ReuseKNN: Neighborhood Reuse for Privacy-Aware Recommendations

PETER MÜLLNER, Know-Center GmbH, Austria
MARKUS SCHEDL, Johannes Kepler University Linz, Austria
ELISABETH LEX, Graz University of Technology, Austria
DOMINIK KOWALD, Know-Center GmbH and Graz University of Technology, Austria

User-based KNN recommender systems (UserKNN) utilize the rating data of a target user’s $k$ nearest neighbors in the recommendation process. This, however, increases the privacy risk of the neighbors since their rating data might be exposed to other users or malicious parties. To reduce this risk, existing work applies differential privacy by adding randomness to the neighbors’ ratings, which reduces the accuracy of UserKNN. In this work, we introduce ReuseKNN, a novel privacy-aware recommender system. The main idea is to identify small but highly reusable neighborhoods so that (i) only a minimal set of users requires protection with differential privacy, and (ii) most users do not need to be protected with differential privacy, since they are only rarely exploited as neighbors. In our experiments on five diverse datasets, we make two key observations: Firstly, ReuseKNN requires significantly smaller neighborhoods, and thus, fewer neighbors need to be protected with differential privacy compared to traditional UserKNN. Secondly, despite the small neighborhoods, ReuseKNN outperforms UserKNN and a fully differentially private approach in terms of accuracy. Overall, ReuseKNN’s recommendation process leads to significantly less privacy risk for users than in the case of UserKNN.

Additional Key Words and Phrases: Neighborhood Reuse, Differential Privacy, Collaborative Filtering, $k$ nearest neighbors, Recommender Systems, Privacy Risk, Popularity Bias

ACM Reference Format:
Peter Müllner, Markus Schedl, Elisabeth Lex, and Dominik Kowald. 2022. ReuseKNN: Neighborhood Reuse for Privacy-Aware Recommendations. 1, 1 (June 2022), 27 pages. https://doi.org/XXXXXXX.XXXXXXX

1 INTRODUCTION

Recommender systems often rely on collaborative filtering to generate recommendations [40, 41]. User-based KNN, i.e., UserKNN, is a variant of collaborative filtering that utilizes the rating data of the $k$ nearest neighbors of a target user to process a rating query. A rating query is a request to a recommender system to predict a rating for a target user to a target item.

However, the way how rating queries are processed by UserKNN can increase the privacy risk of users since the estimated rating scores, i.e., recommendations, are generated based on rating data of users that are used as neighbors. In this regard, existing research [15, 70, 90] finds that these neighbors are susceptible to multiple privacy risks, such as the inference of their private rating data. To mitigate this privacy risk, several works [34, 56, 92] use privacy-preserving
techniques such as differential privacy (DP) [24, 26] to protect users’ rating data by adding a degree of randomness to the data. However, the added randomness typically leads to severe drops in recommendation accuracy [12].

To address this problem, we introduce ReuseKNN, a novel privacy-aware recommender system that reduces the number of neighbors to which differential privacy needs to be applied. Intuitively, instead of utilizing new users as neighbors for processing new rating queries, ReuseKNN reuses useful neighbors from past recommendations queries. Hence, ReuseKNN constructs small but highly reusable neighborhoods for every target user by fostering the neighbors’ reusability for many rating queries. With this, as illustrated in Figure 1, ReuseKNN minimizes the set of users that need to be protected with DP - we call them “vulnerable users”. Plus, most users do not need to be protected with DP, as their rating data is only rarely used in the recommendation process - we call them “secure users”. As shown, we also introduce a privacy risk threshold $\tau$, i.e., a hyperparameter that allows adjusting the maximum privacy risk for a user to be treated as secure. In this way, we leave it to the recommender system provider to specify what privacy risk is tolerated and which users need to be protected.

We evaluate our approach in a two-stage procedure: (i) neighborhood reuse only, i.e., ReuseKNN, and (ii) neighborhood reuse with DP, i.e., ReuseKNN$_{DP}$. In the first stage, ReuseKNN does not use DP at all. With this, we focus on how neighborhood reuse can increase the reusability of neighbors and preserve recommendation accuracy. In the second stage, we combine ReuseKNN with DP, i.e., ReuseKNN$_{DP}$, to protect vulnerable users with DP. This allows investigating how ReuseKNN$_{DP}$ can mitigate all users’ privacy risk while generating accurate recommendations. We evaluate ReuseKNN and ReuseKNN$_{DP}$ on five different datasets, i.e., MovieLens 1M, Douban, LastFM, Ciao, and Goodreads. Plus, we compare ReuseKNN and ReuseKNN$_{DP}$ to five baselines that utilize DP (e.g., [91]) and the concept of neighborhood reuse in different ways, with respect to recommendation accuracy and users’ privacy risk. Additionally, the nature of neighborhood reuse may raise concerns that the generated recommendations are biased towards items preferred by popular users. Thus, we investigate whether our approach is more or less prone to item popularity bias than our baselines.

Manuscript submitted to ACM
Our results indicate that ReuseKNN yields significantly smaller neighborhoods than traditional UserKNN. Despite the smaller neighborhoods, ReuseKNN and ReuseKNN_{DP} outperform our baselines in terms of recommendation accuracy. Moreover, ReuseKNN_{DP} leads to significantly less privacy risk for users than UserKNN with DP. Also, our approach does not increase item popularity bias. Overall, the three main contributions of this paper are as follows:

1. We present a novel ReuseKNN recommender system and compare several neighborhood reuse strategies to substantially foster the reusability of a target user’s neighborhood and effectively reduce the number of vulnerable users.

2. We combine ReuseKNN with DP to realize ReuseKNN_{DP} and show that ReuseKNN_{DP} improves recommendation accuracy over DP-based baselines, and at the same time, does not increase item popularity bias.

3. We show that ReuseKNN_{DP} leads to significantly less privacy risk, since most users are rarely exploited in the recommendation process and only the remaining users, i.e., vulnerable users, are protected with DP.

Our work illustrates how to address privacy risks in recommender systems through neighborhood reuse combined with DP.

