# Collaborative Filtering with Low Regret *

Guy Bresler
IDSS/LIDS/EECS, MIT
32 Vassar Street
Cambridge, Massachusetts
guy@mit.edu

Devavrat Shah
IDSS/LIDS/EECS, MIT
32 Vassar Street
Cambridge, Massachusetts
devavrat@mit.edu

Luis F Voloch
LIDS/EECS, MIT
32 Vassar Street
Cambridge, Massachusetts
voloch@mit.edu

## ABSTRACT

There is much empirical evidence that item-item collaborative filtering works well in practice. Motivated to understand this, we provide a framework to design and analyze various recommendation algorithms. The setup amounts to online binary matrix completion, where at each time a random user requests a recommendation and the algorithm chooses an entry to reveal in the user's row. The goal is to minimize regret, or equivalently to maximize the number of $+1$ entries revealed at any time. We analyze an item-item collaborative filtering algorithm that can achieve fundamentally better performance compared to user-user collaborative filtering. The algorithm achieves good "cold-start" performance (appropriately defined) by quickly making good recommendations to new users about whom there is little information.

## 1. INTRODUCTION

### 1.1 Background

Whenever a business contains a large collection of items for sale, it is of interest to help customers find the items that are of most interest to them. Before the creation and widespread adoption of the Internet, this was done by trained store salesmen, who can recommend items based on experience and the customers' revealed preferences.

After the creation of the Internet, this "recommendation system" has been largely taken off the hands of trained salesmen and is now largely handled by automated, statistically driven policies. For many companies, the efficacy of their recommendation systems stands at the core of their business. Amazon and Netflix are prominent examples.

A natural and clever first idea in designing an automated recommendation system is to use content specific data. In this spirit, one may use words in the title and book's cover, or a user's age and geographic location as inputs to recommendation heuristics. This type of recommendation system, based on content-specific data, is called content filtering.

In contrast to content filtering, a technique called collaborative filtering (CF) provides recommendation in a content-agnostic way. CF works by exploiting patterns in general purchase or usage data. For instance, if 90% of users agree on two items (that is, 90% of users either like both items or dislike both items), a CF algorithm may recommend the second item after a user has expressed positive feedback for the first item.

The term collaborative filtering was coined in [16], and this technique is used in virtually all recommendation systems. There are two main paradigms in neighborhood-based collaborative filtering: the user-user paradigm and the item-item paradigm. To recommend to a user in the user-user paradigm, one first looks for similar users, and then recommends items liked by those similar users. In the item-item paradigm, in contrast, items similar to those liked by the user are found and subsequently recommended. Much empirical evidence exists that the item-item paradigm performs well in many cases [26, 23], and in this paper, motivated to understand the reasons behind this, we introduce a mathematical model and formally analyze the performance of a simple, intuitive algorithm that follows the item-item paradigm. The algorithm, called ITEM-ITEM-CF, is described in Section 3.

### 1.2 Organization

The rest of this paper is organized as follows. For the remainder of Section 1 we formally introduce the model, give an informal overview of the main results, and discuss related works. In Section 2 we describe the assumptions we make, motivated by theoretical lower bounds and empirical observations. In Section 3 we describe our algorithm and in Section 4 we prove the correctness of its main set of routines. In Section 5 we put all the pieces together and give our main results pertaining to the performance of ITEM-ITEM-CF. In Section 6 we further discuss our results and future work.

### 1.3 Model

We consider a system with $N$ users and collection of items $\mathcal{I}$. For each item $i \in \mathcal{I}$, user $u$ has binary preference $L_{u,i}$ equal to $+1$ (like) or $-1$ (dislike). Recommendation systems typically operate in an *online*[1] setting, meaning that when a user logs into a virtual store (such as Amazon), a recommendation must be made immediately. At each dis-

*This work was supported in parts by NSF CMMI-1462158, CNS-1523546 and ARO MURI Award W911NF-11-1-0036.



*SIGMETRICS '16, June 14-18, 2016, Antibes Juan-Les-Pins, France*

DOI: http://dx.doi.org/10.1145/2896377.2901469

[1]Online refers to decisions being made sequentially and with limited information.

crete time step $t = 1, 2, 3, \ldots$ a uniformly random user $U_t \in \{1, .., N\}$ requests a recommendation. The recommendation algorithm selects an item $I_t$ to recommend from the set of available items $\mathcal{I}$, after which $U_t$ gives feedback $L_{U_t,I_t}$. The recommendation must depend only on previous feedback: $I_t$ is required to be measurable with respect to the sigma-field generated by the history $(U_1, I_1, L_{U_1,I_1})$, ...,$(U_{t-1}, I_{t-1}, L_{U_{t-1},I_{t-1}})$.

We impose the constraint that the recommendation algorithm may only recommend each item to a given user at most once. This captures the situation where users do not want to watch a movie or read a book more than once and focuses attention on the ability to recommend new items.

For each recommendation, the algorithm may therefore either recommend an item that has been previously recommended to *other* users (in which case it has some information about the item) or recommend a new item from $\mathcal{I}$.

We are interested in the situation where there are many items, and will assume that $\mathcal{I}$ is infinite. For a given item $i$, the corresponding $i$th column $L_{\cdot,i} \in \{-1, +1\}^N$ containing each user's preference is called the *type* of item $i$. It is convenient to represent the population of items by a probability measure $\mu$ over $\{-1, +1\}^N$. When the algorithm selects an item that has not yet been recommended, the item's type is drawn from this distribution in an i.i.d. manner. Recommending a new item corresponds to adding a column to the rating matrix, with binary preferences jointly distributed according to $\mu$.

## 1.4 Performance measure and main results

As is standard in the online decision-making literature, algorithm performance is measured by regret relative to an all-knowing algorithm that makes no bad recommendations. The regret at time $T$ is therefore

$$\mathcal{R}(T) \triangleq \frac{1}{N} \sum_{t=1}^{T \cdot N} \frac{1}{2} \left(1 - L_{U_t,I_t}\right).$$

Recall that at time $t$ user $U_t \in \{1, \ldots, N\}$ desires a recommendation, $I_t$ is the recommended item, and $L_{u,i}$ is equal to $+1$ (resp. $-1$) if $u$ likes (resp. dislikes) item $i$. The regret $\mathcal{R}(T)$ is the number of bad recommendations per user after having made an average of $T$ recommendations per user. Dependence on the algorithm is implicit through $I_t$.

We now describe two high-level objectives in designing a recommendation system and the corresponding guarantees obtained for our proposed algorithm ITEM-ITEM-CF, which is described in Sections 3 and 3.1. The results are stated in more detail in Section 5.

### 1.4.1 Cold-start time

With no prior information, the algorithm should give reliable recommendations as quickly as possible. The *cold-start time* $T_{\mathrm{cold-start}}$ of a recommendation algorithm is defined as

$$\min \left\{ T+\Gamma : T, \Gamma \geq 0, \mathbb{E}\left[\mathcal{R}(T+\Delta) - \mathcal{R}(T)\right] \leq 0.1\Delta, \forall \Delta > \Gamma \right\}. \tag{1}$$

This is the first time after which the slope of the expected regret is bounded by 0.1: after $T_{\mathrm{cold-start}}$ the algorithm makes a bad recommendation to a randomly chosen user with probability at most 0.1.[2]

[2]The choice 0.1 is arbitrary. We will assume that users like only a small fraction of the items, so the cold-start time is the minimum time after which the algorithm can recommend significantly better than random.

**Our results:** As described in Section 2, we assume that each user likes at least $\nu > 0$ fraction of the items. In Theorem 5.1 we show that algorithm ITEM-ITEM-CF achieves $T_{\mathrm{cold-start}} = \widetilde{\mathcal{O}}(\frac{1}{\nu})$ for $N \geq N_0$.[3] Note that one must typically randomly sample $\Omega(\frac{1}{\nu})$ items to find a *single* liked item, and our results show that this amount of time-investment suffices in order to give consistently good recommendations. Further, in Section 6.2 we show that user-user collaborative filtering cannot achieve such cold-start performance, and give intuition for why that is the case.

### 1.4.2 Improving accuracy

The algorithm should give increasingly more reliable recommendations as it gains information about the users and items. This is captured by having *sublinear expected regret* $\mathbb{E}\left[\mathcal{R}(T)\right] = o(T)$.

**Our results:** Proposition 2.1 shows that without assumptions on the item space, it is impossible for any online algorithm to achieve sublinear regret for any positive length of time. In this paper we assume that the item space has doubling dimension (a measure of complexity of the space, defined and motivated later) bounded by $d$. In Theorem 5.2 we show that after time $T_{\mathrm{cold-start}}$ (until which we incur linear regret), algorithm ITEM-ITEM-CF achieves sublinear expected regret $\widetilde{\mathcal{O}}(T^{\frac{d+1}{d+2}})$ up until a certain time $T_{\max}$. After $T_{\max}$ the expected regret again grows linearly (but with much smaller slope), and this behavior is shown in Theorem 5.3 to be unavoidable. As will be made explicit, performance improves with increasing number of users: $T_{\max}$ (and hence the length of the sublinear time-period) increases with $N$ and the eventual linear slope decreases with $N$, both of which illustrate the so-called collaborative gain.

We would like to note that the mathematical formulation of cold-start time is new, to the best of our knowledge. The strong guarantee we obtain on cold-start time (independent of doubling dimension $d$) is distinct from and does not follow as an implication of the sub-linear regret result (which *does* depend on $d$).

## 1.5 Related Works

In this section we discuss connections to related work. We begin by describing some relevant literature in multi-armed bandits, which deals with explore-exploit trade-offs similar to the ones faced by a recommendation algorithm. We then, for completeness, describe matrix factorization methods of recommendation systems, which is another popular model of recommendation system. We conclude by discussing other relevant works.

### 1.5.1 Multi-armed Bandits

In the multi-armed bandit problem, at each time $t$ the player must choose an arm $i \in \mathcal{X}$ (hence multi-armed) to pull in a slot machine (bandit). Upon pulling arm $i$, the player receives a random reward distributed according to an i.i.d. random variable with mean $r_i$.

The goal of the player is to minimize the expected regret,

[3]Here and throughout, $r = \widetilde{\mathcal{O}}(x)$ means $r \leq Cx \log^c x$ for some numerical constants $c$ and $C$.

which in this context is defined as

$$\bar{\mathcal{R}}(T) = \mathbb{E}\left[\sum_{t=1}^{T} r^* - r_{I_t}\right]. \quad (2)$$

Here $I_t$ is the random variable denoting the arm pulled at time $t$ and $r^* \triangleq \sup_{i\in\mathcal{X}} r_i$ is the supremum of expected reward of all arms. The means $\{r_i\}$ are unknown to the player, so the decision about which arm to pull depends on estimates of the means. Hence, the expected regret is the difference in expected reward compared with an oracle algorithm that always pulls the arm with the highest expected reward.

The survey by [6] thoroughly covers many other variants of multi-armed bandits, and points to [33] as the earliest work in bandits. The multi-armed bandit work that is most relevant to us, however, is when $\mathcal{X}$ is a very large set. In this case we can interpret arms as items, and pulling an arm as recommending an item to a user. When $\mathcal{X}$ is very large (possibly uncountably infinite), however, some structure on $\mathcal{X}$ must be assumed in order for any algorithm to achieve nontrivial regret.

The papers [22, 7] use notions of dimensionality similar to the one in this paper in order to control the structure of the space of arms $\mathcal{X}$: [22] assumes that the arm space is endowed with a metric, and [7] assumes that the arms have a dissimilarity function (which is not necessarily a metric). The expected rewards are then related to this geometry of the arms. In the former work, the difference in expected rewards is Lipschitz[4] in the distance, and in the latter work the dissimilarity function constrains the slope of the reward around its maxima. In contrast, *what is Lipschitz about our setting?* For us, the difference in reward is Lipschitz when averaged over all users. That is

$$\mathbb{E}_u\Big[|L_{u,i} - L_{u,j}|\Big] \le 2 \cdot \gamma_{i,j}, \quad (3)$$

where $\gamma_{ij}$ is a distance between items that will be defined in Section 2.

The expected regret upper bound of the algorithms in [22, 7] is $\widetilde{\mathcal{O}}\big(T^{\frac{d'+1}{d'+2}}\big)$, where $d'$ is a weaker notion (than the one we use) of the covering number of $\mathcal{X}$, and is closely related to the doubling dimension (which we define later) in the case of a metric. The regret bound in Theorem 5.2 for the sublinear regime is of the same form, but two important aspects of our model require a different algorithm and more intricate arguments: $(i)$ in our case, no repeat recommendations (i.e. pulling the same arm) can be made to the same user, and $(ii)$ we do not have an oracle for distances between users and items, and instead we must estimate distances by making carefully chosen exploratory recommendations.

