$\mathcal{D2P}$: Distance-Based Differential Privacy in Recommenders

Rachid Guerraoui
EPFL
rachid.guerraoui@epfl.ch

Anne-Marie Kermarrec
INRIA
anne-marie.kermarrec@inria.fr

Rhicheek Patra
EPFL
rhicheek.patra@epfl.ch

Mahsa Taziki
EPFL
mahsa.taziki@epfl.ch

ABSTRACT

The upsurge in the number of web users over the last two decades has resulted in a significant growth of online information. This information growth calls for recommenders that personalize the information proposed to each individual user. Nevertheless, personalization also opens major privacy concerns.

This paper presents $\mathcal{D2P}$, a novel protocol that ensures a strong form of differential privacy, which we call distance-based differential privacy, and which is particularly well suited to recommenders.

$\mathcal{D2P}$ avoids revealing exact user profiles by creating *altered* profiles where each item is replaced with another one at some *distance*. We evaluate $\mathcal{D2P}$ analytically and experimentally on MovieLens and Jester datasets and compare it with other private and non-private recommenders.

1. INTRODUCTION

In the modern generation of web-based services, the number of users is increasing exponentially. This number bumped up from 16 million users in 1995 to 3 billion users in 2014. The web has become a big storehouse of information (about 2.5 billion GB of data are created everyday), making it impossible for an individual to explore the whole web contents to extract relevant data. This clearly calls for *personalization* [4]. Personalizing the web led in turn to the advent of *recommenders* [17]. These systems filter out user-specific information in real-time, leveraging user activities and behaviors. Recommenders are primarily used in the context of e-commerce to suggest books, DVDs (*Amazon.com*), trips (*TripAdvisor*), music (*last.fm*) and even research papers (*Mendeley*). They are also used to filter user-specific news (*Google News*, *Yahoo News*).

However, the tendency towards personalization has raised a privacy concern [27] as more and more personal data is being collected and used. It is often observed that when an Internet user accesses some service, the provider of this service typically claims the ownership of any personal information provided by the user. The service provider sometimes even distributes the collected information to third parties like advertising and promotional partners [1]. Even the sharing of anonymised user information like the Netflix Prize dataset might end up not being secure. For instance, Narayanan et. al presented a de-anonymization attack that linked the records in the Netflix Prize dataset with the IMDB profiles available publicly [24].

Particularly fragile among recommenders are Collaborative Filtering (CF) ones [31]: these are widespread because of their ability to provide *serendipitous* recommendations (*unexpected* but *desired* recommendations) [27]. CF recommenders make predictions about the preferences of the users by collecting suggestions from similar users (user-based) or finding similar items (item-based) based on $neighborhood$. CF recommenders provide significant advantages over alternatives (e.g. content-based approaches [33]) due to the substantial number of features they use. Yet, CF recommenders are particularly vulnerable to privacy attacks as they rely on direct information about user *profiles* to provide good recommendations. They aggregate *user preferences* [28] in ways analogous to database queries, which can be exploited by adversaries to extract personal identifiable information about a specific user [27].

Clearly, CF recommenders induce an inherent tradeoff between privacy and quality [19]. In this paper, we address this tradeoff by exploring a promising approach where the information used for computing recommendations is *concealed*. We present $\mathcal{D2P}$, a novel protocol that uses a probabilistic substitution technique to create the AlterEgo profile of an original user profile. $\mathcal{D2P}$ ensures a strong form of *differential privacy* [7, 8], which we call **D***istance-based* **D***ifferential* **P***rivacy*. Differential privacy [7, 8] is a celebrated property, originally introduced in the context of databases. Intuitively, it ensures that the removal of a record from a database does not change the result of a query to that database - modulo some arbitrarily small value (ϵ). In this sense, the presence in the database of every single record - possibly revealing some information about some user - is anonymous as no query can reveal the very existence of that record to any other user (modulo ϵ). Applying this notion in the context of recommenders would mean that - modulo ϵ - no user Y would be able to guess - based on the recommendations she gets - whether some other user X has some item I in her profile, e.g., whether X has seen some movie I. Such a guarantee, however, might be considered too weak as nothing would prevent Y from guessing that X has in her profile some item that is very similar to I, e.g., that X has seen some movie similar to I.

We strengthen the notion of differential privacy in the context of CF recommenders to guarantee that any user Y is not only pre-

Proceedings of the VLDB Endowment, Vol. 8, No. 8
Copyright 2015 VLDB Endowment 2150-8097/15/04.

vented from guessing whether the profile of X contains some item I, but also whether the profile of X contains any item I' within some *distance* λ from I (say any movie of the same category of I): hence the name **D***istance-based* **D***ifferential* **P***rivacy* ($\mathcal{D}2\mathcal{P}$). We present a protocol, named $\mathcal{D}2\mathcal{P}$, which ensures this property.

The basic idea underlying $\mathcal{D}2\mathcal{P}$ is the following. We build, for each user profile, an *AlterEgo* profile corresponding to it. The latter profile is based on the former one where we probabilistically replace some of the items with either related or random ones. This poses of course a challenging technical problem. If the *AlterEgo* profile is too far from the original one, the recommendation quality is impacted: we lose the benefits of collaborative filtering. If the profile is too close to the original one, privacy remains weak.

We demonstrate in the paper that the quality of the $\mathcal{D}2\mathcal{P}$ recommendation is still good for values of λ that can hide items within a reasonable distance from the original profile - what might be considered a reasonable distance depends on the dataset as we explain later in the paper.

To illustrate the basic idea, consider traces from Movielens and the scenario of Figure 1, with a total of 5 movies. Consider a user who likes Shawshank Redemption (SR). We compute the distance between the other 4 movies from SR based on their similarity (as shown later in Equation 1 in Section 2.1). $\mathcal{D}2\mathcal{P}$ selects movies (for replacement) with distance less than the upper bound (λ = 0, 1 or 2) with high probability (p) and any random movie from the dataset, including those close to the item to be replaced, with a low probability ($1-p$). If λ is set to 0, then $\mathcal{D}2\mathcal{P}$ satisfies the classical differential privacy (with ϵ given in Equation 12 in Section 3.2.3). Our results in Section 4 show that even if we consider λ as 6.5, we still have a good recommendation quality.

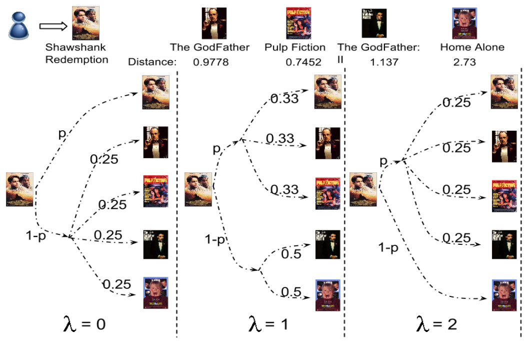


Figure 1: *$\mathcal{D}2\mathcal{P}$ Illustration.*

We perform a thorough evaluation of $\mathcal{D}2\mathcal{P}$: (i) we first analytically compute the guarantees ensured by $\mathcal{D}2\mathcal{P}$, in terms of parameters ϵ and λ, and then (ii) we evaluate experimentally the quality of recommendations provided on real datasets, namely MovieLens and Jester. Our results show that $\mathcal{D}2\mathcal{P}$ provides proved privacy guarantees while preserving the quality of the recommendation. We demonstrate, for instance, that $\mathcal{D}2\mathcal{P}$ achieves 1.5 times the coverage [10] provided by a standard recommender for Movielens dataset. Additionally, we show that the privatization overhead in $\mathcal{D}2\mathcal{P}$ is very small compared to [21], which makes it appealing for real-time workloads.

Interestingly, $\mathcal{D}2\mathcal{P}$ is a generic protocol. As we show through our performance results, it applies well in the context of a user-based as well as an item-based recommender. $\mathcal{D}2\mathcal{P}$ can also be customized for recommendation infrastructures where a *KNN* computation is deployed either on the cloud [26] or on user machines [5].

The rest of the paper is organized as follows. Section 2 presents $\mathcal{D}2\mathcal{P}$. Section 3 discusses its privacy guarantees. Section 4 evaluates the quality and coverage of its recommendations along with the privatization overhead. Section 5 discusses related work. We conclude the paper in Section 6 by discussing future works.

2. $\mathcal{D}2\mathcal{P}$ RECOMMENDER

$\mathcal{D}2\mathcal{P}$ considers a general CF recommendation scheme based on KNN (K Nearest Neighbors [31]). The working principle of such a scheme is twofold. Firstly, the k most similar *neighbors* of any active user are identified in the KNN selection phase. Secondly, the recommendation algorithm is run to suggest items to the users leveraging the profiles obtained through the KNN selection.

2.1 Underlying Scheme

We consider a recommender scheme that stores *user profiles* and *item profiles*. The profile of a user $\mathcal{U}$, denoted by $P_{\mathcal{U}}$, consists of all the items rated (alternatively shared or liked) by $\mathcal{U}$ along with the ratings. In our implementation, we convert the numerical ratings into binary ratings, a *like* (1) or a *dislike* (0). [1] An *item profile* ($P_{\mathcal{I}}$) consists of users who rated item $\mathcal{I}$ along with the ratings.

$\mathcal{D}2\mathcal{P}$ relies on the *distance* between items to create *AlterEgo* profiles, as we discuss below. The recommender in $\mathcal{D}2\mathcal{P}$ operates in four phases as shown in Figure 2.

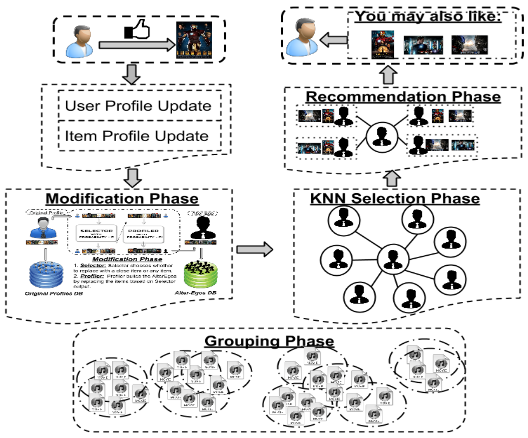


Figure 2: *$\mathcal{D}2\mathcal{P}$ Recommendation Scheme.*

2.1.1 Grouping Phase.

In this phase, groups are formed for each item: group G_i for item i contains all the items with distance less than a predefined upper-bound λ. In our scheme, we define the distance $\Lambda_{i,j}$ between items i and j as:

$$\Lambda_{i,j} = \frac{1}{\Psi(i,j)} - 1 \tag{1}$$

Here, $\Psi(i,j)$ denotes the cosine similarity between items i and j. The neighboring group G_j of a group G_i is defined as a group with which group G_i shares at least one item. Groups can also be formed based on item features (e.g. genres, date-of-release in case of movies) where similarity is measured between the feature vectors of the items. The groups need to be updated periodically to account for newly added items and ratings. In $\mathcal{D}2\mathcal{P}$, the grouping of

[1]Binary ratings are considered for the sake of simplicity: this scheme can be generalized to numerical ratings.

the items in the *Grouping Phase* is performed by the *FormGroups* function shown in Algorithm 1. An item can be included in more than one groups, e.g., an *action-comedy* movie *X* can be present in the group of an *action* movie as well as in the group of a *comedy* movie.

