Online Clustering of Contextual Cascading Bandits

Shuai Li **Shengyu Zhang**
The Chinese University of Hong Kong, Sha Tin, Hong Kong
{shuaili, syzhang}@cse.cuhk.edu.hk

Abstract

We consider a new setting of online clustering of contextual cascading bandits, an online learning problem where the underlying cluster structure over users is unknown and needs to be learned from a random prefix feedback. More precisely, a learning agent recommends an ordered list of items to a user, who checks the list and stops at the first satisfactory item, if any. We propose an algorithm of *CLUB-cascade* for this setting and prove an n-step regret bound of order $\tilde{O}(\sqrt{n})$. Previous work (Li et al. 2016) corresponds to the degenerate case of only one cluster, and our general regret bound in this special case also significantly improves theirs. We conduct experiments on both synthetic and real data, and demonstrate the effectiveness of our algorithm and the advantage of incorporating online clustering method.

Introduction

Most recommendation systems nowadays display items in an ordered list. Examples include typical hotels/restaurants/goods recommendation, search engines, etc. This is especially the case for apps or games recommendations on mobile devices due to the limited size of screen. Click behaviors and feedback in such ordered lists have their distinctive features, and a *cascade model* was recently developed for studying feedback of user click behaviors (Craswell et al. 2008). In the model, after a user receives a list of items, she checks the items in the given order and clicks the first satisfactory one. After the click, she stops checking the rest items in the list. The learning agent receives the feedback of the click and knows that the items before the clicked one have been checked and are unsatisfactory, but whether the user likes any items *after* the clicked one is unknown. The cascade model is straightforward but effective in characterizing user behaviors (Chuklin, Markov, and Rijke 2015).

In this paper, we consider an online learning variant of *cascading bandits* (Kveton et al. 2015a; 2015b). In our model, the learning agent uses exploration-exploitation techniques to learn the preferences of users over items by interactions with users. At each time step, the learning agent recommends a list of items to the current user, observes the click feedback, and receives a reward of 1 if the user clicks on an item (and receives reward 0 otherwise). The learning agent aims to maximize its cumulative rewards after n rounds. Previous work (Li et al. 2016; Zong et al. 2016) considered a setting of *linear cascading bandits* to deal with ever-changing set of items. Roughly speaking, the learning agent adaptively learns a linear mapping between expected rewards and features of items and users.

One important limit of the linear cascading bandit algorithms is that they mainly work in a content-dependent regime, discarding the often useful method of collaborative filtering. One way to utilize the collaborative effect of users is to consider their clustering structure. In this paper, we formulate the problem of online clustering of contextual cascading bandits, and design an algorithm to learn the clustering information and extract user feature vectors adaptively with low cumulative regret. Following the approach in (Gentile, Li, and Zappella 2014), we use a dynamic graph on all users to represent clustering structure, where an edge indicates the similarity between the two users. Edges between different clusters are gradually removed as the algorithms learns from the feedback that the pairs of users are not similar. We prove an upper bound of $O(d\sqrt{mnK}\ln(n))$ for the cumulative regret, where m is the number of clusters, d is the dimension of feature space, n is the number of rounds, and K is the number of recommended items. This extends and improves the existing results in the degenerate setting of only one cluster ($m = 1$). Finally we experiment on both synthetic and real datasets to demonstrate the advantage of the model and algorithm.

The organization of this paper is as follows. We first introduce previous work related to our setting, then formulates the setting of *Online Clustering of Contextual Cascading Bandits* with some appropriate assumptions. Next we give our UCB-like algorithm CLUB-cascade and the cumulative regret bound, which is better than the existing results in the degenerate case. Then we report experimental results on both synthetic data and real data to demonstrate the advantage of incorporating online clustering. Last is the conclusion of the paper.

Related Work

(Kveton et al. 2015a; Katariya et al. 2016) introduced the click model with cascading feedback and DCM feedback to the MAB framework, which describes the random feedback

dependent on the display order of items. In the cascading feedback, a user clicks the first satisfying items and stops checking further, while in the DCM feedback, after a user clicks an item, there is a chance that she is not satisfied and continues checking. (Kveton et al. 2015b) considered the problem where the random feedback stops at the first default position (reward 0), in comparison with the first success position (reward 1) in the cascade setting. Even though the settings are similar, the techniques are totally different because of the asymmetry of the binary OR function and binary AND function. (Zoghi et al. 2017) brought up an online elimination algorithm to deal with different click models. All the above works focused on the setting of fixed item set.

(Li et al. 2016) generalized both the cascade setting and combinatorial cascade setting with contextual information, position discounts and more general reward functions. For the binary OR case, they provided a regret bound for n rounds with order $O(\frac{d}{p^*}\sqrt{nK}\ln(n))$ where p^* is probability to check all recommended items and could be small. At the same time, (Zong et al. 2016) also generalized the cascade setting with linear payoff and brought up a UCB-like algorithm, CascadeLinUCB, as well as a Thompson sampling (TS) algorithm without a proof. They proved a regret bound of n rounds for the CascadeLinUCB algorithm of order $O(dK\sqrt{n}\ln(n))$. In this paper, we consider the basic cascade setting, where the random feedback stops at the first click position, together with the online clustering to explore user structure. We provide a regret bound of order $O(d\sqrt{mnK}\ln(n))$. Cast in this framework, the existing results studied the degenerate case of $m = 1$.

The work (Gentile, Li, and Zappella 2014) first considered online clustering of linear bandits, and maintained a graph among users and used connected components to denote user clusters. A follow-up (Li, Karatzoglou, and Gentile 2016) explored item structures to help cluster users and to improve recommendation performance. (Gentile et al. 2017) considered a variant where the clusters over users are dependent on the current context. In this paper, we employ some idea of the first paper, and manage to make it to work with random feedback in click models. (Combes et al. 2015) considered a similar setting of clustered users with cascade feedback, but in their paper, the clusters are fixed and known to the learning agent. In our paper, the cluster structure is unknown and has to be learned by the learning agent.