2 RELATED WORK

We describe two research strands related to our work: (i) studies on the identification and quantification of users’ privacy risks in recommender systems and (ii) privacy-aware recommender systems that mitigate users’ privacy risks.

2.1 Privacy risks in recommender systems

Previous research [10, 33, 46, 74, 79] describes many severe privacy risks for users of recommender systems. For example, according to Ramakrishnan et al. [70], the use of neighbors’ rating data in the recommendation process can pose a privacy risk to the neighbors. Serendipitous recommendations could reveal unique connections, i.e., sparse relationships [29], between neighbors and items. In this way, the rating data of the neighbors can be uncovered, or the neighbors’ identities can be revealed within the recommendation database. Also, Zhang et al. [90] show that it could be possible to identify users whose data was used in the recommendation process. Their results suggest that the effectiveness of their attack depends on the number of generated recommendations. Moreover, Calandrino et al. [15] propose to generate fake users, i.e., sybils, based on limited knowledge of a victim’s data. These sybils can isolate the victim that is utilized as a neighbor and compromise its privacy. However, this attack’s effectiveness depends on the amount of knowledge of the victim’s data [30] and strategies to weaken the attacks’ effectiveness exist [56].

Privacy risk metrics. To quantify users’ privacy risks in computational systems such as recommender systems, several privacy risk metrics [54, 66, 68, 76, 80, 83] have been proposed. These metrics often rely on the sensitivity of users’ data, i.e., how strong this data puts users’ privacy at risk. For example, Chen et al. [19] detect correlations within the dataset to measure if a piece of data could reveal personal information about the users. Srivastava et al. [76] measure the relative sensitivity of a single piece of data compared to the remaining data of a user. Similarly, Domingo-Ferrer [23] relates the overall sensitivity of a user’s data to the sensitivity of other users’ data. Besides, Liu and Terzi’s privacy score [54] weighs the sensitivity with the degree of visibility of a user’s data (i.e., how often a user’s data is utilized in the recommendation process).

Separation to our work. Evaluating the privacy risk of users based on attacks only measures the privacy risk with respect to the specific attack scenario. Liu and Terzi’s metric measures users’ privacy risk independent of specific attack
scenarios and, thus, allows investigating privacy risk in a recommender system at a more general level. Therefore, in our work, we utilize Liu and Terzi’s metric to measure users’ privacy risk in a general neighborhood-based recommendation scenario. Furthermore, we assume that all pieces of data are equally sensitive, since sensitivity typically depends on the application and the user’s perception of privacy [47, 51, 61].

2.2 Privacy-aware recommender systems

Several works [7, 9, 16, 89] mitigate users’ privacy risks by applying homomorphic encryption [35] to users’ rating data. Here, recommendations are generated based on the encrypted rating data, and thus, users’ rating data remains protected in the recommendation process. Homomorphic encryption, however, has high computational complexity. As a remedy, Kikuchi et al. [50] describe a variant of homomorphic encryption with lower privacy guarantees but improved efficiency. Also, Tang et al. [78] apply homomorphic encryption on the rating data of a target users’ friends, i.e., a small subset of users, to improve computational efficiency. Besides homomorphic encryption, federated learning [59] is used to lower users’ privacy risks [5, 9, 18, 38, 53, 71]. Specifically, not a user’s rating data is utilized in the recommendation process, but parameters of a user’s local recommendation model. However, federated learning could be susceptible to attacks aiming to infer user data [65]. Thus, research proposes to learn a user’s local model by utilizing only a minimal [6, 64] or a user-selected subset of the rating data [20, 88]. Moreover, differential privacy (DP) [24, 26] and perturbation techniques [55, 69] are leveraged for collaborative filtering recommender systems [3, 17, 34, 56, 92] with a focus on matrix factorization [13, 32, 45, 60]. These techniques add randomness to users’ data to hide the actual data. Therefore, they face a trade-off between accuracy and privacy (e.g., [12]). To address this trade-off, Xin and Jaakkola [86] assume a moderate number of public users who tolerate disclosing their rating data. With this unprotected rating data, recommendation accuracy can be preserved while the privacy requirements of the remaining users are respected.

Popularity bias. Reusage of the same neighborhood for many rating queries may raise concerns that the generated recommendations are biased towards items of users that prefer popular items. Specifically, with this popularity bias, users receive more recommendations for popular items, and non-popular items receive less exposure than popular items. This disparate, i.e., unfair, treatment of popular and non-popular items has been thoroughly discussed in recent research [49, 57, 58, 62]. In general, privacy and fairness are seen as two fundamental challenges of recommender systems research [22, 63]. For example, Dwork et al. [25] and Zemel et al. [87] show that formally, there is a close connection between fairness and DP. However, the sole application of DP is not sufficient to ensure fairness due to correlations within the dataset [27]. Moreover, Ekstrand et al. [27] and Agarwal [4] highlight a trade-off between user privacy and algorithmic fairness. In detail, unfairness, such as, e.g., popularity bias, arises due to the fact that algorithms have difficulties to personalize when applied on data that is protected through DP [8, 27]. Overall, related work suggests that DP can severely impact recommendations in different ways, for example, result in popularity bias. Therefore, we believe that it is important to test our approach, i.e., ReuseKNN, for item popularity bias.

Separation to our work. Similar to our work, previous research by Zhu et al. [91] prevents the inference of neighbors’ rating data by applying DP to the users’ rating data in UserKNN. However, to preserve recommendation accuracy, Zhu et al. vary the degree of randomness that is added to all users’ rating data based on the sensitivity of the data. In contrast, ReuseKNN preserves recommendation accuracy by adding randomness only where it is necessary, i.e., to vulnerable users with a high privacy risk. In the remainder of the paper, we denote one variant of the fully differentially private approach by Zhu et al. FullDP and use it as a baseline for our experiments.
Fig. 2. Schematic illustration of the recommendation process for three rating queries in Alice’s query set $Q_{Alice}$ for UserKNN and ReuseKNN. For simplicity, we set all users’ privacy risk to $\tau - 1$. Furthermore, a green shaded item indicates that the rating score for this item is estimated for the target user and a red shaded item indicates that the rating score of a neighbor has been utilized for the rating estimation. Traditional UserKNN selects those users as neighbors, that rated the queried item and have the highest similarity-value; in this toy example, those are Bob and Amy. Thus, Bob and Amy are vulnerable and need to be protected with DP. In contrast, ReuseKNN utilizes Tim as neighbor. As such, ReuseKNN substantially increases reusability (5.15 instead of 1.2 and 0.74), at the price of a slightly reduced similarity (0.90 instead of 0.98 and 0.97). This way, only Tim is vulnerable and is the only neighbor that needs to be protected with DP, as Bob and Amy remain unused.