Algorithm 1 Grouping : *FormGroups(ItemSet): Grouping Phase where* **ItemSet** *is the set of all items in Database*

```
1: Parameter: $\lambda$                          ▷ Distance threshold
2: var $ItemSet$;  ▷ Denotes set of all items in the network
3: var $\lambda$;                                 ▷ Distance Metric
4: for all $iid$ : item in $ItemSet$ do
5:     $Group_{iid}.add(iid)$;
6:     for all $rid$ : item in $(ItemSet \setminus iid)$ do
7:         $S = \Psi(iid, rid)$;               ▷ Compute Similarity
8:         if $S > 0$ then
9:             $\Lambda_{iid,rid} = (1/S) - 1$;
10:            if $\Lambda_{iid,rid} \leq \lambda$ then
11:                $Group_{iid}.add(rid)$;
12:            end if
13:        end if
14:    end for
15: end for
16: return: $Group$;                   ▷ The groups for the items
```

2.1.2 Modification Phase.

$\mathcal{D}2\mathcal{P}$ relies on the above mentioned groups of items to create *AlterEgo* profiles, avoiding to reveal the exact ones. The two core components of $\mathcal{D}2\mathcal{P}$ are the *Selector*, which selects the items to replace and the *Profiler*, which determines by which items those entries should be replaced. The *AlterEgo* profile of a user $\mathcal{U}$ denotes the imitation profile of $\mathcal{U}$ which hides the user preferences by substituting items in the user profile by utilizing $\mathcal{D}2\mathcal{P}$. The *selector* and *profiler* are in charge of computing these *AlterEgo* profiles to preserve (ϵ, λ)*-differential privacy*. Details about *selector* and *profiler* are provided later.

2.1.3 KNN Selection Phase.

In user-based CF recommenders, a K-Nearest Neighbors (KNN) [31] algorithm computes the K most similar users based on some similarity metric. In this phase, we periodically update the top K_{users} similar users for an active user as the $neighbors$ using the *AlterEgo* profiles generated in the modification phase.

2.1.4 Recommendation Phase.

In this final phase, the recommendations are computed using those K_{users} neighbors. In the context of this paper, we select the most popular items among the *neighbors* of $\mathcal{U}$ to be recommended to $\mathcal{U}$.

Some maintenance operations are needed: (i) *Profile update:* When a user $\mathcal{U}$ rates an item $\mathcal{I}$, then both $P_{\mathcal{U}}$ and $P_{\mathcal{I}}$ are updated. Profiles are updated incrementally as in standard online recommenders. (ii) *Group Update:* The static nature of the relationship (similarity) [18, 29] between items stabilizes the grouping phase. So, the frequency of group updates has little impact on the quality of the provided recommendations; The groups are updated periodically after every 10 days in our evaluation. (iii) *Recommendation:* The new recommendations are delivered to the active user incrementally whenever an item is rated by the user. In $\mathcal{D}2\mathcal{P}$, only the *AlterEgo* profiles of the KNN are updated during each recommendation. We take into account the recent ratings provided by the users to compute recommendations.

Privacy breaches occur in a standard user-based CF recommender due to leakage of the information of neighboring profiles to the active user through recommendations provided to her. $\mathcal{D}2\mathcal{P}$ protects the privacy of users in the modification phase through two components: *Selector* and *Profiler* as conveyed by Figure 3. These two components conceal the neighbors' information from the active user, preventing this user to correlate the recommendations to the neighbors' profiles. The *selector* and *profiler* are responsible for forming the *AlterEgo* profiles in such a way that the quality is not impacted too much while privacy is preserved. We now provide details on these two core components.

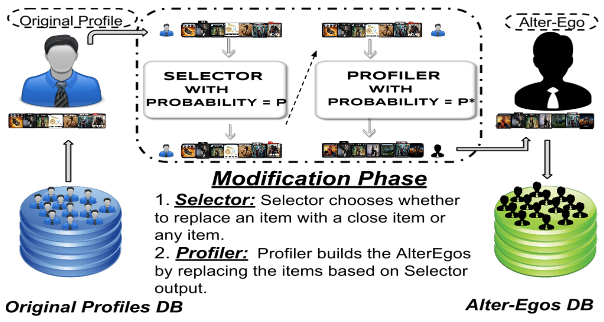

Original Profile
SELECTOR WITH PROBABILITY = P
PROFILER WITH PROBABILITY = P*
Alter-Ego
Modification Phase
1. Selector: Selector chooses whether to replace an item with a close item or any item.
2. Profiler: Profiler builds the AlterEgos by replacing the items based on Selector output.
Original Profiles DB
Alter-Egos DB

Figure 3: $\mathcal{D}2\mathcal{P}$ *Modification Phase.*

2.2 $\mathcal{D}2\mathcal{P}$ Components

2.2.1 $\mathcal{D}2\mathcal{P}$ Selector

The *selector* is responsible for selecting the items to replace by the *profiler* to form the *AlterEgo* profiles. We select an item with a probability p to replace with any possible item at random and with a probability $1 - p$ to replace with some random item from the respective group (and neighboring groups) for that respective item. The *getSelectProb* function mentioned in Algorithms 2 and 3, returns a random real number between 0 and 1. Finally, the *selector* outputs a set of actual items (*GItems*) to be replaced by *GroupItems* and another set of actual items (*RItems*) to be replaced by any item from the set of all possible items at random.

Algorithm 2 Selector Algorithm*: Selector*(P_u) *where* P_u *is the profile of user* u

```
1: Parameter: $p$                        ▷ Selector Probability
2: var $GItems[U] = NULL$;  ▷ Replace with group item
3: var $RItems[U] = NULL$;     ▷ Replace with any item
4: for all $iid$ : item in $P_U.getItems()$ do
5:     if $getSelectProb() > p$ then
6:         $GItems[U] = GItems[U] \cup iid$;
7:     end if
8:     if $getSelectProb() \leq p$ then
9:         $RItems[U] = RItems[U] \cup iid$;
10:    end if
11: end for
12: return: $\{GItems[U],\ RItems[U]\}$;
```

2.2.2 $\mathcal{D}2\mathcal{P}$ Profiler

The *profiler* builds the *AlterEgo* profiles which are used in the KNN selection phase. The *profiler* replaces items in *GItems* with items from their respective group (and neighboring groups) with a probability $1-p^*$ and retains the original item with a probability p^*. We also substitute items in *RItems* with items from the set of all possible items with a probability $1-p^*$ and preserves the actual ones with a probability p^*. The $SRSI$ (*S*elect *R*andom *S*et *I*tem) function in Algorithm 3 selects randomly an item from the respective groups' items. It selects either from *GroupItems* (based on a distance metric between items) for all the items in the set *GItems* or from the *ItemSet* for all the items in *RItems*. In the following sections, we show that $\mathcal{D}2\mathcal{P}$ ensures users' privacy while preserving a good recommendation quality.

Algorithm 3 **Profiler Algorithm**: *$Profiler(P_u)$ where P_u is the profile of user u*

```
1: Parameter: $p^*$                                   ▷ $Profiler\ Probability$
2: var $\{GItems[U],\ RItems[U]\} = Selector(P_U)$;
3: var $Items[U] = GPI(P_U)$;                         ▷ $Get\ items\ from\ P_U$
4: var $ItemSet$;                                     ▷ $Set\ of\ all\ items\ in\ the\ network$
5: for all $iid$ : item in $P_U.getItems()$ do
6:     $GroupID = Group_{iid}$;
7:     $NBGroupIDs = Group_{iid}.getNeighbors()$;
8:     $Groups = GroupID \cup NBGroupIDs$;
9:     $GroupItems = \bigcup_{G \in Groups} Group.get(G)$;
10:    if $(getSelectProb() > p^*\ \&\ iid \in GItems[U])$ then
11:        $rid = SRSI(iid, GroupItems)$;
12:    end if
13:    if $(getSelectProb() > p^*\ \&\ iid \in RItems[U])$ then
14:        $rid = SRSI(iid, ItemSet)$;
15:    end if
16:    $P_U = (P_U \setminus iid) \cup rid$;
17: end for
18: return: $P_U$;                                    ▷ $AlterEgo$ profile for user $U$
```

Interestingly, $\mathcal{D}2\mathcal{P}$ can also be applied in recommendation infrastructures where the KNN is computed by third-party cloud services that act as intermediaries between the recommendation server and users: these servers create the *AlterEgo* profiles, preserving privacy with respect to a server. Moreover, $\mathcal{D}2\mathcal{P}$ can be applied by the users themselves (in P2P or hybrid infrastructures [5]), preserving privacy of users against other users.

3. PRIVACY GUARANTEES

Preserving privacy in CF recommenders is challenging. It was shown using the Netflix Prize dataset that even anonymizing individual data before releasing it publicly is not enough to preserve privacy [24]. Even cryptographic approaches do not preclude the possibility of the output leaking information about the personal input of individuals [32]. The need for stronger and robust privacy guarantees motivated the emergence of the notion of *Differential Privacy* [7, 8, 9]. First introduced in the context of databases, differential privacy provides quantifiable privacy guarantees. We introduce a stronger form of this notion in the context of recommenders by accounting for the concept of distance between items.

3.1 Privacy for Recommenders

3.1.1 Differential Privacy

Differential Privacy (DP) implies that the output of a given function becomes significantly more or less likely - based on some parameter ϵ - if the inputs differ in one record. The basic intuition is that an observer can extract limited information from the output in the absence or presence of a specific record in the database.