Aside from these differences, the nature of the collaborative filtering problem leads to additional novelty relative to existing work on multi-armed bandits. First, we formalize the cold-start problem and prove strong guarantees in this regard. Second, all of our bounds are in terms of system parameters. This allows, for example, to see the role of the number of users $N$ as an important resource allowing for collaboration.

[4]That is, $|\mathbb{E}[r_i] - \mathbb{E}[r_j]| \le C \cdot d(i,j)$, where $d$ is the distance between arms, and $C$ is a constant.

### 1.5.2 Matrix Factorization

Besides neighborhood-based methods of collaborative filtering, matrix factorization is another branch of collaborative filtering that is widely used in practice. The winning team in the Netflix Challenge, for instance, used an algorithm based on matrix factorization, and due to its importance we will briefly cover some aspects of it here[5].

Using the language of recommendations, the set-up for matrix factorization is that we have a (possibly incomplete) $n \times m$ matrix $L$ of ratings (where rows represent users and columns represent rows) that we would like to factor as $L = AW$, where $A$ is $n \times r$ and $W$ is $r \times m$, and $r$ is as small as possible. When $L$ is complete, the singular value decomposition is known to provide the factorization.When $L$ is incomplete, however, a popular approach in practice (cf. [24]) is to find the decomposition by solving the regularized optimization problem

$$\arg\min_{A,W} \sum_{(u,i)\ \text{observed}} \Big(L_{u,i} - A_u^T W_i\Big) + \lambda\left(||A_u||^2 + ||W_i||^2\right) \quad (4)$$

via, for instance, stochastic gradient descent (cf. [37]). This, in effect, is looking for a sparse low-rank factorization of $L$.

In many contexts, including that of recommendations, it also makes sense to want a nonnegative factorization. Intuitively, this enforces that we interpret each item as being composed of different attributes to various extents (rather than as a difference of attributes). [25] provided theoretical guarantees for fixed $r$, and [34] then showed that this problem is NP-hard with $r$ as a variable. The first provable guarantee under nontrivial conditions came only later in [2]. They give an algorithm that runs in time that is polynomial in the $m$, $n$, and the inner rank, granted that $L$ satisfies the so-called separability condition introduced by [14][6].

In Section 2.4 we compare low-rank structure with low doubling dimension. Specifically, we argue that there are simple scenarios where the rank of a matrix is high, but its doubling dimension is small.

### 1.5.3 Other Related Works

The papers [19] and [10] on online learning and matrix completion are also relevant. In their case, however, the matrix entries to be predicted are not chosen by the algorithm and hence there is no explore-exploit trade-off. The paper [21] considers collaborative filtering under a mixture model in the *offline* setting, and they make separation assumptions between the item types (called genres in their paper). The work [11] considers a setting similar to ours (but with finite number of user and item types) and proves certain guarantees on a moving horizon approximation rather than the cumulative anytime regret. The paper [4] proves asymptotic consistency guarantees on estimating the ratings of unrecommended items. The recent paper [27] considers a different model in which repeat recommendations are also not allowed, but they make recommendations by exploiting existing information about users' interests.

It is possible that using the similarities between users, and not just between items as we do, is also useful. This

[5]As noted in [24], matrix factorization techniques have some advantages over neighborhood-based methods, such as the ease of combining it with content-specific data and of including implicit feedback.

[6][30] surveys of all these and more results.

has been studied theoretically in the user-user collaborative filtering framework in [5], via bandits in a wide variety of settings (for instance [1, 32, 9]), with focus on benefits to the cold-start problem [15, 8], and in practice (cf. [12, 3]). In this paper, in order to capture the power of purely item-item collaborative filtering, we intentionally avoid using any user-user similarities.

A latent source model of user types is used by [5] to give performance guarantees for user-user collaborative filtering. The assumptions on users and items are closely related since $K$ items types induce at most $2^K$ user types and vice versa (the $K$ item types liked by a user fully identify the user's preferences, and there are at most $2^K$ such choices). Since we study algorithms that cluster similar items together, in this paper we assume a latent structure of items. We note that unlike the standard mixture model with minimum separation between mixture components (as assumed in [5]), our setup does not have any such gap condition. In contrast, we allow an effectively arbitrary model, and we prove performance guarantees based on a notion of dimensionality of the item space.

# 2. STRUCTURE IN DATA

The main intuition behind all variants of collaborative filtering is that users and items can typically be clustered in a meaningful way even ignoring context specific data. Items, for example, can often be grouped into a few different types that tend to be liked by the same users (see fig. 1). It is with this intuition and empirical observation in mind that the two main paradigms in neighborhood-based collaborative filtering, user-user and item-item, operate.

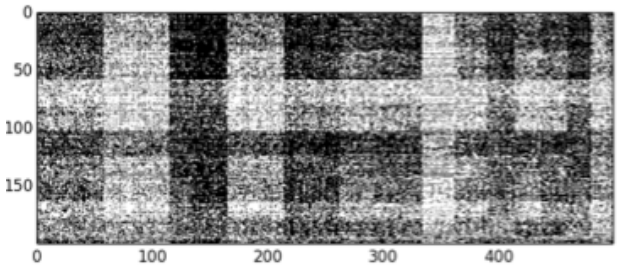


Figure 1: Observed clustering of users and items (from [5]). This is the densest subset of users (rows) and items (columns), where the darker spots indicate likes and the lighter spots indicate dislikes. One can see that the items and users can be grouped in relatively few types, where items of the same type tend to be liked by users of the same type.

In this section we will develop the appropriate notions to capture structure in data. First, in Section 2.1 we show that without structural assumptions, the expected regret of any online algorithm grows linearly with time. In Section 2.2, we then develop our intuition for the structure present in data and define a distance between item types. In Section 2.3 we then define the precise structural assumption that we will make: that the item space has finite doubling dimension. Finally, in Section 2.4 we give an example of how to relate the concepts of doubling dimension and low rank.

## 2.1 Need for Structure

As discussed, a good recommendation algorithm suggests items to users that are liked, but have not been recommended to them before. In order to motivate the need for assumptions on the item space, we begin by stating the intuitive result that in the worst case when $\mu$ has little structure, no online algorithm can do better than recommending random items.

PROPOSITION 2.1 (LOWER BOUND). *Under the uniform distribution $\mu$ over $\{-1,+1\}^N$, for all $T \geq 1$, the expected regret satisfies*

$$\mathcal{R}(T) \geq T/2$$

*for any online recommendation algorithm. Conversely, the algorithm that recommends a random item at each time step achieves $\mathcal{R}(T) = T/2$.*

Proposition 2.1 states that no online algorithm can have sublinear regret for any period of time unless some structural assumptions are made. Hence, to have any *collaborative gain* we need to capture the fact that items tend to come in clusters of similar items. We make two assumptions.

(**A1**) The distribution $\mu$ over the item space has doubling dimension at most $d$ for a given $d \geq 0$.

(**A2**) Each user likes a random item drawn from $\mu$ with probability between $\nu$ and $2\nu$, and each item is liked by a fraction between $\nu$ and $2\nu$ of the users, for a given $\nu \in (0, 1/4)$.

Assumption **A1** captures structure in the item space through the notion of doubling dimension, defined and motivated in Section 2.3. Assumption **A2** is made to avoid the extreme situations where almost no items are liked (in which case recommendation is impossible) or most items are liked (in which case the regret benchmark becomes meaningless).

## 2.2 Item Types

To more formally describe item types and the conditions on $\mu$, let us first define a distance between the item types. We endow the $N$-dimensional Hamming cube $\{-1,+1\}^N$ with the normalized Hamming metric: for any two item types $x, y \in \{-1,+1\}^N$, define their distance

$$\gamma_{x,y} \equiv \gamma(x,y) \stackrel{\triangle}{=} \frac{1}{N}\sum_{k=1}^{N}\frac{1}{2}|x_k - y_k|.$$

We see that $\gamma_{x,y}$ is the fraction of users that disagree on item types $x$ and $y$. Since each item has a unique type, we write $\gamma_{ij}$ for items $i$ and $j$ to denote the distance between their types.

One can plausibly go about capturing the empirical observation that items tend to belong to clusters that are liked by similar set of users by assuming that there are $K$ different item types, i.e., the measure $\mu$ assigns positive mass to only $K$ vectors in $\{-1,+1\}^N$. [5] assumes similar structure, but over users rather than items. In addition, they assume that the types are well-separated (that is, $\gamma_{x,y}$ is lower bounded for each two different types $x$ and $y$ with positive mass). This allows for clustering perfectly with high probability, which in turn leads to a small regret.

However, enforcing that the types are well-separated is counter-intuitive and not necessary for the following reason. If two types $x$ and $y$ are extremely close to each other, that should only make the problem easier: In our setting, two

similar item types have (by definition) the same 'like' or 'dislike' from most users, so for the purpose of recommendation the distinction between type of $x$ and $y$ is insignificant.

It turns out that we can exploit this intuition when the item space has sufficient structure, as captured by a certain notion of dimensionality.

## 2.3 Doubling Dimension

In order to capture this intuition that our mixture assumption should $(i)$ give significant mass around item types and $(ii)$ not have separation assumptions, we define a notion of doubling dimension of $\mu$, and then further discuss its advantages. Let $\mathcal{B}(x,r) = \{y \in \{-1,+1\}^N : \gamma_{x,y} \leq r\}$ be the ball of radius $r$ around $x$ with respect to metric $\gamma$[7].

DEFINITION 2.1. *(Doubling Dimension) A distribution $\mu$ on $\{-1,+1\}^N$ is said to have dimension $d$ if $d$ is the least number such that for each $x \in \{-1,+1\}^N$ with $\mu(x) > 0$ we have*

$$\sup_{r>0} \frac{\mu(\mathcal{B}(x,2r))}{\mu(\mathcal{B}(x,r))} \leq 2^d. \tag{5}$$

*Further, a measure that has finite doubling dimension is called a doubling measure.*

The above definition is a natural adaptation to probability measures on metric spaces of the well-known notion of doubling dimension for metric spaces (cf. [20, 18, 13]). As noted in, for instance [13], this is equivalent to enforcing that $\mu(B_\gamma(x,\alpha r)) \leq \alpha^d \cdot \mu(B_\gamma(x,r))$ for any $r > 0$ and any $x \in \{-1,+1\}^N$ with $\mu(x) > 0$. For Euclidean spaces, the doubling dimension coincides with the ambient dimension, which reinforces the intuition that metric spaces of low doubling dimension have properties of low dimensional Euclidean spaces.

Despite its simplicity, measures of low doubling dimension capture the observed clustering phenomena. Proposition 2.2 below, which follows directly from the definition, shows that a small doubling dimension ensures that the balls around any item type must have a significant mass.

PROPOSITION 2.2. *Let $\mu$ be an item space for $N$ users with doubling dimension $d$. Then for any item type $x \in \{-1,1\}^N$ with $\mu(x) > 0$ we have*

$$\mu\big(\mathcal{B}\,(x,r)\,\big) \geq r^d.$$

Doubling measures also induce many other nice properties on the item space. To that end, let us first define an $\varepsilon$-net for an item space.

DEFINITION 2.1 ($\varepsilon$-NET). *For any $\varepsilon > 0$, a collection of items $\mathcal{C}$ is called an $\varepsilon$-net of the item space represented by distribution $\mu$ on $\{-1,+1\}^N$ if (a) for any pair $i,j \in \mathcal{C}$, we have $\gamma_{ij} > \varepsilon/2$, and (b) for any item $\ell$ with $\mu(\ell) > 0$, there exists $i \in \mathcal{C}$ so that $\gamma_{i\ell} \leq \varepsilon$.*

Proposition 2.3 below shows that an $\varepsilon$-net of items can only have few items close to any given item.

PROPOSITION 2.3. *Let $\mu$ be an item space for $N$ users with doubling dimension $d$ and let $\mathcal{C}$ be an $\varepsilon$-net for $\mu$. For $j$ an arbitrary item, let $c_j \in \mathcal{C}$ be such that $\gamma_{j,c_j} < \varepsilon$, and let $m_{c_j} \triangleq \mu\big(\mathcal{B}(c_j,\varepsilon)\big)$. Then, for each $r \in [\varepsilon/2, 1/2]$, there are at most $m_{c_j}\left(\frac{4}{\varepsilon}\right)^d\left(\frac{4r+5\varepsilon}{4\varepsilon}\right)^d$ items in $\mathcal{C}$ within radius $r$ of $j$.*

[7]We omit the dependence of the ball on $\gamma$ throughout.