Problem Setup

In this section, we formulate the problem of "Online Clustering of Contextual Cascading Bandits". In this problem, there are u users, denoted by set $[u] = \{1, \ldots, u\}$. At each time step t, a user i_t comes to be served with contents and the learning agent receives the user index with a finite feasible content set $D_t \subset \mathbb{R}^{d\times 1}$, where $\|x\|_2 \leq 1$ for all $x \in D_t$. Then the learning agent recommends a ranked list of *distinct* K items $X_t = (x_1, ..., x_K) \in \Pi^K(D_t)$ to the user. The user checks the items in the order from the first one to the last one, clicks the first attractive item, and stops checking after the click. We use the Bernoulli random variable $\boldsymbol{y}_{t,k}$ to indicate whether the item $x_{t,k}$ has been clicked or not. The learning agent receives the feedback of the index of the clicked item, that is

$$\boldsymbol{C}_t = \inf\{k : \boldsymbol{y}_{t,k} = 1\}. \tag{1}$$

Note that $\inf(\emptyset) = \infty$ and $\boldsymbol{C}_t = \infty$ represents that the user does not click on any given item. Let $\boldsymbol{K}_t = \min\{\boldsymbol{C}_t, K\}$. The user checks the first $\boldsymbol{K}_t$ items and the learning agent receives the feedback $\{\boldsymbol{y}_{t,k}, k = 1, \ldots, \boldsymbol{K}_t\}$.

Let $\mathcal{H}_t$ be the entire history information until the end of round t. Then the action $\boldsymbol{X}_t$ is $\mathcal{H}_{t-1}$-adaptive. We will write $\mathbb{E}_t[\cdot]$ for $\mathbb{E}[\cdot|\mathcal{H}_{t-1}]$ for convenience of notation, use the boldface symbols to denote random variables, and denote $[m] = \{1, \ldots, m\}$.

We assume the probability of clicking on an item to be a linear function of item feature vector. Specifically there exists a vector $\theta_{i_t} \in \mathbb{R}^{d\times 1}, \|\theta_{i_t}\|_2 \leq 1$ for user i_t, such that the expectation of the binary click feedback $\boldsymbol{y}$ on the checking item x is given by the inner product of x with θ_{i_t}, i.e.,

$$\mathbb{E}_t[\boldsymbol{y}|x] = \theta_{i_t}^\top x, \tag{2}$$

independently of any other given item.

We assume that there are m clusters among the users, where $m \ll u$, and the partition of the clusters is fixed but unknown. Specifically we use $I_1, \ldots, I_m$ to denote the true clusters and a mapping function $j : [u] \to [m]$ to map user i to its true cluster index $j(i)$ (user i belongs to cluster $I_{j(i)}$). We assume the order of user appearance and the set of feasible items are not under the control of the learning agent. In addition, we assume clusters, users, and items satisfy the following assumptions.

Cluster regularity All users in the same cluster I_j share the same θ, denoted as θ_j. Users in different clusters have a gap between their θ's, that is

$$\|\theta_j - \theta_{j'}\| \geq \gamma > 0$$

for any $j \neq j'$.

User uniformness At each time step t, the user is drawn uniformly from the set of all users $[u]$, independently over past.

Item regularity At each time step t, given the size of D_t, the items in D_t are drawn independently from a fixed distribution $\boldsymbol{x}$ with $\|\boldsymbol{x}\|_2 \leq 1$, and $\mathbb{E}[\boldsymbol{x}\boldsymbol{x}^\top]$ is full rank with minimal eigenvalue $\lambda_x > 0$. Also at all time t, for any fixed unit vector $\theta \in \mathbb{R}^d$, given the size of D_t, $(\theta^\top X)^2$ has sub-Gaussian tail with variance parameter $\sigma^2 \leq \lambda_x^2/(8\log(4|D_t|))$.

Note that the above assumptions follow the settings from previous work (Gentile, Li, and Zappella 2014). We will have more discussions on these in later section.

At each time step t, the reward of $X = (x_1, \ldots, x_K) \in \Pi^K(D_t)$ under the random click result $\boldsymbol{y}_t = (\boldsymbol{y}_t(x))_{x\in D}$ (if known) is

$$f(X, \boldsymbol{y}_t) = 1 - \prod_{k=1}^{K}(1 - \boldsymbol{y}_t(x_k)).$$

By independence assumption, it is easily verified that the expectation of $f(X, \boldsymbol{y}_t)$ is

$$f(X, y_t) = 1 - \prod_{k=1}^{K}(1 - y_t(x_k)),$$

where $y_t(x) = \theta_{j(i_t)}^\top x$ and $y_t = (y_t(x))_{x \in D_t}$. Let

$$X_t^* = \text{argmax}_{X \in \Pi^K(D_t)} f_t(X, y_t)$$

be the optimal action in round t. Then the regret in time step t is

$$R_t(X, \boldsymbol{y}_t) = f(X_t^*, \boldsymbol{y}_t) - f(X, \boldsymbol{y}_t).$$

The goal for the learning agent is to minimize the expected cumulative regret

$$R(n) = \mathbb{E}\Big[\sum_{t=1}^{n} R_t(\boldsymbol{X}_t, \boldsymbol{y}_t)\Big]. \tag{3}$$