DEFINITION 1. (DIFFERENTIAL PRIVACY) *A randomized function $\mathcal{R}$ provides ϵ-differential privacy if for all datasets $\mathcal{D}_1$ and $\mathcal{D}_2$, differing on at most one element, and all $\mathcal{S} \subseteq Range(\mathcal{R})$, the following inequality always holds:*

$$\frac{Pr[\mathcal{R}(\mathcal{D}_1) \in \mathcal{S}]}{Pr[\mathcal{R}(\mathcal{D}_2) \in \mathcal{S}]} \leq e^{\epsilon}$$

Here, e^{ϵ} denotes $exp(\epsilon)$.

3.1.2 Distance-based Differential Privacy

With differential privacy applied in its classical form recalled above to a recommender, an adversary (a curious user) cannot know if one item has been rated by a user. However, the adversary can know about items similar to the rated ones. Hence, the adversary can infer fairly accurate information about user preferences without knowing the exact items rated by that user. In this sense, classical differential privacy is not enough in the context of a recommender.

Our notion of *Distance-based Differential Privacy* is stronger: it extends DP to recommenders. We ensure differential privacy for all the items, rated by that user, and ones that are within a distance of λ. The distance parameter (λ) determines the *closely related items to form the AlterEgo profiles*, thereby *concealing* the actual user profiles and preferences. The distance parameter also aids in tuning the recommendation quality using the *AlterEgo* profiles as shown later in Figures 12a and 12b.

It is important to notice that our notion of Distance-based differential privacy is independent from the underlying recommendation algorithm used. To define this new notion more precisely, we first define the notions of *Distance-based Group* and *Adjacent Profile Sets*.

DEFINITION 2. (GROUP DEFINITION ELEMENT-WISE) *We denote by $\mathbb{E}$ the set of all elements. For every element $x \in \mathbb{E}$, distance function $\Lambda : \mathbb{E} \times \mathbb{E} \rightarrow \mathbb{R}^+ \cup \{0\}$, and fixed distance threshold λ, then $\mathcal{GRP}_\lambda(x)$ is defined as the collection of all elements $x_k \in \mathbb{E}$ such that $\Lambda_{x,x_k} \leq \lambda$. More specifically:*

$$\mathcal{GRP}_\lambda(x) = \{x_k \in \mathbb{E} | \Lambda_{x,x_k} \leq \lambda\}$$

We extend this notion of groups to a set of elements where each element in the set has a *Group* defined by Definition 2.

DEFINITION 3. (GROUP DEFINITION SET-WISE) *For a set of elements $\mathcal{S}$, $\mathcal{GRP}_\lambda(\mathcal{S})$ is the union of all the groups: $\mathcal{GRP}_\lambda(s)$ for each element s$\in$ $\mathcal{S}$. More specifically:*

$$\mathcal{GRP}_\lambda(\mathcal{S}) = \bigcup_{s \in \mathcal{S}} \mathcal{GRP}_\lambda(s)$$

We now introduce the notion of *Neighboring Groups* (used in Section 3.2.3).

DEFINITION 4. (NEIGHBORING GROUP) *We define the $\mathcal{KNN}$ groups ($\mathcal{KNN}(\mathcal{GRP}_\lambda(x))$) of $\mathcal{GRP}_\lambda(x)$ for an element x as the $Top-\mathcal{K}$ groups sorted in decreasing order by the count of shared elements with $\mathcal{GRP}_\lambda(x)$.*

DEFINITION 5. (ADJACENT PROFILE SET) *An event in the context of $\mathcal{D}2\mathcal{P}$ is an interaction between the system and the user when the user provides a rating for some item in the system. Two profile sets $\mathcal{D}_1$ and $\mathcal{D}_2$ as* adjacent profile sets *when $\mathcal{D}_1$ and $\mathcal{D}_2$ differ in only one event, which implies one user-item rating pattern is different in these two profile sets.*

Notations	
$\mathcal{M}$	A mechanism relying on *AlterEgo* profiles to provide recommendations
$\mathcal{U}$	User to whom some recommendation is provided
$\mathcal{P}_i$	Accurate i^{th} profile from profile set $\mathcal{D}$
$\mathcal{P}_i^r$	*AlterEgo* profile of original profile $\mathcal{P}_i$
p_i^r	Any random possible $AlterEgo$ profile for original profile $\mathcal{P}_i$
$\mathcal{D}$	Original Profile Set
$\mathcal{D}'$	Profile Set which differs with $\mathcal{D}$ at an event
$\mathcal{S}$	Arbitrary set of elements
$\mathcal{N}_{\mathbb{E}}$	Total number of elements
π	Any possible permutation of numbers in the range {0,..n}

Table 1: *Notations*

For any arbitrary recommendation mechanism $\mathcal{R}$, which takes a profile set and a specific user as input, the output is the set of items that the algorithm recommends to that specific user.

DEFINITION 6. *(*DISTANCE-BASED DIFFERENTIAL PRIVACY*) For any two adjacent profile sets $\mathcal{D}_1$ and $\mathcal{D}_2$, where $\mathcal{U}$ denotes any arbitrary user and $\mathcal{S}$ denotes any possible subset of elements, then any mechanism $\mathcal{R}$ is (ϵ, λ)-private if the following inequality holds:*

$$\frac{Pr[\mathcal{R}(\mathcal{D}_1,\mathcal{U}) \in \mathcal{GRP}_\lambda(\mathcal{S})]}{Pr[\mathcal{R}(\mathcal{D}_2,\mathcal{U}) \in \mathcal{GRP}_\lambda(\mathcal{S})]} \leq e^\epsilon \tag{2}$$

The result of the recommendations for two profile sets that are close to each other are of the same order probabilistically with a coefficient of e^ϵ. Later in Section 3.2.3, we present the mathematical relationship between ϵ and λ. $\mathcal{D}2\mathcal{P}$ conceals the profiles by anonymizing elements within distance λ from the elements of the original profile. We get the classic notion of differential privacy with λ as 0. If we increase λ then the privacy increases but the quality decreases slightly as shown later in Figure 12a. In a user-level privacy scheme, more than one event can differ for a profile in two adjacent profile sets, whereas in an event-level privacy approach a single event differs for a profile in two adjacent profile sets. The proofs in the following section assume event-level privacy.

3.2 Privacy Analysis

In this section, we analyze our $\mathcal{D}2\mathcal{P}$ protocol based on the privacy parameter ϵ and the distance parameter λ.

3.2.1 Notations

We fix an arbitrary user $\mathcal{U}$ to whom we provide some recommendations. $\mathcal{D}$ and $\mathcal{D}'$ are two adjacent profile sets. Additional notations used later in the proofs are mentioned in Table 1.

3.2.2 Proofs

As $\mathcal{D}$ and $\mathcal{D}'$ are two adjacent profile sets, using Definition 5 we know that $\mathcal{D}$ and $\mathcal{D}'$ differ at one event which is one user-item rating pattern in a profile.$\mathcal{P}_i$ denotes this profile in profile set $\mathcal{D}$ whereas $\mathcal{P}_i'$ denotes this profile in profile set $\mathcal{D}'$. $\mathcal{P}_i$ has an element i_0, for which in $\mathcal{P}_i'$ there is another element i_0' in place of i_0. So, exactly one rating pattern is different in $\mathcal{P}_i$ and $\mathcal{P}_i'$. Let the elements in $\mathcal{P}_i$ be $i_0, i_1, ..., i_n$ and the elements in $\mathcal{P}_i'$ be $i_0', i_1, ..., i_n$.

PROPOSITION 1. *For any given distance metric λ and any two elements i and j, we denote $\mathcal{SUB}(i,j)$ the event of substituting element i with j in a mechanism $\mathcal{M}$. This substitution probability is denoted by $Pr(\mathcal{SUB}(i,j))$. Then, for mechanism $\mathcal{M}$, we propose ϵ as:*

$$\epsilon = \ln(\delta)$$

where $\delta = \max_{i,j,k \in \mathbb{E} \; and \; i \neq j} \left(\frac{Pr(\mathcal{SUB}(i,k))}{Pr(\mathcal{SUB}(j,k))} \right)$

THEOREM 1. *Any mechanism $\mathcal{M}$, that relies on the AlterEgo of a profile $\mathcal{P}_i$, is an (ϵ, λ)-private mechanism for ϵ given in Proposition 1.*

First, we present some lemmas and corollaries needed to prove Theorem 1.

LEMMA 1. *For three arbitrary elements $i, j, k \in \mathbb{E}$ and $i \neq j$, we get the following inequality:*

$$\frac{Pr(\mathcal{SUB}(i,k))}{Pr(\mathcal{SUB}(j,k))} \leq e^\epsilon$$

PROOF. Since, any ratio is upper-bounded by its maximum. Hence, from Proposition 1 we have:

$$\frac{Pr(\mathcal{SUB}(i,k))}{Pr(\mathcal{SUB}(j,k))} \leq \left(\max_{i,j,k \in \mathbb{E} \; and \; i \neq j} \frac{Pr(\mathcal{SUB}(i,k))}{Pr(\mathcal{SUB}(j,k))} \right) = e^\epsilon$$

Therefore, we get:

$$\frac{Pr(\mathcal{SUB}(i,k))}{Pr(\mathcal{SUB}(j,k))} \leq e^\epsilon$$

□

LEMMA 2. *Let $\mathcal{M}$ be a privacy preserving mechanism which creates the AlterEgo ($\mathcal{P}_i^r$) of a profile $\mathcal{P}_i$. $Pr(\mathcal{PS}(\mathcal{P}_i, \mathcal{P}_i^r))$ denotes the probability of the substitution event ($\mathcal{PS}(\mathcal{P}_i, \mathcal{P}_i^r)$) to create the AlterEgo. Then, the following inequality holds:*

$$\frac{Pr(\mathcal{PS}(\mathcal{P}_i, \mathcal{P}_i^r))}{Pr(\mathcal{PS}(\mathcal{P}_i', \mathcal{P}_i^r)} \leq e^\epsilon$$

PROOF. Denote by π any permutation of {0,..,n}. Let $P_{i_\pi}^r$ be a profile with elements $i_{\pi(0)}^r$, $i_{\pi(1)}^r$, ..., $i_{\pi(n)}^r$. $P_{i_\pi}^r$ denotes one possible permutation of the elements of $\mathcal{P}_i^r$. So, for any permutation π, we compute the probability of changing $\mathcal{P}_i$ to $\mathcal{P}_{i_\pi}^r$. Then, we sum over all possible permutations to get the final probability of the substitution event ($\mathcal{PS}(\mathcal{P}_i, \mathcal{P}_i^r)$).