To further illustrate and gain intuition for doubling dimension, let us consider a simple item space with $K$ clusters.

EXAMPLE 2.1. *Consider an item space $\mu$ over $N$ users that assigns probability at least $w > 0$ to $K$ distinct item types with separation at least $\sigma > 0$. Then, since $\mu(\mathcal{B}(x,\alpha)) \leq 1$, and $\mu(\mathcal{B}(x,\alpha/2)) \geq w$, we have that*

$$d = \log_2\left(\max_{x:\mu(x)>0} \sup_r \frac{\mu(\mathcal{B}(x,r))}{\mu(\mathcal{B}(x,r/2))}\right) \leq \log_2\left(\frac{1}{w}\right).$$

*Similarly, if we only know that there are at most $K$ equally likely item types we can bound the doubling dimension as*

$$d = \log_2\left(\max_{x:\mu(x)>0} \sup_r \frac{\mu(\mathcal{B}(x,r))}{\mu(\mathcal{B}(x,r/2))}\right) \leq \log_2 K. \tag{6}$$

With the example above in mind, we would like to emphasize that doubling dimension assumptions are strictly more general than the style of assumptions made in [5] (finite $K$ with separation assumptions) because $(a)$ doubling measures require no separation assumptions (that is, two item types $x$ and $y$ that are arbitrarily close to each other can have positive mass) and $(b)$ the number of types of positive mass is not bounded by a finite $K$ anymore, but instead can grow with the number of users.

Finally, note that doubling dimension is not only a proof technique: it can be estimated from data and tends to be small in practice. To illustrate this point, we calculated the doubling dimension on the Jester Jokes Dataset[8] and for the MovieLens 1M Dataset[9]. For the MovieLens dataset we considered only movies that have been rated by at least 750 users (to ensure some density).

In both datasets we calculated the empirical doubling dimension $d_i$ (that is, the smallest $d_i$ such that $\mu(\mathcal{B}(i,2r)) \leq 2^{d_i}\mu(\mathcal{B}(i,r))$ for each $r$) around each item $i$. Under a simple noise assumption, figs. 2 and 3 show that all the $d_i$ tend to be small. The appendix B describes the precise experiments.

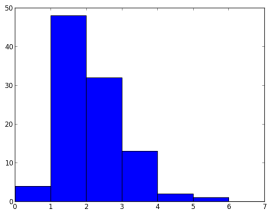


Figure 2: Jester Doubling Dimensions

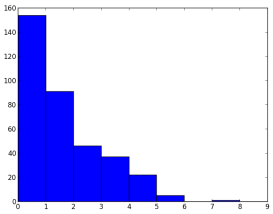


Figure 3: MovieLens Doubling Dimensions

## 2.4 Low Rank and Doubling Dimension

As mentioned in Section 1.5, a common assumption behind matrix factorization methods is that the matrix has low rank. In this section we would like to draw a connection between rank properties of the rating matrix $L$ and the doubling dimension of the item space *induced* by $L$. In particular, we will show that low doubling dimension can be a weaker requirement than low rank of $L$. To that end, we

[8][17], and data available on http://goldberg.berkeley.edu/jester-data/

[9][31], and data available on http://grouplens.org/datasets/movielens/

will first define what we mean by the item space induced by a rating matrix.

DEFINITION 2.2. *(Item Space Induced by Rating Matrix) Let $L$ be an $N \times M$ binary rating matrix. Then the item space $\mu$ induced by $L$ is*

$$\mu(x) = \frac{1}{M}\sum_{j=1}^{M} \mathbf{1}_{L_j = x}, \text{ for each } x \in \{-1,+1\}^N,$$

*where $L_j$ is the $j^{th}$ column of $L$.*

That is, the item space induced by a rating matrix assigns mass to item types according to the empirical frequency of the item type in $L$.

The example below shows that there are rating matrices of high binary rank, but whose corresponding doubling dimension is constant (at most 2). Consider the $N \times N$ matrix $L_N$ whose $j^{th}$ column's first $j$ entries are $+1$ and the remaining $-1$. That is, each of its columns differ from its adjacent columns in exactly one entry. For instance, for $N = 4$ we have

$$L_4 = \begin{pmatrix} +1 & +1 & +1 & +1 \\ -1 & +1 & +1 & +1 \\ -1 & -1 & +1 & +1 \\ -1 & -1 & -1 & +1 \end{pmatrix}.$$

The matrix $L_N$ clearly has rank $N$. However, the doubling dimension of its induced item space is at most 2. We can see this because for each $x$ and $r \in \{0, ..., N-1\}$

$$\frac{1+r}{N} \le \mu\left(\mathcal{B}\left(x, \frac{r}{N}\right)\right) \le \frac{1+2r}{N},$$

which we can use in turn to conclude that

$$d \le \max_x \max_r \log_2 \left(\frac{\mu\left(\mathcal{B}\left(x, \frac{2r}{N}\right)\right)}{\mu\left(\mathcal{B}\left(x, \frac{r}{N}\right)\right)}\right) \le \log_2 \left(\frac{\frac{1+4r}{N}}{\frac{1+r}{N}}\right) \le 2.$$

Hence, the doubling dimension of the induced item space can be substantially smaller than the rank of the rating matrix. Since the rank is a way of counting the number of item types, this reinforces the fact that *the number of types is not particularly important, but the geometric structure between them is.*

# 3. ITEM-ITEM COLLABORATIVE FILTERING ALGORITHM

In this section we describe our algorithm, ITEM-ITEM-CF. The algorithm carries out a certain procedure over increasingly longer epochs (blocks of time), where the epoch index is denoted by $\tau \ge 1$. In each epoch the algorithm carefully balances *Explore* and *Exploit* steps.

In the Explore steps of epoch $\tau$, a partition $\{P_k^{(\tau+1)}\}$ of a set of items is created for use in the subsequent epoch. Each epoch has a target precision $\varepsilon_\tau$ (specified below) such that if two items $i$ and $j$ are in the same block $P_k^{(\tau+1)}$, then usually $\gamma_{ij} \le \varepsilon_{\tau+1}$.

In the Exploit steps of epoch $\tau$, the partition $\{P_k^{(\tau)}\}$ created in the previous epoch is used for recommendation. Exploit recommendations to a user $u$ are made as follows: $u$ samples a random item $i$ from a random block $P_k^{(\tau)}$, and if $u$ likes $i$ ($L_{u,i} = +1$) then the rest of $P_k^{(\tau)}$ is recommended to $u$ in subsequent Exploit steps. After all items in $P_k^{(\tau)}$ have been recommended to $u$, the user repeats the process by sampling random items in random blocks until liking some item $j$ in $P_{k'}^{(\tau)}$, upon which the rest of $P_{k'}^{(\tau)}$ is recommended.

In the first epoch there is no possibility of exploiting a partition created in a previous epoch, so the algorithm begins with a purely exploratory "cold-start" period. The pseudocode of the algorithm is as follows.

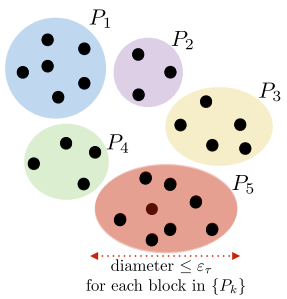


Figure 4: Partition resulting from exploration in the previous epoch

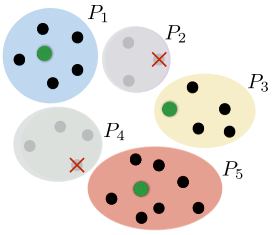


Figure 5: During exploitation, items in $P_k$ are recommended only when from a block when a user likes an item in the block

ITEM-ITEM-CF($N$)

1 **Algorithm parameters:**
$\varepsilon_N = \left(\frac{2^{5d+18}}{\nu} \cdot 630(2d+11)(d+2)^4 \frac{1}{N}\right)^{\frac{1}{d+5}}$, $C = \frac{\nu}{148}\frac{1}{20}$
$\varepsilon_\tau = \max\left(\frac{1}{2^\tau}, \varepsilon_N\right) \cdot C$, for $\tau \ge 1$ (target accuracy for epoch)
$M_\tau = \frac{2^{\max(3.5d,8)}}{\nu} \frac{(3d+1)}{\varepsilon_\tau^{d+2}} \ln(\frac{2}{\varepsilon_\tau})$, for $\tau \ge 1$ (items per epoch)
$D_\tau = \frac{\nu}{2} M_\tau$, for $\tau \ge 1$ (duration of epoch)
2 **Cold-Start:**
$\{P_k^{(1)}\}$ = MAKE-PARTITION$(M_1, \varepsilon_1, \varepsilon_1)$
3 **Subsequent epochs:**
**for** $\tau \ge 1$
**do for** $t = 1$ to $N \cdot D_\tau$
**do** $U_t$ = random user
w.p. $1 - \varepsilon_\tau$: *exploit* $\{P_k^{(\tau)}\}$ to recommend an item to $U_t$
w.p. $\varepsilon_\tau$: $U_t$ *explores* to help construct partition $\{P_k^{(\tau+1)}\}$ (see Section 3.1)

## 3.1 Explore: making a partition

Recall that during epoch $\tau$ the goal of the explore recommendations is to create a partition $\{P_k^{(\tau+1)}\}$ of items such that whenever $i, j \in P_k^{(\tau+1)}$ then $\gamma_{ij} \le \varepsilon_{\tau+1}$. We later prove that this can be done by executing the routine MAKE-PARTITION$(M_{\tau+1}, \varepsilon_{\tau+1}, \varepsilon_{\tau+1})$ described below, which at any point makes recommendations to a randomly chosen user. Hence, given the random user making the recommendation, ITEM-ITEM-CF provides explore recommendations in whatever order MAKE-PARTITION would have recommended (had it been run sequentially)[10].

[10]For instance, suppose that time $t$ is the first in some epoch $\tau$. We might have that times $t+5$ and $t+30$ are the first two explore recommendations of the epoch, then for those two recommendations the algorithm makes whatever the first

MAKE-PARTITION first finds a *net* $\mathcal{C}$ for the item space (using the subroutine GET-NET described later). To each item in the net there is associated a block in a partition. $M$ randomly sampled items are assigned to the blocks as follows: for each sampled item $j$, an item $i \in \mathcal{C}$ is found that is similar to $j$, and $j$ is assigned to the partition block $P_i$ (if there is more than one item $i$ similar to $j$, the algorithm chooses among the relevant blocks at random). Finally, the algorithm breaks up large blocks into blocks of size on the order of $1/\varepsilon$. This guarantees that there will be many blocks in the partition, which turns out to be important in Theorem 5.1 showing brief cold-start time [11].

MAKE-PARTITION$(M, \varepsilon, \delta)$

$\mathcal{C} =$ GET-NET$(\varepsilon/2, \delta/2)$
$\mathcal{M} = M$ randomly drawn items from item space
**for** each $i \in \mathcal{C}$, let $P_i = \varnothing$
**for** each $j \in \mathcal{M}$, let $S_j = \{i \in \mathcal{C} \mid \text{SIMILAR}(i, j, 0.6\varepsilon, \frac{\delta}{4M|\mathcal{C}|})\}$
**if** $|S_j| > 0$ **then** $P_i = P_i \cup \{j\}$, for $i$ chosen u.a.r. from $S_j$
**for** each $i$, **if** $|P_i| > 1/\varepsilon$, **then** partition $P_i$ into blocks of size $\geq \frac{1}{2\varepsilon}$ and $\leq 1/\varepsilon$
**return** $\{P_i\}$

The subroutine SIMILAR is used in MAKE-PARTITION; it determines whether most users have the same preference (like or dislike) for two given items $i$ and $j$.This is accomplished by sampling many random users and counting the number of disagreements on the two items.

SIMILAR$(i, j, \varepsilon, \delta)$

$q_{\varepsilon,\delta} = \lceil 630 \frac{d+1}{\varepsilon} \ln\left(\frac{1}{\delta}\right) \rceil$
**for** $n = 1$ **to** $q_{\varepsilon,\delta}$
    **do** sample a uniformly random user $u$
        let $X_u = \mathbf{1}_{L_{u,i} \neq L_{u,j}}$
**if** $\frac{1}{q_{\varepsilon,\delta}} \sum_{\text{sampled } u} X_u \geq 0.9\varepsilon$ **then**
        **return** FALSE
**return** TRUE

The subroutine GET-NET below is a natural greedy procedure for constructing an $\varepsilon$-net. Given parameters $\varepsilon$ and $\delta$, it finds a set of items $\mathcal{C}$ that is an $\varepsilon$-net for $\mu$ with probability at least $1 - \delta$ (proven in the appendix). It does so by keeping a set of items $\mathcal{C}$ and whenever it samples an item $i$ that currently has no similar item in $\mathcal{C}$, it adds $i$ to $\mathcal{C}$.

