 2012; Lattimore and Szepesvári, 2018] has been widely studied in the field of statistics and machine learning, where the learner selects an action each round with the goal of maximizing the cumulative rewards over all rounds (or equivalently minimize the cumulative regret). One successful application of MAB algorithms is recommendation systems [Aggarwal, 2016], such as recommendations of movies, restaurants and news articles. For the setting where new users and items constantly appear where simultaneous exploration and exploitation are naturally needed, bandit algorithms can cope with the "cold start" problem and improve itself as time goes on.

To adjust bandit algorithms for large-scale applications, structural assumptions are usually added on actions and reward functions, among which linear structure is one of the most common due to its simplicity and effectiveness. Each item in the stochastic linear bandits [Abe and Long, 1999; Rusmevichientong and Tsitsiklis, 2010; Dani *et al.*, 2008; Abbasi-Yadkori *et al.*, 2011] is represented by a feature vector whose expected reward is an unknown linear function on the feature vector. In many systems certain side information about users and items are employed as the feature vectors to guarantee recommendation quality [Li *et al.*, 2010; Chu *et al.*, 2011].

Usually user features are not accurate enough or even do not exist to characterize user preferences. When user features exist, these static features are usually combined together with item features to be actions in linear bandit models. This processing might have implacable modeling bias and also neglect the often useful tool of collaborative filtering. One way to utilize the collaborative effect of users is to discover their clustering structure. Clustering methods have been proved to be important in supervised/unsupervised recommendations [Linden *et al.*, 2003; Woo *et al.*, 2014]. The works [Gentile *et al.*, 2014; Li *et al.*, 2016] start to bring these useful techniques to bandit algorithms. These online clustering of bandit algorithms adaptively learn the clustering structure over users based on the collaborative recommendation results to gather information of users' similarity and still keep certain diversity across users.

There are two main challenges in the existing works on online clustering of bandits [Gentile *et al.*, 2014; Li *et al.*, 2016; Korda *et al.*, 2016; Gentile *et al.*, 2017; Li, 2016]. (a) The existing works assume users come to be served in a uniform manner. However, as stated by long tail effect and $80/20$ rule, the imbalance between users always exist especially with a large number of low-frequency users. The low-frequency users deteriorate the performance of existing algorithms. (b) The existing works use graphs to represent users, use edges to denote the similarity between users and use connected components to represent clusters. As a result, the learner may split two dissimilar users into two clusters only when it cuts every path between these two users. Even when the edge (representing the similarity) between two users are cut, the two users might still need to stay in the same cluster for a long time. Therefore these algorithms are not efficient in identifying the underlying clusters.

To overcome the first problem, we include frequency properties into the criterion of an underlying clustering structure. To overcome the second problem, we split a user out of her current cluster once we find the current cluster contains other users dissimilar to this one. Since the split operation might be a bit radical and premature, we add a new merge operation to reconcile the radical effect of the split operation. Based on these two building operations, complex graph structure can be discarded and replaced by simple set structures where each cluster is represented by a set of users.

This paper makes four major contributions:

1. We generalize the setting of online clustering of bandits to allow non-uniform frequency distributions over users.

2. We design both split and merge operations on clusters together with set representations of clusters to accelerate the process of identifying underlying clusters.

3. We analyze the regret of our new algorithm to get a regret bound free of the term $1/p_{\min}$, where $p_{\min}$ is the minimal frequency probability among all users and the existing algorithms could not avoid it.
4. We compare our algorithm to the existing algorithms on both synthetic and real datasets. A series of experiments consistently show the advantage of our algorithm over the existing ones.

1.1 Related work

The most related works are a series of studies on online clustering of bandits. Gentile *et al.* [2014] first establish the setting of online clustering of bandits and first show the effectiveness of using graph structure to represent clustering over users. A follow-up [Li *et al.*, 2016] considers the collaborative filtering effects on both users and items but only under the setting of static item pool. They maintain a clustering over items and for each item cluster there is a clustering over all users. Their analysis is only built on a special case that the items form an orthonormal basis in the feature space. Based on [Gentile *et al.*, 2014], Korda *et al.* [2016] design a distributed version of confidence ball algorithms in peer to peer networks with limited communications where the peers in the same cluster solve the same bandit problem. Gentile *et al.* [2017] present a context-aware clustering bandit algorithm where the clustering structure depends on items, though with a high computational cost. All these works assume the user frequency distribution is uniform and use connected components of graphs to represent clusters.

Besides these works with new algorithms and theoretical guarantees, there are many progress of applying these online algorithms in real applications. Nguyen and Lauw [2014] design a variant of clustering of bandit algorithms by performing k-means clustering method [MacQueen, 1967] on the estimates of user weight vectors with a known number of clusters. Christakopoulou and Banerjee [2018] present a variant of clustering of bandit algorithms based on Thompson sampling and allow recommending a list of items to users. Kwon [2018] allows the underlying clustering has overlapping. All these works focus on experimental performance and do not include theoretical guarantees, especially on regret bounds. Li and Zhang [2018] study an application of using clustering of bandit algorithms to improve the performance of recommending a list of items under a specific click model, and they provide a regret guarantee.

There is a group of works on stochastic low-rank bandits [Katariya *et al.*, 2017b; Katariya *et al.*, 2017a; Kveton *et al.*, 2017]. They assume the reward matrix of users and items are of low-rank and the learner selects a user-item pair each time. The goal of the learner is to find the user-item pair that has the maximum reward. This is different from the setting of online clustering of bandits where there is no control over user appearances and the goal of the learner is to maximize satisfaction of each appearing user.

The works on linear bandits are cornerstones of clustering of bandits, where the latter has different weight vectors and adaptively clusters users. It is first studied by [Abe and Long, 1999] and refined by [Dani *et al.*, 2008; Abbasi-Yadkori *et al.*, 2011] and many others. Turgay *et al.* [2018] present a similar work to make use of collaborative results but they assume the similarity information is known.