Algorithm and Results

Notations

Our main algorithm is given in **Algorithm 1**. Before diving into details, let us define some useful notations used in later analysis. For any time step t and user i, define

$$\boldsymbol{S}_{i,t} = \sum_{\substack{s \le t \\ i_s = i}} \sum_{k=1}^{\boldsymbol{K}_s} \boldsymbol{x}_{s,k} \boldsymbol{x}_{s,k}^\top, \quad \boldsymbol{b}_{i,t} = \sum_{\substack{s \le t \\ i_s = i}} \sum_{k=1}^{\boldsymbol{K}_s} \boldsymbol{y}_{s,k} \boldsymbol{x}_{s,k},$$
$$T_{i,t} = \sum_{s \le t,\ i_s = i} \boldsymbol{K}_s$$

to be the Gramian matrix, the moment matrix of regressand by regressors, and the number of effective feedbacks for user i up to time t, respectively. Let $\emptyset \neq I \subset [u]$ be any nonempty user index subset and

$$\begin{aligned} \boldsymbol{M}_{I,t} &= \lambda I_d + \sum_{i \in I} \boldsymbol{S}_{i,t}, \quad & \boldsymbol{b}_{I,t} &= \sum_{i \in I} \boldsymbol{b}_{i,t}, \\ \hat{\boldsymbol{\theta}}_{I,t} &= \boldsymbol{M}_{I,t}^{-1} \boldsymbol{b}_{I,t}, & T_{I,t} &= \sum_{i \in I} T_{i,t} \end{aligned} \tag{5}$$

be the regularized Gramian matrix, the moment matrix of regressand, the estimate by ridge regressors, and the frequency associated with user set I and regularization parameter $\lambda > 0$ up to time t, respectively.

Algorithm

The algorithm maintains an undirected graph structure on all users $G_t = ([u], E_t)$, where an edge exists between a pair of users if they are similar. The collection of the connected components represents a partition of the users.

The learning agent starts with a complete graph over all users and initializes Gramian matrix and the moment matrix of regressand for each user i (Line 2). At each time step t, the learning agent receives a user index i_t and a feasible finite content set D_t (Line 4), where $\|x\|_2 \le 1$ for all $x \in D_t$. From the current graph structure on users

Algorithm 1 CLUB-cascade

1: **Input:** $\lambda, \alpha, \beta > 0$
2: **Initialize:** $G_0 = ([u], E_0)$ is a complete graph over all users, $\boldsymbol{S}_{i,0} = 0_{d \times d}, \boldsymbol{b}_{i,0} = 0_{d \times 1}, T_{i,0} = 0$ for all $i \in [u]$.
3: **for all** $t = 1, 2, \ldots, n$ **do**
4: Receive user index i_t, and the feasible context set $D_t \subset \mathbb{R}^{d \times 1}$;
5: Find the connected component V_t for user i_t in the current graph $G_{t-1} = ([u], E_{t-1})$, and compute

$$\begin{aligned} \boldsymbol{M}_{V_t,t-1} &= \lambda I + \sum_{i \in V_t} \boldsymbol{S}_{i,t-1}, \\ \boldsymbol{b}_{V_t,t-1} &= \sum_{i \in V_t} \boldsymbol{b}_{i,t-1}, \\ \hat{\boldsymbol{\theta}}_{V_t,t-1} &= \boldsymbol{M}_t^{-1} \boldsymbol{b}_t; \end{aligned}$$

6: For all $x \in D_t$, compute

$$\boldsymbol{U}_t(x) = \min\{\hat{\boldsymbol{\theta}}_{V_t,t-1}^\top x + \beta \sqrt{x^\top \boldsymbol{M}_{V_t,t-1}^{-1} x}, 1\}; \tag{4}$$

7: Recommend a list of K items $\boldsymbol{X}_t = (\boldsymbol{x}_{t,1}, \ldots, \boldsymbol{x}_{t,K})$ with largest $\boldsymbol{U}_t(\cdot)$ values and receive feedback $\boldsymbol{C}_t \in \{1, \ldots, K, \infty\}$;
8: Update statistics

$$\begin{aligned} \boldsymbol{S}_{i_t,t} &= \boldsymbol{S}_{i_t,t-1} + \sum_{k=1}^{\boldsymbol{K}_t} \boldsymbol{x}_{t,k} \boldsymbol{x}_{t,k}^\top, \\ \boldsymbol{b}_{i_t,t} &= \boldsymbol{b}_{i_t,t-1} + \sum_{k=1}^{\boldsymbol{K}_t} \boldsymbol{x}_{t,k} \boldsymbol{1}\{\boldsymbol{C}_t = k\}, \\ T_{i_t,t} &= T_{i_t,t-1} + \boldsymbol{K}_t, \end{aligned}$$

where $\boldsymbol{K}_t = \min\{\boldsymbol{C}_t, K\}$ and update

$$\hat{\boldsymbol{\theta}}_{i_t,t} = (\lambda I + \boldsymbol{S}_{i_t,t})^{-1} \boldsymbol{b}_{i_t,t};$$

9: Update $\boldsymbol{S}_{\ell,t} = \boldsymbol{S}_{\ell,t-1}, \boldsymbol{b}_{\ell,t} = \boldsymbol{b}_{\ell,t-1}, T_{\ell,t} = T_{\ell,t-1}, \hat{\boldsymbol{\theta}}_{\ell,t} = \hat{\boldsymbol{\theta}}_{\ell,t-1}$ for all $\ell \neq i_t$;
10: Delete the edge $(i_t, \ell) \in E_{t-1}$, if

$$\Big\|\hat{\boldsymbol{\theta}}_{i_t,t} - \hat{\boldsymbol{\theta}}_{\ell,t}\Big\|_2 \ge \alpha\Big(\frac{\sqrt{1 + \ln(1 + T_{i_t,t})}}{1 + T_{i_t,t}} + \frac{\sqrt{1 + \ln(1 + T_{\ell,t})}}{1 + T_{\ell,t}}\Big)$$

and obtain a new graph $G_t = ([u], E_t)$;
11: **end for** t

$G_{t-1} = ([u], E_{t-1})$, the agent finds the connected component V_t containing user i_t and computes the Gramian matrix, the moment matrix of regressand, and the estimates $\hat{\boldsymbol{\theta}}_{V_t,t-1}$ by ridge regressor associated with set V_t up to time $t-1$ (Line 5). Then it uses this $\hat{\boldsymbol{\theta}}_{V_t,t-1}$ as the estimate for the true weight vector $\theta_{j(i_t)}$ to compute the upper confidence bound of the expected reward $\theta_{j(i_t)}^\top x$ for each item $x \in D_t$ (Line 6). This step relies on the following lemma, which gives the theoretical guarantee of the ridge regression estimate for the true weight vector.