$$Pr(\mathcal{PS}(\mathcal{P}_i, \mathcal{P}_i^r)) = \sum_{\forall \pi} Pr(\mathcal{PS}(\mathcal{P}_i, \mathcal{P}_{i_\pi}^r)) \tag{3}$$

Now, based on the fact that every element is replaced independently of the replacement of other elements, we compute:

$$Pr(\mathcal{PS}(\mathcal{P}_i, \mathcal{P}_{i_\pi}^r)) = \prod_{k=0}^{n} Pr(\mathcal{SUB}(i_k, i_{\pi(k)}^r)) \tag{4}$$

Based on Equations 3 and 4, we get:

$$Pr(\mathcal{PS}(\mathcal{P}_i, \mathcal{P}_i^r)) = \sum_{\forall \pi} \left(\prod_{k=0}^{n} Pr(\mathcal{SUB}(i_k, i_{\pi(k)}^r)) \right) \tag{5}$$

In the same way, for $\mathcal{P}_i'$, we have:

$$Pr(\mathcal{PS}(\mathcal{P}_i', \mathcal{P}_i^r))$$

$$= \sum_{\forall \pi} Pr(\mathcal{SUB}(i_0', i_{\pi(0)}^r)). \prod_{k=1}^{n} Pr(\mathcal{SUB}(i_k, i_{\pi(k)}^r)) \tag{6}$$

Now, from Equations 5 and 6, we get:

$$\frac{Pr(\mathcal{PS}(\mathcal{P}_i,\mathcal{P}_i^r))}{Pr(\mathcal{PS}(\mathcal{P}_i',\mathcal{P}_i^r))}$$

$$= \frac{\sum_{\forall\pi} Pr(\mathcal{SUB}(i_0,i^r_{\pi(0)})).\prod_{k=1}^{n} Pr(\mathcal{SUB}(i_k,i^r_{\pi(k)}))}{\sum_{\forall\pi} Pr(\mathcal{SUB}(i_0',i^r_{\pi(0)})).\prod_{k=1}^{n} Pr(\mathcal{SUB}(i_k,i^r_{\pi(k)}))} \quad (7)$$

From Lemma 1, we get for any three arbitrary elements i_0, i_0' and i_0^r:

$$\frac{Pr(\mathcal{SUB}(i_0,i_0^r))}{Pr(\mathcal{SUB}(i_0',i_0^r))} \leq e^{\epsilon} \quad (8)$$

So, from Equations 7 and 8, we get:

$$\frac{Pr(\mathcal{PS}(\mathcal{P}_i,\mathcal{P}_i^r))}{Pr(\mathcal{PS}(\mathcal{P}_i',\mathcal{P}_i^r))}$$

$$\leq \frac{\sum_{\forall\pi} e^{\epsilon}.Pr(\mathcal{SUB}(i_0',i^r_{\pi(0)})).\prod_{k=1}^{n} Pr(\mathcal{SUB}(i_k,i^r_{\pi(k)}))}{\sum_{\forall\pi} Pr(\mathcal{SUB}(i_0',i^r_{\pi(0)})).\prod_{k=1}^{n} Pr(\mathcal{SUB}(i_k,i^r_{\pi(k)}))}$$

$$= e^{\epsilon}.\frac{\sum_{\forall\pi} Pr(\mathcal{SUB}(i_0',i^r_{\pi(0)})).\prod_{k=1}^{n} Pr(\mathcal{SUB}(i_k,i^r_{\pi(k)}))}{\sum_{\forall\pi} Pr(\mathcal{SUB}(i_0',i^r_{\pi(0)})).\prod_{k=1}^{n} Pr(\mathcal{SUB}(i_k,i^r_{\pi(k)}))}$$

$$= e^{\epsilon}$$

Hence, we can conclude that:

$$\frac{Pr(\mathcal{PS}(\mathcal{P}_i,\mathcal{P}_i^r))}{Pr(\mathcal{PS}(\mathcal{P}_i',\mathcal{P}_i^r))} \leq e^{\epsilon}$$

□

COROLLARY 1. *For any two adjacent profile sets $\mathcal{D}$, $\mathcal{D}'$ and any arbitrary AlterEgo profile set $\mathcal{D}^r$, we denote by $\mathcal{DS}(\mathcal{D},\mathcal{D}^r)$ the Set Substitution event for $\mathcal{D}$ with $\mathcal{D}^r$. We assume the profile sets $\mathcal{D}$ and $\mathcal{D}'$ differ at the i^{th} profile (P_i). Then, we have:*

$$\frac{Pr(\mathcal{DS}(\mathcal{D},\mathcal{D}^r))}{Pr(\mathcal{DS}(\mathcal{D}',\mathcal{D}^r))} \leq e^{\epsilon}$$

PROOF. *Using the same approach as in Lemma 2, we get:*

$$\frac{Pr(\mathcal{DS}(\mathcal{D},\mathcal{D}^r))}{Pr(\mathcal{DS}(\mathcal{D}',\mathcal{D}^r))} = \frac{\sum\limits_{\forall\pi} Pr(\mathcal{PS}(P_i,P^r_{\pi(i)})).\prod\limits_{k=1,k\neq i}^{m} Pr(\mathcal{PS}(P_k,P^r_{\pi(k)}))}{\sum\limits_{\forall\pi} Pr(\mathcal{PS}(P_i',P^r_{\pi(i)})).\prod\limits_{k=1,k\neq i}^{m} Pr(\mathcal{PS}(P_k,P^r_{\pi(k)}))}$$

From Lemma 2, we have:

$$\frac{Pr(\mathcal{PS}(P_i,P^r_{\pi(i)}))}{Pr(\mathcal{PS}(P_i',P^r_{\pi(i)}))} \leq e^{\epsilon}$$

Therefore, using Lemma 2, we get:

$$\frac{Pr(\mathcal{DS}(\mathcal{D},\mathcal{D}^r))}{Pr(\mathcal{DS}(\mathcal{D}',\mathcal{D}^r))} \leq \frac{\sum\limits_{\forall\pi} \mathbf{e}^{\epsilon}.Pr(\mathcal{PS}(P_i',P^r_{\pi(i)})).\prod\limits_{k=1,k\neq i}^{m} Pr(\mathcal{PS}(P_k,P^r_{\pi(k)}))}{\sum\limits_{\forall\pi} Pr(\mathcal{PS}(P_i',P^r_{\pi(i)})).\prod\limits_{k=1,k\neq i}^{m} Pr(\mathcal{PS}(P_k,P^r_{\pi(k)}))}$$

Hence, we get:

$$\frac{Pr(\mathcal{DS}(\mathcal{D},\mathcal{D}^r))}{Pr(\mathcal{DS}(\mathcal{D}',\mathcal{D}^r))} \leq e^{\epsilon}$$

□

PROOF OF THEOREM 1. Let $\mathcal{D}$ and $\mathcal{D}'$ be any two adjacent profile sets and $\mathcal{D}^r$ be any arbitrary *AlterEgo* profile set. We define $\mathcal{M}'$ as any mechanism which takes as input the *AlterEgo* profiles and the target user to whom we provide recommendations as shown in Figure 4. So, we can rewrite:

$$Pr[\mathcal{M}(\mathcal{D},\mathcal{U}) \in \mathcal{GRP}_{\lambda}(\mathcal{S})]$$
$$= \sum_{\mathcal{D}^r} Pr(DS(\mathcal{D},\mathcal{D}^r)).Pr[\mathcal{M}'(\mathcal{D}^r,\mathcal{U}) \in \mathcal{GRP}_{\lambda}(\mathcal{S})] \quad (9)$$

Using the same approach for the profile set $\mathcal{D}'$, we get the following equation:

$$Pr[\mathcal{M}(\mathcal{D}',\mathcal{U}) \in \mathcal{GRP}_{\lambda}(\mathcal{S})]$$
$$= \sum_{\mathcal{D}^r} Pr(DS(\mathcal{D}',\mathcal{D}^r)).Pr[\mathcal{M}'(\mathcal{D}^r,\mathcal{U}) \in \mathcal{GRP}_{\lambda}(\mathcal{S})] \quad (10)$$

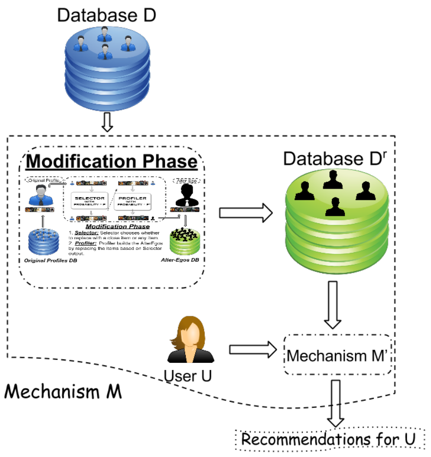

Database D
Modification Phase
Original Profile
SELECTOR WITH PROBABILITY = P
PROFILER WITH PROBABILITY = P'
Alter Ego
Modification Phase
1. Selector: Selector chooses whether to replace with a close item or any item.
2. Profiler: Profiler builds the AlterEgos by replacing the items based on Selector output.
Original Profiles DB
Alter-Egos DB
Database D^r
User U
Mechanism M'
Mechanism M
Recommendations for U

Figure 4: *Relation between mechanisms M and M'.*

From Equations 9 and 10, we arrive at:

$$\frac{Pr[\mathcal{M}(\mathcal{D},\mathcal{U}) \in \mathcal{GRP}_{\lambda}(\mathcal{S})]}{Pr[\mathcal{M}(\mathcal{D}',\mathcal{U}) \in \mathcal{GRP}_{\lambda}(\mathcal{S})]}$$
$$= \frac{\sum_{\mathcal{D}^r} Pr(DS(\mathcal{D},\mathcal{D}^r)).Pr[\mathcal{M}'(\mathcal{D}^r,\mathcal{U}) \in \mathcal{GRP}_{\lambda}(\mathcal{S})]}{\sum_{\mathcal{D}^r} Pr(DS(\mathcal{D}',\mathcal{D}^r)).Pr[\mathcal{M}'(\mathcal{D}^r,\mathcal{U}) \in \mathcal{GRP}_{\lambda}(\mathcal{S})]}$$