2 Model and Problem Definitions

In this section, we formulate the setting of "online clustering of bandits". There are n_u users, denoted by the set $[n_u] = \{1, \ldots, n_u\}$, with a fixed but *unknown* distribution $p = (p_1, \ldots, p_{n_u}) \in \Delta_{n_u}$ over users. Here $\Delta_n = \{x \in [0,1]^n : \sum_{i=1}^n x_i = 1\}$ denotes the simplex in $\mathbb{R}^n$. There is also a fixed but *unknown* distribution ρ over $\{x \in \mathbb{R}^d : \|x\| \leq 1\}$, which is the set of feature vectors for all items, and we simply call such a feature vector an item. At each time t, a user $\boldsymbol{i}_t \in [n_u]$ is randomly drawn from the distribution p, and L items are drawn independently from distribution ρ to form a feasible item set $\boldsymbol{D}_t$. The learning agent receives the user index $\boldsymbol{i}_t$ and the feasible item set $\boldsymbol{D}_t$, selects an item $\boldsymbol{x}_t \in \boldsymbol{D}_t$ and recommends it to the user. After the user checks the item $\boldsymbol{x}_t$, the learning agent receives a reward $\boldsymbol{y}_t$. The setting follows the previous work [Gentile *et al.*, 2014] but allows non-uniform distribution over users.

Let $\boldsymbol{\mathcal{H}}_t = \{\boldsymbol{i}_1, \boldsymbol{x}_1, \boldsymbol{y}_1, \ldots, \boldsymbol{i}_{t-1}, \boldsymbol{x}_{t-1}, \boldsymbol{y}_{t-1}, \boldsymbol{i}_t\}$ be all the information after receiving the user index in time t. Then the action $\boldsymbol{x}_t$ is $\boldsymbol{\mathcal{H}}_t$-adaptive. Henceforth, we will write $\mathbb{E}_t[\cdot]$ for $\mathbb{E}[\cdot \mid \boldsymbol{\mathcal{H}}_t]$ for the sake of notational convenience, use the boldface symbols to denote random variables, and denote $[n]$ to be the set $\{1, \ldots, n\}$.

For any time t, given a fixed user i at time t and a fixed item $x \in \boldsymbol{D}_t$ selected for user i, we define the random reward of item x for user i to be $\boldsymbol{y}_t(i,x) = \theta_i^\top x + \boldsymbol{\varepsilon}_{i,t,x}$, where (a) θ_i is a fixed but *unknown* weight vector in $\mathbb{R}^{d\times 1}$ with $\|\theta_i\| \leq 1$, independently of other items and other user behaviors; and (b) $\boldsymbol{\varepsilon}_{i,t,x}$ is an $\boldsymbol{\mathcal{H}}_t$-conditional random noise with zero mean and R-sub-Gaussian tail, i.e. $\mathbb{E}[\exp(\nu\boldsymbol{\varepsilon}_{i,t,x})] \leq \exp(R^2\nu^2/2)$ for every $\nu \in \mathbb{R}$. Then the mean reward is $\mathbb{E}[\boldsymbol{y}_t(i,x)] = \theta_i^\top x$. At time t, the learning agent recommends $\boldsymbol{x}_t$ to user $\boldsymbol{i}_t$ and receives reward $\boldsymbol{y}_t(\boldsymbol{i}_t, \boldsymbol{x}_t)$, which satisfies

$$\mathbb{E}_t[\boldsymbol{y}_t(\boldsymbol{i}_t, \boldsymbol{x}_t) \mid \boldsymbol{i}_t, \boldsymbol{x}_t] = \mathbb{E}[\boldsymbol{y}_t(\boldsymbol{i}_t, \boldsymbol{x}_t) \mid \boldsymbol{\mathcal{H}}_t, \boldsymbol{i}_t, \boldsymbol{x}_t] = \theta_{\boldsymbol{i}_t}^\top \boldsymbol{x}_t \,.$$

Suppose there are m (unknown) different weight vectors in the set $\{\theta_i : i \in [n_u]\}$. An underlying clustering exists over the users according to the weight vectors, where the users with the same weight vector form a cluster. The goal is to learn the underlying clustering to help the recommendation process. We make the following assumptions on the weight vectors $\{\theta_i : i \in [n_u]\}$, the distribution p over users, and the distribution ρ on items.

Gap between weight vectors. For any two different weight vectors $\theta_{i_1} \neq \theta_{i_2}$, there is a (fixed but unknown) gap γ between them: $\|\theta_{i_1} - \theta_{i_2}\| \geq \gamma > 0$.

Item regularity. For item distribution ρ, $\mathbb{E}_{\boldsymbol{x}\sim\rho}[\boldsymbol{x}\boldsymbol{x}^\top]$ is full rank with minimal eigenvalue $\lambda_x > 0$. Also at all time t, for any fixed unit vector $\theta \in \mathbb{R}^d$, $(\theta^\top \boldsymbol{x})^2$ has sub-Gaussian tail with variance parameter $\sigma^2 \leq \lambda_x^2/(8\log(4L))$. We assume a lower bound for λ_x is known.