GET-NET$(\varepsilon, \delta)$

$\mathcal{C} = \varnothing$, COUNT $= 0$, MAX-SIZE $= (4/\varepsilon)^d$
MAX-WAIT $= (\frac{5}{\varepsilon})^d \ln\left(\frac{2 \cdot \text{MAX-SIZE}}{\delta}\right)$, $\delta' = \frac{\delta}{4 \cdot \text{MAX-WAIT} \cdot \text{MAX-SIZE}^2}$
**while** COUNT $\leq$ MAX-WAIT and $|\mathcal{C}| <$ MAX-SIZE
    **do** draw item $i$ from $\mu$
        **if** SIMILAR$(i, j, \varepsilon, \delta')$ for any $j \in \mathcal{C}$ **then**
            COUNT = COUNT + 1
        **else** $\mathcal{C} = \mathcal{C} \cup i$, COUNT $= 0$
**return** $\mathcal{C}$

two recommendations would have been in MAKE-PARTITION. If the execution of MAKE-PARTITION has finished, the algorithm resorts to an exploit recommendation instead.

[11] It is crucial that blocks in the partition are not too small because we would like the reward for exploration to be large when a user finds a likable item. Although the algorithm does not explicitly ensure that blocks are not too small (as it did in ensuring the blocks are not too large) it comes as a byproduct of a property proven in Proposition 2.3: there are not many items in the net close to any given item.

# 4. CORRECTNESS OF EXPLORE

This section establishes correctness of the explore procedure as well as some of its properties that will be utilized for establishing the main result of the paper. Concretely, we will prove that with high probability the procedure MAKE-PARTITION produces a partition of similar items during each epoch. To that end, in Section 4.1, we prove that SIMILAR succeeds in deciding whether two items are close to each other. In Section 4.2 we prove that the procedure GET-NET succeeds in finding a set of items that is an $\varepsilon$-net for $\mu$. We then put all the pieces together and prove that MAKE-PARTITION, the routine at which the explore recommendations are aimed at completing, succeeds in creating a partition of similar items. Finally, in Section 4.3 we prove that with high probability during any given epoch there will be enough explore recommendations.

## 4.1 Guarantees for SIMILAR

The procedure SIMILAR is used throughout GET-NET and MAKE-PARTITION, and it is aimed at testing whether two items are approximately $\varepsilon$-close to each other. Lemma 4.1 below shows that given two items $i$ and $j$, SIMILAR indeed succeeds in determining that the items are similar when $\gamma_{i,j} \leq 0.8\varepsilon$, and that the items are not similar when $\gamma_{i,j} \geq \varepsilon$.

LEMMA 4.1. *Let $i$ and $j$ be arbitrary items, $\delta, \varepsilon \in (0, 1)$, and $S_{i,j}$ be the event that* SIMILAR$(i, j, \varepsilon, \delta)$ *returns* TRUE. *Then we have that*

*(i) if $\gamma_{i,j} \leq 0.8\varepsilon$, then $\mathbb{P}(S_{ij}) \geq 1 - \delta$, and*

*(ii) if $\gamma_{ij} \in [k\varepsilon, (k+1)\varepsilon)$ where $k \in \{1, ..., \lfloor \frac{1}{\varepsilon} \rfloor\}$, then $\mathbb{P}(S_{ij}) \leq \frac{\delta}{4}\left(\frac{1}{4k}\right)^d \frac{1}{k^2}$.*

The lemma above bounds the probability of false positive and missed detection for deciding whether or not two items are similar. Further, the lemma states that the probability of a false-positive decreases quickly as the items get further apart. Lemma 4.2 below shows that, when one of the items is drawn from $\mu$, SIMILAR still works and that the false positive rate is small, despite the possibility that it may be much more likely to draw an item that is far from $i$. Lemma 4.2 uses the doubling dimension of $\mu$ for the first time, and in this context the doubling dimension guarantees that SIMILAR (which is a random projection) preserves relative distances.

LEMMA 4.2. *Let $i$ be an arbitrary item, let $J$ be a randomly drawn item from an item space $\mu$ of doubling dimension $d$, and let $S_{iJ}$ be the event that* SIMILAR$(i, J, \varepsilon, \delta)$ *returns* TRUE. *Then we have $\mathbb{P}(\gamma_{iJ} \geq \varepsilon \mid S_{iJ}) \leq \delta$.*

PROOF. By Bayes' rule we get

$$\mathbb{P}(\gamma_{i,J} \geq \varepsilon \mid S_{iJ}) = \frac{\mathbb{P}(\gamma_{i,J} \geq \varepsilon, S_{iJ})}{\mathbb{P}(\gamma_{i,J} \geq \varepsilon, S_{iJ}) + \mathbb{P}(\gamma_{i,J} < \varepsilon, S_{iJ})}, \tag{7}$$

where the probability is with the respect to the random choice of $J$ and the random users in SIMILAR. Now if

$$\delta \mathbb{P}(S_{iJ}, \gamma_{iJ} < \varepsilon) \geq (1 - \delta)\mathbb{P}(S_{iJ}, \gamma_{iJ} \geq \varepsilon) \qquad (\star)$$

holds, we get

$$\mathbb{P}(\gamma_{i,J} \geq \varepsilon \mid S_{iJ}) = \frac{1}{1 + \frac{\mathbb{P}(\gamma_{i,J} < \varepsilon, S_{iJ})}{\mathbb{P}(\gamma_{i,J} \geq \varepsilon, S_{iJ})}} \leq \frac{1}{1 + \frac{1-\delta}{\delta}} = \delta. \tag{8}$$

Hence, it suffices to show $(\star)$. Recall that $\mathcal{B}(i,r)$ is the ball of radius $r$ centered at $i$. Note that

$$\mathbb{P}(\gamma_{iJ} < \varepsilon, S_{iJ}) \geq \mathbb{P}(S_{iJ} \mid \gamma_{iJ} \leq \varepsilon/2)\, \mu(\mathcal{B}(i,\varepsilon/2)),$$

and

$$\begin{aligned}
&\mathbb{P}(\gamma_{iJ} \geq \varepsilon, S_{iJ}) \\
&= \sum_{k=0}^{\lceil \log_2(\frac{1}{\varepsilon}) \rceil} \mathbb{P}\left(S_{iJ} \mid \gamma_{iJ} \in [2^k\varepsilon, 2^{k+1}\varepsilon)\right) \\
&\qquad \times \mu\left(\mathcal{B}\left(i, 2^{k+1}\varepsilon\right) \setminus \mathcal{B}\left(i, 2^k\varepsilon\right)\right) \\
&\leq \sum_{k=0}^{\lceil \log_2(\frac{1}{\varepsilon}) \rceil} \mathbb{P}\left(S_{iJ} \mid \gamma_{iJ} \in [2^k\varepsilon, 2^{k+1}\varepsilon)\right) \mu\left(\mathcal{B}\left(i, 2^{k+1}\varepsilon\right)\right).
\end{aligned}$$

Let us first lower bound $\mathbb{P}(S_{iJ} \mid \gamma_{ij} \leq \varepsilon/2)\, \mu(\mathcal{B}(i,\varepsilon/2))$. Letting $p \triangleq \mu(\mathcal{B}(i,\varepsilon/2))$, Lemma 4.1 gives

$$\mathbb{P}(S_{iJ} \mid \gamma_{iJ} \leq \varepsilon/2)\, \mu(\mathcal{B}(i,\varepsilon/2)) \geq (1-\delta)\, p. \tag{9}$$

We will now upper bound $\mathbb{P}(S_{iJ}, \gamma_{iJ} \geq \varepsilon)$. Using the doubling dimension of the item space, which implies that $\mu\left(\mathcal{B}_\gamma\left(i, 2^{k+1}\varepsilon\right)\right) \leq \left(2^{k+2}\right)^d p$, we have that

$$\begin{aligned}
&\mathbb{P}(\gamma_{iJ} \geq \varepsilon, S_{iJ}) \\
&\leq \sum_{k=0}^{\lceil \log_2(\frac{1}{\varepsilon}) \rceil} \mathbb{P}\left(S_{iJ} \mid \gamma_{iJ} \in [2^k\varepsilon, 2^{k+1}\varepsilon)\right) \mu\left(\mathcal{B}_\gamma\left(i, 2^{k+1}\varepsilon\right)\right) \\
&\leq \sum_{k=0}^{\lceil \log_2(\frac{1}{\varepsilon}) \rceil} \mathbb{P}\left(S_{iJ} \mid \gamma_{iJ} \in [2^k\varepsilon, 2^{k+1}\varepsilon)\right) \left(2^{k+2}\right)^d p.
\end{aligned}$$

We now use the second half of Lemma 4.1, and arrive at

$$\mathbb{P}(\gamma_{iJ} \geq \varepsilon, S_{iJ}) \leq \sum_{k=0}^{\lceil \log_2(\frac{1}{\varepsilon}) \rceil} \left(\frac{\delta}{4}\left(\frac{1}{4\cdot 2^k}\right)^d \frac{1}{2^{2k}}\right)\left(2^{k+2}\right)^d p, \tag{10}$$

which in turn is at most $p\frac{\delta}{4}\sum_{k=0}^{\infty}\frac{1}{2^{2k}} \leq p\frac{\delta}{2}$. We can now check that the sufficient condition from eq. $(\star)$ is satisfied:

$$\begin{aligned}
\delta\mathbb{P}(S_{iJ}, \gamma_{iJ} < \varepsilon) \geq \delta p(1-\delta) &\geq \frac{\delta}{2} p(1-\delta) \\
&\geq (1-\delta)\,\mathbb{P}(S_{iJ}, \gamma_{iJ} \geq \varepsilon),
\end{aligned}$$

which completes the proof. □

## 4.2 Making the Partition

In the previous section we proved that the procedure SIMILAR works well in deciding whether two items are similar to each other at some desired precision. In this section, we will prove that with SIMILAR as a building block we can partition items into blocks of similar items.

We will begin with a lemma showing that the subroutine GET-NET, used in the beginning of MAKE-PARTITION, succeeds at producing an $\varepsilon$-net of items with high probability. The proof is omitted due to space constraints.

LEMMA 4.3. *With probability at least $1-\delta$ the routine* GET-NET$(M,\varepsilon,\delta)$ *returns an $\varepsilon$-net for $\mu$ that contains at most $\left(\frac{4}{\varepsilon}\right)^d$ items.*

It is now only left to prove that the main tool used during exploration, MAKE-PARTITION, indeed produces a partition of similar items. This is done in Lemma 4.4 below. The additional properties stated in the Lemma regarding size of the blocks will be crucial later in ensuring good cold-start performance.

LEMMA 4.4. *Let $\varepsilon, \delta \in (0,1)$, and let $M$ be at least $12 \cdot \left(\frac{12}{\varepsilon}\right)^{d+1} \ln\left(\frac{2}{\delta}\left(\frac{8}{\varepsilon}\right)^d\right)$. Then with probability at least $1-\delta$ the subroutine* MAKE-PARTITION$(M,\varepsilon,\delta)$ *returns a partition $\{P_k\}$ of a subset of $M$ randomly drawn items such that*

*(i) For each block $P_k$ and $i,j \in P_k$ we have $\gamma_{i,j} \leq 1.2\cdot\varepsilon$,*

*(ii) Each block $P_k$ contains at least $\frac{1}{2\varepsilon}$ items,*

*(iii) Each block $P_k$ contains at most $1/\varepsilon$ items,*

*(iv) There are at most $2M\varepsilon$ blocks.*

PROOF. We will show that properties $(i)$ and $(ii)$ hold with probability at least $1-\delta$, and note that $(iii)$ follows directly from the algorithm and that $(iv)$ follows from $(ii)$.

Let $C$ be the event that the set $\mathcal{C}$ returned by GET-NET is not an $\frac{\varepsilon}{2}$-net for $\mu$, and let $\mathcal{M}$ be the set of $M$ items sampled. Let $E_{i,j}$ be the event $\{S_{i,j}, \gamma_{ij} > 0.6\varepsilon\} \cup \{S^c_{i,j}, \gamma_{ij} < 0.5\varepsilon\}$, where $S_{ij}$ is the event that SIMILAR$(i,j,0.6\varepsilon,\delta/(4M|\mathcal{C}|))$ returns TRUE. Intuitively, event $E_{i,j}$ occurs when SIMILAR is incorrect. Furthermore, let $E = \bigcup_{c\in\mathcal{C}}\bigcup_{j\in\mathcal{M}} E_{c,j}$.