Let $\mathcal{Q}_{\mathcal{D}^r}$ denote the event probability: $Pr[\mathcal{M}'(\mathcal{D}^r,\mathcal{U}) \in \mathcal{GRP}_{\lambda}(\mathcal{S})]$, so we can reformulate the above equation as:

$$\frac{Pr[\mathcal{M}(\mathcal{D},\mathcal{U}) \in \mathcal{GRP}_{\lambda}(\mathcal{S})]}{Pr[\mathcal{M}(\mathcal{D}',\mathcal{U}) \in \mathcal{GRP}_{\lambda}(\mathcal{S})]} = \frac{\sum_{\mathcal{D}^r} Pr(DS(\mathcal{D},\mathcal{D}^r)).\mathcal{Q}_{\mathcal{D}^r}}{\sum_{\mathcal{D}^r} Pr(DS(\mathcal{D}',\mathcal{D}^r)).\mathcal{Q}_{\mathcal{D}^r}} \quad (11)$$

From Equation 11 and Corollary 1, we have:

$$\frac{Pr[\mathcal{M}(\mathcal{D},\mathcal{U}) \in \mathcal{GRP}_{\lambda}(\mathcal{S})]}{Pr[\mathcal{M}(\mathcal{D}',\mathcal{U}) \in \mathcal{GRP}_{\lambda}(\mathcal{S})]} = \frac{\sum_{\mathcal{D}^r} Pr(DS(\mathcal{D},\mathcal{D}^r)).\mathcal{Q}_{\mathcal{D}^r}}{\sum_{\mathcal{D}^r} Pr(DS(\mathcal{D}',\mathcal{D}^r)).\mathcal{Q}_{\mathcal{D}^r}}$$

$$\leq \frac{\sum_{\mathcal{D}^r} e^{\epsilon}.Pr(DS(\mathcal{D}',\mathcal{D}^r)).\mathcal{Q}_{\mathcal{D}^r}}{\sum_{\mathcal{D}^r} Pr(DS(\mathcal{D}',\mathcal{D}^r)).\mathcal{Q}_{\mathcal{D}^r}} = e^{\epsilon}$$

Hence, using ϵ from Proposition 1 for mechanism $\mathcal{M}$, we get:

$$\frac{Pr[\mathcal{M}(\mathcal{D},\mathcal{U}) \in \mathcal{GRP}_{\lambda}(\mathcal{S})]}{Pr[\mathcal{M}(\mathcal{D}',\mathcal{U}) \in \mathcal{GRP}_{\lambda}(\mathcal{S})]} \leq e^{\epsilon}$$

Therefore, using Definition 6, we can conclude that mechanism $\mathcal{M}$ satisfies (ϵ, λ)-privacy. □

3.2.3 Privacy Analysis of $\mathcal{D2P}$

Now, we analyze our $\mathcal{D2P}$ privacy in the abstract model which we introduced in Section 2.

First, we denote the $GroupItems$ for an item i in Algorithm 3 as:

$$\mathcal{G}_\lambda(i) = (\cup_{j \in \mathcal{KNN}(\mathcal{GRP}_\lambda(i))} \mathcal{GRP}_\lambda(j)) \cup \mathcal{GRP}_\lambda(i)$$

As mentioned in Section 2, the $selector$ selects to replace an element s with any random element from $\mathbb{E}$ with a probability p and with any random element from $\mathcal{G}_\lambda(s)$ with a probability $1-p$. So, it finally outputs two sets of elements $GItems$ and $RItems$ for each user profile. For both of these sets ($GItems$ and $RItems$), the $profiler$ retains the original elements with probability p^*. It replaces elements in $GItems$ with elements from $\mathcal{G}_\lambda(s)$ and elements in $RItems$ with any possible element $e \in \mathbb{E}$ with probability $1-p^*$. Here $\mathcal{N}_\mathbb{E}$ is the total number of elements in $\mathbb{E}$.

So, we compute $Pr(\mathcal{SUB}(s,t))$, for any two arbitrary elements s and t, in this abstract model. We get the following:

$$Pr(\mathcal{SUB}(s,t)) = \begin{cases} p^* + \frac{(1-p)(1-p^*)}{|\mathcal{G}_\lambda(s)|} + \frac{p(1-p^*)}{\mathcal{N}_\mathbb{E}} & \text{if } s = t \\ \frac{(1-p)(1-p^*)}{|\mathcal{G}_\lambda(s)|} + \frac{p(1-p^*)}{\mathcal{N}_\mathbb{E}} & \text{if } t \in \mathcal{G}_\lambda(s) \setminus s \\ \frac{p(1-p^*)}{\mathcal{N}_\mathbb{E}} & \text{if } t \notin \mathcal{G}_\lambda(s) \,. \end{cases}$$

Let $\epsilon_{\mathcal{D2P}}^{(p,p^*,\lambda)}$ denote the ϵ for $\mathcal{D2P}$ with privacy parameters (p, p^* *and* λ) and $|\mathcal{G}_\lambda|$ denote $\min_{s \in E}(|\mathcal{G}_\lambda(s)|)$. Then, using the above substitution probabilities, we get:

$$\epsilon_{\mathcal{D2P}}^{(p,p^*,\lambda)} = \ln(1 + \frac{p^* + \frac{(1-p)(1-p^*)}{|\mathcal{G}_\lambda|}}{\frac{p(1-p^*)}{\mathcal{N}_\mathbb{E}}}) \tag{12}$$

So, when we compute using the original profile, we have $p^* = 1$, which implies $\epsilon_{\mathcal{D2P}}^{(p,1,\lambda)} = \infty$ (*no privacy*). When $p^* = 0$ in Equation 12, so all the items are replaced with some items. Then we have $\epsilon_{\mathcal{D2P}}^{(p,0,\lambda)}$ as :

$$\epsilon_{\mathcal{D2P}}^{(p,0,\lambda)} = \ln(1 + \frac{\frac{(1-p)}{|\mathcal{G}_\lambda|}}{\frac{p}{\mathcal{N}_\mathbb{E}}}) = \ln(1 + \frac{(1-p).\mathcal{N}_\mathbb{E}}{p.|\mathcal{G}_\lambda|}) \tag{13}$$

From this $\epsilon_{\mathcal{D2P}}^{(p,0,\lambda)}$, we see that when p increases, the probability to replace an item with a random item increases leading to more privacy and that is evident from the decreasing value of $\epsilon_{\mathcal{D2P}}^{(p,0,\lambda)}$ in Equation 13. When $p = 1$ in Equation 13, $\mathcal{D2P}$ achieves $\epsilon_{\mathcal{D2P}}^{(1,0,\lambda)} = 0$ (*perfect privacy*). For larger λ, the size of the groups becomes larger, hence privacy increases resulting in smaller $\epsilon_{\mathcal{D2P}}$.

4. EXPERIMENTAL EVALUATION

This section presents an exhaustive experimental evaluation of our $\mathcal{D2P}$ protocol using two real datasets namely Jester and MovieLens. In particular, we compare the recommendation quality and coverage [10] of $\mathcal{D2P}$ with that of a non-private protocol directly relying on the original user profiles. We also provide a comparison with [21], one of the closest to our work. Additionally, we discuss an item-based version of $\mathcal{D2P}$: *i-*$\mathcal{D2P}$ which we implemented and evaluated.

4.1 Experimental Setup

4.1.1 Evaluation Metrics

We measure the recommendation quality as follows: we divide the dataset into a training set (80% of the dataset trace) and a test set (20%). For each rating in the test set, a set of top recommendations is selected as the *Recommendation Set* (RS). We denote the size of the recommendation set as N. The quality is measured in terms of standard classification accuracy metrics (CAM) [30]. More precisely, we evaluate the extent to which the recommender is able to predict the content of the test set while having computed the KNN on the training set. We use $Precision$ and $Recall$ as classification accuracy metrics for they are conventionally used in top-N recommenders [6]. Table 2 shows the terms needed for defining *Precision* and *Recall*: *True Positives* (tp), *True Negatives* (tn), *False Positives* (fp), *False Negatives* (fn).

	Relevant	**Irrelevant**	**Total**
Recommended	tp	fp	$tp+fp$
Not Recommended	fn	tn	$fn+tn$
Total	$tp+fn$	$fp+tn$	N

Table 2: *Confusion Matrix for true/false positive/negative recommendations.*

Precision or *True Positive Accuracy (TPA)* is the ratio of the number of relevant recommended items to the total number of recommended items.

$$Precision = TPA = \frac{tp}{tp+fp}$$

Recall or *True Positive Rate (TPR)* is the ratio of the number of relevant recommended items to the total number of relevant items.

$$Recall = TPR = \frac{tp}{tp+fn}$$

To get an estimate of the drop in quality, we measure the decrease in precision for Top-5 recommendations [22] (denoted by Pr@5), as most recommenders follow Top-N recommendations, e.g: *IMDB* uses *Top-6* list to suggest similar movies, *Amazon* uses *Top-4* list to suggest similar products and *last.fm* uses *Top-5* list to suggest similar music.

F_1*-Score* is used to access precision and recall simultaneously. Mathematically, it is the harmonic mean of *Precision* and *Recall.*

$$F_1 - Score = 2.\frac{Precision.Recall}{Precision+Recall}$$

4.1.2 Datasets

We evaluate $\mathcal{D2P}$ with two datasets: the MovieLens (ML) dataset [23] and the Jester one [14]. The ML dataset consists of 100, 000 (100K) ratings given by 943 users over 1682 movies. The Jester dataset [14] contains 4.1 million ratings of 100 jokes from 73,421 users. We use a subset of the Jester dataset with around 36K ratings given by 500 users over 100 jokes. The Jester subset consists of 500 users selected uniformly at random among all users who rated at least 50 jokes. $\mathcal{D2P}$ relies on the item-replacement technique, so the quality of the recommendation provided by $\mathcal{D2P}$ depends on how much two items are connected in the dataset. We thus consider datasets with diverse characteristics to evaluate $\mathcal{D2P}$.

Diversity: We created 4 diverse datasets from the ML $100K$ dataset to cover a variety of characteristics (typically sparsity).

The ratings are stored in a user-item matrix where the rows of the matrix contain the user-ids and the columns contain the item-ids. Then, the rows are sorted based on the total number of ratings given by the users and the columns are sorted based on the total number of times the items have been rated by different users. The partitioning of the dataset is shown in Figure 5 as *users* $\times$ *items* matrix.