Gap between user frequencies. Users with same weight vectors will have same frequencies: $\theta_{i_1} = \theta_{i_2}$ implies that

Algorithm 1 SCLUB

```
1: Input: exploration parameter $\alpha_\theta, \alpha_p > 0$, and $\beta > 0$;
2: Initialize information for each user $i \in [n_u]$ by $S_i = \boldsymbol{I}_{d\times d}, b_i = \boldsymbol{0}_{d\times 1}, T_i = 0$;
3: Initialize the set of cluster indexes by $J = \{1\}$ and initialize the single cluster by $S^1 = \boldsymbol{I}_{d\times d}, b^1 = \boldsymbol{0}_{d\times 1}, T^1 = 0, C^1 = [n_u], j(i) = 1, \forall i$;
4: for $s = 1, 2, \dots$ do
5:     Mark every user unchecked for each cluster;
6:     For each cluster $j$, compute $\tilde{\boldsymbol{T}}^j = \boldsymbol{T}^j$, and $\tilde{\boldsymbol{\theta}}^j = (\boldsymbol{S}^j)^{-1}\boldsymbol{b}^j$;
7:     for $t = 1, \dots, 2^s$ do
8:         Compute the total time step $\tau = 2^s - 2 + t$;
9:         Receive a user $\boldsymbol{i}_\tau$ and an item set $\boldsymbol{D}_\tau \subset \mathbb{R}^d$;
10:        Get the cluster index $\boldsymbol{j} = \boldsymbol{j}(\boldsymbol{i}_\tau)$ and its associated information $(\boldsymbol{S^j}, \boldsymbol{b^j}, \boldsymbol{T^j})$;
11:        Recommend item $\boldsymbol{x}_\tau = \mathrm{argmax}_{x\in \boldsymbol{D}_\tau} \boldsymbol{U}(x)$ where
               $$\boldsymbol{U}(x) = (\boldsymbol{b^j})^\top (\boldsymbol{S^j})^{-1} x + \beta \left\|x\right\|_{(\boldsymbol{S^j})^{-1}}$$
           to user $\boldsymbol{i}_\tau$ and receive feedback $\boldsymbol{y}_\tau$
12:        Run Update
13:        Run Split
14:        Mark user $\boldsymbol{i}_\tau$ has been checked;
15:        Run Merge
16:    end for $t$
17: end for $s$
```

Algorithm 2 Update

```
Update the information for user $\boldsymbol{i}_\tau$ and cluster $\boldsymbol{j}$
    $\boldsymbol{S}_{\boldsymbol{i}_\tau} = \boldsymbol{S}_{\boldsymbol{i}_\tau} + \boldsymbol{x}_\tau \boldsymbol{x}_\tau^\top, \quad \boldsymbol{b}_{\boldsymbol{i}_\tau} = \boldsymbol{b}_{\boldsymbol{i}_\tau} + \boldsymbol{y}_\tau \boldsymbol{x}_\tau,$
    $\boldsymbol{T}_{\boldsymbol{i}_\tau} = \boldsymbol{T}_{\boldsymbol{i}_\tau} + 1, \quad \hat{\boldsymbol{p}}_{\boldsymbol{i}_\tau} = \boldsymbol{T}_{\boldsymbol{i}_\tau}/\tau,$
    $\hat{\boldsymbol{\theta}}_{\boldsymbol{i}_\tau} = \boldsymbol{S}_{\boldsymbol{i}_\tau}^{-1} \boldsymbol{b}_{\boldsymbol{i}_\tau},$
    $\boldsymbol{S^j} = \boldsymbol{S^j} + \boldsymbol{x}_\tau \boldsymbol{x}_\tau^\top, \quad \boldsymbol{b^j} = \boldsymbol{b^j} + \boldsymbol{y}_\tau \boldsymbol{x}_\tau,$
    $\boldsymbol{T^j} = \boldsymbol{T^j} + 1, \quad \hat{\boldsymbol{p}}_{i'} = \boldsymbol{T}_{i'}/\tau, \ \forall i' \in \boldsymbol{C^j},$
    $\hat{\boldsymbol{\theta}}^{\boldsymbol{j}} = (\boldsymbol{S^j})^{-1} \boldsymbol{b^j}\,.$
```

Algorithm 3 Split

```
$F(T) = \sqrt{\frac{1+\ln(1+T)}{1+T}}$;
if $\left\|\hat{\boldsymbol{\theta}}_{\boldsymbol{i}_\tau} - \tilde{\boldsymbol{\theta}}^{\boldsymbol{j}}\right\| > \alpha_\theta \left(F(\boldsymbol{T}_{\boldsymbol{i}_\tau}) + F(\tilde{\boldsymbol{T}}^{\boldsymbol{j}})\right)$ or there exists user $i' \in \boldsymbol{C^j}$ with $\left|\hat{\boldsymbol{p}}_{\boldsymbol{i}_\tau} - \hat{\boldsymbol{p}}_{i'}\right| > 2\alpha_p F(\tau)$ then
    // Split user $\boldsymbol{i}_\tau$ from cluster $\boldsymbol{j}$ and form a new cluster $j' (= \max J + 1)$ of user $\boldsymbol{i}_\tau$
    $\boldsymbol{S^j} = \boldsymbol{S^j} - \boldsymbol{S}_{\boldsymbol{i}_\tau} + \boldsymbol{I}, \quad \boldsymbol{b^j} = \boldsymbol{b^j} - \boldsymbol{b}_{\boldsymbol{i}_\tau},$
    $\boldsymbol{T^j} = \boldsymbol{T^j} - \boldsymbol{T}_{\boldsymbol{i}_\tau}, \quad \boldsymbol{C^j} = \boldsymbol{C^j} - \{\boldsymbol{i}_\tau\};$
    $\boldsymbol{S}^{\boldsymbol{j}'} = \boldsymbol{S}_{\boldsymbol{i}_\tau}, \quad \boldsymbol{b}^{\boldsymbol{j}'} = \boldsymbol{b}_{\boldsymbol{i}_\tau}, \quad \boldsymbol{T}^{\boldsymbol{j}'} = \boldsymbol{T}_{\boldsymbol{i}_\tau}, \quad \boldsymbol{C}^{\boldsymbol{j}'} = \{\boldsymbol{i}_\tau\}$
end if
```

$p_{i_1} = p_{i_2}$. Also for any two different user frequency probabilities $p_{i_1} \neq p_{i_2}$, there is a (fixed but unknown) gap γ_p between them: $|p_{i_1} - p_{i_2}| \geq \gamma_p > 0$.

