On Context-Dependent Clustering of Bandits

Claudio Gentile [1] **Shuai Li** [2] **Purushottam Kar** [3] **Alexandros Karatzoglou** [4] **Giovanni Zappella** [5] **Evans Etrue** [1]

Abstract

We investigate a novel cluster-of-bandit algorithm CAB for collaborative recommendation tasks that implements the underlying feedback sharing mechanism by estimating user neighborhoods in a context-dependent manner. CAB makes sharp departures from the state of the art by incorporating collaborative effects into inference, as well as learning processes in a manner that seamlessly interleaves explore-exploit tradeoffs and collaborative steps. We prove regret bounds for CAB under various data-dependent assumptions which exhibit a crisp dependence on the expected number of clusters over the users, a natural measure of the statistical difficulty of the learning task. Experiments on production and real-world datasets show that CAB offers significantly increased prediction performance against a representative pool of state-of-the-art methods.

1. Introduction

In many prominent applications of bandit algorithms, such as computational advertising, web-page content optimization and recommendation systems, one of the main sources of information is embedded in the preference relationships between users and the items served. Preference patterns, emerging from clicks, views or purchase of items, are typically exploited through collaborative filtering techniques.

In fact, it is common knowledge in recommendation systems practice, that collaborative effects carry more information about user preferences than say, demographic metadata (Pilaszy & Tikk, 2009). Yet, as content recommendation functionalities are incorporated in diverse online services, the requirements differ vastly too. For instance, in a movie recommendation system, where the catalog is relatively static and ratings for items accumulate, one can easily deploy collaborative filtering methods such as matrix factorization or restricted Boltzmann machines. However, the same methods become practically impossible to use in more dynamic environments such as in news or YouTube video recommendation, where one has to deal with a near-continuous stream of new items to be recommended, along with new users to be served. These dynamic environments pose a dual challenge to recommendation methods: 1) How to present the new items to the users (or, vice versa, which items to present to new users), in order to optimally gather preference information on the new content (exploration), and 2) How to use all the available user-item preference information gathered so far (exploitation). Ideally, one would like to exploit both the content information but also, and more importantly, the collaborative effects that can be observed across users and items.

When the users to serve are many and the content universe (or content popularity) changes rapidly over time, recommendation services have to show both strong adaptation in matching user preferences and high degree of algorithmic scalability/responsiveness so as to allow effective online deployment. In typical scenarios like social networks, where users are engaged in technology-mediated interactions influencing each other's behavior, it is often possible to single out a few groups or *communities* made up of users sharing similar interests and/or behavior. Such communities are not static over time and, more often than not, are clustered around specific content *types*, so that a given set of users can in fact host a multiplex of interdependent communities depending on specific content items, which can change dramatically on the fly. We call this multiplex of interdependent clusterings over users induced by the content universe, a *context-dependent* clustering. In addition to the above, the set of users itself can change over time, for new users get targeted by the service, others may sign out or unregister. Thus, a recommendation method has to readily adapt to a changing set of both users and items.

In this paper, we introduce and analyze the CAB (Context-Aware clustering of Bandits) algorithm, a simple and flexible algorithm rooted in the linear contextual bandit framework that does the above by incorporating collaborative effects which traditional approaches to contextual bandits ig-

[1]DiSTA, University of Insubria, Italy [2]University of Cambridge, United Kingdom [3]IIT Kanpur, India [4]Telefonica Research, Spain [5]Amazon Dev Center, Germany (work done while at the University of Milan, Italy). Correspondence to: Claudio Gentile <claudio.gentile@uninsubria.it>.

Proceedings of the 34^{th} International Conference on Machine Learning, Sydney, Australia, PMLR 70, 2017. Copyright 2017 by the author(s).

nore (e.g., (Auer, 2002; Li et al., 2010; Chu et al., 2011; Abbasi-Yadkori et al., 2011)). CAB adapts to match user preferences in the face of a constantly evolving content universe and set of targeted users and implements the context-dependent clustering intuition by computing clusterings of bandits which allows each content item to cluster users into groups (which are few relative to the total number of users), where within each group, users tend to react similarly when that item gets recommended. CAB distinguishes itself in allowing distinct items to induce distinct clusterings which is frequently observed in practice (Sutskever et al., 2009). These clusterings are in turn suggestive of a natural context-dependent feedback sharing mechanism across users. CAB is thus able to exploit collaborative effects in contextual bandit settings in a manner similar to neighborhood techniques in batch collaborative filtering.

We analyze CAB from both, theoretical and experimental standpoints. We show that CAB enjoys a regret bound wherein the number of users engaged essentially enters in the regret bound only through the *expected* number of context-dependent clusters over the users, a natural measure of the predictive hardness of learning these users. We extend this result to provide a sharper bound under sparsity assumptions on the user model vectors. Experimentally, we present comparative evidence on production and real-world datasets that CAB significantly outperforms, in terms of predictive performance, state-of-the-art contextual bandit algorithms that either do not leverage any clustering at all or do so in a context-*in*dependent fashion.

1.1. Related Work

The literature on contextual bandit algorithms is too vast to be surveyed here. In the sequel, we point out works that most closely relate to ours. The technique of sequentially clustering users in the bandit setting was introduced by Gentile et al. (2014) and Maillard & Mannor (2014), but has also been inspired by earlier works such as (Azar et al., 2013) on transfer learning for stochastic bandits, and (Djolonga et al., 2013) on low-rank (Gaussian Process) bandits. This led to further developments such as (Nguyen & Lauw, 2014) which relies on k-means clustering, (Korda et al., 2016) which proposes distributed clustering of confidence ball algorithms for solving linear bandit problems in peer to peer networks, and (Zhou & Brunskill, 2016) that learns the latent classes of users (the clusters) so as to better serve new users. Related papers that implement feedback sharing mechanisms by leveraging (additional) social information among users include (Cesa-Bianchi et al., 2013; Wu et al., 2016). In all these cases, the way users are grouped is not context-dependent. Even more related to our work is the recent work of Li et al. (2016) which proposes to simultaneously cluster users as well as items, with item clusters dictating user clusters. However, a significant limitation of this approach is that the content universe has to be finite and known in advance, and in addition to the resulting algorithm being somewhat involved.

It is worth stressing that having context-dependent clusters not only changes the model considerably, as compared to previous works, but the algorithms as well. All previous works have a clustering process which is context-oblivious, or else assume a static item universe. A specific drawback of works such as (Gentile et al., 2014; Li et al., 2016) is that the clustering is *unidirectional*, in that users/items once (erroneously) separated into different clusters cannot be joined again, even if future evidence suggests so. Compared to previous works, our approach distinguishes itself for being simple and flexible (e.g., we can seamlessly accommodate the inclusion/exclusion of items and users), as well as for performing feedback propagation among users in a context-dependent manner. As will be demonstrated in Section 5, this offers significant performance boosts in real-world recommendation settings.

2. Notation and Preliminaries

We will consider the bandit clustering model standard in the literature, but with the crucial difference that we will allow user behavior similarity to be represented by a family of clusterings that depend on the specific item context under consideration. In particular, we let $\mathcal{U} = \{1, \ldots, n\}$ represent the set of n users. An item, represented by its feature vector $\boldsymbol{x} \in \mathbb{R}^d$ will be seen as inducing a (potentially unique) partition of the user set $\mathcal{U}$ into a small number $m(\boldsymbol{x})$ of clusters $\{U_1(\boldsymbol{x}), U_2(\boldsymbol{x}), \ldots, U_{m(\boldsymbol{x})}(\boldsymbol{x})\}$, where $m(\boldsymbol{x}) \ll n$. Users belonging to the same cluster $U_j(\boldsymbol{x})$ share similar behavior w.r.t. $\boldsymbol{x}$ (i.e. they both like or both dislike the item represented by $\boldsymbol{x}$), while users lying in different clusters have significantly different behavior.