Let $F$ be the event that for some block $P_k$ there exists $i,j \in P_k$ such that $\gamma_{ij} > 1.2\varepsilon$, and let $B$ be the event that all blocks in the partition have size at least $\frac{1}{2\varepsilon}$. Hence, $B^c = \bigcup_k B^c_k$, where $B^c_k$ is the even that $P_k$ has size less than $\frac{1}{2\varepsilon}$.

Since event $F$ guarantees condition $(i)$ and event $B$ guarantees condition $(ii)$ it suffices to show that

$$\mathbb{P}(F^c \cup B^c) \leq \delta.$$

We will do so by conditioning on $C^c$ and $E^c$ where after a couple of union bounds we arrive at

$$\mathbb{P}(F^c \cup B^c) \leq \mathbb{P}(F^c \mid C^c, E^c) + \mathbb{P}(B^c \mid C^c, E^c) + \mathbb{P}(C) + \mathbb{P}(E).$$

Now showing $(i)\ \mathbb{P}(B^c \mid C^c, E^c) \leq \delta/2$, $(ii)\ \mathbb{P}(F^c \mid C^c, E^c) = 0$, $(iii)\ \mathbb{P}(E) \leq \delta/4$, and $(iv)\ \mathbb{P}(C) \leq \delta/4$ (of which we omit the proof) completes the argument. □

## 4.3 Sufficient Exploration

Any given recommendation in epoch $\tau$ is used for exploration with probability $\varepsilon_\tau$. In the lemma below, we show that during each epoch there are enough Explore recommendations for the procedure MAKE-PARTITION to terminate.

LEMMA 4.5. *With probability at least $1-\varepsilon_{\tau+1}$, during the $\tau^{th}$ epoch the algorithm has enough* explore *recommendations for* MAKE-PARTITION $(M_{\tau+1}, \varepsilon_{\tau+1}, \varepsilon_{\tau+1})$ *to terminate.*

# 5. REGRET ANALYSIS

In this section we prove the main results of the paper. In Section 5.1 we prove the Quick Recommendations Lemma, which is the main lemma that will that will be used in Theorems 5.1 and 5.2, which are proved in Section 5.2.

## 5.1 Quick Recommendations Lemma

In Section 3 we described the algorithm, which starts recommending items to a user as soon as it knows one item that the user likes. Below we show that indeed shortly after the beginning of the epoch the slope of the regret is small.

LEMMA 5.1 (QUICK RECOMMENDATIONS LEMMA). *For $\tau \geq 1$ let $\mathcal{R}^{(\tau)}(T) = \frac{1}{N}\sum_{t=T_\tau}^{T_\tau+TN}\frac{1}{2}(1-L_{U_t,I_t})$ denote the number of bad recommendations made to users during the first $TN$ recommendations of epoch $\tau$. Then we have*

$$\mathbb{E}\left[\mathcal{R}^{(\tau)}(T)\right] \leq \frac{148\varepsilon_\tau}{\nu}T \tag{11}$$

*whenever $T \in [T_{\min,\tau}, D_\tau]$ and where $T_{\min,\tau} \triangleq \frac{12}{\varepsilon_\tau}\ln(\frac{1}{\varepsilon_\tau})$. For $T < T_{\min,\tau}$, we trivially have $\mathbb{E}\left[\mathcal{R}^{(\tau)}(T)\right] \leq T$.*

PROOF. Let $\mathcal{R}'^{(\tau)}(T)$ denote the number of bad recommendations made to users during the first $TN$ *exploit* recommendations of epoch $\tau$. Then, since the expected number of explore recommendations by time $TN$ of epoch $\tau$ is $\varepsilon_\tau TN$, we get that

$$\mathbb{E}\left[\mathcal{R}^{(\tau)}(T)\right] \leq \varepsilon_\tau T + \mathbb{E}\left[\mathcal{R}'^{(\tau)}(T)\right].$$

Furthermore, as described in the algorithm, during epoch $\tau-1$ the algorithm spends a small fraction of the recommendations (in the explore part) to create a partition $\{P_k\}$ (which we call $\{P_k^{(\tau)}\}$ in the pseudocode) of $M_\tau$ random items to be exploited during epoch $\tau$. Let $\mathcal{E}_\tau$ be the event that the partition $\{P_k^{(\tau)}\}$ to be used during epoch $\tau$ satisfies the conditions specified in Lemma 4.4 (with $M = M_\tau$ $\varepsilon = \varepsilon_\tau$, $\delta = \varepsilon_\tau$). Then we get

$$\mathbb{E}\left[\mathcal{R}^{(\tau)}(T)\right] \leq \varepsilon_\tau T + \mathbb{E}\left[\mathcal{R}'^{(\tau)}(T)\right] \leq 2\varepsilon_\tau T + \mathbb{E}\left[\mathcal{R}'^{(\tau)}(T) \mid \mathcal{E}_\tau\right],$$

where the last inequality is due to Lemma 4.4, which guarantees that $\mathbb{P}(\mathcal{E}_\tau) \leq \varepsilon_\tau$.

For the rest of the proof, we will show that $\mathbb{E}\left[\mathcal{R}'^{(\tau)}(T)\right]$ is at most $\frac{45}{\nu}\varepsilon_\tau T$. We will do so by first rewriting this in terms of the number of bad exploit recommendations to each user

$$\mathbb{E}\left[\mathcal{R}'^{(\tau)} \mid \mathcal{E}_\tau\right] \leq \frac{1}{N}\mathbb{E}\left[\sum_u \mathcal{R}_u'^{(\tau)}(T) \mid \mathcal{E}_\tau\right],$$

where $\mathcal{R}_u'^{(\tau)}(T)$ is the number of bad recommendations made to user $u$ during the first $TN$ exploit recommendations of epoch $\tau$. We will now bound the latter term by conditioning on a nice property of users (denoted by $g_{u,T}$), and showing that this property holds for most users. Let $g_{u,T}$ be the event that user $u$ has tried at most $16\frac{T}{D_\tau}P^{(\tau)}$ blocks during the first $TN$ recommendations of epoch $\tau$ (we omit $\tau$ in the notation of $g_{u,T}$ since it is clear from the context here). Here we use notation $P^{(\tau)} = |\{P_k^{(\tau)}\}|$ to denote the total number of blocks in the partition for epoch $\tau$. Then we get that $\frac{1}{N}\mathbb{E}\left[\sum_u \mathcal{R}_u'^{(\tau)}(T) \mid \mathcal{E}_\tau\right]$ is at most

$$\frac{1}{N}\sum_u \mathbb{E}\left[\mathcal{R}_u'^{(\tau)}(T) \mid \mathcal{E}_\tau, g_{u,T}\right] + T\frac{1}{N}\sum_u \mathbb{P}\left(g_{u,T}^c \mid \mathcal{E}_\tau\right).$$

We dedicate Lemma 5.3 to showing that $\frac{1}{N}\sum_u \mathbb{P}\left(g_{u,T}^c \mid \mathcal{E}_\tau\right) \leq \frac{42}{\nu}\varepsilon_\tau$. Hence, it suffices to show that for each $T > T_{\min,\tau}$ we have

$$\frac{1}{N}\sum_u \mathbb{E}\left[\mathcal{R}_u'^{(\tau)}(T) \mid \mathcal{E}_\tau, g_{u,T}\right] \leq \frac{104}{\nu}\varepsilon_\tau T, \qquad (\bigstar)$$

which we prove now. We will first rewrite the regret by summing over the number of bad recommendations due to each of the blocks as

$$\sum_u \mathbb{E}\left[\mathcal{R}_u'^{(\tau)}(T) \mid \mathcal{E}_\tau, g_{u,T}\right] = \sum_u \mathbb{E}\left[\sum_k W_{u,k,T} \mid \mathcal{E}_\tau, g_{u,T}\right],$$

where $W_{u,k,T}$ is the random variable denoting the number of bad exploit recommendations to user $u$ from block $P_k$ among the first $TN$ exploit recommendations of epoch $\tau$. We can further rewrite to get that

$$\begin{aligned}&\frac{1}{N}\sum_u \mathbb{E}\left[\sum_k W_{u,k,T} \mid \mathcal{E}_\tau, g_{u,T}\right] \qquad (12)\\ &\leq \frac{1}{N}\sum_u\sum_k \mathbb{E}\left[W_{u,k,T} \mid \mathcal{E}_\tau, g_{u,T}, s_{u,k,T}\right]\mathbb{P}\left(s_{u,k,T} \mid \mathcal{E}_\tau, g_{u,T}\right),\end{aligned}$$

where $s_{u,k,T}$ denotes the event that by time $T$ user $u$ has sampled an item from block $P_k$. Here we used the fact that $W_{u,k,T}$ is identically equal to zero on $s_{u,k,T}^c$ because the user hasn't sampled an item from the block, so the expectation $\mathbb{E}\left[W_{u,k,T} \mid \mathcal{E}_\tau, g_{u,T}, s_{u,k,T}^c\right]$ is also zero.

Now note that by conditioning on $g_{u,T}$, we know that user $u$ has sampled at most $16\frac{T}{D_\tau}P^{(\tau)}$ blocks. Now given $g_{u,T}$ as well as $\mathcal{E}_\tau$, the indices of the sampled blocks are not revealed. Let $K$ be a random variable that selects one of the indices of the blocks uniformly at random. Then, it follows that with respect to randomness in $K$,

$$\mathbb{P}\left(s_{u,K,T} \mid \mathcal{E}_\tau, g_{u,T}\right) \leq \frac{16\frac{T}{D_\tau}P^{(\tau)}}{P^{(\tau)}} = 16\cdot\frac{T}{D_\tau}.$$

We can re-write (12) in this notation and apply the above discussed bound to obtain

$$\begin{aligned}&\frac{1}{N}\sum_u \mathbb{E}\left[\sum_k W_{u,k,T} \mid \mathcal{E}_\tau, g_{u,T}\right]\\ &\leq \frac{P^{(\tau)}}{N}\sum_u \mathbb{E}\left[W_{u,K,T} \mid \mathcal{E}_\tau, g_{u,T}, s_{u,K,T}\right]\mathbb{P}\left(s_{u,K,T} \mid \mathcal{E}_\tau, g_{u,T}\right)\\ &\leq \frac{P^{(\tau)}}{N}\sum_u \mathbb{E}\left[W_{u,K,T} \mid \mathcal{E}_\tau, g_{u,T}, s_{u,K,T}\right]\frac{16T}{D_\tau}\\ &= \frac{16T}{ND_\tau}\sum_u\sum_k \mathbb{E}\left[W_{u,k,T} \mid \mathcal{E}_\tau, g_{u,T}, s_{u,k,T}\right].\end{aligned}$$

The last quantity above can be bounded as

$$\underbrace{\leq}_{Lemma\ 5.2}\ \frac{16T}{ND_\tau}\Big(\sum_k(1+1.2\varepsilon_\tau|P_k^{(\tau)}|)N\Big)$$

$$\underbrace{\leq}_{Lemma\ 4.4}\ 52\varepsilon_\tau T\frac{M_\tau}{D_\tau}\overbrace{=}^{D_\tau=\frac{\nu}{2}M_\tau}\frac{104}{\nu}\varepsilon_\tau T.$$

The first inequality follows from Lemma 5.2 by realizing that each block, $P_k^{(\tau)}$ corresponds to a collection of items such that for any $i,j \in P_k^{(\tau)}$, we have $\gamma_{ij} < 1.2\varepsilon_\tau$. The second inequality can be justified as: from Lemma 4.4, $1/2\varepsilon_\tau \leq |P_k^{(\tau)}| \leq 1/\varepsilon_\tau$ and hence

$$\sum_k(1+1.2\varepsilon_\tau|P_k^{(\tau)}|) \leq \sum_k(3.2\varepsilon_\tau|P_k^{(\tau)}|) \leq 3.2\varepsilon_\tau\sum_k|P_k^{(\tau)}|,$$

which is $3.2\varepsilon_\tau M_\tau$. This, along with some arithmetic, completes the proof of eq. ($\bigstar$) and hence the lemma. □

The lemma below was used in the proof of Lemma 5.1. Informally, it says that our recommendation policy, which recommends the whole block to a user after the user likes an item in the block, succeeds in finding most likable items to recommend and in not recommending many bad items.