All the assumptions except the last one follow the previous work [Gentile *et al.*, 2014]. The first part of last assumption could be relaxed. Discussions are provided in a later section.

The optimal item for user $\boldsymbol{i}_t$ in round t is $x^*_{\boldsymbol{i}_t, \boldsymbol{D}_t} = \mathrm{argmax}_{x \in \boldsymbol{D}_t} \theta_{\boldsymbol{i}_t}^\top x$. Then the expected regret for user $\boldsymbol{i}_t$ in time t is $\boldsymbol{R}_t = \theta_{\boldsymbol{i}_t}^\top x^*_{\boldsymbol{i}_t, \boldsymbol{D}_t} - \theta_{\boldsymbol{i}_t}^\top \boldsymbol{x}_t$. The goal of the learning agent is to minimize the expected cumulative regret

$$R(T) = \mathbb{E}\left[\sum_{t=1}^T \boldsymbol{R}_t\right] = \mathbb{E}\left[\sum_{t=1}^T \left(\theta_{\boldsymbol{i}_t}^\top x^*_{\boldsymbol{i}_t, \boldsymbol{D}_t} - \theta_{\boldsymbol{i}_t}^\top \boldsymbol{x}_t\right)\right],$$

where the expectation is taken over the randomness of users $\boldsymbol{i}_1, \dots, \boldsymbol{i}_T$, the randomness of the item sets $\boldsymbol{D}_1, \dots, \boldsymbol{D}_T$ and the possible randomness in selected items $\boldsymbol{x}_1, \dots, \boldsymbol{x}_T$.

3 Algorithm

In this section, we introduce our algorithm of "set-based clustering of bandits (SCLUB)" to deal with the online clustering of bandit problem. Recall that the related previous work [Gentile *et al.*, 2014; Li *et al.*, 2016] adopted connected components in graphs to represent clusters and the learning process only split clusters. In contrast, our algorithm uses sets to represent clusters and allow both split and merge operations on sets in the learning process. The pseudocode is provided in Algorithm 1.

In the algorithm, we store a profile of $(\boldsymbol{S}_i, \boldsymbol{b}_i, \boldsymbol{T}_i)$ for each user i, where $\boldsymbol{T}_i$ is the number of times that the user i has appeared, $\boldsymbol{S}_i$ is the Gramian matrix and $\boldsymbol{b}_i$ is the moment vector of regressand by regressors. For example, if $(x_1, y_1), \dots, (x_n, y_n)$ are pairs of items and corresponding rewards collected for user i before time t, then

$$T_i = n\,, \quad S_i = \sum_{k=1}^n x_k x_k^\top\,, \quad b_i = \sum_{k=1}^n x_k y_k\,.$$

The user profiles are initialized in the beginning of the algorithm (line 2), and the clustering is initialized to be a single set with index 1 containing all users (line 3). We store a profile of $(\boldsymbol{S}^j, \boldsymbol{b}^j, \boldsymbol{T}^j, \boldsymbol{C}^j)$ for each cluster j, where $\boldsymbol{C}^j$ is the set of all user indexes in this cluster and

$$\boldsymbol{S}^j = \boldsymbol{I} + \sum_{i\in \boldsymbol{C}^j} (\boldsymbol{S}_i - \boldsymbol{I}), \quad \boldsymbol{b}^j = \sum_{i \in \boldsymbol{C}^j} \boldsymbol{b}_i, \quad \boldsymbol{T}^j = \sum_{i\in\boldsymbol{C}^j} \boldsymbol{T}_i$$

are the aggregate information in the cluster j. As a convention, we use subscript $\boldsymbol{S}_i, \boldsymbol{b}_i, \boldsymbol{T}_i$ to denote the information related to a user i and use superscript $\boldsymbol{S}^j, \boldsymbol{b}^j, \boldsymbol{T}^j$ to denote the information related to a cluster j.

The learning agent proceeds in phases (line 4), where the s-th phase contains 2^s rounds (line 7). Each phase has an accuracy level such that the learner could put accurate clusters aside and focus on exploring inaccurate clusters. In the beginning of each phase, mark every user to be unchecked (line 5), which is used to identify good clusters in the current accuracy level. If all users in a cluster are checked, then

Algorithm 4 Merge

```
for any two checked clusters $j_1 < j_2$ satisfying
    $\left\|\hat{\boldsymbol{\theta}}^{j_1} - \hat{\boldsymbol{\theta}}^{j_2}\right\| < \frac{\alpha_\theta}{2}\left(F(\boldsymbol{T}^{j_1}) + F(\boldsymbol{T}^{j_2})\right)$
  and
    $\left|\hat{\boldsymbol{p}}^{j_1} - \hat{\boldsymbol{p}}^{j_2}\right| < \alpha_p F(\tau)$
 do
    // Merge them: add information of cluster $j_2$ to cluster $j_1$ and remove cluster $j_2$
        $\boldsymbol{S}^{j_1} = \boldsymbol{S}^{j_1} + \boldsymbol{S}^{j_2} - \boldsymbol{I}, \quad \boldsymbol{b}^{j_1} = \boldsymbol{b}^{j_1} + \boldsymbol{b}^{j_2}$,
        $\boldsymbol{T}^{j_1} = \boldsymbol{T}^{j_1} + \boldsymbol{T}^{j_2}, \quad \boldsymbol{C}^{j_1} = \boldsymbol{C}^{j_1} \bigcup \boldsymbol{C}^{j_2}$;
        $j(i) = j_1 \ \forall i \in \boldsymbol{C}^{j_2}$ and delete cluster $j_2$
end for
```

this cluster is *checked* to be an accurate cluster in the current phase. Compute the estimated weight vectors for each cluster (line 6), which serve as the pivots to be compared for splitting its containing users. Note that the pivot weight vectors are denoted as $\tilde{\boldsymbol{\theta}}^j$ instead of $\hat{\boldsymbol{\theta}}^j$.