This is a very flexible model that allows users to agree on their opinion of certain items and disagree on others, something that often holds in practice. It is important to note that the mapping $\boldsymbol{x} \to \{U_1(\boldsymbol{x}), U_2(\boldsymbol{x}), \ldots, U_{m(\boldsymbol{x})}(\boldsymbol{x})\}$ specifying the partitioning of $\mathcal{U}$ into the clusters determined by $\boldsymbol{x}$ (including the number of clusters $m(\boldsymbol{x})$), and the common user behavior within each cluster are *unknown* to the learner, and have to be inferred based on user feedback.

For the sake of simplicity, we assume that the context-dependent clustering is determined by linear functions $\boldsymbol{x} \to \boldsymbol{u}_i^\top \boldsymbol{x}$, each one parameterized by an unknown vector $\boldsymbol{u}_i \in \mathbb{R}^d$ hosted at user $i \in \mathcal{U}$, with $\|\boldsymbol{u}_i\| = 1$ for all i, in such a way that if users $i, i' \in \mathcal{U}$ are in the same cluster w.r.t. $\boldsymbol{x}$ then $\boldsymbol{u}_i^\top \boldsymbol{x} = \boldsymbol{u}_{i'}^\top \boldsymbol{x}$, and if $i, i' \in \mathcal{U}$ are in different clusters w.r.t. $\boldsymbol{x}$ then $|\boldsymbol{u}_i^\top \boldsymbol{x} - \boldsymbol{u}_{i'}^\top \boldsymbol{x}| \geq \gamma$, for some *gap parameter* $\gamma > 0$.[1] We will henceforth call this assumption the

[1] As usual, this hypothesis may be relaxed by assuming the

γ-gap assumption. For user vectors $\boldsymbol{u}_1, \ldots, \boldsymbol{u}_n \in \mathbb{R}^d$ corresponding to the n users (note that these vectors are unknown to the algorithm), context $\boldsymbol{x} \in \mathbb{R}^d$, and user index $i \in \mathcal{U}$, we denote by $N_i(\boldsymbol{x})$ the *true neighborhood* of i w.r.t. $\boldsymbol{x}$, i.e., $N_i(\boldsymbol{x}) = \{j \in \mathcal{U} \,:\, \boldsymbol{u}_j^\top \boldsymbol{x} = \boldsymbol{u}_i^\top \boldsymbol{x}\}$. Hence, $N_i(\boldsymbol{x})$ is simply the cluster (over $\mathcal{U}$) that i belongs to w.r.t. $\boldsymbol{x}$. Notice that $i \in N_i(\boldsymbol{x})$ for any i and any $\boldsymbol{x}$. We will henceforth assume that all items vectors $\boldsymbol{x}$ satisfy $\|\boldsymbol{x}\| \le 1$.

As is standard in linear bandit settings (Auer, 2002; Chu et al., 2011; Abbasi-Yadkori et al., 2011; Krause & Ong, 2011; Crammer & Gentile, 2011; Yue et al., 2012; Djolonga et al., 2013; Cesa-Bianchi et al., 2013; Agrawal & Goyal, 2013; Gentile et al., 2014; Li et al., 2016; Korda et al., 2016), the (unknown) user vector $\boldsymbol{u}_i$ determines the average behavior of user i. More precisely, upon encountering an item context vector $\boldsymbol{x}$, user i "reacts" by delivering a payoff value $y_i(\boldsymbol{x}) = \boldsymbol{u}_i^\top \boldsymbol{x} + \epsilon_i(\boldsymbol{x})$, where $\epsilon_i(\boldsymbol{x})$ is a conditionally zero-mean sub-Gaussian error variable with (conditional) variance parameter $\sigma^2(\boldsymbol{x}) \le \sigma^2$ for all $\boldsymbol{x}$.[2] Hence, conditioned on the past, the quantity $\boldsymbol{u}_i^\top \boldsymbol{x}$ is indeed the expected payoff observed at user i for context vector $\boldsymbol{x}$. For the sake of concreteness, we will assume that for all $i \in \mathcal{U}$ and $\boldsymbol{x} \in \mathbb{R}^d$ we have $y_i(\boldsymbol{x}) \in [-1, +1]$ a.s..

Learning takes place over a discrete sequence of time steps (or *rounds*). At each time $t = 1, 2, \ldots$, the learner receives a user index $i_t \in \mathcal{U}$, representing the user to serve content. Notice that the user to serve may change from round to round, and the same user may recur several times. Together with i_t, the learner receives a set of context vectors $C_t = \{\boldsymbol{x}_{t,1}, \boldsymbol{x}_{t,2}, \ldots, \boldsymbol{x}_{t,c_t}\} \subseteq \mathbb{R}^d$, such that $\|\boldsymbol{x}_{t,k}\| \le 1$ for all t and $k = 1, \ldots, c_t$, encoding the content which is currently available for recommendation to user i_t. The learner is compelled to pick some $\bar{\boldsymbol{x}}_t = \boldsymbol{x}_{t,k_t} \in C_t$ to recommend to i_t, and then observes i_t's feedback in the form of a payoff $y_t \in [-1, +1]$ whose (conditional) expectation is $\boldsymbol{u}_{i_t}^\top \bar{\boldsymbol{x}}_t$. The sequence of pairings $\{i_t, C_t\}_{t=1}^T = \{(i_1, C_1), (i_2, C_2), \ldots, (i_T, C_T)\}$ are generated by an exogenous process and, in a sense, represents the "data at hand". As we shall see in Section 4, the performance of our algorithm will depend on the properties of these data.

The practical goal of the learner is to maximize its total payoff $\sum_{t=1}^T y_t$ over T time steps. From a theoretical standpoint, we are instead interested in bounding the cumulative *regret* incurred by our algorithm. More precisely, let the regret r_t of the learner at time t be the extent to which the average payoff of the best choice in hindsight at user i_t exceeds the average payoff of the algorithm's choice, i.e.,

$$r_t = \left(\max_{\boldsymbol{x} \in C_t} \boldsymbol{u}_{i_t}^\top \boldsymbol{x}\right) - \boldsymbol{u}_{i_t}^\top \bar{\boldsymbol{x}}_t \, .$$

We are aimed at bounding with high probability (over the noise variables $\epsilon_{i_t}(\bar{\boldsymbol{x}}_t)$, and any other possible source of randomness) the cumulative regret $\sum_{t=1}^T r_t$. As a special case of the above model, when the set of items do not possess informative features, we can always resort to the non-contextual bandit setting (e.g., (Auer et al., 2002; Audibert et al., 2009)). To implement this approach, we simply take the set of all items (which must be finite for this technique to work), and apply a *one-hot* encoding by assigning to the i-th item, the i-th canonical basis vector $\boldsymbol{e}_i$, with one at the i-th position and zero everywhere else as the context vector. It is easy to see that the expected payoff given by user i on item j will simply be the j-th component of vector $\boldsymbol{u}_i$.