Lemma 1 *Suppose $(x_1, y_1), \ldots, (x_t, y_t), \ldots$ are generated sequentially from a linear model such that $\|x_t\| \leq 1$ for all t, $\mathbb{E}[y_t|x_t] = \theta_*^\top x_t$ for fixed but unknown θ_* with norm at most 1, and $\{y_t - \theta_*^\top x_t\}_{t=1,2,\ldots}$ have R-sub-Gaussian tails. Let $M_t = \lambda I + \sum_{s=1}^t x_s x_s^\top$, $b_t = \sum_{s=1}^t x_s y_s$, and $\delta > 0$. If $\hat{\theta}_t = M_t^{-1} b_t$ is the ridge regression estimator of θ_*, then with probability at least $1-\delta$, for all $t \geq 0$,*

$$\left\|\hat{\theta}_t - \theta_*\right\|_{M_t} \leq R\sqrt{d\ln\left(1+\frac{t}{\lambda d}\right) + 2\ln\frac{1}{\delta}} + \sqrt{\lambda} =: \beta(t,\delta).$$

This Lemma is by Theorem 2 of (Abbasi-Yadkori, Pál, and Szepesvári 2011).

When the current cluster is correct (which is guaranteed after $O(\ln(n))$ rounds and to be proved later), i.e. $V_t = I_{j(i_t)}$,

$$\left\|\theta_{j(i_t)} - \hat{\boldsymbol{\theta}}_{V_t,t-1}\right\|_{\boldsymbol{M}_{V_t,t-1}} \leq \boldsymbol{\beta}(T_{V_t,t-1},\delta) \leq \boldsymbol{\beta}(n,\delta).$$

(Here for a positive-definite matrix M, define the norm $\|x\|_M = \sqrt{x^T M x}$. It is not hard to verify that if $M \succ 0$, the dual norm $\|x\|_M$ is $\|x\|_{M^{-1}}$.) Then by Cauchy-Schwarz inequality, we have

$$\left|\hat{\boldsymbol{\theta}}_{V_t,t-1}^\top x - \theta_{j(i_t)}^\top x\right| \leq \left\|\hat{\boldsymbol{\theta}}_{V_t,t-1} - \theta_{j(i_t)}\right\|_{\boldsymbol{M}_{V_t,t-1}} \|x\|_{\boldsymbol{M}_{V_t,t-1}^{-1}} \leq \boldsymbol{\beta}(n,\delta)\,\|x\|_{\boldsymbol{M}_{V_t,t-1}^{-1}},$$

which results in a confidence interval for the expected reward $\theta_{j(i_t)}^\top x$ on each item $x \in D_t$.

Next the learning agent recommends a list of K items $\boldsymbol{X}_t = (\boldsymbol{x}_1, \ldots, \boldsymbol{x}_K)$ which have the largest upper confidence bounds. The user i_t checks the recommended items from the first one, clicks on the first satisfactory item, and stops checking anymore. Then the learning agent receives feedback $\boldsymbol{C}_t \in \{1, \ldots, K, \infty\}$ (Line 7). $\boldsymbol{C}_t \in \{1, \ldots, K\}$ means that the user clicks $\boldsymbol{C}_t$-th item and the first $\boldsymbol{C}_t - 1$ items are not satisfactory, while the items after $\boldsymbol{C}_t$-th position are not checked by the user. $\boldsymbol{C}_t = \infty$ means that the user has checked all recommended items but none of them is satisfactory. Based on the feedbacks, the learning agent updates its statistics on user i_t (Line 8) but not on other users (Line 9).

Based on the updates, the weight vector estimate for the user i_t might change and the similarity with other users might be verified false. The learning agent checks the edge of $(i_t, \ell) \in E_{t-1}$ for any user ℓ that is linked to user i_t and deletes it if the distance between the two estimated weight vectors is large enough (Line 10).

Analysis

The following theorem gives a bound on the cumulative regret achieved by our algorithm CLUB-cascade.

Theorem 2 *Suppose the cluster structure on the users, user appearance, and items satisfy the assumptions stated in the section of Problem Setup with gap parameter $\gamma > 0$ and item regularity parameter $0 < \lambda_x \leq 1$. Let λ, K be the regularization constant and the number of recommended items in each round. Let $\lambda \geq K, \beta = \sqrt{d\ln\left(1+\frac{n}{\lambda d}\right) + 2\ln(4mn)} + \sqrt{\lambda}$ and $\alpha = 4\sqrt{d}/\lambda_x$, where d, m, u denotes the feature dimension, the number of clusters and the number of users, respectively. Then the cumulative regret of CLUB-cascade algorithm for n rounds satisfies*

$$\begin{aligned} R(n) \leq & 2\beta\sqrt{2dmnK\ln\left(1+\frac{nK}{\lambda d}\right)} \\ & + O\left((u+\frac{1}{\lambda_x^2})\ln(n) + \frac{\sqrt{d}}{\gamma\lambda_x}\sqrt{\ln(n)}\right) \qquad (6) \\ \leq & O\left(d\sqrt{mnK}\ln(n)\right). \end{aligned}$$

For the degenerate case when $m = 1$, our result improves the existing regret bounds.