At each time t in the s-th phase, a user $\boldsymbol{i}_\tau$ comes to be served with candidate item set $\boldsymbol{D}_\tau$ (line 9), where τ denotes the index of the total time step (line 8). First obtain the information $(\boldsymbol{S^j}, \boldsymbol{b^j}, \boldsymbol{T^j})$ for the cluster $\boldsymbol{j}$ of user $\boldsymbol{i}_\tau$ (line 10) and then recommend based on these information (line 11). Here $\|x\|_A = \sqrt{x^\top A x}$ for any positive definite matrix A. After receiving feedback $\boldsymbol{y}_\tau$, the learner will *update* information (Algorithm 2) and check if a split is possible (Algorithm 3).

By the assumptions in Section 2, users in one cluster have same weight vector and same frequency. Thus if a cluster is a good cluster, which only contains users of same underlying cluster, the estimated parameters for containing users will be close to estimated parameters of the cluster. We call a user i is *consistent* with the current cluster j if $\hat{\theta}_i$ is close to $\hat{\theta}^j$ and $\hat{p}_i$ is close to $\hat{p}_{i'}$ for any other user i' of cluster j. If a user is inconsistent with the current cluster, the learner will split her out (Algorithm 3).

We call two checked clusters *consistent* if their estimated weight vectors and estimated frequencies are close enough. The algorithm will merge two checked clusters if they are consistent (Algorithm 4). Here $\hat{p}^j = T^j/(\left|C^j\right|\tau)$ is the average frequency of cluster j. The condition of clusters to be *checked* is to avoid merging two clusters that are not accurate enough in the current level. The operations of split and merge, together with the conditions, can guarantee that the clusters are good at the end of each phase in the corresponding accuracy level with high probability (the analysis is part of the regret bound proof).

4 Results

Recall that d is the feature dimension, m is the (unknown) number of clusters and n_u is the number of users. Let n^j be the (unknown) number of users in cluster j and p^j be the (unknown) frequency for cluster j, which is the sum of (unknown) user frequencies in cluster j. The following main theorem bounds the cumulative regret of our algorithm SCLUB.

Theorem 1 *Suppose the clustering structure over the users, user frequencies, and items satisfy the assumptions stated in Section 2 with gap parameters $\gamma, \gamma_p > 0$ and item regularity parameter $0 < \lambda_x \le 1$. Let $\alpha_\theta = 4R\sqrt{d/\lambda_x}$, $\alpha_p = 2$ and $\beta = R\sqrt{d\ln(1+T/d) + 2\ln(4mn_u)}$. Then the cumulative regret of the algorithm* SCLUB *after T rounds satisfies*

$$R(T) \le \sum_{j=1}^{m} 4\beta\sqrt{dp^j T \ln(T/d)} + O\left(\left(\frac{1}{\gamma_p^2} + \frac{n_u}{\gamma^2\lambda_x^3}\right)\ln(T)\right) \quad (1)$$

$$= O(d\sqrt{mT}\ln(T)) \,. \quad (2)$$

Proof. [sketch] The proof is mainly based on two parts. The first part bounds the exploration rounds to guarantee the clusters partitioned correctly. The second part is to estimate regret bounds for linear bandits after the clusters are partitioned correctly.

By Chernoff-Hoeffding inequality, we prove when $\tau \ge O\left(\frac{1}{\gamma_p^2}\ln(T) + \frac{1}{p_i}\ln(T)\right)$ the distance of p_i and $\hat{p}_i$ is less than $\gamma_p/2$. Then the users with different frequencies will be assigned to different clusters. Thus, the cluster containing user i only contains users with the same frequency probabilities, for all user i.

Then by the assumption of item regularity, when $\tau \ge O\left(\frac{1}{\gamma_p^2}\ln(T) + \frac{1}{p_i}\ln(T) + \frac{1}{p_i\gamma^2\lambda_x^3}\ln(T)\right)$ the estimated weight vectors will be accurate enough for all the users with the frequency probability same as p_i. The 2-norm radius for the confidence ellipsoid of the weight vector will be less than $\gamma/2$. Thus the split and merge operations will function correctly for users. Next, we prove that if two clusters containing the users of same frequency probabilities have accurate estimates for weight vectors and frequencies, the combined cluster inherits the accuracies with high probability. Thus, the combination steps can continue until it contains all users of an underlying cluster.

After the cluster j with frequency probability p^j is correctly partitioned, the recommendation is based on the estimates of cluster weight vector. The regret for the second part reduces to linear bandits with $p^j T$ rounds. □

The proof is much different from that of CLUB [Gentile *et al.*, 2014] with the additional frequency probability and merge operation. The full proof is put in the supplementary materials.

4.1 Discussions

First, we compare our regret bound with that of CLUB. Since CLUB is designed and proved only for uniform distribution over users, we first generalize the regret bound of CLUB to the setting of arbitrary distribution over users.

Theorem 2 *Under the same setting with* SCLUB*, the cumulative regret of the algorithm,* CLUB*, after T rounds satisfies*

$$R(T) \le \sum_{j=1}^{m} 4\beta\sqrt{dp^jT\ln(T/d)} + O\left(\frac{1}{p_{\min}\gamma^2\lambda_x^3}\ln(T)\right) = O(d\sqrt{mT}\ln(T))\,.$$

Note that in the degenerate setting where the distribution p over users is uniform, $p_{\min} = 1/n_u$ and the regret bound recovers the one in the previous work [Gentile *et al.*, 2014].