Our aim would be to obtain a regret bound that gracefully improves as the context-dependent clustering structure over the users becomes stronger. More specifically, values taken by the number of clusters $m(\boldsymbol{x})$ would be of particular interest since we expect to reap the strongest collaborative effects when $m(\boldsymbol{x})$ is small whereas not much can be done by way of collaborative analysis if $m(\boldsymbol{x}) \approx n$. Consequently, a desirable regret bound would be one that scales with $m(\boldsymbol{x})$. Yet, recall that $m(\boldsymbol{x})$ is a function of the context vector $\boldsymbol{x}$, which means that we expect our regret bound to also depend on the properties of the actual data $\{i_t, C_t\}_{t=1}^T$. We will see in Section 4 that, under suitable stochastic assumptions on the way $\{i_t, C_t\}_{t=1}^T$ is generated, our regret analysis essentially replaces the dependence on the total number of users n by the (possibly much) smaller quantity $\mathbb{E}[m(\boldsymbol{x})]$, the expected number of clusters over users, the expectation being over the draw of context vectors $\boldsymbol{x}$.

3. The Context-Aware Bandit Algorithm

We present Context-Aware (clustering of) Bandits (dubbed as CAB, see Algorithm 1), an upper-confidence bound-based algorithm for performing recommendations in the context-sensitive bandit clustering model. Similar to previous works (Cesa-Bianchi et al., 2013; Gentile et al., 2014; Nguyen & Lauw, 2014; Li et al., 2016; Wu et al., 2016), CAB maintains a vector estimate $\boldsymbol{w}_{i,t}$ to serve as a proxy to the unknown user vector $\boldsymbol{u}_i$ at time t. CAB also maintains standard correlation matrices $M_{i,t}$. The standard confidence bound function for user i for item $\boldsymbol{x}$ at time t is derived as $\mathrm{CB}_{i,t}(\boldsymbol{x}) = \alpha(t)\sqrt{\boldsymbol{x}^\top M_{i,t}^{-1} \boldsymbol{x}}$, for a suitable function $\alpha(t) = \mathcal{O}(\sqrt{d \log t})$.

However, CAB makes sharp departures from previous works both in the way items are recommended, as well as in they way the proxy estimates $\boldsymbol{w}_{i,t}$ are updated.

Item Recommendation: At time t, we are required to serve user $i_t \in \mathcal{U}$ by presenting an item out of a set of items

existence of two thresholds, one for the within-cluster distance of $\boldsymbol{u}_i^\top \boldsymbol{x}$ and $\boldsymbol{u}_{i'}^\top \boldsymbol{x}$, the other for the between-cluster distance.

[2] Recall that a zero-mean random variable X is sub-Gaussian with variance parameter σ^2 if $\mathbb{E}[\exp(sX)] \le \exp(s^2\,\sigma^2/2)$ for all $s \in \mathbb{R}$. Any variable X with $\mathbb{E}[X] = 0$ and $|X| \le b$ is sub-Gaussian with variance parameter upper bounded by b^2.

Algorithm 1 Context-Aware clustering of Bandits (CAB).

```
1:  Input: Separation parameter $\gamma$, exploration parameter $\alpha(t)$.
2:  Init: $\boldsymbol{b}_{i,0} = \mathbf{0} \in \mathbb{R}^d$ and $M_{i,0} = I \in \mathbb{R}^{d \times d},\ i \in \mathcal{U}$.
3:  for $t = 1, 2, \ldots, T$ do
4:    Set $\boldsymbol{w}_{i,t-1} = M_{i,t-1}^{-1} \boldsymbol{b}_{i,t-1}$,    for all $i \in \mathcal{U}$;
5:    Use $\mathrm{CB}_{i,t}(\boldsymbol{x}) = \alpha(t) \sqrt{\boldsymbol{x}^\top M_{i,t-1}^{-1} \boldsymbol{x}}$, for all $\boldsymbol{x}, i \in \mathcal{U}$;
6:    Receive user $i_t \in \mathcal{U}$ to be served, and context vectors $C_t = \{\boldsymbol{x}_{t,1}, \ldots, \boldsymbol{x}_{t,c_t}\}$ for items to be recommended from;
7:    for $k = 1, \ldots, c_t$ do
8:      Compute neighborhood $\widehat{N}_k := \widehat{N}_{i_t,t}(\boldsymbol{x}_{t,k})$ for this item
        $\widehat{N}_k = \Big\{ j \in \mathcal{U} \ : |\boldsymbol{w}_{i_t,t-1}^\top \boldsymbol{x}_{t,k} - \boldsymbol{w}_{j,t-1}^\top \boldsymbol{x}_{t,k}|$
            $\le \mathrm{CB}_{i_t,t-1}(\boldsymbol{x}_{t,k}) + \mathrm{CB}_{j,t-1}(\boldsymbol{x}_{t,k}) \Big\}$.
9:      Set $\boldsymbol{w}_{\widehat{N}_k,t-1} = \frac{1}{|\widehat{N}_k|} \sum_{j \in \widehat{N}_k} \boldsymbol{w}_{j,t-1}$;
10:     Set $\mathrm{CB}_{\widehat{N}_k,t-1}(\boldsymbol{x}_{t,k}) = \frac{1}{|\widehat{N}_k|} \sum_{j \in \widehat{N}_k} \mathrm{CB}_{j,t-1}(\boldsymbol{x}_{t,k})$;
11:   end for
12:   Recommend item $\bar{\boldsymbol{x}}_t = \boldsymbol{x}_{t,k_t} \in C_t$ such that
        $k_t = \operatorname*{argmax}_{k=1,\ldots,c_t} \Big( {\boldsymbol{w}_{\widehat{N}_k,t-1}}^\top \boldsymbol{x}_{t,k} + \mathrm{CB}_{\widehat{N}_k,t-1}(\boldsymbol{x}_{t,k}) \Big)$;
13:   Observe payoff $y_t \in [-1, 1]$;
14:   if $\mathrm{CB}_{i_t,t-1}(\bar{\boldsymbol{x}}_t) \ge \gamma/4$ then
15:     Set $M_{i_t,t} = M_{i_t,t-1} + \bar{\boldsymbol{x}}_t \bar{\boldsymbol{x}}_t^\top$,
16:     Set $\boldsymbol{b}_{i_t,t} = \boldsymbol{b}_{i_t,t-1} + y_t \bar{\boldsymbol{x}}_t$,
17:     Set $M_{j,t} = M_{j,t-1},\ \boldsymbol{b}_{j,t} = \boldsymbol{b}_{j,t-1}$ for all $j \ne i_t$;
18:   else
19:     for all $j \in \widehat{N}_{k_t}$ such that $\mathrm{CB}_{j,t-1}(\bar{\boldsymbol{x}}_t) < \gamma/4$ do
20:       $M_{j,t} = M_{j,t-1} + \bar{\boldsymbol{x}}_t \bar{\boldsymbol{x}}_t^\top$,
21:       $\boldsymbol{b}_{j,t} = \boldsymbol{b}_{j,t-1} + y_t \bar{\boldsymbol{x}}_t$;
22:     end for
23:     Set $M_{j,t} = M_{j,t-1},\ \boldsymbol{b}_{j,t} = \boldsymbol{b}_{j,t-1}$ for all $j \notin \widehat{N}_{k_t}$ and
        for $j \in \widehat{N}_{k_t}$ such that $\mathrm{CB}_{j,t-1}(\bar{\boldsymbol{x}}_t) \ge \gamma/4$ .
24:   end if
25: end for
```

$C_t = \{\boldsymbol{x}_{t,1}, \ldots, \boldsymbol{x}_{t,c_t}\}$ available at time t. To do so, CAB first computes for each item $\boldsymbol{x}_{t,k}$ in C_t, the set of users that are likely to give the item a similar payoff as i_t. This set $\widehat{N}_{i_t,t}(\boldsymbol{x}_{t,k})$ is the *estimated neighborhood* of user i_t with respect to item $\boldsymbol{x}_{t,k}$. A user j is included in $\widehat{N}_{i_t,t}(\boldsymbol{x}_{t,k})$ if the estimated payoff it gives to the item $\boldsymbol{x}_{t,k}$ is sufficiently close to that given to the item by user i_t (see step 8).