Corollary 3 *When the number of clusters $m = 1$, that is all users are treated as one, let $\lambda = K$ and $\beta = \sqrt{d\ln\left(1+\frac{n}{\lambda d}\right) + 2\ln(4n)} + \sqrt{\lambda}$. Then the cumulative regret of CLUB-cascade after n rounds satisfies*

$$\begin{aligned} R(n) \leq & 2\left(\sqrt{dn\ln(1+\frac{n}{\lambda d}) + 2\ln(4n)} + \sqrt{\lambda}\right) \\ & \cdot \sqrt{2dK\ln\left(1+\frac{nK}{\lambda d}\right)} \\ \leq & O(d\sqrt{nK}\ln(K)). \qquad (7) \end{aligned}$$

Note this result improves the existing results(Li et al. 2016; Zong et al. 2016). Discussions about the results, problem assumptions and implementations are given later.

Next we give a proof sketch for the Theorem 2.

Proof. [Sketch for Theorem 2] The proof for the main theorem is mainly based on two parts. The first part proves the exploration rounds needed to guarantee the clusters partitioned correctly. And the second part is to estimate regret bounds for linear cascading bandits after the clusters are partitioned correctly.

Under the assumption of item regularity, we prove when $T_{i,t} \geq O\left(\frac{1}{\gamma\lambda_x}\sqrt{d\ln(n)}\right)$, the $\|\cdot\|_2$ confidence radius for weight vector associated with user i will be smaller than $\gamma/2$, where the γ is the gap constant raised in the assumption of cluster regularity. Suppose user i and user ℓ belong

to different clusters and the effective number of feedbacks associated to both user i and ℓ meet the requirements. Then the condition in the Algorithm 1 of deleting an edge (i, ℓ) (Line 10) will be satisfied, thus the edge between user i and ℓ will be deleted under our algorithm with high probability. On the other hand, if the condition of deleting an edge (i, ℓ) is satisfied, then the $\|\cdot\|_2$ difference between the weight vectors is greater than 0, thus the two users belong to different clusters, by the assumption of cluster regularity.

By the assumption of item regularity and Bernstein's inequality, after $t \geq O\left((u + \frac{1}{\lambda_x^2})\ln(n) + \frac{\sqrt{d}}{\gamma\lambda_x}\sqrt{\ln(n)}\right)$ rounds, we could gather enough information for every user, thus resulting a correct clustering with high probability.

After the clusters are correctly partitioned, the recommendation is based on the estimates of cluster weight vector with the cascade feedback collected so far. After decomposing, the instaneous regret can be bounded by the individual difference between expected rewards of best items and checked items, which can be bounded with $2\beta \|x\|_{\boldsymbol{M}}$, by the definition of $\boldsymbol{U}_t(x)$. Then it remains to bound the sum of self-normalized sequence $\sum_{t=1}^{n} \|x_t\|_{M_{t-1}^{-1}}$, where $M_t = M_{t-1} + xx^\top$. □

Extensions to Generalized Linear Rewards

In this section, we consider a general case that the expected reward of recommending item x to user i_t at round t is

$$\mathbb{E}_t[\boldsymbol{y}|x] = \mu(\theta_{i_t}^\top x),$$

where μ is a strictly increasing link function, continuously differentiable, and Lipschitz with constant κ_μ. This definition arises from exponential family distributions (Filippi et al. 2010) and incorporates a large class of problems, like Poisson or logistic regression. Let $c_\mu = \inf_{a\in[-2,2]} \mu'(a)$ and assume $c_\mu > 0$.

In this setting, let the estimator $\hat{\boldsymbol{\theta}}_{I,t-1}$ for the set of users I be maximum likelihood estimator, or equivalently (Filippi et al. 2010; Li, Lu, and Zhou 2017) the unique solution of

$$\sum_{s=1}^{t-1} \mathbf{1}\{i_s \in I\} \sum_{k=1}^{\boldsymbol{K}_s} \left(\boldsymbol{y}_{s,k} - \mu(\theta^\top \boldsymbol{x}_{s,k})\right) \boldsymbol{x}_{s,k} = 0, \quad (8)$$

which can be found efficiently using Newton's algorithm. Note that the original samples $(\boldsymbol{x}_{s,k}, \boldsymbol{y}_{s,k})$ are stored instead of only aggregation $\boldsymbol{S}, \boldsymbol{b}$ in the linear case. With a slightly modified version of Algorithm 1, a result of the cumulative regret bound is obtained and provided in the following theorem.

Theorem 4 *Under the same assumptions and notations in linear setting, let* $\beta = \frac{1}{c_\mu}\sqrt{\frac{8}{\lambda_x} + d\ln(n/d) + 2\ln(4mn)}$ *and* $\alpha = 16\sqrt{d}/(\lambda_x c_\mu)$*, where* d, m, u *denotes the feature dimension, the number of clusters, and the number of users, respectively. Then the cumulative regret of CLUB-cascade with generalized linear rewards, after* n *rounds, satisfies*

$$\begin{aligned} R(n) \leq & 2\kappa_\mu\beta\sqrt{2dmnK\ln\left(1 + \frac{nK}{\lambda d}\right)} \\ & + O\left((u + \frac{1}{\lambda_x^2})\ln(n) + \frac{\sqrt{d}}{\gamma\lambda_x}\sqrt{\ln(n)}\right) \quad (9) \\ \leq & O\left(\frac{\kappa_\mu d}{c_\mu}\sqrt{mnK}\ln(n)\right). \end{aligned}$$