In real applications, there might be a lot of infrequent users with frequency close to 0. Since the algorithm CLUB initializes with a complete graph or a random connected graph, there would be a lot of infrequent users connecting with frequent users. Infrequent users receive a recommendation once a long time. Thus it takes a long time to differentiate them from frequent users, or to be split into a different cluster (connected component) from frequent users. Then the recommendations for frequent users would be polluted by these infrequent and dissimilar users for a long time. The length of rounds to separate infrequent dissimilar users is proportional to $1/p_{\min}$, the minimal frequency over all users. $p_{\min}$ could be arbitrarily small in real applications, the log term in regret bound for CLUB is not satisfactory. Due to an additional care on frequencies, SCLUB could get rid of the uncontrollable term $1/p_{\min}$.

The assumption that users with same weight vectors have same frequencies is based on the intuition that frequent users and infrequent users usually have different preferences. It can be relaxed to that users with same weight vectors have close frequencies. For more general cases, a new structure of nested clusters could help where the infrequent users could use the information of frequent users but not vice versa. We leave this as interesting future work.

Second, our result can be generalized by allowing multiple users each round. In applications the recommender system will update itself every a fixed time slot, and during each time slot, multiple users will be served using the same history information.

Theorem 3 *Under the same setting with* SCLUB*, each user appear with frequency $p_i \in [0,1]$, then the cumulative regret of the algorithm after T rounds satisfies*

$$R(T) \le \sum_{j=1}^{m} 4\beta\sqrt{dp^jT\ln(T/d)} + O\left(\left(\frac{\bar{p}}{\gamma_p^2} + \frac{n_u}{\gamma^2\lambda_x^3}\right)\ln(T)\right)$$

where $\bar{p} = \sum_{i=1}^{n_u} p_i$ is the sum of all user frequencies.

Note this recovers the regret bound in Theorem 1 when $\bar{p} = 1$. The regret bound for CLUB can also be generalized in a similar way.

Third, if the clustering structure is known, the setting would be equivalent to m independent linear bandits, each with the expected number of rounds p^jT. Then the regret bound would be $O(d\sum_{j=1}^{m}\sqrt{p^jT}\ln(T))$ by [Abbasi-Yadkori *et al.*, 2011] which matches the main term of ours (1). The upper bound reaches its maximum $O(d\sqrt{mT}\ln(T))$ when $p^1 = \cdots = p^m = 1/m$. Also the regret lower bound in this case is $\Omega(\sum_{j=1}^{m}\sqrt{dp^jT})$ by [Dani *et al.*, 2008] which matches the main term of (1) up to a term of $\sqrt{d}\ln(T)$.

Fourth, we would like to state the additional benefit of SCLUB in privacy protection. To protect user privacy, when a user i comes to be served, the learning agent will have the only access to the information of user i and some aggregate information among users, but does not have the access to other individual users' information. In this sense, lack of other individual users' information will make existing graph-based methods [Gentile *et al.*, 2014; Li *et al.*, 2016] inapplicable. Our algorithm SCLUB only uses information of user i and aggregate information of clusters to split and merge.

Due to space limit, more discussions are put in the supplementary materials.

5 Experiments

SCLUB algorithm is compared with CLUB [Gentile *et al.*, 2014], LinUCB-One which uses a single estimated weight vector for all users and LinUCB-Ind which uses a separate estimated weight vector for each user on both synthetic and real datasets.

5.1 Synthetic experiments

We consider a setting of $n_u = 10^3$ users with $m = 10$ clusters where each cluster contains equal number of users. The weight vectors $\theta_1, \ldots, \theta_m$ and item vectors at each round are first randomly drawn in $d-1$ ($d = 20$) dimension with each entry a standard Gaussian variable, then normalized, added one more dimension with constant 1, and divided by $\sqrt{2}$. The transformation is as follows

$$x \mapsto \left(\frac{x}{\sqrt{2}\,\|x\|}, \frac{1}{\sqrt{2}}\right). \tag{3}$$

This transformation on both the item vector x and weight vector θ is to guarantee the mean $\langle\theta, x\rangle$ lies in $[0,1]$. The parameters in all algorithms take theoretical values. The comparisons on theoretical computational complexities of these algorithms is put in supplementary materials. The evolution of the regret as a function of time is shown in the first row of Figure 1 [1] . The regrets at the end and total running times are given in supplementary materials. The left figure (a) is under the setting of uniform distribution over all users where each user has a frequency $1/n_u$; middle figure (b) is under the setting of arbitrary distribution over clusters where the users in the same cluster have the same frequency probability; right figure (c) is under the setting of arbitrary distribution over users.

The performance of SCLUB improves over CLUB by 5.94% (Figure 1(1a)) even in the setting of CLUB where the frequency distribution over users is uniform. The reason is that CLUB uses connected components of graphs to represent clusters and if it wants to identify the underlying cluster of

[1]The parameters of CLUB are theoretical values and might be suboptimal. We do not optimize the parameters of any algorithm and just use theoretical values of the parameters in all algorithms including SCLUB.