Characterization. To evaluate $\mathcal{D2P}$ in different settings, we characterize the datasets according to *Rating Density* metric. The Rating Density (RD) is the ratio of the number of ratings given by

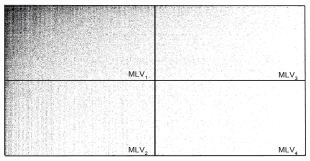


Figure 5: *ML1 Dataset Partitions based on rating density.*

Dataset	#Users	#Items	Ratings	RD(%)
Jester	500	100	36000	71.01
ML1	940	1680	99647	6.31
$\mathbf{MLV_1}$	470	840	76196	19.3
$\mathbf{MLV_2}$	470	840	16187	4.1
$\mathbf{MLV_3}$	470	840	6317	1.6
$\mathbf{MLV_4}$	470	840	750	0.19

Table 3: *Datasets characteristics.*

the users in the dataset to the total number of ratings possibly given (number of users multiplied by the number of items).

Table 3 depicts the rating densities of different datasets.

4.2 Results

4.2.1 Impact of Rating Density

Figure 6 shows the recall measured with varying size of the recommendation set in $\mathcal{D2P}$ with parameters $p = 0.5$, $p^* = 0.5$ and $\lambda = 1$. We observe that higher rating density results in better recall using $\mathcal{D2P}$. As shown in Table 3, the rating density of the Movielens 100K dataset is 6.31% and that of its 4 subsets varies with a maximum of 19.3% and minimum of about 0.19%. From Figure 6, we observe that $\mathcal{D2P}$ is not suitable for datasets with too low rating densities, like MLV_3 and MLV_4, as these result in lower $recall$. However, we observe, for MLV_2, $\mathcal{D2P}$ provides slightly better recall compared to a more dense dataset (like MLV_1). This happens because the number of items relevant to a user (in the test set) is less in MLV_2 (more sparse) compared to MLV_1 (less sparse). However, for more sparse datasets like MLV_3 or MLV_4, collaborative filtering is not effective because the ratings are insufficient to identify similarities in user interests.

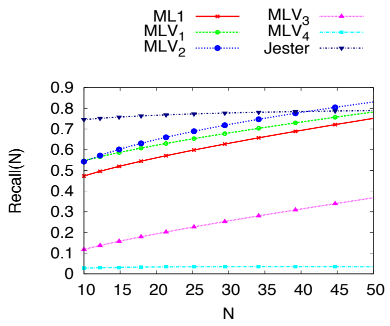


Figure 6: *Recall@N with varying Dataset Characteristics.*

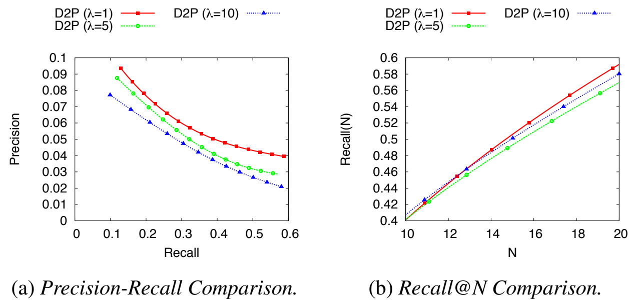
4.2.2 Privacy-Quality Trade-off

Effect of Profiler probability (p^*).
We vary the value of parameter p^* from the *Profiler* algorithm from a minimum of 0 to a maximum of 1 (*no privacy*) with other parameters $\lambda = 1$, $p = 0.5$.

Movielens. Figures 7a, 7b and 7c demonstrate the performance of the $\mathcal{D2P}$ over several values of p^* on the Movielens dataset. In Figure 7a, we observe that the quality drops only by 3.24%, in terms of Pr@5, when compared to a non-private approach ($p^* = 1$).
Jester. Figures 8a, 8b and 8c show the results of the performance of the $\mathcal{D2P}$ over several values of p^* on Jester workload. In Figure 8a, we observe that the quality drops only by 2.9% in terms of Pr@5. Interestingly, we observe in Figure 8b that the recall of a non-private approach ($p^* = 1$) is very similar to the one achieved by $\mathcal{D2P}$ (e.g, at $N = 20$, the *recall* values differ by 0.02 only). This observation also means that $\mathcal{D2P}$ provides good recommendation quality in datasets with higher rating densities. The higher the profiler probability, the better the recommendation quality.

Effect of Selector Probability (p).
Here, we vary the probability p from the *Selector* algorithm from a minimum of 0 to a maximum of 0.5 (with $\lambda = 1$, $p^* = 0$).
Movielens. Figures 9a, 9b and 9c demonstrate the performance of $\mathcal{D2P}$ over several values of p on Movielens.
Jester. Figures 10a, 10b and 10c show the results of the performance of $\mathcal{D2P}$ over several values of p on Jester dataset. The lower the selector probability, the better the recommendation quality.

Effect of distance metric (λ).
We also analyzed the effect of varying the level of privacy using the distance parameter: λ. We observed the quality of recommendations provided by $\mathcal{D2P}$ with several values of λ (with $p = 0.5$, $p^* = 0$). The results of these experiments are given in Figures 12a and 12b. We observe that a lower λ provides better quality because items gets replaced by closer items for lower λ.

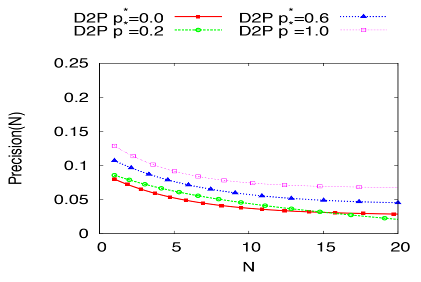


(a) *Precision-Recall Comparison.* (b) *Recall@N Comparison.*

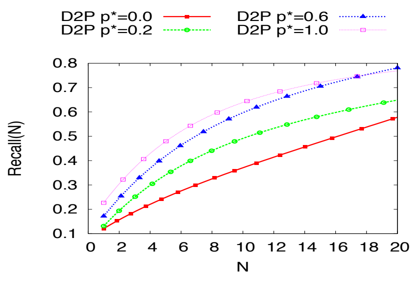
Figure 12: Effect of Distance Metric (λ) on Quality for the ML Dataset (User-based CF).

4.2.3 Parameter Selection

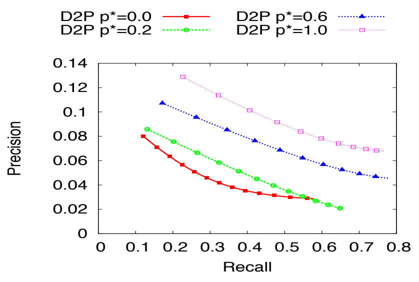
The distance parameter λ is used to protect user's privacy. We now illustrate its usage on two examples. The first one is depicted in Figure 13. We consider 3 categories (A,B,C), 3 users (U_1, U_2, U_3) and 5 movies (I_1, I_2, I_3, I_4, I_5). We assume that each user wants to hide some specific category. To hide a Category A for user U_1, we anonymize it with at least one different Category (B or C). We can achieve this by computing the minimum distance for items from Category A in U_1's profile (I_1,I_3) to items in different categories. For item I_1, we get the distance is 2.8 to I_2 in Category B and 3 to I_4 in Category C. So, the minimum distance for I_1 is 2.8 to I_2 in Category B. We get the same for I_3 in U_1's profile. Now, to satisfy the distance for both of these items, we choose the maximum among them which is 2.8. This gives us the λ_{U_1} to hide Category A for U_1. We do the same for users U_2 and U_3. Finally, to set the λ for the system, we get the maximum from all users (which is 2.8 in the example).



(a) *Precision@N Comparison.*

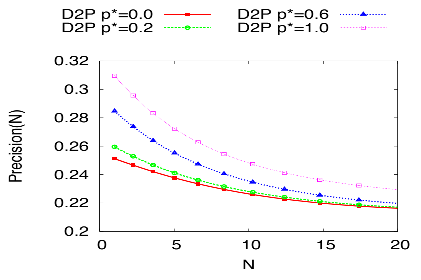


(b) *Recall@N Comparison.*

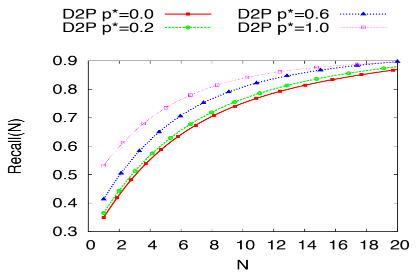


(c) *Precision-Recall Comparison.*

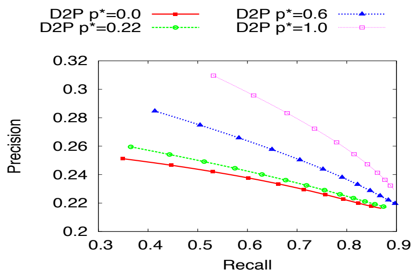
Figure 7: Effect of Profiler Probability (p^*) on Quality for the ML Dataset (User-based CF).



(a) *Precision@N Comparison.*

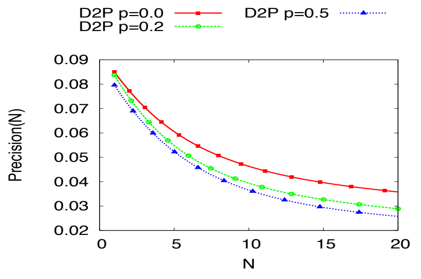


(b) *Recall@N Comparison.*

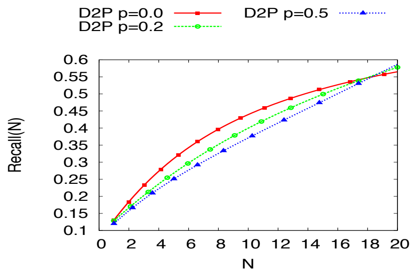


(c) *Precision-Recall Comparison.*

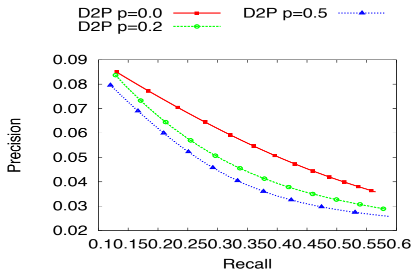
Figure 8: Effect of Profiler Probability (p^*) on Quality for the Jester Dataset (User-based CF).