LEMMA 5.2 (PARTITION LEMMA). *Let $P_k$ be a set of items such that for each $i, j \in P_k$ we have $\gamma_{ij} < \varepsilon$, and consider the usual recommendation policy that* ITEM-ITEM-CF *uses during its "exploit" steps (where when user $u$ samples a random item $i \in_R P_k$, only if $u$ likes $i$ will $u$ be recommended the remaining items). Let $s_{u,k}$ be the event that user $u$ has sampled an item from $P_k$, let $W_{u,k}$ (W for wrong) denote the number of wrong recommendations made to $u$ from $P_k$, and let $A_{u,k}$ (A for absent) denote the number of items in $P_k$ that $u$ likes that are not recommended to $u$. Then we have*

$$\sum_u \mathbb{E}\left[A_{u,k} + W_{u,k} \mid s_{u,k}\right] \leq \big(1 + \varepsilon|P_k|\big)N.$$

PROOF. For each block $P_k$ and user $u$, let $\ell_{u,k} = |\{i \in P_k \mid L_{u,i} = +1\}|$ denote the number of items in $P_k$ that $u$ likes. Note that $\mathbb{E}\left[A_{u,k} \mid s_{u,k}\right] = \ell_{u,k} \cdot \frac{\left(|P_k| - \ell_{u,k}\right)}{|P_k|}$ and $\mathbb{E}\left[W_{u,k} \mid s_{u,k}\right] = \left(|P_k| - \ell_{u,k}\right) \cdot \frac{\ell_{u,k}}{|P_k|} + 1 \cdot \frac{\left(|P_k| - \ell_{u,k}\right)}{|P_k|}$. This is because with probability $\ell_{u,k}/|P_k|$ user $u$ will sample an item from $P_k$ that $u$ likes and will then be recommended $(|P_k| - \ell_{u,k})$ bad items, and with probability $(|P_k| - \ell_{u,k})/|P_k|$ the first (and thus only) item from $P_k$ recommended to $u$ is bad. Likewise, with probability $\left(|P_k| - \ell_{u,k}\right)/|P_k|$ the user will sample an item that the user dislikes, and then fail to be recommended $\ell_{u,k}$ items that the user likes. Hence we have that $\mathbb{E}\left[\sum_u A_{u,k} + W_{u,k} \mid s_{u,k}\right]$ is equal to

$$\underbrace{\sum_u \frac{\left(|P_k| - \ell_{u,k}\right)}{|P_k|}}_{\leq N} + 2\sum_u \frac{\ell_{u,k}\left(|P_k| - \ell_{u,k}\right)}{|P_k|}.$$

Now,

$$2\sum_u \frac{\ell_{u,k}\left(|P_k| - \ell_{u,k}\right)}{|P_k|} \overbrace{=}^{\text{by definition}} \frac{2}{|P_k|}\sum_u \sum_{i,j \in P_k} \mathbf{1}_{L_{u,i} \neq L_{u,j}}$$

$$= \frac{2}{|P_k|}\sum_{i,j \in P_k} \underbrace{\sum_u \mathbf{1}_{L_{u,i} \neq L_{u,j}}}_{=\gamma_{ij} \cdot N} \underbrace{\leq}_{\gamma_{ij} < \varepsilon} \frac{2}{|P_k|}\sum_{i,j \in P_k} \varepsilon N$$

$$\leq \frac{2}{|P_k|}\binom{|P_k|}{2}\varepsilon N \leq \varepsilon|P_k|N. \tag{13}$$

Putting it all together we get $\mathbb{E}\left[\sum_u A_{u,k} + W_{u,k} \mid s_{u,k}\right]$ is equal to

$$\underbrace{\sum_u \frac{\left(|P_k| - \ell_{u,k}\right)}{|P_k|}}_{\leq N} + \underbrace{2\sum_u \frac{\ell_{u,k}\left(|P_k| - \ell_{u,k}\right)}{|P_k|}}_{\varepsilon|P_k|N} \leq \big(\varepsilon|P_k| + 1\big)N. \quad \square$$

The lemma below was also needed in the proof of Lemma 5.1.

LEMMA 5.3 (AUXILIARY CLAIM). *Consider an arbitrary epoch $\tau$, and let $g_{u,T}$ be the event that by the $(TN)^{th}$ exploit recommendation of epoch $\tau$ user $u$ has tried at most $16\frac{T}{D_\tau}|\{P_k^{(\tau)}\}|$ blocks from the partition $\{P_k^{(\tau)}\}$ constructed during the* MAKE-PARTITION$(M_\tau, \varepsilon_\tau, \varepsilon_\tau)$ *of the previous epoch, and let $\mathcal{E}_\tau$ be the event that $\{P_k^{(\tau)}\}$ satisfies the conditions specified in Lemma 4.4. Then*

$$\frac{1}{N}\sum_u \mathbb{P}\left(g^c_{u,T} \mid \mathcal{E}_\tau\right) \leq \frac{42}{\nu}\varepsilon_\tau$$

*holds for any $T \in \left(\frac{12}{\varepsilon_\tau}\ln(1/\varepsilon_\tau), D_\tau\right]$.*

PROOF. Let us make a few definitions. Let $N_{u,T}$ be the event that by the $TN^{th}$ exploit recommendation user $u$ has been recommended at most $1.1T$ items, *and* that $u$ likes at least $0.9\nu M_\tau$ among the items in $\{P_k^{(\tau)}\}$. Also, let $H_{u,T}$ be the event that by the $TN^{th}$ exploit recommendation there are still at least $\frac{\nu}{5}M_\tau$ items liked by $u$ in blocks that haven't been sampled by $u$. Then we get $\mathbb{P}\left(g^c_{u,T} \mid \mathcal{E}_\tau\right)$ is at most

$$\mathbb{P}\left(g^c_{u,T} \mid H_{u,T}, N_{u,T}, \mathcal{E}_\tau\right) + \mathbb{P}\left(H^c_{u,T} \mid N_{u,T}, \mathcal{E}_\tau\right) + \mathbb{P}\left(N^c_{u,T} \mid \mathcal{E}_\tau\right).$$

By showing that for $T \in \left(\frac{12}{\varepsilon_\tau}\ln(1/\varepsilon_\tau), D_\tau\right]$ the following points (of which we omit the proofs) hold the lemma follows:
$(A)$ $\mathbb{P}\left(g^c_{u,T} \mid H_{u,T}, N_{u,T}, \mathcal{E}_\tau\right) \leq \varepsilon_\tau$,
$(B)$ $\frac{1}{N}\sum_u \mathbb{P}\left(H^c_{u,T} \mid N_{u,T}, \mathcal{E}_\tau\right) \leq \frac{40}{\nu}\varepsilon_\tau$,
$(C)$ $\mathbb{P}\left(N^c_{u,T} \mid \mathcal{E}_\tau\right) \leq \varepsilon_\tau$. $\square$

## 5.2 Main Results

In Section 3 we described how ITEM-ITEM-CF starts recommending items to a user as soon as it finds one item that the user likes. This leads to a short cold-start time. We are now ready to bound the cold-start time of ITEM-ITEM-CF (the proof is presented in the Appendix).

THEOREM 5.1 (COLD-START PERFORMANCE). *Suppose assumptions* **A1** *and* **A2** *are satisfied. Then the algorithm* ITEM-ITEM-CF *has cold-start time $T_{\text{cold}-\text{start}} = \frac{f(\nu,d)}{N} + \widetilde{\mathcal{O}}(1/\nu)$.*

It follows that the algorithm ITEM-ITEM-CF has cold-start time $\widetilde{\mathcal{O}}(1/\nu)$ for $N$ sufficiently large. This differs from the results of [5] for the user-user paradigm, where the cold-start time increases with user space complexity and the effect is not counteracted with more users present. (Section 6.2 contains a more in-depth discussion.)

The next result shows that after the cold-start period and until a time $T_{\max}$, the expected regret is sublinear.

THEOREM 5.2 (UPPER BOUND ON REGRET ). *Suppose assumptions* **A1** *and* **A2** *are satisfied. Then the algorithm* ITEM-ITEM-CF *achieves expected regret for $T_{\min} < T \leq T_{\max}$ given by*

$$\mathbb{E}\left[\mathcal{R}(T)\right] \leq T_{\min} + \alpha(\nu, d) \cdot \left(T - T_{\min}\right)^{\frac{d+1}{d+2}} \log_2(T - T_{\min}),$$

*and for $T \geq T_{\max}$*

$$\mathbb{E}\left[\mathcal{R}(T)\right] \leq \beta + \varepsilon_N\left(T - T_{\max}\right).$$

*Here $T_{\min} = \widetilde{\mathcal{O}}\left(\frac{1}{\nu}\right) + \frac{f(d,\nu)}{N}$, $T_{\max} = g(\nu, d)N^{\frac{d+2}{d+5}}$, $\varepsilon_{N,d,\nu} = h(d,\nu)\left(\frac{1}{N}\right)^{\frac{1}{d+5}}$, and*

$$\beta = T_{\min} + \alpha(\nu, d) \cdot \left(T_{\max} - T_{\min}\right)^{\frac{d+1}{d+2}} \log_2(T_{\max} - T_{\min}).$$

The reader is directed to the proof (presented in the Appendix) for the exact constants. Also note that $T_{\max}$ increases with $N$ and the asymptotic slope $\varepsilon_N$ decreases as a function with $N$, both of which illustrate the so-called collaboration gain. Note that the regret bound in Theorem 5.2

has an asymptotic linear regime. The next result (whose proof is in the Appendix) shows that with a finite number of users such linear regret is unavoidable.

THEOREM 5.3 (LINEAR REGRET IS UNAVOIDABLE). *Let $\mu$ be an item space satisfying assumptions* **A1** *and* **A2**. *Then any online algorithm must have expected asymptotic regret $\mathbb{E}[\mathcal{R}(T)] \geq C(\nu, N) \cdot T$, where $C(\nu, N) = (1 - 2\nu)/N$.*

# 6. CONCLUSION

In this section we further discuss our results and give suggestions for future work in the subject.

## 6.1 Discussion

In this paper we provided a formal expected regret analysis of ITEM-ITEM-CF, a simple recommendation algorithm following the item-item paradigm in collaborative filtering. We first proved that unless some structural assumption is made, no online recommendation algorithm can have sublinear expected for any period of time.

We then motivated using the doubling dimension $d$ of the item space as a measure of structure in the data, and showed that the algorithm achieves expected regret $\tilde{\mathcal{O}}\left(T^{\frac{d+1}{d+2}}\right)$ for a period of time that increases with the number of users. Furthermore, we proved that the asymptotic linear regime following the sublinear regime is unavoidable.

## 6.2 Comparison with user-user CF

In this section we will contrast the cold-start performance of user-user collaborative filtering to that of our item-item algorithm. In particular, we give a heuristic argument showing that the cold-start time for user-user algorithms grows with the complexity of the user space. This is in contrast to our Theorem 5.1, where for any doubling dimension of the item space, if there are sufficiently many users then the cold-start time is independent of system complexity.

We consider a simple scenario with $K$ user clusters. First, let $\widetilde{\gamma}_{uv}$ denote the probability that users $u$ and $v$ agree on an item randomly drawn from the item space. We have $K$ equally sized clusters of users, such that $\widetilde{\gamma}_{uv} = 0$ for users $u, v$ in the same cluster, and $\widetilde{\gamma}_{uv} \in (0.1\nu, 0.2\nu)$ for users $u, v$ in different clusters.

Consider now a given user $u$. A user-user algorithm seeks to find another user $v$ who is similar to $u$, so that the items liked by $v$ can be recommended to $u$. In order to recommend with at most (say) 0.1 probability of error, the similar user $v$ should have distance $\widetilde{\gamma}_{uv}$ at most $0.1\nu$. The extra factor $\nu$ is present because inference can only effectively be made from the $\nu$ fraction of liked items.

Concretely, we sample a random user $v$, and attempt to decide if it is from the same cluster as $u$. Suppose $u$ and $v$ have rated $q$ items in common. The problem then reduces to a classical hypothesis test: after observing $q$ items in common from two users, determine whether or not they are from the same cluster. The goal is to understand what is the minimal value of $q$ needed so that the above procedure works with at least probability 1/2.

We consider the maximum a posteriori rule for deciding that $v$ is from $u$'s cluster. If $u$ and $v$ disagree on any single item, then they cannot be from the same cluster. Conversely, if $u$ and $v$ agree on all $q$ sampled items, the MAP rule declares $v$ to be from $u$'s cluster only if

$$\frac{K-1}{K}(1 - 0.2\nu)^q \leq \frac{1}{K}\,.$$

This means that if $q$ is too small, we will never declare $v$ to be from $u$'s cluster and therefore will be unable to make recommendations.