3 APPROACH

In the following, we first describe traditional UserKNN (Section 3.1) and then outline the four neighborhood reuse strategies of our ReuseKNN recommender system (Section 3.2). Finally, we present ReuseKNN$_{DP}$, i.e., ReuseKNN with differential privacy (DP) (Section 3.3). Moreover, a summary of the notation used in this paper is given in Table 1.

In Figure 2, we provide a schematic illustration of UserKNN’s and ReuseKNN’s recommendation process based on a simple example. For simplicity, we set all users’ privacy risk to $\tau - 1$. This means that with one additional rating query for which Bob, Amy, or Tim are utilized as neighbors, their privacy risk reaches the privacy risk threshold $\tau$ and thus, DP needs to be applied. To process Alice’s rating queries for items $i_l$ and $i_m$, UserKNN selects Bob and Amy as...
neighbors, as they have the highest similarity-values across all users that rated the queried items. This causes Bob’s and Amy’s privacy risk to exceed the privacy risk threshold $\tau$, which means that both are vulnerable, and, thus, need to be protected with DP. For the rating query for item $i_n$, again, Amy is utilized in the recommendation process. Since she is already protected with DP, her privacy risk does not increase. This is different to how ReuseKNN processes rating queries. For the rating queries for items $i_l$, $i_m$, and $i_n$, ReuseKNN selects Tim as neighbor, as Tim has a substantially higher reusability-value and only marginally smaller similarity than Bob and Amy. By using Tim for many rating queries, Tim’s privacy risk exceeds $\tau$, and DP is needed to protect Tim.

In summary, in this simple example, UserKNN leads to two vulnerable users, i.e., Bob and Amy, that need to be protected with DP. In contrast, ReuseKNN leads to only one vulnerable user, i.e., Tim, to which DP has to be applied.

### 3.1 UserKNN

Typically, a user-based KNN recommender system $R^k$, i.e., UserKNN, generates an estimated rating score for a rating query of a target user $u$ and a target item $i$ by utilizing the ratings of $k$ other users that have rated $i$, i.e., the $k$ nearest neighbors, as they have the highest similarity-values across all users that rated the queried items. This causes Bob’s and Amy’s privacy risk to exceed the privacy risk threshold $\tau$, which means that both are vulnerable, and, thus, need to be protected with DP. For the rating query for item $i_n$, again, Amy is utilized in the recommendation process. Since she is already protected with DP, her privacy risk does not increase. This is different to how ReuseKNN processes rating queries. For the rating queries for items $i_l$, $i_m$, and $i_n$, ReuseKNN selects Tim as neighbor, as Tim has a substantially higher reusability-value and only marginally smaller similarity than Bob and Amy. By using Tim for many rating queries, Tim’s privacy risk exceeds $\tau$, and DP is needed to protect Tim.

In summary, in this simple example, UserKNN leads to two vulnerable users, i.e., Bob and Amy, that need to be protected with DP. In contrast, ReuseKNN leads to only one vulnerable user, i.e., Tim, to which DP has to be applied.

### 3.1 UserKNN

Typically, a user-based KNN recommender system $R^k$, i.e., UserKNN, generates an estimated rating score for a rating query of a target user $u$ and a target item $i$ by utilizing the ratings of $k$ other users that have rated $i$, i.e., the $k$ nearest neighbors, as they have the highest similarity-values across all users that rated the queried items. This causes Bob’s and Amy’s privacy risk to exceed the privacy risk threshold $\tau$, which means that both are vulnerable, and, thus, need to be protected with DP. For the rating query for item $i_n$, again, Amy is utilized in the recommendation process. Since she is already protected with DP, her privacy risk does not increase. This is different to how ReuseKNN processes rating queries. For the rating queries for items $i_l$, $i_m$, and $i_n$, ReuseKNN selects Tim as neighbor, as Tim has a substantially higher reusability-value and only marginally smaller similarity than Bob and Amy. By using Tim for many rating queries, Tim’s privacy risk exceeds $\tau$, and DP is needed to protect Tim.

In summary, in this simple example, UserKNN leads to two vulnerable users, i.e., Bob and Amy, that need to be protected with DP. In contrast, ReuseKNN leads to only one vulnerable user, i.e., Tim, to which DP has to be applied.

### 3.1 UserKNN

Typically, a user-based KNN recommender system $R^k$, i.e., UserKNN, generates an estimated rating score for a rating query of a target user $u$ and a target item $i$ by utilizing the ratings of $k$ other users that have rated $i$, i.e., the $k$ nearest neighbors, as they have the highest similarity-values across all users that rated the queried items. This causes Bob’s and Amy’s privacy risk to exceed the privacy risk threshold $\tau$, which means that both are vulnerable, and, thus, need to be protected with DP. For the rating query for item $i_n$, again, Amy is utilized in the recommendation process. Since she is already protected with DP, her privacy risk does not increase. This is different to how ReuseKNN processes rating queries. For the rating queries for items $i_l$, $i_m$, and $i_n$, ReuseKNN selects Tim as neighbor, as Tim has a substantially higher reusability-value and only marginally smaller similarity than Bob and Amy. By using Tim for many rating queries, Tim’s privacy risk exceeds $\tau$, and DP is needed to protect Tim.

In summary, in this simple example, UserKNN leads to two vulnerable users, i.e., Bob and Amy, that need to be protected with DP. In contrast, ReuseKNN leads to only one vulnerable user, i.e., Tim, to which DP has to be applied.