(a) *Precision@N Comparison.*

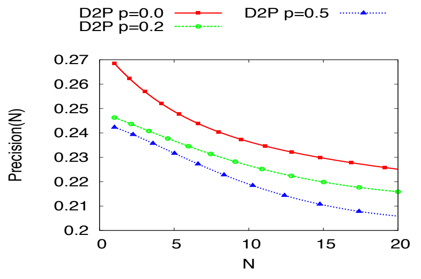


(b) *Recall@N Comparison.*

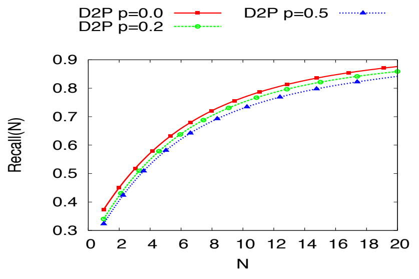


(c) *Precision-Recall Comparison.*

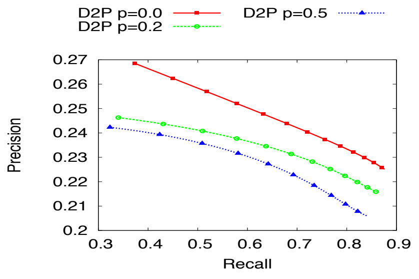
Figure 9: Effect of Selector Probability (p) on Quality for the ML Dataset (User-based CF).



(a) *Precision@N Comparison.*



(b) *Recall@N Comparison.*

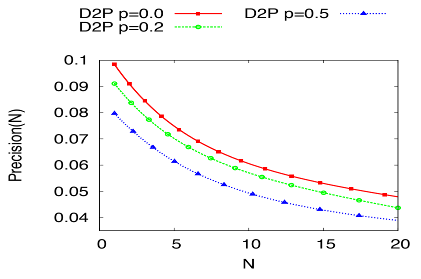


(c) *Precision-Recall Comparison.*

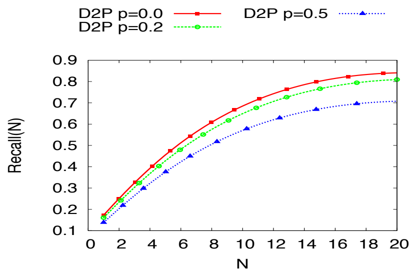
Figure 10: Effect of Selector Probability (p) on Quality for the Jester Dataset (User-based CF).

The distance parameter can be also selected as the average distance for each user profile (λ_i). Here, λ_i for user U_i is computed as the average value of the distance between all pairs of items rated by user U_i. Figure 14 provides an intuition for this distance parameter. For the datasets used for evaluation, we get $\lambda_{ML1} = 6.5$, $\lambda_{Jester} = 1.5$.

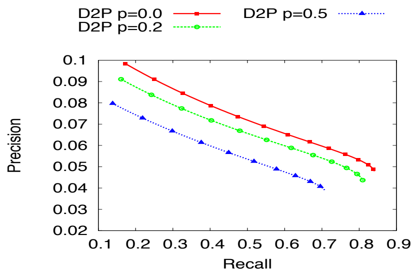
To demonstrate the degradation of ϵ based on parameters, p and p^*, we fix the distance parameter ($\lambda_{ML1} = 6.5$, $\lambda_{Jester} = 1.5$). Figure 15 demonstrates the degradation of ϵ based on the privacy parameters (p, p^*). For Movielens, we obtain good privacy (ϵ= 2.9) and good quality (F_1-Score=8.5%) with p =0.7, p^*=0.03, λ=6.5.



(a) *Precision@N Comparison.*



(b) *Recall@N Comparison.*



(c) *Precision-Recall Comparison.*

Figure 11: Effect of Selector Probability (p) on Quality for the ML Dataset (Item-based CF).

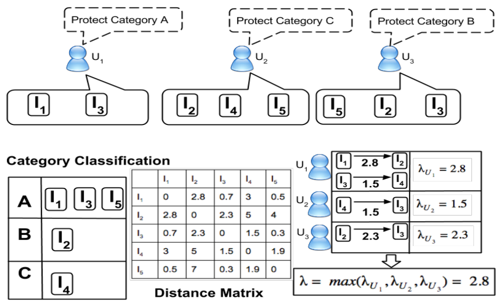


Figure 13: *Distance for Personal Choice.*

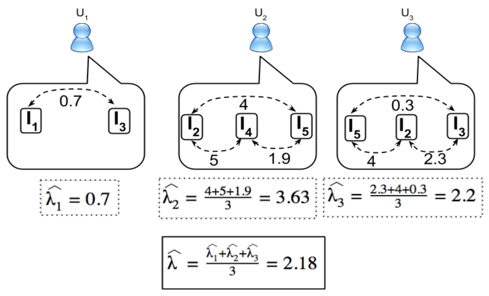


Figure 14: *Distance for Average.*

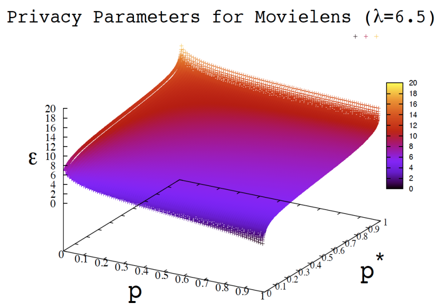
For Jester, we obtain good privacy (ϵ= 0.97) and good quality (F_1-Score=23.1%) with p =0.8, p^*=0.01, λ=1.5.

4.2.4 Coverage Evaluation

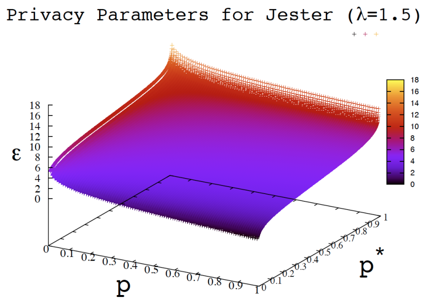
Beyond accuracy, there is a variety of other metrics that should be used to evaluate a recommender [10, 12]. The *Coverage* of a recommender is a metric that captures the domain of items over which it can make recommendations. In particular, we evaluate *Catalog Coverage* [10] of $\mathcal{D}2\mathcal{P}$ and compare it to the coverage provided by a standard non-private recommender. Consider a set of items I_K^j contained in the *Top-K* list during the j^{th} recommendation instance. Also, denote the total number of items by N. Hence, *Catalog Coverage* after M recommendation instances can be mathematically represented as follows:

$$Catalog\ Coverage = \frac{|\cup_{j=1\ldots M} I_K^j|}{N}$$

Figure 16 demonstrates the *Catalog Coverage* for $\mathcal{D}2\mathcal{P}$ and compares it with the coverage in a standard recommender for Movielens. We observe that $\mathcal{D}2\mathcal{P}$ provides 1.5 times better coverage than a standard recommender when the size of recommendation set is 1.



(a) *Privacy Parameters for Movielens (ML1).*



(b) *Privacy Parameters for Jester.*

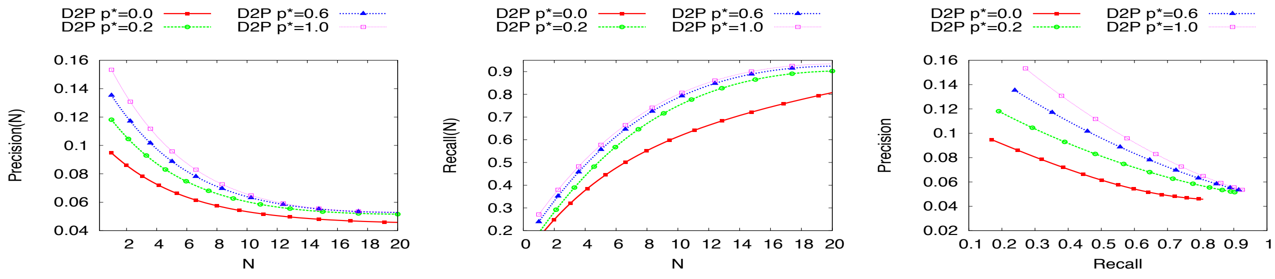
Figure 15: *Privacy Parameters Comparison.*

4.2.5 Overhead Evaluation

We evaluate here the computational overhead of $\mathcal{D}2\mathcal{P}$'s privacy and compare it to the one of [21] which we denote as DP_δ. We call the computations performed for every recommendation as *Online* computations and the computations done periodically as *Offline* computations. We compare the privacy overhead with the Recommendation Latency (RL) in $\mathcal{D}2\mathcal{P}$. Additionally, we compare the privacy overhead in $\mathcal{D}2\mathcal{P}$ with the privacy overhead in DP_δ. As shown in Table 4, the overhead for the offline computations in $\mathcal{D}2\mathcal{P}$ is around 26.4 times smaller than that of [21] for Movielens and around 4.5 times smaller for Jester. All offline computations are parallelised on a 8-core machine.

4.2.6 Item-based $\mathcal{D}2\mathcal{P}$

$\mathcal{D}2\mathcal{P}$ can be used with any collaborative filtering technique. We evaluate $\mathcal{D}2\mathcal{P}$ in another context to illustrate the genericity of $\mathcal{D}2\mathcal{P}$.



(a) *Precision@N Comparison.* (b) *Recall@N Comparison.* (c) *Precision-Recall Comparison.*

Figure 17: Effect of Profiler Probability (p^*) on Quality for the ML Dataset (Item-based CF).

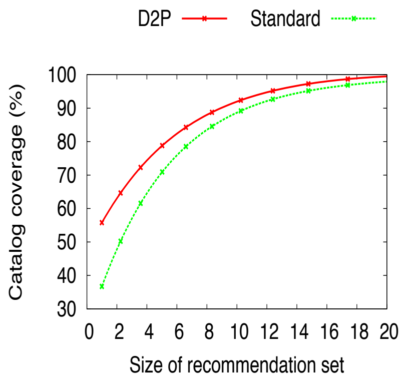


Figure 16: *Catalog Coverage Comparison.*

Datasets	$\mathcal{D}2\mathcal{P}$ Overhead			DP_δ Overhead
	RL	Online	Offline	Offline
$ML1$	196ms	32ms	4.54s	120s
$Jester$	24ms	12ms	162ms	740ms

Table 4: *Overhead of Privacy.*

We implemented an item-based version of $\mathcal{D}2\mathcal{P}$: *i*-$\mathcal{D}2\mathcal{P}$. In *i*-$\mathcal{D}2\mathcal{P}$, the grouping phase is responsible for creating groups of similar *users* based on the distance metric λ. The selector and profiler components in *i*-$\mathcal{D}2\mathcal{P}$ create *AlterReplica* profiles of the items using the same approach as in $\mathcal{D}2\mathcal{P}$. Finally, the item recommendations are computed using these *AlterReplica* profiles during the recommendation phase in *i*-$\mathcal{D}2\mathcal{P}$. Figures 11a, 11b and 11c convey the quality of recommendations provided by *i*-$\mathcal{D}2\mathcal{P}$ for varying values of parameter p (with $\lambda = 1$, $p^* = 0$). Figures 17a, 17b and 17c convey the quality of recommendations provided by *i*-$\mathcal{D}2\mathcal{P}$ for several values of parameter p^* (with $\lambda = 1$, $p = 0.5$). In Figure 17a, we observe that the quality drops by 1.89% in terms of Pr@5 for the ML dataset. This shows that $\mathcal{D}2\mathcal{P}$ also provides good quality of recommendations in item-based CF recommenders.