CAB incorporates collaborative effects by lifting the notions of the user proxy and confidence bounds to a set of users $N \subseteq \mathcal{U}$. CAB uses an inexpensive, flat averaging lift: $\mathrm{CB}_{N,t}(\boldsymbol{x}) = \frac{1}{|N|} \sum_{j \in N} \mathrm{CB}_{j,t}(\boldsymbol{x})$ and $\boldsymbol{w}_{N,t} = \frac{1}{|N|} \sum_{j \in N} \boldsymbol{w}_{j,t}$. Next, CAB uses (see step 12) aggregated confidence bounds $\mathrm{CB}_{\widehat{N}_{i_t,t}(\boldsymbol{x}_{t,k})}(\boldsymbol{x}_{t,k})$ and aggregated proxy vectors $\boldsymbol{w}_{\widehat{N}_{i_t,t}(\boldsymbol{x}_{t,k}),t-1}$ to select an item $\bar{\boldsymbol{x}}_t = \boldsymbol{x}_{t,k_t} \in C_t$ based on an upper confidence estimation step.

Proxy Updates: Classical approaches update the user proxies $\boldsymbol{w}_{i,t}$ by solving a regularized least squares problem involving (feature representations of) items served previously to user i and payoffs received. However, CAB remains fully committed to the collaborative approach (see steps 14-24) by allowing a user i to inherit updates due to an item $\boldsymbol{x}$ served to another user j if the two users are indeed deemed to agree on their opinion on item $\boldsymbol{x}$ with a sufficiently high degree of confidence. The proxies $\boldsymbol{w}_{j,t}$ are updated after receiving the feedback y_t from user i_t.

If CAB is not too confident regarding the opinion i_t has along the direction $\bar{\boldsymbol{x}}_t$, formally $\mathrm{CB}_{i_t,t-1}(\bar{\boldsymbol{x}}_t) \ge \gamma/4$, then only the proxy at user i_t is updated (see step 15-17). However, if CAB is confident, i.e., if $\mathrm{CB}_{i_t,t-1}(\bar{\boldsymbol{x}}_t) < \gamma/4$ then the proxy updates are performed (see steps 19-23) for all users j in i_t's estimated neighborhood with respect to $\bar{\boldsymbol{x}}_t$ about whose opinions CAB is confident too. Notice that all such users j undergo the same update, which is motivated by the algorithm's belief that $\widehat{N}_{i_t,t}(\bar{\boldsymbol{x}}_t) = N_{i_t}(\bar{\boldsymbol{x}}_t)$, i.e., that the conditional expectation $\boldsymbol{u}_{i_t}^\top \bar{\boldsymbol{x}}_t$ of y_t given $\bar{\boldsymbol{x}}_t$ is actually also equal to $\boldsymbol{u}_j^\top \bar{\boldsymbol{x}}_t$ for all users $j \in \widehat{N}_{i_t,t}(\bar{\boldsymbol{x}}_t)$ such that $\mathrm{CB}_{j,t-1}(\bar{\boldsymbol{x}}_t) < \gamma/4$.

It is worth noting that CAB is extremely flexible in handling a fluid set of users $\mathcal{U}$. Due to its context-sensitive user aggregation step, which is repeated at every round, CAB allows users to be added or dropped on the fly, in a seamless manner. This is in strike contrast to past approaches to bandit aggregation, such as GobLin (Cesa-Bianchi et al., 2013), CLUB (Gentile et al., 2014), and COFIBA (Li et al., 2016), where more involved feedback sharing mechanisms across the users are implemented which are based either on static network Laplacians or on time-evolving connected components of graphs over a given set of users.

4. Regret Analysis

Our regret analysis depends on a specific measure of hardness of the data at hand: for an observed sequence of users $\{i_t\}_{t=1}^T = \{i_1, \ldots, i_T\}$ and corresponding sequence of item sets $\{C_t\}_{t=1}^T = \{C_1, \ldots, C_T\}$, where $C_t = \{\boldsymbol{x}_{t,1}, \ldots, \boldsymbol{x}_{t,c_t}\}$, the *hardness* $\mathrm{HD}(\{i_t, C_t\}_{t=1}^T, \eta)$ of the pairing $\{i_t, C_t\}_{t=1}^T$ at level $\eta > 0$ is defined as

$$
\begin{aligned}
&\mathrm{HD}(\{i_t, C_t\}_{t=1}^T, \eta) \\
&\quad = \max\Big\{ t = 1, \ldots, T \ :\ \exists j \in \mathcal{U},\ \exists k_1, k_2, \ldots, k_t,\ : \\
&\qquad I + \sum_{s \le t \,:\, i_s = j} \boldsymbol{x}_{s,k_s} \boldsymbol{x}_{s,k_s}^\top \text{ has smallest eigenvalue} \le \eta \Big\}.
\end{aligned}
$$

Given a data sequence $\{i_t, C_t\}$, $\mathrm{HD}(\{i_t, C_t\}_{t=1}^T, \eta)$ measures the number of rounds we need to wait until correlation matrices $M_{j,t}$, corresponding to *all* users $j \in \mathcal{U}$ have eigenvalues lower bounded by η. For sake of convenience the above quantity is calculated using the worst possible way of updating the matrices $M_{j,t}$ through rank-one adjustments based on the data. Based on the above hardness definition, the following result summarizes our main efforts in this section.

Theorem 1 *Suppose CAB is executed on $\{i_t, C_t\}_{t=1}^T$, such that $c_t \le c$ for all t, and the condition $|\boldsymbol{u}_j^\top \boldsymbol{x} - \boldsymbol{w}_{j,t}^\top \boldsymbol{x}| \le \mathrm{CB}_{j,t}(\boldsymbol{x})$ holding for all $j \in \mathcal{U}$, $\boldsymbol{x} \in \mathbb{R}^d$, along with the γ-gap assumption. Then the cumulative regret $\sum_{t=1}^T r_t$ of CAB can be deterministically upper bounded as follows:*

$$\sum_{t=1}^{T} r_t \le 9\alpha(T)\Bigg(c\,\mathrm{HD}\Big(\{i_t, C_t\}_{t=1}^T, \frac{16\,\alpha^2(T)}{\gamma^2}\Big) + \sqrt{d\,\log T \sum_{t=1}^{T} \frac{n}{|N_{i_t}(\bar{\boldsymbol{x}}_t)|}}\Bigg),$$

where we set $\alpha(T) = \mathcal{O}(\sqrt{\log T})$. Some comments are in order. Theorem 1 delivers a *deterministic* regret bound on the cumulative regret, and is composed of two terms. The first term is a measure of hardness of the data sequence $\{i_t, C_t\}_{t=1}^T$ at hand whereas the second term is the usual $\sqrt{T}$-style term in linear bandit regret analyses (Auer, 2002; Chu et al., 2011; Abbasi-Yadkori et al., 2011). However, note that the dependence of the second term on the total number n of users to be served gets replaced by a much smaller quantity $\frac{n}{|N_{i_t}(\bar{\boldsymbol{x}}_t)|}$ that depends on the actual size of context-dependent clusters of the served users. The statement of the theorem requires $|\boldsymbol{u}_j^\top \boldsymbol{x} - \boldsymbol{w}_{j,t}^\top \boldsymbol{x}| \le \mathrm{CB}_{j,t}(\boldsymbol{x})$, which (see Lemma 3 below) holds from standard results.