### 3.1 UserKNN

Typically, a user-based KNN recommender system $R^k$, i.e., UserKNN, generates an estimated rating score for a rating query of a target user $u$ and a target item $i$ by utilizing the ratings of $k$ other users that have rated $i$, i.e., the $k$ nearest neighbors, as they have the highest similarity-values across all users that rated the queried items. This causes Bob’s and Amy’s privacy risk to exceed the privacy risk threshold $\tau$, which means that both are vulnerable, and, thus, need to be protected with DP. For the rating query for item $i_n$, again, Amy is utilized in the recommendation process. Since she is already protected with DP, her privacy risk does not increase. This is different to how ReuseKNN processes rating queries. For the rating queries for items $i_l$, $i_m$, and $i_n$, ReuseKNN selects Tim as neighbor, as Tim has a substantially higher reusability-value and only marginally smaller similarity than Bob and Amy. By using Tim for many rating queries, Tim’s privacy risk exceeds $\tau$, and DP is needed to protect Tim.

In summary, in this simple example, UserKNN leads to two vulnerable users, i.e., Bob and Amy, that need to be protected with DP. In contrast, ReuseKNN leads to only one vulnerable user, i.e., Tim, to which DP has to be applied.
neighbors $N_{u,i}^k$:

$$R^k(u, i) = \frac{\sum_{n \in N_{u,i}^k} \text{sim}(u, n) \cdot r_{n,i}}{\sum_{n \in N_{u,i}^k} \text{sim}(u, n)}$$  \hspace{1cm} (1)$$

where $\text{sim}(u, n)$ is a similarity metric (e.g., cosine similarity) between target user $u$ and neighbor $n$. For UserKNN, the neighborhood $N_{u,i}^k$ used for generating recommendations for target user $u$ and item $i$, comprises the $k$ most similar neighbors:

$$N_{u,i}^k = \arg \max_{c \in U_i} \text{sim}(u, c)$$  \hspace{1cm} (2)$$

where $U_i$ are all users that have rated item $i$, $\text{sim}$ measures the similarity between two users. ReuseKNN differs from traditional UserKNN in exactly this point, i.e., how the $k$ neighbors are selected. In contrast to UserKNN, ReuseKNN also incorporates the neighbors’ reusability for many rating queries of a target user, in addition to their similarity.

### 3.2 ReuseKNN

The key feature of ReuseKNN is to reuse neighbors from a target user $u$’s previous rating queries to minimize the cardinality of the neighborhood $N_u = \bigcup_{i \in Q_u} N_{u,i}^k$ across all rating queries $Q_u$. As illustrated in Figure 1, this means that ReuseKNN decreases the privacy risk of most users, i.e., secure users, and decreases the number of highly reused neighbors, i.e., vulnerable users.

In addition to the similarity, ReuseKNN also considers the extent to which a target user $u$ could reuse candidate neighbor $c$ as a neighbor for many rating queries, i.e., $\text{reusability}(c|u)$. Since both, similarity and reusability scores are differently distributed across their respective numeric ranges, we convert them to ranks. Formally, for a function $f$, i.e., similarity or reusability, it holds that $\text{rank}(u) > \text{rank}(v)$ if $f(u) > f(v)$ for users $u$ and $v$. In case that $f(u) = f(v)$, we assign both users the same rank, i.e., $\text{rank}(u) = \text{rank}(v)$. With this, the $k$ neighbors $N_{u,i}^k$ are selected based on similarity and also, reusability, i.e., their merit in improving privacy. Formally:

$$N_{u,i}^k = \arg \max_{c \in U_i} \left[ \text{rank}(\text{sim}(u, c)) + \text{rank}(\text{reusability}(c|u)) \right]$$  \hspace{1cm} (3)$$

where $U_i$ are all users that rated item $i$, $\text{sim}$ measures the similarity between two users, and $\text{reusability}$ depends on the given neighborhood reuse strategy of ReuseKNN. In case multiple candidate neighbors have equal values for $\text{rank}(\text{sim}(u, c)) + \text{rank}(\text{reusability}(c|u))$ we choose these neighbors at random.

To estimate a candidate neighbor’s reusability, ReuseKNN utilizes our four neighborhood reuse strategies given in Figure 3. Implicit strategies create an artificial preference for selecting reusable neighbors by balancing candidate neighbors’ similarity and reusability, as given in Equation 3. With explicit strategies, the recommender system prioritizes neighbors from previous rating queries over new neighbors. Hence, the neighborhood is only expanded when the pool of previous neighbors, i.e., the neighborhood, is insufficient to generate a new recommendation. Unpersonalized strategies measure a candidate neighbor’s reusability for an average target user, whereas personalized strategies measure the reusability for a specific target user.

#### 3.2.1 Implicit Neighborhood Reuse

In the following, we discuss two strategies to increase the reusability of a target user’s neighbors based on implicit neighborhood reuse.

**Unpersonalized neighborhood reuse: Expect.** The more users rated an item, the more likely it is that a random target user will query a rating estimation for this item. Following this intuition, Expect promotes candidate neighbors that rated
Fig. 3. Overview of our four neighborhood reuse strategies. For every strategy, there also exists a variant that protects vulnerable users with DP (in green). Implicit strategies create an artificial preference towards reusable neighbors, whereas explicit strategies only expand the neighborhood, if the neighborhood is insufficient to generate a new recommendation. Unpersonalized strategies measure a candidate neighbor’s reusability for an average target user, whereas personalized strategies measure the candidate neighbor’s reusability for a specific target user.

many popular items and penalizes candidate neighbors that either rated only a few items or a lot of unpopular items. For Expect, the reusability score of candidate neighbor $c$ is defined by:

$$\text{reusability}(c|u) = \sum_{i \in I_c} \frac{|U_i|}{|U|}$$

where $u$ is the target user, $I_c$ are the items $c$ rated, $U_i$ are the users that rated an item $i$, and $U$ is the set of all users. In this case, $\text{reusability}(c)$ is the summed-up popularity of $c$’s rated items and measures the expected number of a random user’s rating queries for which $c$ could be used as a neighbor. This means that the reusability of a candidate neighbor is estimated for an average user and not for a specific target user (i.e., unpersonalized).