Rearranging gives $q \geq \Omega(\log(K)/\nu)$. Hence, an algorithm based on user similarity needs at least $T = \Omega(\log(K)/\nu)$ steps simply to determine if two users are similar to each other, a prerequisite to making good recommendations. In contrast, we have shown that ITEM-ITEM-CF achieves cold start time $\widetilde{\mathcal{O}}(1/\nu)$, which in particular does not increase with the complexity of the item space.

This contrast between cold-start times highlights the *asymmetry between item-item and user-user* collaborative filtering. The intuition is that it is much faster to compare two items than two users: it takes a long time to make many recommendations to two particular users, but comparing two items can be done in *parallel* by sampling different users.

## 6.3 Two roles of Doubling Dimension

In Section 2.1 we make the assumption that $\mu$ has small doubling dimension. This is crucial in the proofs for two distinct reasons. First, it guarantees that the *$\varepsilon$-net grows slowly* as $\varepsilon$-decreases (polynomially in $1/\varepsilon$). This is important in Lemma 4.4 for ensuring that the blocks of $\varepsilon$ similar items are large enough and the reward for exploration pays off (as when the algorithm finds an item liked by the user, it can now recommend the many other items in the block). It is in this "slowly-growing $\varepsilon$-net" sense that doubling dimension/covering numbers are used, for instance, as [22, 7]. Second, the doubling dimension ensures that SIMILAR, which is a random projection, works with high probability (as proved in Lemma 4.2) in preserving relative distances. It is in this *random projection preserves relative distances* sense that it is used, for instance, in [13].

## 6.4 Future Works

This paper analyzes a collaborative filtering algorithm based on item similarity, and proves guarantees on its regret. Our algorithm exploits structure only in the item space. It would be desirable to have a matching lower bound, in the spirit of the lower bound for multi-armed bandits in metric spaces shown in [22] and [7]. Furthermore, many practitioners use a hybrid of user-user and item-item paradigms [36] and [35], and formally analyzing such algorithms is an open problem.

Finally, the main challenge of the cold-start problem is that initially we do not have any information about item-item similarities. In practice, however, some similarity can be inferred via content specific information. For instance, two books with similar words in the title can have a prior for having a higher similarity than books with no similar words in the title. In practice such hybrid content/collaborative filtering algorithms have had good performance [29]. Formally analyzing such hybrid algorithms has not been done and can shed light onto how to best combine content information with the collaborative filtering information.

# 7. REFERENCES


[1] N. Alon, N. Cesa-Bianchi, C. Gentile, S. Mannor, Y. Mansour, and O. Shamir. Nonstochastic multi-armed bandits with graph-structured feedback. *arXiv Technical Report arXiv:1409.8428*, 2014.

[2] S. Arora, R. Ge, R. Kannan, and A. Moitra. Computing a nonnegative matrix factorization–provably. In *STOC*, 2012.

[3] A. Bellogin and J. Parapar. Using graph partitioning techniques for neighbour selection in user-based collaborative filtering. In *RecSys*, 2012.

[4] G. Biau, B. Cadre, and L. Rouviere. Statistical analysis of k-nearest neighbor collaborative recommendation. *The Annals of Statistics*, 38(3):1568–1592, 2010.

[5] G. Bresler, G. Chen, and D. Shah. A latent source model for online collaborative filtering. In *NIPS*, 2014.

[6] S. Bubeck and N. Cesa-Bianchi. Regret analysis of stochastic and nonstochastic multi-armed bandit problems. *arXiv preprint arXiv:1204.5721*, 2012.

[7] S. Bubeck, R. Munos, G. Stoltz, and C. Szepesvari. X-armed bandits. *JMLR*, 2011.

[8] S. Caron and S. Bhagat. Mixing bandits: A recipe for improved cold-start recommendations in a social network. In *Workshop on Social Network Mining and Analysis*, 2013.

[9] N. Cesa-Bianchi, C. Gentile, and G. Zappella. A gang of bandits. In *NIPS*, 2013.

[10] N. Cesa-Bianchi and O. Shamir. Efficient transductive online learning via randomized rounding. In *Empirical Inference*, pages 177–194. Springer, 2013.

[11] O. Dabeer. Adaptive collaborating filtering: The low noise regime. In *ISIT*, 2013.

[12] A. S. Das, M. Datar, A. Garg, and S. Rajaram. Google news personalization: scalable online collaborative filtering. In *WWW*, 2007.

[13] S. Dasgupta and K. Sinha. Randomized partition trees for nearest neighbor search. *Algorithmica*, 2014.

[14] D. Donoho and V. Stodden. When does non-negative matrix factorization give a correct decomposition into parts? In *NIPS*, 2003.

[15] C. Gentile, S. Li, and G. Zappella. Online clustering of bandits. *ICML*, 2014.

[16] D. Goldberg, D. Nichols, B. M. Oki, and D. Terry. Using collaborative filtering to weave an information tapestry. *Commun. ACM*, 1992.

[17] K. Goldberg, T. Roeder, D. Gupta, and C. Perkins. Eigentaste: A constant time collaborative filtering algorithm. *Information Retrieval*, 2001.

[18] S. Har-Peled and M. Mendel. Fast construction of nets in low-dimensional metrics and their applications. *SIAM Journal on Computing*, 35(5):1148–1184, 2006.

[19] E. Hazan, S. Kale, and S. Shalev-Shwartz. Near-optimal algorithms for online matrix prediction. *COLT*, 2012.

[20] J. Heinonen. *Lectures on analysis on metric spaces*. Springer, 2001.

[21] J. Kleinberg and M. Sandler. Using mixture models for collaborative filtering. In *STOC*, 2004.

[22] R. Kleinberg, A. Slivkins, and E. Upfal. Bandits and experts in metric spaces. *arXiv preprint arXiv:1312.1277*, 2013.

[23] Y. Koren and R. Bell. Advances in collaborative filtering. In *Recommender Systems Handbook*, pages 145–186. Springer US, 2011.

[24] Y. Koren, R. Bell, and C. Volinsky. Matrix factorization techniques for recommender systems. *IEEE*, 2009.

[25] D. D. Lee and H. S. Seung. Algorithms for non-negative matrix factorization. In *NIPS*, 2001.

[26] G. Linden, B. Smith, and J. York. Amazon.com recommendations: Item-to-item collaborative filtering. *IEEE Internet Computing*, 7(1):76–80, 2003.

[27] L. Massoulie, M. Ohannessian, and A. Proutiere. Greedy-bayes approach for targeted news dissemination. *Sigmetrics*, 2015.

[28] C. McDiarmid. Concentration. In *Probabilistic methods for algorithmic discrete mathematics*. Springer, 1998.

[29] P. Melville, R. J. Mooney, and R. Nagarajan. Content-boosted collaborative filtering for improved recommendations. In *AAAI/IAAI*, 2002.

[30] A. Moitra. Algorithmic aspects of machine learning. *Online Manuscript*, 2014.

[31] J. Riedl and J. Konstan. Movielens dataset, 1998.

[32] A. Slivkins. Contextual bandits with similarity information. *The Journal of Machine Learning Research*, 2014.

[33] W. R. Thompson. On the likelihood that one unknown probability exceeds another in view of the evidence of two samples. *Biometrika*, pages 285–294, 1933.

[34] S. A. Vavasis. On the complexity of nonnegative matrix factorization. *SIAM Journal on Optimization*, 2009.

[35] K. Verstrepen and B. Goethals. Unifying nearest neighbors collaborative filtering. In *Recsys*, 2014.

[36] J. Wang, A. De Vries, and M. Reinders. Unifying user-based and item-based collaborative filtering approaches by similarity fusion. In *ACM SIGIR*, 2006.

[37] T. Zhang. Solving large scale linear prediction problems using stochastic gradient descent algorithms. In *ICML*, 2004.

# APPENDIX

# A. PROOFS

## A.1 Proof of Theorem 5.1

First recall the usual definitions: $D_\tau = \frac{\nu}{2} M_\tau$, $T_\tau = T_{MP} + \sum_{\tau' < \tau} D_{\tau'}$, and $T_{MP} = f(\nu, d)/N$, where $f(\nu, d)$ is the number of recommendations required for the initial MAKE-PARTITION call (as stated in Lemma 4.4), and $T_{min,\tau} = \frac{12}{\varepsilon_\tau} \ln\left(\frac{1}{\varepsilon_\tau}\right)$ (as stated in Lemma 5.1). We will show the bound in the definition of cold-start time with $T = T_{MP}$ and $\Gamma = T_{min,1} = \widetilde{\mathcal{O}}(1/\nu)$, which implies $T_{\text{cold}-\text{start}} = T_{MP} + T_{min,1}$.

We shall establish the following two properties:

(*i*) For any $\Delta > 0$, $\mathbb{E}\left[\mathcal{R}(T_{\text{MP}} + T_{\text{min},1} + \Delta) - \mathcal{R}(T_{\text{MP}})\right] \le 0.1(T_{\text{min},1} + \Delta)$, for $T_{\text{MP}} + T_{min,1} + \Delta \le T_2$. This condition says that the desired property holds for times involving the first epoch.

(*ii*) $\mathbb{E}\left[\mathcal{R}(T_\tau + \Delta) - \mathcal{R}(T_\tau)\right] \le 0.05(\Delta + D_{\tau-1})$, for $\Delta \le D_\tau$ and any $\tau \ge 2$.

Before we show how the above two properties imply the desired result, we note that (*i*) follows directly from Lemma 5.1, and (*ii*) will be proved at the end.

Now let us complete the proof using (i) and (ii). Consider a time of the form $T_{\text{cold}-\text{start}} + \Delta = T_{MP} + T_{min,1} + \Delta$, for any $\Delta > 0$. Let $\tau^* \ge 1$ be the epoch to which $T_{MP} + T_{min,1} + \Delta$ belongs, i.e.

$$T_{\tau^*} < T_{MP} + T_{min,1} + \Delta \le T_{\tau^*+1}.$$

Define $t = T_{MP} + T_{min,1} + \Delta - T_{\tau^*} > 0$. We separately deal with $\tau^* = 1$ and $\tau^* > 1$. If $\tau^* = 1$, (i) implies the desired result. For $\tau^* > 1$, we use (ii) as follows:

$$\begin{aligned}
&\mathbb{E}\left[\mathcal{R}\left(T_{\text{MP}} + T_{\text{min},1} + \Delta\right) - \mathcal{R}\left(T_{\text{MP}}\right)\right] \underbrace{\le}_{t = T_{\text{cs}} + \Delta - T_{\tau^*}} \\
&\underbrace{\mathbb{E}\left[\mathcal{R}\left(T_2\right) - \mathcal{R}\left(T_{\text{MP}}\right)\right]}_{\le 0.1 \cdot D_1 \text{ by } (i)} \\
&+ \mathbf{1}_{\tau^* \ge 3} \left( \sum_{\tau=2}^{\tau^*-1} \underbrace{\mathbb{E}\left[\mathcal{R}\left(T_\tau\right) - \mathcal{R}\left(T_{\tau-1}\right)\right]}_{\le 0.05(D_\tau + D_{\tau-1}) \text{ by } (ii)} \right) \\
&+ \overbrace{\mathbb{E}\left[\mathcal{R}\left(T_{\tau^*} + t\right) - \mathcal{R}\left(T_{\tau^*}\right)\right]}^{\le 0.05(t + D_{\tau-1}) \text{ by } (ii)} \\
&\le 0.05 \left( t + 2 \sum_{\tau=1}^{\tau^*-1} D_\tau \right) \\
&\le 0.1 \cdot (\Delta + T_{\text{min},1}).
\end{aligned}$$

This establishes the desired result that $T_{\text{cold}-\text{start}} = T_{MP} + T_{min,1} = f(\nu, d)/N + \widetilde{\mathcal{O}}(1/\nu)$.