We will shortly see that if the pairings $\{i_t, C_t\}_{t=1}^T$ are generated in a favorable manner, such as sampling the item vectors $\boldsymbol{x}_{t,k}$ i.i.d. from a (possibly unknown) distribution over the instance space (see Lemma 1 below), the hardness measure can be upper bounded with high probability by a term of the form $\frac{\log T}{\gamma^2}$. Similarly, for the second term, in the simple case when $N_{i_t}(\bar{\boldsymbol{x}}_t) = B$ for all t, the second term has the form $\sqrt{\frac{n}{B}T}$, up to log factors. Notice that $\sqrt{T}$ is roughly the regret effort for learning a single bandit, and $\sqrt{\frac{n}{B}T}$ is the effort for learning $\frac{n}{B}$-many (unrelated) clusters of bandits when the clustering is known. Thus, in this example, it is the ratio $\frac{n}{B}$ that quantifies the hardness of the problem, insofar clustering is concerned. Again, under favorable circumstances (see Lemma 2 below), we can relate the quantity $\sum_{t=1}^T \frac{n}{|N_{i_t}(\bar{\boldsymbol{x}}_t)|}$ to the *expected* number of context-dependent clusters of users, the expectation being w.r.t. the random draw of item context vectors.

On the other hand, making no assumptions whatsoever on the way $\{i_t, C_t\}_{t=1}^T$ is generated makes it hard to exploit the cluster structure. For instance, if $\{i_t, C_t\}_{t=1}^T$ is generated by an adaptive adversary, this might cause $\mathrm{HD}\left(\{i_t, C_t\}_{t=1}^T, \eta\right)$ to become linear in T for any constant $\eta > 1$, thereby making the bound in Theorem 1 vacuous. However, a naive algorithm that disregards the cluster structure, making no attempts to incorporate collaborative effects, and running n-many independent LinUCB-like algorithms (Auer, 2002; Abbasi-Yadkori et al., 2011; Chu et al., 2011), will still easily yield a $\sqrt{n\,T}$ regret bound[3].

A sufficient condition for controlling the hardness term in Theorem 1 is provided by the following lemma.

Lemma 1 *For each round t, let the context vectors $C_t = \{\boldsymbol{x}_{t,1}, \dots, \boldsymbol{x}_{t,c_t}\}$ be generated i.i.d. (conditioned on i_t, c_t, past data $\{i_s, C_s\}_{s=1}^{t-1}$ and rewards $y_1, \dots, y_{t-1}$) from a sub-Gaussian random vector $X \in \mathbb{R}^d$ with (conditional) variance parameter ν^2, such that $\|X\| \le 1$ a.s., and $\mathbb{E}[XX^\top]$ is full rank with smallest eigenvalue $\lambda > 0$. Let $c_t \le c$ for all t, and $\nu^2 \le \frac{\lambda^2}{8\ln(4c)}$. Finally, let the sequence $\{i_t\}_{t=1}^T$ be generated uniformly at random,* [4] *independent of all other variables. Then with probability at least $1-\delta$,*

$$\mathrm{HD}\Big(\{i_t, C_t\}_{t=1}^T, \eta\Big) = \mathcal{O}\left(\frac{n\,\eta}{\lambda^2}\,\log\left(\frac{Tnd}{\delta}\right)\right).$$

The following lemma handles the second term in the bound of Theorem 1.

Lemma 2 *For each round t, let the context vectors $C_t = \{\boldsymbol{x}_{t,1}, \dots, \boldsymbol{x}_{t,c_t}\}$ be generated i.i.d. (conditioned on i_t, c_t, past data $\{i_s, C_s\}_{s=1}^{t-1}$ and rewards $y_1, \dots, y_{t-1}$) from a random vector $X \in \mathbb{R}^d$ with $\|X\| \le 1$. Let also $c_t \le c$ for all t. Then, with probability at least $1-\delta$,*

$$\sum_{t=1}^{T} \frac{1}{|N_{i_t}(\bar{\boldsymbol{x}}_t)|} \le \frac{2Tc\,\mathbb{E}[m(X)]}{n} + 12\,\log\left(\frac{\log T}{\delta}\right).$$

Remark 1 *The linear dependence on c can be turned to logarithmic, e.g., at the cost of an extra sub-Gaussian assumption on variables $\frac{1}{|N_i(\boldsymbol{x})|}$, $i \in \mathcal{U}$.*

Finally, we recall the following upper confidence bound, from (Abbasi-Yadkori et al., 2011).

Lemma 3 *Let $\mathrm{CB}_{j,t}(\boldsymbol{x}) = \alpha(t)\sqrt{\boldsymbol{x}^\top M_{j,t}^{-1}\boldsymbol{x}}$, with $\alpha(t) = \mathcal{O}\left(\sqrt{d\,\log\frac{Tn}{\delta}}\right)$.*[5] *Then, under the payoff noise model defined in Section 2, $|\boldsymbol{u}_j^\top \boldsymbol{x} - \boldsymbol{w}_{j,t}^\top \boldsymbol{x}| \le \mathrm{CB}_{j,t}(\boldsymbol{x})$ holds uniformly for all $j \in \mathcal{U}$, $\boldsymbol{x} \in \mathbb{R}^d$, and $t = 1, 2, \dots$.*

A straightforward combination of Theorem 1 with Lemmata 1, 2, and 3 yields the following result.

Corollary 1 *Let $\mathrm{CB}_{j,t}(\boldsymbol{x})$ be defined with $\alpha(t)$ as in Lemma 3, and let the γ-gap assumption hold. Assume context vectors are generated as in Lemma 1 in such a way that*

[3] To see this, simply observe that each of the n LinUCB-like algorithms has a regret bound of the form $\sqrt{T_i}$, where T_i is the number of rounds where $i_t = i$. Then $\sum_{t=1}^T r_t \le \sum_{i=1}^n \sqrt{T_i} \le \sqrt{n\,T}$, with equality if $T_i = T/n$ for all i.

[4] Any (non-uniform) process $\{i_t\}_{t=1}^T$ that nevertheless ensures that each user i gets served a number of times that is likely to grow unbounded with T would suffice here.