**Personalized neighborhood reuse: Gain.** In contrast to unpersonalized neighborhood reuse, Gain measures a candidate neighbor’s reusability for a specific target user. In detail, Gain quantifies how many of a target user’s ratings a candidate neighbor could have covered in the past, i.e., how many ratings the target user could have gained from the candidate neighbor. Thus, Gain gives the fraction of a target user $u$’s rated items, for which a candidate neighbor $c$ could have served as a neighbor:

$$\text{reusability}(c|u) = \frac{|I_u \cap I_c|}{|I_u|}$$

where $I_u$ are the items rated by $u$ and $I_c$ are the items rated by $c$. In contrast to our unpersonalized Expect strategy, Gain’s reusability score depends on a specific target user (i.e., personalized).

3.2.2 Explicit Neighborhood Reuse. Explicit neighborhood reuse strategies extend our implicit neighborhood reuse strategies by prioritizing previous neighbors over new neighbors. This extension results in two explicit neighborhood reuse strategies: Expect+Reuse (unpersonalized), and Gain+Reuse (personalized).

Specifically, explicit neighborhood reuse strategies only expand a user’s neighborhood if it comprises too few neighbors to generate a recommendation. More formally, if $k$ neighbors are necessary to generate an estimated rating score for an item $i$ and the neighborhood of $u$, i.e., $N_u = \bigcup_{i \in Q_u} N_{u,i}^k$ among all rating queries $Q_u$, already includes at least $k$ neighbors that rated $i$, i.e., $|N_{u,i}| \geq k$, then ReuseKNN does not add any new neighbors and selects those users...
as neighbors that are already in $N_{u,i}$:

$$N^k_{u,i} = \arg\max_{n \in N_{u,i}} \text{sim}(u, n)$$ (6)

If too few neighbors could be reused for the rating query for item $i$, i.e., $|N_{u,i}| < k$, then only the minimal required number of new neighbors $k' = \max\{k - |N_{u,i}|, 0\}$ is added by using the reusability($c|u$) definition of either our Expect (cf. Equation 4) or Gain (cf. Equation 5) strategy. Formally:

$$N^{k'}_{u,i} = N_{u,i} \cup \arg\max_{c \in U \setminus N_{u,i}} \left[ \text{rank}\left(\text{sim}(u, c)\right) + \text{rank}(\text{reusability}(c|u)) \right]$$ (7)

where $U_i$ are all users that rated item $i$. In this way, only $k'$ new neighbors are selected as in implicit neighborhood reuse strategies, i.e., by balancing similarity and reusability (cf. Equation 3).

### 3.3 ReuseKNN$\_\text{DP}$

ReuseKNN leads to a minimal number of highly reused neighbors, i.e., vulnerable users, with a privacy risk that exceeds the privacy risk threshold $\tau$. ReuseKNN$\_\text{DP}$ addresses this high privacy risk of vulnerable users by adding DP to our neighborhood reuse strategies in Figure 3. Specifically, for a rating query for user $u$ and item $i$, a privacy mechanism $m_{\text{DP}}$ is applied to the ratings for $i$ of vulnerable users $V$ that are used as neighbors, i.e., $\hat{R}_V = \{m_{\text{DP}}(r_{n,i}) : n \in N^k_{u,i} \cap V\}$. In this way (see Equation 8), ReuseKNN$\_\text{DP}$ utilizes real ratings of secure users $S$, i.e., $R_S = \{r_{n,i} : n \in N^k_{u,i}\}$, plus the modified ratings $\hat{R}_V$ of vulnerable users, to generate the estimated rating score $R^k(u, i)$.

$$R^k(u, i) = \frac{\sum_{n \in N^k_{u,i} \cap S} \text{sim}(u, n) \cdot r_{n,i} + \sum_{n \in N^k_{u,i} \cap V} \text{sim}(u, n) \cdot m_{\text{DP}}(r_{n,i})}{\sum_{n \in N^k_{u,i}} \text{sim}(u, n)}$$ (8)

In detail, privacy mechanism $m_{\text{DP}}$ utilizes randomized responses [36, 84] to achieve DP [26]. With this, intuitively, neighbors can plausibly deny that their real rating was used in the recommendation process. Privacy mechanism $m_{\text{DP}}$ flips a fair coin and if the coin is heads, the vulnerable neighbor’s real rating is utilized in the recommendation process. If the coin is tails, $m_{\text{DP}}$ flips a second fair coin to decide whether to utilize the vulnerable neighbor’s real rating or a random rating drawn from a uniform distribution over the range of ratings. With this, an attacker is unaware whether the utilized rating is real, or random, which leads to the privacy-guarantees within the DP-framework [26].

### 4 EXPERIMENTAL SETUP

We utilize a two-stage evaluation procedure to compare and evaluate the four neighborhood reuse strategies of (i) ReuseKNN, and (ii) ReuseKNN$\_\text{DP}$ (cf. Figure 3):

**Neighborhood reuse only: ReuseKNN.** In the first stage, we evaluate ReuseKNN without protecting vulnerable neighbors with DP in order to better understand the advantages and disadvantages of different neighborhood reuse strategies. Hence, we compare Expect, Gain, Expect+Reuse, and Gain+Reuse, and with this, we distill the impact of neighborhood reuse for recommendations.