5. RELATED WORK

The notion of differential privacy was introduced by Cynthia Dwork [7, 8, 9]. Most of the research focused on theoretical aspects and provided feasibility and infeasibility results [15]. In this paper, we extend differential privacy to the context of recommenders. We appended the original definition with a distance metric (λ) and presented an effective way to achieve it through our $\mathcal{D}2\mathcal{P}$ protocol.

Polat et. al. [25] proposed a randomized perturbation technique to protect user's privacy. Zhang et. al. [34] showed however that a considerable amount of information can be derived from randomly perturbed ratings. Instead of adding perturbations to user profiles, $\mathcal{D}2\mathcal{P}$ uses the $AlterEgo$ profiles which are created based on a distance threshold (λ). Privacy breaches (compromised user identities) occur when e-commerce sites release their databases to third-parties for data-mining or statistical reporting [2]. The fact that with $\mathcal{D}2\mathcal{P}$, the third-parties have only access to the $AlterEgo$ profiles alleviates the risk of revealing user's identity to those third parties.

In fact, although, there had been a lot of research work related to privacy in online recommenders [16, 20] and differential privacy [7, 8, 9], only a few of these combined these two notions [13, 21]. McSherry et. al. designed a relaxed version of differential privacy in the context of recommenders [21]. In short, the idea is to add to the ratings - a limited amount of - Gaussian noise. Our notion of distance-based differential privacy provides a stronger form of classical differential privacy in the context of recommender systems. In our case, we replaced items in users profiles with others at some distance. Other differences between the two approaches include the way dynamic updates are addressed as well as the underlying overhead. McSherry et. al. does not consider updates to the covariance matrix, and hence is not applicable to a dynamic system without jeopardizing the privacy guarantee. The *AlterEgo* profiles used in $\mathcal{D}2\mathcal{P}$ can grow naturally without the need to recompute from scratch like in [21]. Also, the underlying overhead in $\mathcal{D}2\mathcal{P}$ is lower. As shown in Table 4, the overhead in $\mathcal{D}2\mathcal{P}$ is around 26.4 times smaller than that of [21] for Movielens and around 4.5 times smaller for Jester. The additional overhead in [21] stems from the compute-intensive preprocessing steps: (i) removal of per-movie global effects and (ii) centering and clamping process.

6. CONCLUDING REMARKS

While personalization has become crucial on the web, it raises however privacy concerns as its quality relies on leveraging user profiles. In this paper, we present an extension of the notion of differential privacy to the context of recommenders: systems that personalize recommendations based on similarities between users. We introduced $\mathcal{D}2\mathcal{P}$ which ensures this strong form of privacy. $\mathcal{D}2\mathcal{P}$ addresses the tradeoff between privacy and quality of recommendation: it can be applied to any collaborative recommender.

The main intuition behind $\mathcal{D}2\mathcal{P}$ is to rely on a distance metric between items so that groups of similar items can be identified. $\mathcal{D}2\mathcal{P}$ leverages this notion of group to generate, from real user profiles, alternative ones, called $AlterEgo$ profiles. These represent differentially private versions of the exact profiles. Such profiles are then used to compute the KNN and provide recommendations. We analyze $\mathcal{D}2\mathcal{P}$ and evaluate experimentally the impact of the privacy mechanism on the quality of the recommendation in the context of two datasets: MovieLens and Jester. Our results show that privacy can be ensured without significantly impacting the quality of the recommendation. Our experiments demonstrate that $\mathcal{D}2\mathcal{P}$ can provide 1.5 times better coverage than a standard recommender for Movielens. Additionally, $\mathcal{D}2\mathcal{P}$ incurs a small privatization overhead compared to other privacy-preserving system like [21] which makes it comparatively more practical for dealing with real-time workloads.

$\mathcal{D}2\mathcal{P}$ could be further extended to other filtering techniques that rely on user profiles for their recommendation computations. It would also be interesting to incorporate a hybrid approach in $\mathcal{D}2\mathcal{P}$ where the item groups would be formed using content-based filtering [33] while the actual recommendations would be made based on collaborative filtering techniques. Another interesting direction for future work is to introduce $\mathcal{D}2\mathcal{P}$ in matrix factorisation techniques [18].

One limitation of $\mathcal{D}2\mathcal{P}$ stems from the fact that the users trust the service-providers with the original user profiles. Privacy could hence be compromised by online spying on users' activities [11]. For future work, we would like to study the impact on privacy and recommendation quality of probabilistically altering or encrypting user rating [3]: the goal would be to preserve the profile anonymity even from service-providers. Combining such techniques with $\mathcal{D}2\mathcal{P}$ would result in a recommender which is robust to malicious users and even untrusted service-providers engaged in spying activities.

Acknowledgements. This work was funded by the Web Alter-Ego Google Focused Award.

7. REFERENCES

[1] Shopzilla, inc. privacy policy. 2014. http://about.bizrate.com/privacy-policy.

[2] M. S. Ackerman and D. T. Davis Jr. Privacy and security issues in e-commerce. In *New Economy Handbook*, pages 911–930. Academic Press/ Elsevier, 2003.

[3] N. Ahituv, Y. Lapid, and S. Neumann. Processing encrypted data. pages 777–780. In *CACM*, 1987.

[4] M. Bonett. Personalization of web services: opportunities and challenges. In *Ariadne Journal*, 2001.

[5] A. Boutet, D. Frey, R. Guerraoui, A.-M. Kermarrec, and R. Patra. Hyrec: Leveraging browsers for scalable recommenders. pages 85–96. In *MIDDLEWARE*, 2014.

[6] P. Cremonesi, Y. Koren, and R. Turrin. Performance of recommender algorithms on top-n recommendation tasks. pages 39–46. In *RecSys*, 2010.

[7] C. Dwork. Differential privacy. pages 1–12. In *ICALP*, 2006.

[8] C. Dwork and J. Lei. Differential privacy and robust statistics. pages 371–380. In *STOC*, 2009.

[9] A. Friedman and A. Schuster. Data mining with differential privacy. pages 493–502. In *SIGKDD*, 2010.

[10] M. Ge, C. Delgado-Battenfeld, and D. Jannach. Beyond accuracy: evaluating recommender systems by coverage and serendipity. pages 257–260. In *RECSYS*, 2010.

[11] S. Gorman. Nsa.s domestic spying grows as agency sweeps up data. In *The Wall Street Journal*, 2008.

[12] J. L. Herlocker, J. A. Konstan, L. G. Terveen, and J. T. Riedl. Evaluating collaborative filtering recommender systems. pages 5–53. In *TOIS*, 2004.

[13] P. Jain, P. Kothari, and A. Thakurta. Differentially private online learning. In *COLT*, 2011.

[14] *Jester, Online Joke Recommender System and Dataset*, 2001.

[15] S. P. Kasiviswanathan, H. K. Lee, K. Nissim, S. Raskhodnikova, and A. Smith. What can we learn privately? pages 793–826. In *SIAM*, 2011.

[16] S. Katzenbeisser and M. Petkovic. Privacy-preserving recommendation systems for consumer healthcare services. pages 889–895. In *ARES*, 2008.

[17] J. A. Konstan and J. Riedl. Recommender systems: from algorithms to user experience. pages 101–123. In *UMUAI*, 2012.

[18] Y. Koren, R. Bell, and C. Volinsky. Matrix factorization techniques for recommender systems. pages 30–37. In *IEEE Computer Journal*, 2009.

[19] T. Li and T. Unger. Willing to pay for quality personalization? trade-off between quality and privacy. pages 621–642. In *European Journal of Information Systems*, 2012.

[20] Y. Luo, J. Le, and H. Chen. A privacy-preserving book recommendation model based on multi-agent. pages 323–327. In *WCSE*, 2009.

[21] F. McSherry and I. Mironov. Differentially private recommender systems: Building privacy into the net. pages 627–636. In *SIGKDD*, 2009.

[22] R. J. Mooney and L. Roy. Content-based book recommending using learning for text categorization. pages 195–204. In *JCDL*, 2000.

[23] *MovieLens dataset*, 2003. http://grouplens.org/datasets/movielens/.

[24] A. Narayanan and V. Shmatikov. Robust de-anonymization of large sparse datasets. pages 111–125. In *SP*, 2008.

[25] H. Polat and W. Du. Privacy-preserving collaborative filtering using randomized perturbation techniques. In *ICDM*, 2003.

[26] Z. Qian, X. Chen, N. Kang, M. Chen, Y. Yu, T. Moscibroda, and Z. Zhang. Madlinq: large-scale distributed matrix computation for the cloud. pages 197–210. In *EUROSYS*, 2012.

[27] N. Ramakrishnan, B. J. Keller, B. J. Mirza, A. Y. Grama, and G. Karypis. Privacy risks in recommender systems. pages 54–62. In *Internet Computing*, 2001.

[28] A. M. Rashid, I. Albert, D. Cosley, S. K. Lam, S. M. McNee, J. A. Konstan, and J. Riedl. Getting to know you: learning new user preferences in recommender systems. pages 127–134. In *IUI*, 2002.

[29] B. Sarwar, G. Karypis, J. Konstan, and J. Riedl. Item-based collaborative filtering recommendation algorithms. pages 285–295. In *WWW*, 2001.

[30] G. Schröder, M. Thiele, and W. Lehner. Setting goals and choosing metrics for recommender system evaluations. In *UCERSTI 2*, 2011.

[31] X. Su and T. M. Khoshgoftaar. A survey of collaborative filtering techniques. page 4. In *AAI*, 2009.

[32] M. Van Dijk and A. Juels. On the impossibility of cryptography alone for privacy-preserving cloud computing. pages 1–8. In *HotSec*, 2010.

[33] R. Van Meteren and M. Van Someren. Using content-based filtering for recommendation. In *MLnet/ECML*, 2000.

[34] S. Zhang, J. Ford, and F. Makedon. Deriving private information from randomly perturbed ratings. pages 59–69. In *SIAM*, 2006.