[5] The big-oh notation here hides the dependence on the variance σ^2 of the payoff values.

the sub-Gaussian assumption therein holds with $c_t \le c$. Finally, let the sequence $\{i_t\}_{t=1}^T$ be generated as described in Lemma 1. Then, with probability at least $1 - \delta$, the regret of CAB (Algorithm 1) satisfies

$$\sum_{t=1}^{T} r_t = R + \widetilde{\mathcal{O}}\left(d\sqrt{Tc\,(\mathbb{E}[m(X)])}\right) ,$$

where the $\widetilde{\mathcal{O}}$-notation hides logarithmic factors in $\frac{TNd}{\delta}$, and R is of the form[6]

$$R = \frac{c\,n\,d\sqrt{d}}{\lambda^2\,\gamma^2}\,\log^{2.5}\left(\frac{Tnd}{\delta}\right).$$

Sparse user models. We conclude with a pointer to an extension of the CAB framework for sparse linear models that extends past analyses on sparse linear bandits for a single user (Abbasi-Yadkori et al., 2012; Carpentier & Munos, 2012; Carpentier, 2015), to include collaborative effects. If all user vectors are sparse, i.e. for all $i \in \mathcal{U}$, $\|\boldsymbol{u}_i\|_0 \le s$, for $s \ll d$, then by replacing the least-squares solution in Step 4 of Algorithm 1 with the solution computed by a fully corrective sparse recovery method (Dai & Milenkovic, 2009; Jain et al., 2014; Needell & Tropp, 2008), we can obtain an improved regret bound. Specifically, we can replace the factor $d\sqrt{d}$ in the term R above by $s^2\sqrt{s}$, and the factor d multiplying the $\sqrt{T}$-term by a term of the form $\sqrt{s\,d}$.

5. Experiments

We tested CAB on production and real-world datasets, and compared them to standard baselines as well as to state-of-the-art bandit and clustering of bandit algorithms. For datasets with no item features, a one-hot encoding was used. We attempted to follow, as closely as possible, previous experimental settings, such as those as described in (Cesa-Bianchi et al., 2013; Gentile et al., 2014; Korda et al., 2016; Li et al., 2016).

5.1. Dataset Description

Tuenti. Tuenti (owned by Telefonica) is a Spanish social network website that serves ads on its site, the data contains ad impressions viewed by users along with a variable that registers a click on an ad. The dataset contains $d = 105$ ads, $n = 14,612$ users, and 15M records/timesteps. We adopted a one hot encoding scheme for the items, hence items are described by the unit-norm vectors $\boldsymbol{e}_1, \ldots, \boldsymbol{e}_d$. Since the available payoffs are those associated with the items served by the system, we performed offline policy evaluation through a standard importance sampling technique: we discarded on the fly all records where the system's recommendation (the logged policy) did not coincide with the algorithms' recommendations. The resulting number of retained records was around $T = 1M$, loosely depending on the different algorithms and runs. Yet, because this technique delivers reliable estimates when the logged policy makes random choices (e.g., (Li et al., 2010)), we actually simulated a random logged policy as follows. At each round t, we retained the ad served to the current user i_t with payoff value a_t (1 = “clicked”, 0 = “not clicked”), but also included 14 extra items (hence $c_t = 15$ for all t) drawn uniformly at random in such a way that, for any item $\boldsymbol{e}_j$, if $\boldsymbol{e}_j$ occurs in some set C_t, this item will be the one served by system only $1/15$ of the times. Notice that this random selection was independent of the available payoff a_t.

KDD Cup. This dataset was released for the KDD Cup 2012 Online Advertising Competition[7] where the instances were derived from the session logs of the search engine soso.com. A search session included user, query and ad information, and was divided into multiple instances, each being described using the ad impressed at that time at a certain depth and position. Instances were aggregated with the same user ID, ad ID, and query. We took the chronological order among all the instances, and seeded the algorithm with the first $c_t = 20$ instances (the length of recommendation lists). Payoffs a_t are again binary. The resulting dataset had $n = 10,333$ distinct users, and $d = 6,780$ distinct ads. Similar to the Tuenti dataset, we generated random recommendation lists, and a random logged policy. We employed one-hot encoding as well in this dataset. The number of retained records was around $T = 0.1M$.

Avazu. This dataset was released for the Avazu Click-Through Rate Prediction Challenge on Kaggle[8]. Here click-through data were ordered chronologically, and non-clicks and clicks were subsampled according to different strategies. As before, we simulated a random logged policy over recommendation lists of size $c_t = 20\ \forall t$. Payoffs are once again binary. The final dataset had $n = 48,723$ users, $c_t = 20$ for all t, $d = 5,099$ items, while the number of retained records was around $T = 1.1M$. Again, we took the one-hot encoding for the items.

LastFM and Delicious. These two datasets[9] are extracted from the music streaming service Last.fm and the social bookmarking web service Delicious. The LastFM dataset includes n = 1,892 users, and 17,632 items (the artists). Delicious refers to n = 1,861 users, and 69,226 items (URLs). Preprocessing of data followed previous experimental settings where these datasets have been used, e.g., (Cesa-Bianchi et al., 2013; Gentile et al., 2014). Specifically, using a tf-idf representation of the available items, the context vectors $\boldsymbol{x}_{t,i}$ were generated by retaining only the first

[6] We note that no special efforts have been devoted here to obtain sharper upper bounds on R.

[7] http://www.kddcup2012.org/c/kddcup2012-track2

[8] https://www.kaggle.com/c/avazu-ctr-prediction

[9] www.grouplens.org/node/462

Table 1. Dataset statistics. Here, n is the number of users, d is the dimension of the item vectors (which corresponds to the number of items for Tuenti, KDD Cup and Avazu), c_t is the size of the recommendation lists, and T is the number of records (or just *retained* records, in the case of Tuenti, KDD Cup and Avazu).

DATASET	n	d	c_t	T
TUENTI	14,612	105	15	≃1,000,000
KDD CUP	10,333	6,780	20	≃100,000
AVAZU	48,723	5,099	20	≃1,100,000
LASTFM	1,892	25	25	50,000
DELICIOUS	1,861	25	25	50,000

$d = 25$ principal components. Binary payoffs were created as follows. LastFM: If a user listened to an artist at least once the payoff is 1, otherwise it is 0. Delicious: the payoff is 1 if the user bookmarked the URL, and 0 otherwise. We processed the datasets to make them suitable for use with multi-armed bandit algorithms. Recommendation lists C_t of size $c_t = 25\ \forall t$ were generated at random by first selecting index i_t at random over the n users, and then padding with 24 vectors chosen at random from the available items up to that time step, in such a way that at least one of these 25 items had payoff 1 for the current user i_t. This was repeated for $T = 50,000$ times for the two datasets.

5.2. Algorithms

We used the first 20% of each dataset to tune the algorithms' parameters through a grid search, and report results on the remaining 80%. All results are averaged over 5 runs. We compared to a number of state-of-the art bandit and clustering-of-bandit methods:

- CLUB (Gentile et al., 2014): sequentially refines user clusters based on their confidence ellipsoid balls; We seeded the graph over users by an initial random Erdos-Renyi graphs with sparsity parameter $p = (3 \log n)/n$. Since this is a randomized algorithm, each run was repeated five times, and the results averaged (the observed variance turned out to be small).
- DynUCB (Nguyen & Lauw, 2014): uses a traditional k-Means algorithm to cluster bandits.
- LinUCB-SINGLE: uses a single instance of LinUCB (Chu et al., 2011) to serve all users, i.e., all users belong to the same cluster, independent of the items.
- LinUCB-MULTIPLE: uses an independent instance of LinUCB per user with no user interactions. Each user forms his own cluster, independent of the items.
- The following variant of CAB (see Algorithm 1): each user j is considered for addition to the estimated neighborhoods $\widehat{N}_k$ of the currently served user i_t only if $\boldsymbol{w}_{j,t-1}$ has been updated at least once in the past.
- RAN: recommends a uniformly random item from C_t.