**Neighborhood reuse with DP: ReuseKNN$\_\text{DP}$.** In the second stage, we combine ReuseKNN with DP to protect vulnerable users, i.e., ReuseKNN$\_\text{DP}$, and compare the neighborhood reuse strategies Expect$\_\text{DP}$, Gain$\_\text{DP}$, Expect+Reuse$\_\text{DP}$, and Gain+Reuse$\_\text{DP}$. With this, we investigate how ReuseKNN$\_\text{DP}$ can address the trade-off between accuracy and privacy.
Table 2. Overview of the seven evaluation metrics used in this work. \(\downarrow\) indicates that lower values are better and \(\uparrow\) indicates that higher values are better.

| Evaluation Criterion          | Evaluation Metric | Objective | Short description                                      |
|-------------------------------|-------------------|-----------|-------------------------------------------------------|
| Neighborhood Growth           | Neighbors@\(q\)   | \(\downarrow\) | Neighborhood size after \(q\) queries                |
| Neighborhood Reliability      | CoRatings@\(q\)   | \(\uparrow\) | No. of co-rated items after \(q\) queries            |
| Accuracy                      | MAE@\(k\)         | \(\downarrow\) | Mean absolute error                                   |
| Privacy Risk                  | \(|V|\)            | \(\downarrow\) | Percentage of vulnerable users                        |
|                               | PrivacyRisk_{DPP}@\(k\) | \(\downarrow\) | Privacy risk when DP is applied                       |
| Popularity Bias               | PP-Corr@\(k\)     | \(\downarrow\) | Pearson positivity-popularity correlation             |
|                               | Coverage@\(k\)    | \(\uparrow\) | Percentage of item coverage                           |

4.1 Baselines

We compare ReuseKNN and ReuseKNN_{DPP} to in total five different baselines. Concretely, for ReuseKNN, i.e., neighborhood reuse only, we use two non-DP baselines:

1. UserKNN [41]: Traditional UserKNN without neighborhood reuse. No users are protected with DP. This corresponds to a privacy risk threshold \(\tau = \infty\).
2. UserKNN+Reuse: A variant of UserKNN [41] with explicit neighborhood reuse. Neighbors from previous rating queries are prioritized over new neighbors.

For ReuseKNN_{DPP}, i.e., neighborhood reuse with DP, we use three DP baselines:

1. UserKNN_{DPP}: A variant of UserKNN without neighborhood reuse, but DP is applied only to vulnerable users - in contrast to Full_{DPP} [91], where it is applied to all users. The privacy threshold \(\tau\) is set to the same value as in the case of our ReuseKNN_{DPP} variants (see Section 4.5).
2. UserKNN+Reuse_{DPP}: Explicit neighborhood reuse is combined with UserKNN_{DPP}. Neighbors from previous rating queries are prioritized over new neighbors. DP is applied to only vulnerable users. Please see Section 4.5 for the exact \(\tau\) values per dataset.
3. Full_{DPP}: Fully differentially private UserKNN without neighborhood reuse, based on [91]. DP is applied to all users - in contrast to UserKNN_{DPP}, where it is applied only to vulnerable users. This corresponds to \(\tau = 0\).

In addition to our three DP baselines, we use our non-DP UserKNN to evaluate ReuseKNN_{DPP}. With this, we can compare ReuseKNN_{DPP} to two contrastive baselines: Full_{DPP}, which protects all users with DP, and UserKNN, which does not protect any user with DP.

4.2 Evaluation Criteria

We test our approach using the following evaluation criteria and metrics (see Table 2 for an overview):

4.2.1 Neighborhood Growth. For every rating query of a target user \(u\), \(k\) neighbors are required to generate the recommendation. In the worst case, no neighbors from previous rating queries can be reused. Thus, after \(q\) queries, \(|N_u| = \min\{q \cdot k, |U| - 1\}\) for \(U\) being the set of all users. In the best case, \(u\) reuses the same \(k\) neighbors for all \(q\) queries, i.e., \(|N_u| = k\). To quantify how many of \(u\)’s neighbors are reused, we measure the size of \(u\)’s neighborhood after \(q\) rating queries...
queries:
\[
\text{Neighbors}@q(u) = |N_u^{(q)}|
\]  \hspace{1cm} (9)

where \(N_u^{(q)}\) is \(u\)’s set of neighbors after \(q\) rating queries. With that, we test how well the neighborhood reuse strategies of ReuseKNN, i.e., neighborhood reuse only, can reuse a target user’s neighbors for many rating queries.

4.2.2 Neighborhood Reliability. As discussed earlier, our ReuseKNN recommender system limits the growth of a target user’s neighborhood. However, this utilization of fewer neighbors across many rating queries might impact the accuracy of recommendations. Therefore, we test if a target user’s neighborhood includes neighbors that are beneficial for recommendation accuracy, i.e., “reliable” neighbors. As several works suggest, one important characteristic of these reliable neighbors is the number of co-rated items with the target user \([2, 40, 67]\). Thus, we measure the average number of co-rated items between a target user \(u\) and its neighbors \(n \in N_u\) after \(q\) rating queries:
\[
\text{CoRatings}@q(u) = \frac{1}{|N_u^{(q)}|} \sum_{n \in N_u^{(q)}} |I_u \cap I_n|\]

where \(I_u\) are the items rated by target user \(u\) and \(I_n\) are the items rated by neighbor \(n\). With this, we test how reliable the neighborhoods are for generating accurate recommendations.

4.2.3 Accuracy. To quantify the accuracy of a target user’s recommendations, we rely on the widely-used mean absolute error metric (MAE) \([85]\). According to \([40, 41]\), the number of neighbors \(k\) has an impact on the recommendation accuracy. Thus, we test the accuracy of \(u\)’s recommendations for \(k \in \{5, 10, 15, 20, 25, 30\}\). Therefore, MAE@\(k(u)\) quantifies the accuracy of \(u\)’s recommendations when \(k\) neighbors are used to generate a recommendation. More formally:
\[
\text{MAE}@k(u) = \frac{1}{|R_u^{\text{test}}|} \sum_{r_{u,i} \in R_u^{\text{test}}} |r_{u,i} - R^k(u, i)|\]

where the predicted rating score \(R^k(u, i)\) is compared to the real rating scores \(r_{u,i} \in R_u^{\text{test}}\) in \(u\)’s test set. We note that the items for which \(R_u^{\text{test}}\) comprises ratings are the ones that are in \(u\)’s set of rating queries \(Q_u\). We use the MAE@\(k(u)\) metric for evaluating both, ReuseKNN, i.e., neighborhood reuse only, and ReuseKNN\(DP\), i.e., neighborhood reuse with DP.