Discussions

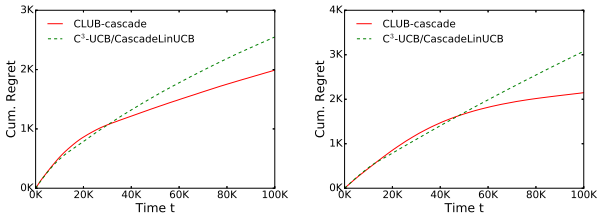
The degenerate case where the number of clusters $m = 1$, or equivalently all users are treated as the same type, has the same setting with Section 4.2.2 of (Li et al. 2016) and the setting in (Zong et al. 2016). The regret proved in the first paper is $O(\frac{d}{p^*}\sqrt{nK}\ln(n))$, which has an additional term $1/p^*$ compared to ours. The parameter p^* denotes the minimal probability that a user has checked all items, which could be quite small. The reason that we can get rid of such a $1/p^*$ term is because we have a better regret decomposition formula than theirs. The regret presented in the second paper has the bound of $O(dK\sqrt{n}\ln(n))$, which has an additional term $\sqrt{K}$ than ours. This reason is that we have a tighter bound for the sum of self-normalized sequence.

For the assumption on the true cluster structure over users, we assume there is a gap $\gamma > 0$ between the weight vectors associated with different clusters. The parameter γ is a trade-off between personalization and collaborative filtering, where $\gamma = 2$ corresponds to the case of only one cluster containing all users and γ taking the value of minimal distance between different user weight vectors corresponds to the case that each user is one cluster. Also, the assumption of γ can be further relaxed by modifying γ along running the algorithm. Our algorithm explores clustering structures adaptively: It starts with one cluster, then finds finer and finer clustering until the cluster distance reaches γ. As more data flows in, the parameter γ can be changed smaller and our algorithm can continue working without the need to restart. By a similar analysis, we could derive an asymptotic regret bound (with parameters in the algorithm changed accordingly). We omit this part and simply assume a $\gamma > 0$ gap exists.

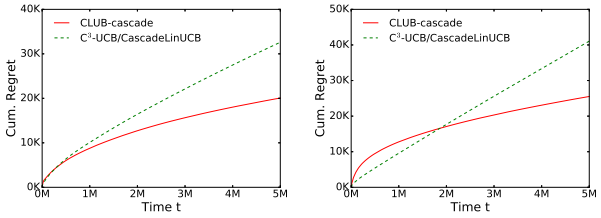
For the users, we assume the learning agent has no control over user appearances and at each time step, a user is drawn uniformly from all users, and independently from the past. If the learning agent has the access to sample users, the setting becomes active learning in online clustering and should have a better regret bound because the learning agent does not need to wait for collecting enough information. The uniform user appearance assumption means that the users we take care of are on the same activity level, which is easier for us to deal with. If there is some activity structure over users, we might need further assumptions and corresponding strategies on the activity structure. For example, if there are a large amount of new users, or users who only come a few time, additional assumptions like that those users share the same prediction vectors θ might be brought up. We leave the relaxations of the two assumptions as future work.

Experiments

In this section, we compare our algorithm with C^3-UCB (Li et al. 2016) and CascadeLinUCB (Zong et al. 2016),

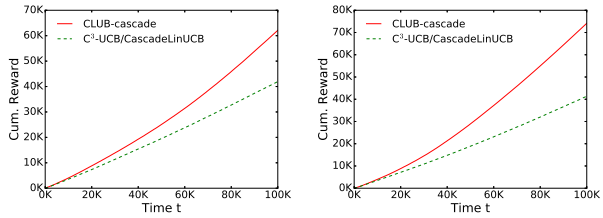


(a) θ orthogonal, $m = 2$ (b) θ orthogonal, $m = 5$

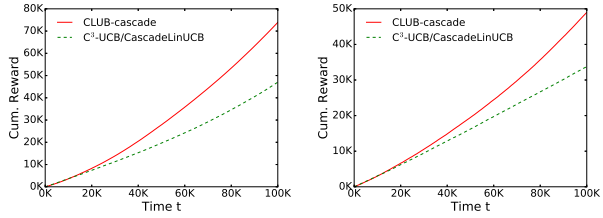


(c) $\gamma = 0.2,\ m = 2$ (d) $\gamma = 0.2, m = 5$

Figure 1: Synthetic Data Set, $u = 40,\ L = 200,\ K = 4,\ d = 20$



(a) $u = 40,\ L = 200$ (b) $u = 40,\ L = 1k$



(c) $u = 200,\ L = 200$ (d) $u = 200,\ L = 1k$

Figure 2: Cumulative clicks on Yelp dataset, $d = 20, K = 4$

which are the most related works. In both synthetic and real datasets, the results demonstrate the advantage of incorporating online clustering in the setting of online recommendations with cascade model. We focus on linear rewards for all experiments. To accelerate our algorithm, we use a sparse initialization instead of the complete graph initialization, similar in (Gentile, Li, and Zappella 2014).

Synthetic Data

In this section, we compare our algorithm, CLUB-cascade, with C^3-UCB/CascadeLinUCB on the synthetic data. The results are shown in Figure 1.

In all the four settings, we randomly choose a content set with $L = 200$ items, each of which has a feature vector $x \in \mathbb{R}^d$ with $\|x\|_2 \leq 1$ and $d = 20$. We use $u = 40$ users and assign them randomly to $m = 2, 5$ clusters. For each cluster $j \in [m]$, we fix a weight vector θ_j with norm 1 and use it to generate Bernoulli random variable, whose mean is the inner product of θ_j with the corresponding item vector. In each round, a random user comes and the algorithm recommends $K = 4$ items to the user. According to the Bernoulli random variables, the algorithm receives the cascading feedback and updates its statistics accordingly. In the synthetic setting, since we know the true weight vector θ_j, the best action can be computed and thus the cumulative regret for algorithms. The vertical axis denotes the cumulative regret and the horizontal axis denotes time step t.