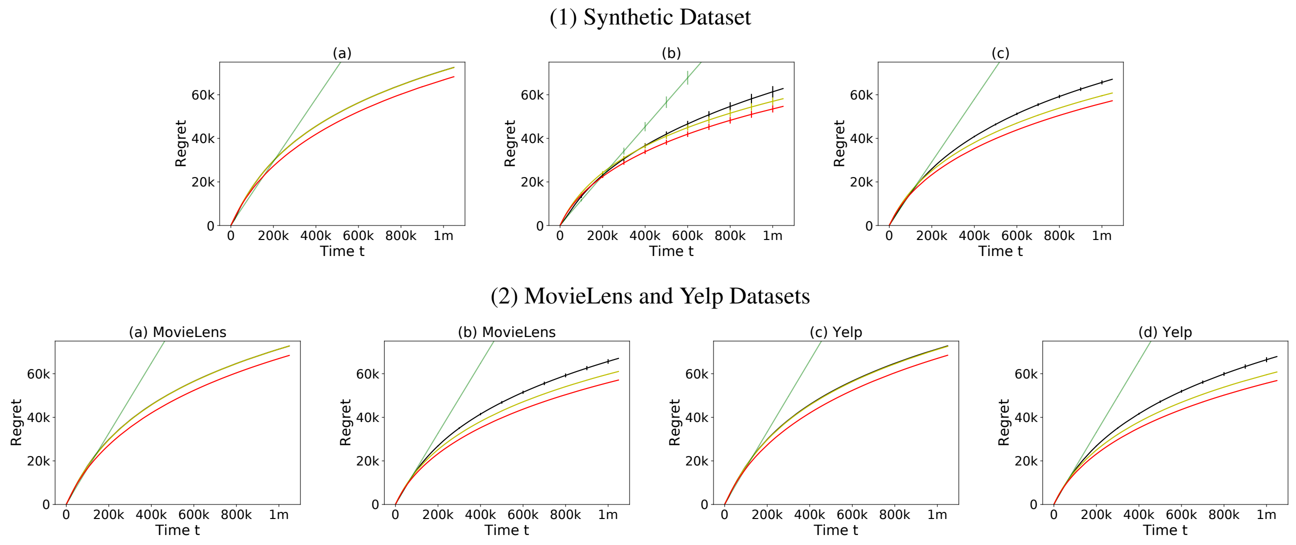

Figure 1: The figures compare `SCLUB` (red) with `CLUB` (black), `LinUCB-One` (green) and `LinUCB-Ind` (yellow). The first row is for synthetic experiments and the second row is for real datasets, MovieLens and Yelp. All the experiments are of $n_u = 10^3$ users with $d = 20, L = 20$. We set $m = 10$ for (1) synthetic experiments. (1a)(2a)(2c) are of uniform distribution over users, (1b) is of arbitrary distribution over clusters, where the users in the same cluster have the same frequency probabilities, (1c)(2b)(2d) are of arbitrary distribution over users. All results are averaged under 10 random runs and the errorbars are computed by standard errors, which are standard deviations divided by $\sqrt{10}$.

a user i, it has to delete every bad edge of its neighbors (and neighbors of neighbors, etc.). In comparison, `SCLUB` is much faster, will split a user i out if it finds any inconsistency from her current running cluster, and merge her to any consistent running cluster. The improvement is enlarged to be 13.02% (Figure 1(1b)) in the setting of Section 2 where users in the same cluster have the same frequency probability but users in different clusters might have different frequencies. `SCLUB` is robust in the setting of arbitrary distribution over users where the assumptions in Section 2 might fail and can still improve over `CLUB` by 14.69% (Figure 1(1c)).

5.2 Real datasets

We use the $20m$ MovieLens dataset [Harper and Konstan, 2016] which contains 20 million ratings for 2.7×10^4 movies by 1.38×10^5 users and Yelp dataset[2] which contains 4.7 million ratings of 1.57×10^5 restaurants from 1.18 million users. The Yelp dataset is more sparse than MovieLens. For each of the two real datasets, we extract 10^3 items with most ratings and $n_u = 10^3$ users who rate most. Then use the rating matrix to derive feature vectors of $d - 1$ ($d = 20$) dimension for all users by singular-value decomposition (SVD). The feature vectors are also processed as (3) and the resulting vectors are regarded as the underlying weight vectors of each user.

The performances are shown in Figure 1(2) where (2a)(2b) are for MovieLens dataset and (2c)(2d) are for Yelp dataset. (2a)(2c) are under the setting of uniform distribution over users and (2b)(2d) are under the setting of arbitrary distribution over users. Since there is no underlying clustering over the users, we do not experiment on the setting of arbitrary distribution over clusters like Figure 1(1b). The algorithms of `SCLUB` and `CLUB` adaptively find finer and finer clustering as more data flows in. The performance of `SCLUB` improves over `CLUB` by $5.94\%, 14.84\%, 5.94\%, 16.24\%$ respectively.

[2]http://www.yelp.com/dataset_challenge

6 Conclusions and Future Work

In this paper, we extend the existing setting for online clustering of bandits to include non-uniform distribution over users. The new set-based algorithm together with both split and merge operations clusters users adaptively and is proven with a better theoretical guarantee on cumulative regrets than the existing works. Our algorithm also has an additional benefit of privacy protection. The experiments on both synthetic and real datasets show that our algorithm is consistently better then the existing works.

One interesting future direction is to explore the asymmetric relationships between users, whereas all existing works use symmetric relationships including ours. For example, recommendations of low-frequency users can use information (or feedback) from high-frequency users, but not vice versa. This idea can be extended by using nested clusters where an underlying cluster of frequent users can be a subset of another underlying cluster containing many infrequent users. Another interesting problem is to generalize the idea of our method to collaborative filtering of both users and items such as Li *et al.* [2016]. Also extending [Li *et al.*, 2016] to the general setting of changing item set would be a challenge one.

Acknowledgements

This work was partially supported by The Chinese University of Hong Kong Direct Grants [ID: 4055073 and 4055104]. Wei Chen is partially supported by the National Natural Science Foundation of China (Grant No. 61433014).

References

[Abbasi-Yadkori *et al.*, 2011] Yasin Abbasi-Yadkori, Dávid Pál, and Csaba Szepesvári. Improved algorithms for linear stochastic bandits. In *Advances in Neural Information Processing Systems*, pages 2312–2320, 2011.