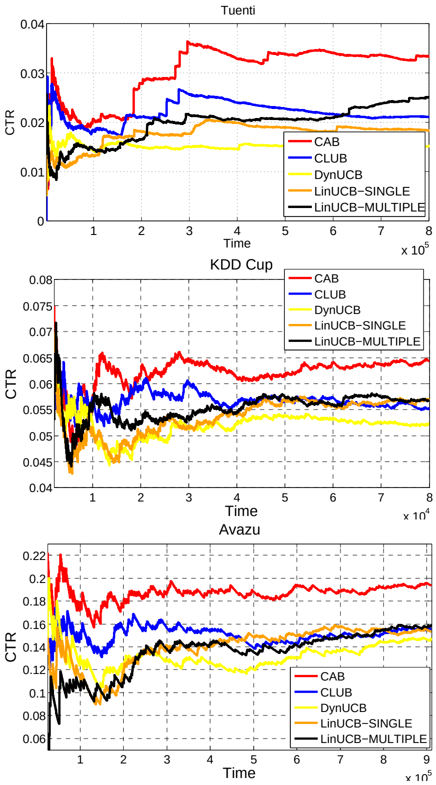

Figure 1. Clickthrough Rate vs. retained records ("time") on three online advertising datasets. The higher the curves the better.

All tested algorithms (except RAN) are based on upper-confidence bounds of the form $\text{CB}_{i,t}(\boldsymbol{x}) = \alpha \sqrt{\boldsymbol{x}^\top N_{i,t}\boldsymbol{x}\ \log(1+t)}$. We tuned α for all algorithms across the grid $\{0, 0.01, 0.02, \ldots, 0.2\}$. The α_2 parameter in CLUB was tuned within $\{0.1, 0, 2, \ldots, 0.5\}$. The number of clusters in DynUCB was increased in an exponential progression, starting from 1, and ending to n. Finally, the γ parameter in CAB was simply set to 0.2. In fact, the value of γ did not happen to have a significant influence on the performance of the version of CAB we tested.

5.3. Results

Figures 1, 2, 3 summarize our experimental results. For the online advertising datasets Tuenti, KDD Cup, and Avazu (Figure 1), we report performance using the Click-Through Rate (CTR), hence the higher the curves the better. For the LastFM and Delicious datasets (Figure 2), we report the ratio of the cumulative regret of the tested algorithm to the cumulative regret of RAN, hence the lower the better.

Our experimental setting, as well some results reproduced here, are in line with past work in the area (Li et al., 2010; Cesa-Bianchi et al., 2013; Gentile et al., 2014; Li et al., 2016). Given the way data have been prepared, our findings give reliable estimates of the actual CTR (Figure 1) or regret (Figure 2) performance of the tested algorithms.

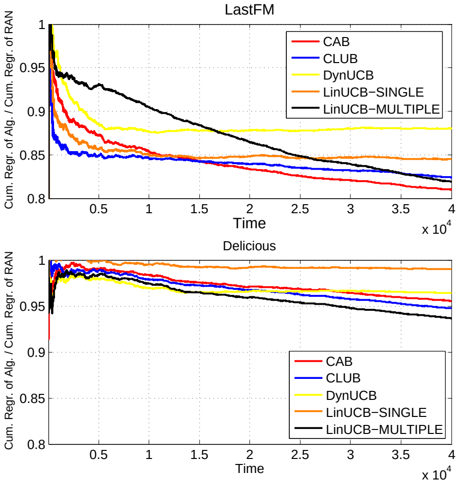

Figure 2. Ratio of the cumulative regret of the algorithms to the cumulative regret of RAN against time on the two datasets LastFM and Delicious. The lower the curves the better.

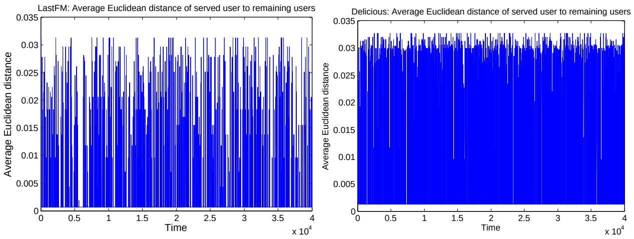

Figure 3. Average (estimated) Euclidean distance between the served user i_t and all other users, as a function of t for the two datasets LastFM (left) and Delicious (right). The distance is computed by associating with each user a model vector obtained through a regularized least-squares solution based on all available data for that user (instance vectors and corresponding payoffs).

In four out of five datasets, CAB was found to offer performance superior to all other baselines. CAB performed particularly well on the Tuenti dataset where it delivered almost double the CTR compared to some of the baselines. CAB's performance advantage was more moderate on the KDD Cup and Avazu datasets. This is expected since exploiting collaborative effects is more important on a dataset like Tuenti, where users are exposed to a few ($\approx$ 100) ads, as compared to the KDD Cup dataset (where ads are modulated by a user query) and the Avazu dataset, both of which have a much broader ad base ($\approx$ 7000). This provides a strong indication that CAB effectively exploits collaborative effects. In general, on Tuenti, KDD Cup, and Avazu (Figure 1), CAB was found to offer benefits in the cold-start region (i.e., the initial relatively small fraction of time horizon), but also continued to maintain a lead throughout.

On the LastFM and Delicious datasets (Figure 2), the results we report are consistent with (Gentile et al., 2014). On LastFM all methods are again outperformed by CAB. The overall performance of all bandit methods seems however, to be relatively poor; this can be attributed to the way the LastFM dataset was generated. Here users typically have little interaction with the music serving system and a lot of the songs played were generated by a recommender. Hence while there are collaborative effects, they are relatively weak compared to datasets such as Tuenti.

On the other hand, on the Delicious dataset the best performing strategy seems to be LinUCB-MULTIPLE, which deliberately avoids any feedback sharing mechanism among the users. This dataset reflects user web-browsing patterns, as evinced by their bookmarks. As noted previously (Gentile et al., 2014), this dataset does not seem to contain any collaborative information, hence we can hardly expect to take advantage of clustering efforts. To shed further light, in Figure 3 we plot the average distance between a linear model for user i_t and the corresponding linear models for all other users $j \neq i_t$, as a function of t. For each user of these two datasets, these linear models were computed by taking the whole test set and treating each pairing $(\boldsymbol{x}_{t,k}, y_{t,k})$ with $i_t = i$, and $y_{t,k} = 1$ as a training sample for a (regularized) least-squares estimator for user i. The conclusion we can draw after visually comparing the left and the right plots in Figure 3 is that on Delicious these estimated user models tend to be significantly more separated than on LastFM, which readily explains the effectiveness of LinUCB-MULTIPLE. Moreover, on Delicious, studies have shown that tags which are used as item features are generally chosen by users to reflect their interests and for personal use, hence we can expect these features to diverge even for similar websites. On the other hand, LastFM tags are typically indicative of the genre of the song.

6. Conclusions and Ongoing Research

In this paper we proposed CAB, a novel contextual bandit algorithm for personalized recommendation systems. CAB effectively incorporates collaborative effects by implementing a simple context-dependent feedback sharing mechanism which greatly relaxes restrictions imposed by earlier works. We established crisp regret bounds for CAB which scale gracefully with the expected number of clusters over the users. We further found CAB to outperform existing approaches in practice, using extensive experimentation on a number of production and real-world datasets.

We have begun preliminary experiments with Thompson sampling versions of CAB and its competitors but have thus far not observed statistically significant deviations from results in Section 5. From a theoretical standpoint, it would be nice to complement our upper bound in Corollary 1 with a lower bound helping characterize the regret complexity of our problem. We are also planning to experimentally benchmark with performance of the sparse bandit version of CAB that infers sparse user models.