4.2.4 Privacy Risk. Liu and Terzi \([54]\) provide a framework to measure a user’s privacy risk in computational systems such as recommender systems based on the visibility of the user’s data. In our work, we relate this visibility to how often a user’s rating data was utilized in the recommendation process. As such, our PrivacyRisk@\(k(u)\) metric counts for how many rating queries a user \(u\) was used as a neighbor. Similar to MAE@\(k(u)\), we also relate \(u\)’s privacy risk to the number of neighbors \(k\) used to generate recommendations. Formally:
\[
\text{PrivacyRisk}@k(u) = \sum_{u \in U} \sum_{i \in Q_v} \mathbb{1}_{N_u^{(i)}(u)} (u)
\]

where \(U\) is the set of all users, \(Q_v\) is the set of items for which user \(v\) queries estimated ratings, and \(\mathbb{1}_{N_u^{(i)}(u)}\) is the indicator function of user \(u\) being in \(v\)’s set of neighbors \(N_u^{(i)}\) for an item \(i\).

Percentage of Vulnerable Users. As mentioned earlier, the main goal of neighborhood reuse is to reduce the number of users that need to be protected with DP. Our PrivacyRisk@\(k\) definition allows us to identify these vulnerable users \(V\).
i.e., the set of users whose privacy risk exceeds the adjustable privacy risk threshold \( \tau \) given by:

\[
V = \{ u \in U : \text{PrivacyRisk}_k(u) > \tau \}
\]

(13)

where \( U \) is the set of all users. Thus, the percentage of vulnerable users relates to what fraction of users DP has to be applied to (i.e., \( |V|/|U| \)). We use this metric to evaluate ReuseKNN, i.e., neighborhood reuse only.

**Privacy Risk When DP is Applied.** We apply DP to a user \( u \)’s data if \( \text{PrivacyRisk}_k(u) > \tau \) (cf. Section 3.3). In practice, the privacy risk of protected neighbors cannot increase any further. To account for this, we set the upper limit of a user \( u \)’s privacy risk when DP is applied to \( \tau \), i.e.,

\[
\text{PrivacyRisk}_{DP}_k(u) = \min[\tau, \text{PrivacyRisk}_k(u)]
\]

(14)

We use \( \text{PrivacyRisk}_{DP}_k(u) \) to measure the users’ privacy risk when neighborhood reuse is combined with DP, i.e., \( \text{ReuseKNN}_{DP} \).

### 4.2.5 Item Popularity Bias

One might be concerned that reusing the same set of neighbors for many rating queries could lead to using only users as neighbors who have rated many popular items, which could result in more positive estimated rating scores for popular items. To test for this item popularity bias, we analyze all items for which the recommender system estimates high rating scores. In the remainder of this work, we refer to these items as “top items”. For a recommender system model \( R \) and \( k \) neighbors, a user \( u \)’s set of top items is given by:

\[
R^k_{\text{top}}(u) = \arg \max_{i \in Q_u} R^k(u, i)
\]

where \( Q_u \) are the items in user \( u \)’s query set. In our case, we set \( n = 10 \).

**Positivity-Popularity Correlation.** To study if \( \text{ReuseKNN} \) exhibits a pattern in which higher estimated rating scores are given to popular items, we follow the methodology of Kowald et al. [52]. Specifically, we correlate an item’s popularity with the number of times the item occurs in users’ set of positive items:

\[
\text{ItemFreq}_{\text{top}}^k(i) = \sum_{u \in U} 1_{R^k_{\text{top}}(u)}(i)
\]

(15)

where \( 1_{R^k_{\text{top}}(u)}(i) \) indicates whether item \( i \) is in user \( u \)’s set of top items \( R^k_{\text{top}}(u) \). Plus, the popularity of item \( i \) is given by: \( \text{ItemPop}(i) = |U_i|/|U| \), where \( U \) is the set of all users and \( U_i \) are the users that rated \( i \). We compute the Pearson correlation coefficient [11] between the two variables \( \text{ItemFreq}^k_{\text{top}} \) and \( \text{ItemPop}^k \) to identify item popularity bias:

\[
\text{PP-Corr}^k = \frac{\sigma_{\text{ItemFreq}^k_{\text{top}} \cdot \text{ItemPop}^k}}{\sigma_{\text{ItemFreq}^k_{\text{top}}} \cdot \sigma_{\text{ItemPop}^k}}
\]

(16)

where \( \sigma_{\text{ItemFreq}^k_{\text{top}} \cdot \text{ItemPop}^k} \) is the sample covariance between \( \text{ItemFreq}^k_{\text{top}} \) and \( \text{ItemPop}^k \). Furthermore, the sample standard deviations are given by \( \sigma_{\text{ItemFreq}^k_{\text{top}}} \) and \( \sigma_{\text{ItemPop}^k} \).

**Item Coverage.** In addition to evaluating the correlation between an item’s estimated rating score and its popularity, we also measure for what fraction of the item catalog the recommender system is able to estimate high rating scores. For this, we use the Item Space Coverage metric [42, 48, 75] given by:

\[
\text{Coverage}^k = \frac{1}{|I|} \left| \bigcup_{u \in U} R^k_{\text{top}}(u) \right|
\]

(17)

where \( k \) is the number of neighbors, \( I \) is the set of items, \( U \) is the set of users, and \( R^k_{\text{top}}(u) \) is the set of top items for user \( u \). This way, we can test whether parts of the item catalog always receive low estimated rating scores. We use
Table 3. Descriptive statistics of our five datasets. $|U|$ is the number of users, $|I|$ is the number of items, $|R|$ is the number of ratings, $|R|/|U|$ is the rating-to-user ratio, $|U|/|I|$ is the user-to-item ratio, and Density is given by $|R|/(|U||I|)$.

| Dataset   | $|U|$  | $|I|$  | $|R|$  | $|R|/|U|$ | $|U|/|I|$ | Density   |
|-----------|-------|-------|-------|----------|--------|-----------|
| ML 1M     | 6,040 | 3,706 | 1,000,209 | 165.60  | 1.6298 | 4.47%     |
| Douban    | 2,509 | 39,576| 893,575 | 356.15  | 0.0634 | 0.90%     |
| LastFM    | 3,000 | 352,805| 1,755,361 | 585.12  | 0.0085 | 0.17%     |
| Ciao      | 7,375 | 105,096| 282,619 | 38.32   | 0.0702 | 0.04%     |
| Goodreads | 20,000| 508,696| 2,569,177| 128.46  | 0.0394 | 0.03%     |

PP-Corr@k and Coverage@k to evaluate $\text{ReuseKNN}_{DP}$. Additionally, we also use these metrics to evaluate UserKNN to explore the impact of DP [27].