*Proof of (ii):* Now we show the remaining property (ii). Lemma 5.1 tells us that for $\Delta \in (T_{min,\tau}, D_\tau)$ we have that $\mathbb{E}\left[\mathcal{R}\left(T_\tau + \Delta\right) - \mathcal{R}\left(T_\tau\right)\right] \le \frac{148}{\nu} \varepsilon_\tau \Delta$, i.e.

$$\mathbb{E}\left[\mathcal{R}\left(T_\tau + T_{min,\tau}\right) - \mathcal{R}\left(T_\tau\right)\right] \le \frac{148}{\nu} \varepsilon_\tau T_{min,\tau}.$$

Thus, for $\Delta < D_\tau$, we have that

$$\mathbb{E}\left[\mathcal{R}\left(T_\tau + \Delta\right) - \mathcal{R}\left(T_\tau\right)\right] \le \frac{148}{\nu} \varepsilon_\tau T_{min,\tau} + \underbrace{\frac{148}{\nu} \varepsilon_\tau \Delta}_{\le 0.05 \text{ for } \tau \ge 2}. \tag{14}$$

In above we used the fact that for $\Delta < T_{min,1}$, $\mathcal{R}\left(T_\tau + \Delta\right) \le \mathcal{R}\left(T_\tau + T_{min,1}\right)$. Using the fact that $T_{min,\tau} \triangleq \frac{12}{\varepsilon_\tau} \ln\left(\frac{1}{\varepsilon_\tau}\right) \le 0.05\left(\frac{300}{\varepsilon_\tau} \ln(\frac{1}{\varepsilon_\tau})\right) = 0.05 D_{\tau-1}$, we conclude

$$\mathbb{E}\left[\mathcal{R}\left(T_\tau + \Delta\right) - \mathcal{R}\left(T_\tau\right)\right] \le 0.05\left(\Delta + D_{\tau-1}\right). \quad \square$$

## A.2 Proof of Theorem 5.2

Recall that ITEM-ITEM-CF starts by running the routine MAKE-PARTITION$(M_1, \varepsilon_1, \varepsilon_1)$. This consumes at most

$$MP(1) \triangleq \left(\frac{8}{\varepsilon_1}\right)^{d+1} 4 \cdot 630 \left(d+1\right)^3 M_1 \ln^2\left(\frac{8}{\varepsilon_1}\right) \ln\left(M_1\right) \tag{15}$$

recommendations (by Lemma 4.4), and hence finishes in at most $T_{MP} \triangleq MP(1)/N$ time steps. For this initial exploratory period $T \le T_{MP}$ we will bound the regret with the trivial bound $\mathcal{R}(T) \le T$.

Let us now deal with the regime between $T_{min}$ and $T_{max}$. Recall that the target $\varepsilon_\tau$ used in the $\tau^{th}$ epoch is decreasing as $\frac{C}{2^\tau}$, until it plateaus at $\varepsilon_N$ when $\frac{C}{2^\tau} \le \varepsilon_N$, where $C = \frac{\nu}{148 \cdot 20}$. Hence $\tau^* \triangleq \lceil \log_2 \frac{C}{\varepsilon_N} \rceil$is the first epoch in which $\varepsilon_N$ is used. For a function $g$ defined later, we will show that

$$T_{MP(1)} + \sum_{\tau'=1}^{\tau^*-1} D_\tau \ge g(\nu, d) N^{\frac{d}{d+5}} \triangleq T_{max}. \tag{16}$$

Now since $\varepsilon_N = \left(\frac{2^{5d+18}}{\nu} \cdot 630(2d+11)(d+2)^4 \frac{1}{N}\right)^{\frac{1}{d+5}}$, we get that

$$\tau^* \ge \frac{1}{d+5} \log_2 \left( \left(\frac{\nu}{148 \cdot 20}\right)^{d+5} \frac{\nu}{630(2d+11)(d+2)^4} \frac{1}{2^{5d+18}} \cdot N \right). \tag{17}$$

Also,

$$T_{MP(1)} + \sum_{\tau=1}^{\tau^*-1} D_\tau \ge \sum_{\tau=1}^{\tau^*-1} D_\tau = \sum_{\tau=1}^{\tau^*-1} \frac{\nu}{2} M_\tau, \tag{18}$$

where we used the fact that $D_\tau = \frac{\nu}{2} M_\tau$. Recall $M_\tau \triangleq C_M \frac{1}{\varepsilon_\tau^{d+2}} \ln(\frac{2}{\varepsilon_\tau})$, where $C_M = \frac{2^{\max(3.5d, 8)}}{\nu}(3d+1)$, and for $\tau \le \tau^*$ we have $\varepsilon_\tau = C/2^\tau$, where $C = \nu/(148 \cdot 20)$. Then we get

$$\begin{aligned}
T_{MP(1)} + \sum_{\tau=1}^{\tau^*-1} D_\tau &\ge \frac{\nu C_M}{2C^{d+2}} \sum_{\tau=1}^{\tau^*-1} 2^{\tau(d+2)} \ge \frac{\nu C_M}{4C^{d+2}} 2^{\tau^*(d+2)} \\
&\ge \underbrace{\frac{\nu C_M}{4C^{d+2}} \left( \frac{\nu}{630(2d+11)(d+2)^4} \frac{1}{2^{5d+18}} \right)^{\frac{d+2}{d+5}}}_{g(\nu,d)} \cdot N^{\frac{d+2}{d+5}} \triangleq T_{max},
\end{aligned} \tag{19}$$

as wished. Hence, between $T_{min}$ and $T_{max}$ the target $\varepsilon_\tau$ for the epochs is indeed halving for each subsequent epoch. Let $\tau(T)$ be the epoch of time $T$. Then, by Lemma 5.1, for $T \in [T_{min}, T_{max}]$, where $T_{min} = T_{MP} + T_{min,1}$, the expected regret satisfies $\mathcal{R}(T) - T_{MP} \le \frac{148}{\nu} \sum_{\tau=1}^{\tau(T)} \varepsilon_\tau D_\tau$,which we can further bound as

$$\begin{aligned}
\mathcal{R}(T) - T_{MP} &\le \frac{C_M}{2} \log_2 \left( \frac{2^{\tau(T)}}{2C} \right) \sum_{\tau=1}^{\tau(T)} 2^{\tau(d+1)} \\
&\le \frac{C_M}{2} \log_2 \left( \frac{2^{\tau(T)}}{2C} \right) 2^{(\tau(T)+1)(d+1)}.
\end{aligned} \tag{20}$$

Now, since for $T > T_{min}$ epoch

$$\tau(T) \leq 1 + \frac{1}{d+2}\log_2\left(\frac{T-T_{MP}}{C_M}\frac{1}{\log(2/C)}\right),$$

we get that

$$\mathcal{R}(T) \leq T_{MP} + \tag{21}$$

$$\frac{C_M}{2}\log_2\left(\frac{T-T_{MP}}{C_M C(d+2)}\frac{1}{\log(2/C)}\right)2^{2(d+1)}2^{(\tau(T)+1)(d+1)} \tag{22}$$

$$\leq T_{MP} + \underbrace{\frac{C_M}{2}\log_2\left(\frac{1}{CC_M(d+2)}\frac{1}{\log(2/C)}\right)2^{4(d+1)}}_{\triangleq C'} \tag{23}$$

$$\cdot \log_2(T-T_{MP})\, 2^{\frac{d+1}{d+2}\log_2\left(\frac{T-T_{MP}}{C_M}\frac{1}{\log(2/C)}\right)} \tag{24}$$

$$\leq T_{MP} + \underbrace{C'\left(\frac{1}{C_M}\frac{1}{\log(2/C)}\right)^{\frac{d+1}{d+2}}}_{\triangleq \alpha(\nu,d)}(T-T_{MP})^{\frac{d+1}{d+2}}\log_2(T-T_{MP}),$$

as we wished, which completes the proof of the sublinear regret regime.

The case $T > T_{max}$ now follows. Recall that by Lemma 5.1 we get $\mathcal{R}(T) \leq T_{MP} + \frac{148}{\nu}\sum_{\tau=1}^{\tau(T)}\varepsilon_\tau D_\tau$,which we can in turn split between before $T_{max}$ and after $T_{max}$ as

$$\mathcal{R}(T) \leq T_{MP} + \frac{148}{\nu}\sum_{\tau=1}^{\tau^*-1}\varepsilon_\tau D_\tau + \frac{148}{\nu}\sum_{\tau=\tau^*}^{\tau(T)}\varepsilon_\tau D_\tau \tag{25}$$

$$\leq \underbrace{T_{MP} + \alpha(\nu,d)T_{max}^{\frac{d+1}{d+2}}\log_2(T_{max})}_{\triangleq \beta} + \varepsilon_N(T-T_{max}), \tag{26}$$

as claimed, and where the last inequality is due to the sublinear regime proved above. □

## A.3 Proof of Theorem 5.3

Let $\{i_1, ..., i_{k_T}\}$ be the set of distinct items that have been recommended up to time $TN$. Then we have

$$\mathbb{E}[\mathcal{R}(T)] = \frac{1}{N}\mathbb{E}\left[\sum_{t=1}^{TN}\frac{1}{2}(1-L_{U_t,I_t})\right]$$
$$= \frac{1}{N}\mathbb{E}\left[\sum_{k=1}^{k_T}\sum_{t=1}^{TN}\frac{1}{2}\mathbf{1}_{I_t=i_k}(1-L_{U_t,i_k})\right]$$
$$\geq \frac{1}{N}\mathbb{E}\left[\sum_{k=1}^{k_T}\frac{1}{2}(1-L_{U_{T_k},i_k})\right],$$

where $T_k$ is the first time in which the item $i_k$ is recommended to any user. Now note that for each $k$ by $(A_2)$ we have that $\mathbb{E}\left[\frac{1}{2}(1-L_{U_{T_k},i_k})\right] \geq 1-2\nu$, since when we have no prior information about $i_k$ the best we can do is to recommend it to the user that likes the largest fraction of items. Hence we get

$$\mathbb{E}[\mathcal{R}(T)] \geq \frac{1-2\nu}{N}k_t. \tag{27}$$

Since each item can be recommended to each user at most once, we see that by the $TN^{th}$ recommendation at least $T$ different items must have been recommended (that is, $k_t \geq T$). We can then conclude that

$$\mathbb{E}[\mathcal{R}(T)] \geq \underbrace{\frac{1-2\nu}{N}}_{C(\nu,d)}, \tag{28}$$

as we wished. □

## A.4 Chernoff Bound

We state a standard version of the Chernoff Bound [28]:

THEOREM A.1 (CHERNOFF BOUND). *Let $X_1, \cdots, X_n$ be independent random variables that take value in $[0,1]$. Let $X = \sum_{i=1}^n X_i$, and let $\bar{X} = \sum_{i=1}^n \mathbb{E}X_i$. Then, for any $\varepsilon \geq 0$,*

$$\mathbb{P}\left(X \geq (1+\varepsilon)\bar{X}\right) \leq \exp\left(-\frac{\varepsilon^2}{2+\varepsilon}\bar{X}\right), \text{ and}$$
$$\mathbb{P}\left(X \leq (1-\varepsilon)\bar{X}\right) \leq \exp\left(-\frac{\varepsilon^2}{2}\bar{X}\right).$$

# B. EMPIRICAL DOUBLING DIMENSION EXPERIMENTS

The jester dataset contains ratings of one hundred jokes by over seventy thousand users. The dataset is fairly dense (as the average number of ratings per user is over fifty), which makes it a great dataset for calculating the doubling dimension. For the MovieLens 1M Dataset we consider the only movies that have been rated by at least 750 users (to ensure some density).

The Jester ratings are in $[-10, 10]$, with an average of 2, so we make ratings greater than 2 a $R_{u,i} = +1$, and ratings at most 2 a $R_{u,i} = -1$. For the MovieLens 1M Dataset we make ratings $1, 2, 3$ into $-1$, and $4, 5$ into $+1$. We then estimate the doubling dimension as follows:

- For each pair of items $(i, j)$, we calculate $\hat{d}_{i,j,\Delta}$ as fraction of users that agree on them, where the $\Delta$ subscript is put to denote our assumption that each entry has a noise probability of $\Delta$ (that is, $\mathbb{P}(R_{u,i} \neq L_{u,i}) = \Delta$), where $R$ is the empirical ratings matrix and $L$ is the true, noiseless, ratings matrix.
- Assuming that each entry has a noise probability of $\Delta = 0.20$, we estimate the true distance $d_{i,j}$ as the solution to $\hat{d}_{i,j,\Delta} = (1-d_{ij})(2\Delta(1-\Delta)) + d_{i,j}(\Delta^2 + (1-\Delta)^2)$.
- For each item $i$ and $r$ in $\{0, \frac{1}{N}, ..., \frac{N-1}{N}, 1\}$, let $N_{i,r}$ be the number of items such that $d_{i,j} \leq r$.
- For each item $i$ let $d_i$ be the least such that $N_{i,2r}/N_{i,r} \leq 2^{d_i}$ for each $r$ in $\{0, \frac{1}{N}, ..., \frac{1}{2}\}$.
- Figs 2 and 3 show the histogram of the $\{d_i\}$.