Acknowledgments

The authors thank the ICML reviewers for useful comments. CG acknowledges partial support from Criteo through a Faculty Research Award. PK is supported by the Deep Singh and Daljeet Kaur Faculty Fellowship and the Research-I foundation at IIT Kanpur, and thanks Microsoft Research India for a research grant. GZ would like to thank N. Cesa-Bianchi for inspiring discussions regarding bandit algorithms and related topics.

References

Abbasi-Yadkori, Yasin, Pal, David, and Szepesvari, Csaba. Improved Algorithms for Linear Stochastic Bandits. In *Proceedings of the 25th Annual Conference on Neural Information Processing Systems (NIPS)*, 2011.

Abbasi-Yadkori, Yasin, Pál, Dávid, and Szepesvári, Csaba. Online-to-Confidence-Set Conversions and Application to Sparse Stochastic Bandits. In *Proceedings of the 15th International Conference on Artificial Intelligence and Statistics (AISTATS)*, 2012.

Agrawal, Shipra and Goyal, Navin. Thompson Sampling for Contextual Bandits with Linear Payoffs. In *Proceedings of the 30th International Conference on Machine Learning (ICML)*, 2013.

Audibert, Jean Yves, Munos, Remi, and Szepesvari, Csaba. Exploration-exploitation Tradeoff using Variance Estimates in Multi-armed Bandits. *Theoretical Computer Science*, 410(19):1876–1902, 2009.

Auer, Peter. Using Confidence Bounds for Exploration-Exploitation Trade-Offs. *Journal of Machine Learning Research*, 3:397–422, 2002.

Auer, Peter, Cesa-Bianchi, Nicolo, and Fischer, Paul. Finite-time Analysis of the Multiarmed Bandit Problem. *Machine Learning*, 47:235–256, 2002.

Azar, Mohammad Gheshlaghi, Lazaric, Alessandro, and Brunskill, Emma. Sequential Transfer in Multi-armed Bandit with Finite Set of Models. In *Proceedings of the 27th Annual Conference on Neural Information Processing Systems (NIPS)*, 2013.

Carpentier, Alexandra. Implementable Confidence Sets in High Dimensional Regression. In *Proceedings of the 18th International Conference on Artificial Intelligence and Statistics (AISTATS)*, 2015.

Carpentier, Alexandra and Munos, Remi. Bandit Theory meets Compressed Sensing for High-dimensional Stochastic Linear Bandit. In *Proceedings of the 15th International Conference on Artificial Intelligence and Statistics (AISTATS)*, 2012.

Cesa-Bianchi, Nicolo, Gentile, Claudio, and Zappella, Giovanni. A Gang of Bandits. In *Proceedings of the 27th Annual Conference on Neural Information Processing Systems (NIPS)*, 2013.

Chu, Wei, Li, Lihong, Reyzin, Lev, and Schapire, Robert. Contextual Bandits with Linear Payoff Functions. In *Proceedings of the 14th International Conference on Artificial Intelligence and Statistics (AISTATS)*, 2011.

Crammer, Koby and Gentile, Claudio. Multiclass Classification with Bandit Feedback using Adaptive Regularization. In *Proceedings of the 28th International Conference on Machine Learning (ICML)*, 2011.

Dai, Wei and Milenkovic, Olgica. Subspace Pursuit for Compressive Sensing Signal Reconstruction. *IEEE Transactions on Information Theory*, 55(5):2230–2249, 2009.

Dekel, Ofer, Gentile, Claudio, and Sridharan, Karthik. Selective Sampling and Active Learning from Single and Multiple Teachers. *Journal of Machine Learning Research*, 13:2655–2697, 2012.

Djolonga, Josip, Krause, Andreas, and Cevher, Volkan. High-Dimensional Gaussian Process Bandits. In *Proceedings of the 27th Annual Conference on Neural Information Processing Systems (NIPS)*, 2013.

Gentile, Claudio, Li, Shuai, and Zappella, Giovanni. Online Clustering of Bandits. In *Proceedings of the 31st International Conference on Machine Learning (ICML)*, 2014.

Jain, Prateek, Tewari, Ambuj, and Kar, Purushottam. On Iterative Hard Thresholding Methods for High-dimensional M-Estimation. In *Proceedings of the 28th Annual Conference on Neural Information Processing Systems (NIPS)*, 2014.

Kakade, S. and Tewari, A. On the Generalization Ability of Online Strongly Convex Programming Algorithms. In *Proceedings of the 22nd Annual Conference on Neural Information Processing Systems (NIPS)*, 2008.

Korda, Nathan, Szorenyi, Balazs, and Li, Shuai. Distributed Clustering of Linear Bandits in Peer to Peer Networks. In *Proceedings of the 33rd International Conference on Machine Learning (ICML)*, 2016.

Krause, Andreas and Ong, Cheng Soon. Contextual Gaussian Process Bandit Optimization. In *Proceedings of the 25th Annual Conference on Neural Information Processing Systems (NIPS)*, 2011.

Li, Lihong, Chu, Wei, Langford, John, and Schapire, Robert. A Contextual-Bandit Approach to Personalized

News Article Recommendation. In *Proceedings of the 19th International World Wide Web Conference (WWW)*, 2010.

Li, Shuai, Karatzoglou, Alexandros, and Gentile, Claudio. Collaborative Filtering Bandits. In *39th Annual International ACM-SIGIR Conference on Research and Development in Information Retrieval (SIGIR)*, 2016.

Maillard, Odalric-Ambrym and Mannor, Shie. Latent Bandits. In *Proceedings of the 31st International Conference on Machine Learning (ICML)*, 2014.

Needell, Deanna and Tropp, Joel A. CoSaMP: Iterative Signal Recovery from Incomplete and Inaccurate Samples. *Applied Computational Harmonic Analysis*, 26: 301–321, 2008.

Nguyen, Trong and Lauw, Hady. Dynamic Clustering of Contextual Multi-Armed Bandits. In *Proceedings of the 23rd ACM International Conference on Information and Knowledge Management (CIKM)*, 2014.

Pilaszy, Istvan and Tikk, Domonkos. Recommending New Movies: Even a Few Ratings Are More Valuable Than Metadata. In *Proceedings of the 3rd ACM Conference on Recommender Systems (RecSys)*, 2009.

Sutskever, Ilya, Salakhutdinov, Ruslan, and Tenenbaum, Joshua. Modelling Relational Data using Bayesian Clustered Tensor Factorization. In *Proceedings of the 23rd Annual Conference on Neural Information Processing Systems (NIPS)*, 2009.

Tropp, Joel A. Freedman's Inequality for Matrix Martingales. *Electronic Communications in Probability*, 16: 262–270, 2011.

Wu, Qingyun, Wang, Huazheng, Gu, Quanquan, and Wang, Hongning. Contextual Bandits in A Collaborative Environment. In *39th Annual International ACM-SIGIR Conference on Research and Development in Information Retrieval (SIGIR)*, 2016.

Yue, Yisong, Hong, Sue Ann, and Guestrin, Carlos. Hierarchical Exploration for Accelerating Contextual Bandits. In *Proceedings of the 29th International Conference on Machine Learning (ICML)*, 2012.

Zhou, Li and Brunskill, Emma. Latent Contextual Bandits and their Application to Personalized Recommendations for New Users. In *Proceedings of the 25th International Joint Conference on Artificial Intelligence (IJCAI)*, 2016.