In the four subfigures, we explore the distance gap γ between different θ's and the number of clusters m. When the gap γ between weight vectors θ is fixed, our algorithm has a better advantage over theirs when the number of clusters m is bigger. The θ's in subfigures (a)(c) are orthogonal, that is, the difference gap between them is $\gamma = \sqrt{2}$. The difference gap γ in (b)(d) is set to be 0.2, thus the cosine similarity between the different θ's is 0.98, which is quite high. Thus under the same number of clusters, our algorithm needs more time to learn well in the setting of a smaller γ. Because $\gamma = 0.2$ means near-1 cosine similarity, to regard all users as a whole might have advantages in early rounds. However, after our algorithm learns out the true cluster structure, their advantage depreciates very fast.

Although our algorithm needs more steps to achieve an obvious advantage with a smaller gap γ, typically it is not required to differentiate θ's with near-1 cosine similarity. The purpose we use this setting is to demonstrate the extreme case. In real applications, the estimated weight vectors will not be too similar so that our algorithm can easily outperform theirs, which we will see in the next experiments.

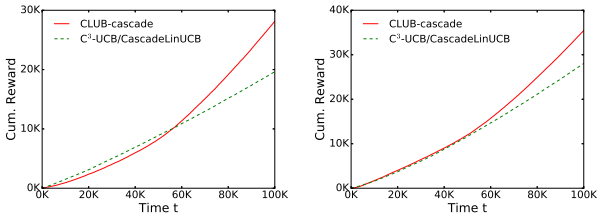
Note that our algorithm is not as good as theirs in the beginning. The reason is that we use a random sparse graph initialization and this initialization might result in inaccurate clustering for early rounds. However, after collecting enough information, our algorithm can still learn out the correct clustering (which might be a little finer for true clustering).

Yelp Dataset

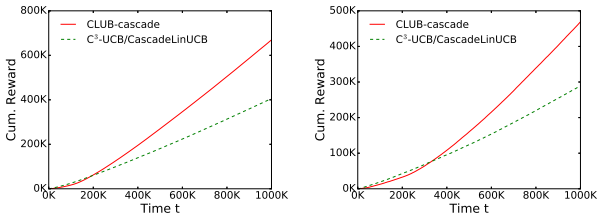
In this section, we compare our algorithm, CLUB-cascade, with C^3-UCB/CascadeLinUCB on restaurant recommendations with Yelp dataset[1]. The dataset contains user ratings for several businesses. For restaurants, it contains 1579523 ratings of 26629 restaurants from 478841 users. We extract $1k$ restaurants with most reviews and $1k$ users who review most for experiments.

Before we start, we randomly choose 100 users and formulate a binary matrix $H \in \mathbb{R}^{100\times 1k}$ (stands for 'history') where $H(i,k) = 1$ denotes the user i has rated restaurant k and $H(i,k) = 0$ denotes otherwise. We want to construct feature vectors for $1k$ restaurants from the records of 100 users and then use them to conduct experiments on the

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



(a) $u = 40,\ L = 200$ (b) $u = 40,\ L = 1k$



(c) $u = 200,\ L = 200$ (d) $u = 200,\ L = 1k$

Figure 3: Cumulative clicks on MovieLens, $d = 20, K = 4$

records of the remaining 900 users. Then, we perform SVD on H to get a $d = 20$ feature vectors for each of the chosen restaurants. The remaining ratings form another binary matrix $F \in \mathbb{R}^{900\times 1k}$ (stands for 'future'), which is used for online experiments.

For each of the following settings, we randomly choose $L = 200$ (or $1k$) restaurants and $u = 40$ (or 200) users. At each time step t, a user is selected uniformly and the learning agent recommends $K = 4$ restaurants to the user. By referring the binary matrix F, the learning agent receives a feedback $\boldsymbol{C}_t \in \{1, ..., K, \infty\}$ and updates its statistics. The objective is to maximize the cumulative clicks of the learning agent[2]. The results are shown in Figure 2, where the vertical axis denotes the cumulative rewards and the horizontal axis is the time step t. From the results, the performance of our algorithm has a clear advantage over theirs.

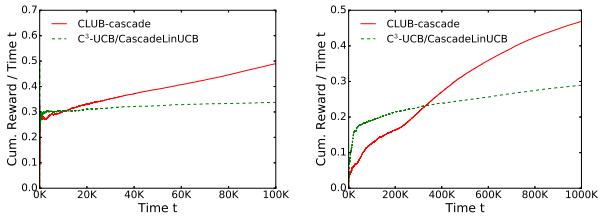
MovieLens Dataset

In this experiment, we compare our algorithm, CLUB-cascade, with C^3-UCB/CascadeLinUCB on the real dataset MovieLens (Harper and Konstan 2016). We use the processed 20m dataset[3], in which there are 20 million ratings for $27k$ movies by $138k$ users.

Since the MovieLens dataset has been processed and all users and movies have records with similar density, we randomly draw $1k$ movies and $1k$ users for experiments. After that, we randomly draw 100 users from the $1k$ users and formulate a binary matrix $H \in \mathbb{R}^{100\times 1k}$, where $H(i, j) = 1$ denotes the user i has rated movie j and $H(i, j) = 0$ denotes otherwise. Then we perform SVD on H to get a $d = 20$ feature vectors for all the $1k$ chosen movies. The

[2]Since there is no universal truth about correct clustering and choosing the gap parameter γ is quite subjective, to avoid disputes and be consistent with previous works (Li et al. 2016), we adopt the measure of cumulative rewards here.