[Abe and Long, 1999] Naoki Abe and Philip M Long. Associative reinforcement learning using linear probabilistic concepts. In *Proceedings of the Sixteenth International Conference on Machine Learning*, pages 3–11. Morgan Kaufmann Publishers Inc., 1999.

[Aggarwal, 2016] Charu Aggarwal. *Recommender Systems*. Springer, 2016.

[Bubeck *et al.*, 2012] Sébastien Bubeck, Nicolo Cesa-Bianchi, et al. Regret analysis of stochastic and nonstochastic multi-armed bandit problems. *Foundations and Trends® in Machine Learning*, 5(1):1–122, 2012.

[Christakopoulou and Banerjee, 2018] Konstantina Christakopoulou and Arindam Banerjee. Learning to interact with users: A collaborative-bandit approach. In *Proceedings of the 2018 SIAM International Conference on Data Mining*, pages 612–620. SIAM, 2018.

[Chu *et al.*, 2011] Wei Chu, Lihong Li, Lev Reyzin, and Robert E Schapire. Contextual bandits with linear payoff functions. In *AISTATS*, volume 15, pages 208–214, 2011.

[Dani *et al.*, 2008] Varsha Dani, Thomas P Hayes, and Sham M Kakade. Stochastic linear optimization under bandit feedback. In *COLT*, pages 355–366, 2008.

[Gentile *et al.*, 2014] Claudio Gentile, Shuai Li, and Giovanni Zappella. Online clustering of bandits. In *Proceedings of the 31st International Conference on Machine Learning*, pages 757–765, 2014.

[Gentile *et al.*, 2017] Claudio Gentile, Shuai Li, Purushottam Kar, Alexandros Karatzoglou, Giovanni Zappella, and Evans Etrue. On context-dependent clustering of bandits. In *International Conference on Machine Learning*, pages 1253–1262, 2017.

[Harper and Konstan, 2016] F Maxwell Harper and Joseph A Konstan. The movielens datasets: History and context. *ACM Transactions on Interactive Intelligent Systems (TiiS)*, 5(4):19, 2016.

[Katariya *et al.*, 2017a] Sumeet Katariya, Branislav Kveton, Csaba Szepesvári, Claire Vernade, and Zheng Wen. Bernoulli rank-1 bandits for click feedback. In *Proceedings of the 26th International Joint Conference on Artificial Intelligence*, pages 2001–2007. AAAI Press, 2017.

[Katariya *et al.*, 2017b] Sumeet Katariya, Branislav Kveton, Csaba Szepesvari, Claire Vernade, and Zheng Wen. Stochastic rank-1 bandits. In *Artificial Intelligence and Statistics*, pages 392–401, 2017.

[Korda *et al.*, 2016] Nathan Korda, Balazs Szorenyi, and Shuai Li. Distributed clustering of linear bandits in peer to peer networks. In *The 33rd International Conference on Machine Learning (ICML)*, 2016.

[Kveton *et al.*, 2017] Branislav Kveton, Csaba Szepesvari, Anup Rao, Zheng Wen, Yasin Abbasi-Yadkori, and S Muthukrishnan. Stochastic low-rank bandits. *arXiv preprint arXiv:1712.04644*, 2017.

[Kwon, 2018] Jin Kyoung Kwon. Overlapping clustering of contextual bandits with nmf techniques. 2018.

[Lattimore and Szepesvári, 2018] Tor Lattimore and Csaba Szepesvári. *Bandit Algorithms*. preprint, 2018.

[Li and Zhang, 2018] Shuai Li and Shengyu Zhang. Online clustering of contextual cascading bandits. In *Thirty-Second AAAI Conference on Artificial Intelligence*, 2018.

[Li *et al.*, 2010] Lihong Li, Wei Chu, John Langford, and Robert E Schapire. A contextual-bandit approach to personalized news article recommendation. In *Proceedings of the 19th international conference on World wide web*, pages 661–670. ACM, 2010.

[Li *et al.*, 2016] Shuai Li, Alexandros Karatzoglou, and Claudio Gentile. Collaborative filtering bandits. In *Proceedings of the 39th International ACM SIGIR conference on Research and Development in Information Retrieval*, pages 539–548. ACM, 2016.

[Li, 2016] Shuai Li. *The art of clustering bandits.* PhD thesis, Università degli Studi dell'Insubria, 2016.

[Linden *et al.*, 2003] Greg Linden, Brent Smith, and Jeremy York. Amazon. com recommendations: Item-to-item collaborative filtering. *IEEE Internet computing*, 7(1):76–80, 2003.

[MacQueen, 1967] James MacQueen. Some methods for classification and analysis of multivariate observations. In *Proceedings of the fifth Berkeley symposium on mathematical statistics and probability*, volume 1, pages 281–297. Oakland, CA, USA, 1967.

[Nguyen and Lauw, 2014] Trong T Nguyen and Hady W Lauw. Dynamic clustering of contextual multi-armed bandits. In *Proceedings of the 23rd ACM International Conference on Conference on Information and Knowledge Management*, pages 1959–1962. ACM, 2014.

[Rusmevichientong and Tsitsiklis, 2010] Paat Rusmevichientong and John N Tsitsiklis. Linearly parameterized bandits. *Mathematics of Operations Research*, 35(2):395–411, 2010.

[Turgay *et al.*, 2018] Eralp Turgay, Doruk Oner, and Cem Tekin. Multi-objective contextual bandit problem with similarity information. In *International Conference on Artificial Intelligence and Statistics*, pages 1673–1681, 2018.

[Woo *et al.*, 2014] Choong-Wan Woo, Anjali Krishnan, and Tor D Wager. Cluster-extent based thresholding in fmri analyses: pitfalls and recommendations. *Neuroimage*, 91:412–419, 2014.