[3]https : //grouplens.org/datasets/movielens/20m/



(a) Yelp (b) MovieLens

Figure 4: Comparisons of CTR on Yelp and MovieLens, $d = 20, K = 4, u = 200, L = 1000$

records for the remaining 900 users form another binary matrix $F \in \mathbb{R}^{900\times 1k}$, which is used for online evaluations.

For each of the four settings, we randomly choose $L = 200$ (or $1k$) movies and $u = 40$ (or 200) users. At each time step t, a user is selected uniformly and the learning agent recommends $K = 4$ movies to the user. By referring to the binary matrix F, the learning agent receives a feedback $\boldsymbol{C}_t \in \{1, ..., K, \infty\}$ and updates its statistics. The objective is to maximize the cumulative clicks of the learning agent. The results are shown in Figure 3, where the vertical axis denotes the cumulative rewards and the horizontal axis is the time step t. From the results, the performance of our algorithm has a clear advantage over theirs.

Comparing the performances on two datasets, our algorithm seems to need more steps to obtain an obvious advantage in the MovieLens dataset. This is because the MovieLens dataset has been processed and the user-movie matrix we are dealing with is quite dense, thus users are more similar. To see this phenomenon more clearly, we draw the results on the average rewards, the cumulative rewards up to time t divided by t, for both datasets in Figure 4. In earlier rounds, their algorithm taking all users as one will have a temporary advantage in MovieLens dataset since the users are similar. At the same time, our algorithm pays the cost of exploring clusters and starts with low average rewards. However, as the explored cluster structure becomes more and more accurate, our algorithm benefits from it and keeps a high increasing rate. As time goes by, the cost for regarding users as one is not negligible and our algorithm outperforms theirs. In most real applications, the user-item matrix would be very sparse and the users are tending to be dissimilar, resulting in a more advantaged environment for our algorithm.

Conclusions

In this paper, we bring up a new problem of online clustering of contextual cascading bandits, where the algorithm has to explore the unknown cluster structure on users under a prefix feedback of the recommended item list. We propose a CLUB-cascade algorithm based on the principle of optimism in face of uncertainty and prove a cumulative regret bound, whose degenerate case improves the existing results. The experiments conducted on both synthetic and real dataset demonstrate the advantage of incorporating online clustering.

Acknowledgments

This research was sponsored by Research Grants Council of the Hong Kong S.A.R. (Project no. CUHK14239416), as well as Huawei Innovation Research Program.

References

Abbasi-Yadkori, Y.; Pál, D.; and Szepesvári, C. 2011. Improved algorithms for linear stochastic bandits. In *Advances in Neural Information Processing Systems*, 2312–2320.

Chuklin, A.; Markov, I.; and Rijke, M. d. 2015. Click models for web search. *Synthesis Lectures on Information Concepts, Retrieval, and Services* 7(3):1–115.

Combes, R.; Magureanu, S.; Proutiere, A.; and Laroche, C. 2015. Learning to rank: Regret lower bounds and efficient algorithms. *ACM SIGMETRICS Performance Evaluation Review* 43(1):231–244.

Craswell, N.; Zoeter, O.; Taylor, M.; and Ramsey, B. 2008. An experimental comparison of click position-bias models. In *Proceedings of the 2008 International Conference on Web Search and Data Mining*, 87–94. ACM.

Filippi, S.; Cappe, O.; Garivier, A.; and Szepesvári, C. 2010. Parametric bandits: The generalized linear case. In *Advances in Neural Information Processing Systems*, 586–594.

Gentile, C.; Li, S.; Kar, P.; Karatzoglou, A.; Zappella, G.; and Etrue, E. 2017. On context-dependent clustering of bandits. In *International Conference on Machine Learning*, 1253–1262.

Gentile, C.; Li, S.; and Zappella, G. 2014. Online clustering of bandits. In *Proceedings of the 31st International Conference on Machine Learning*, 757–765.

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

Katariya, S.; Kveton, B.; Szepesvari, C.; and Wen, Z. 2016. Dcm bandits: Learning to rank with multiple clicks. In *Proceedings of The 33rd International Conference on Machine Learning*, 1215–1224.

Kveton, B.; Szepesvári, C.; Wen, Z.; and Ashkan, A. 2015a. Cascading bandits: learning to rank in the cascade model. In *Proceedings of the 32nd International Conference on Machine Learning*.

Kveton, B.; Wen, Z.; Ashkan, A.; and Szepesvari, C. 2015b. Combinatorial cascading bandits. In *Advances in Neural Information Processing Systems*, 1450–1458.

Li, S.; Wang, B.; Zhang, S.; and Chen, W. 2016. Contextual combinatorial cascading bandits. In *Proceedings of The 33rd International Conference on Machine Learning*, 1245–1253.

Li, S.; Karatzoglou, A.; and Gentile, C. 2016. Collaborative filtering bandits. In *Proceedings of the 39th International ACM SIGIR conference on Research and Development in Information Retrieval*, 539–548. ACM.

Li, L.; Lu, Y.; and Zhou, D. 2017. Provably optimal algorithms for generalized linear contextual bandits. In *International Conference on Machine Learning*, 2071–2080.

Zoghi, M.; Tunys, T.; Ghavamzadeh, M.; Kveton, B.; Szepesvari, C.; and Wen, Z. 2017. Online learning to rank in stochastic click models. In *International Conference on Machine Learning*, 4199–4208.

Zong, S.; Ni, H.; Sung, K.; Ke, N. R.; Wen, Z.; and Kveton, B. 2016. Cascading bandits for large-scale recommendation problems. In *the 32nd Conference on Uncertainty in Artificial Intelligence*.