### 4.3 Datasets

In this work, we conduct experiments on five different datasets: MovieLens 1M (ML 1M) [39], Douban [44], LastFM User Groups (LastFM) [52], Ciao [37], and Goodreads [81, 82].

All five datasets exhibit different properties, as illustrated in Table 3. For example, the movie rating dataset ML 1M (integer ratings in $\{1 \ldots 5\}$) is the densest dataset. Similarly, also Douban (integer ratings in $\{1 \ldots 5\}$) and Ciao (integer ratings in $\{1 \ldots 5\}$) are movie rating datasets. Moreover, in Ciao, users have the smallest number of ratings per user (i.e., $|R|/|U|$) on average. LastFM includes implicit feedback data (i.e., listening counts) from the online music streaming service Last.fm. However, in this dataset, the authors of [52] transfer the implicit feedback to decimal ratings in $\{1 \ldots 1000\}$. Plus, users have the largest number of ratings per users. The book rating dataset Goodreads (integer ratings $\in \{1 \ldots 5\}$) is our largest and least dense dataset.

Overall, our datasets cover (i) the movie, music, and book domain, (ii) implicit and explicit feedback, and (iii) different descriptive statistics. Moreover, we utilize datasets used in both, recommender systems and information retrieval research [53].

### 4.4 Evaluation Protocol & Statistical Tests

We perform all experiments using 5-fold cross-validation, and randomly split all folds into 80% training sets $R^{train}$ and 20% test sets $R^{test}$. The ratings in $R^{train}$ are used to train the recommendation algorithms, and the ratings in $R^{test}$ represent the rating queries used for evaluation. Also, we perform several statistical tests to measure the significance of our results. Specifically, after close inspection of our results, we resort to the Mann-Whitney-U-Test. For our query-based metrics Neighbors@$q$ and CoRatings@$q$, we evaluate significance for all rating queries $q \in [1; 100]$ when utilizing $k = 10$ neighbors. For other metrics, i.e., MAE@$k$, PrivacyRisk$_{DP}@k$, PP-Corr@$k$, and Coverage@$k$, we evaluate significance after all queries have been processed by the recommender system. Again, here, we utilize $k = 10$ neighbors to generate recommendations. Importantly, we only report statistical significance if we observe significance for each of our five folds.

### 4.5 Parameter Settings

Our approach relies on two adjustable hyperparameters, i.e., (i) the number of neighbors $k$ used for generating recommendations and (ii) the privacy risk threshold $r$. To test the performance of $\text{ReuseKNN}$ and $\text{ReuseKNN}_{DP}$ for
different values of \( k \), we vary \( k \in \{5, 10, 15, 20, 25, 30\} \). Plus, we set \( r \) to the approximate starting value of the tail of UserKNN’s privacy risk distribution, which is given by its maximal second derivative (see Figure 1).

This way, we assume that only the tail’s small privacy risk is tolerable and give an example of how \( r \) can be defined by the recommender system provider. Also, \( r \) is the same for all users. This leads to the following \( r \) values for \( k = 10 \): 92.89 (ML 1M), 91.54 (Douban), 104.32 (LastFM), 95.79 (Ciao), and 94.90 (Goodreads). For the similarity function \( sim \), we use cosine similarity.

5 RESULTS & DISCUSSION

We structure our results into two parts: (i) neighborhood reuse only (ReuseKNN), and (ii) neighborhood reuse with DP (ReuseKNN\_DP).

5.1 ReuseKNN

In this section, we present our evaluation results for ReuseKNN, i.e., neighborhood reuse only.

5.1.1 Neighborhood Growth. Figure 4 illustrates the average size of target users’ neighborhood after querying \( q \) recommendations for a model with \( k = 10 \) neighbors. For all of our five datasets, the size of a user’s neighborhood increases more strongly for traditional UserKNN than for all of our neighborhood reuse strategies. For MovieLens 1M, Douban, LastFM, and Goodreads, a one-tailed Mann-Whitney-U-Test (\( \alpha = 0.01 \)) shows that all our neighborhood reuse strategies yield significantly smaller neighborhoods than traditional UserKNN for \( q \in [2; 100] \) rating queries. This means that ReuseKNN can already reuse neighbors after an initial neighborhood is generated for the very first rating query.
ReuseKNN: Neighborhood Reuse for Privacy-Aware Recommendations

Fig. 5. Avg. number of co-rated items between the target user and its neighbors. All our strategies generate neighborhoods, in which the neighbors’ rated items overlap more with the target users’ than in the case of UserKNN. With this, neighbors are more reliable.

However, for Ciao, we find that multiple rating queries are needed to generate a reusable neighborhood. For Gain applied to the Ciao dataset, we do not observe significant differences. In contrast, our remaining neighborhood strategies tend to yield significantly smaller neighborhoods for a few rating queries. We attribute this to the on average small user profiles in Ciao (cf. Table 3). Reusable neighbors are scarce and thus, ReuseKNN cannot reduce the neighborhood size as effectively as in the case of the other datasets.

Interestingly, for all datasets, our explicit neighborhood reuse strategies generate smaller neighborhoods than their implicit counterparts. We underline that explicit neighborhood reuse prioritizes neighbors from previous rating queries over new neighbors and, thus, limits the neighborhood size more strictly than implicit neighborhood reuse.

5.1.2 Neighborhood Reliability. Similar to the neighborhood growth in Figure 4, Figure 5 illustrates the average number of co-rated items between the target user and its neighbors after querying $q$ rating queries. For all our five datasets, all our neighborhood reuse strategies can substantially increase the co-ratings over traditional UserKNN. A one-tailed Mann-Whitney-U-Test ($\alpha = 0.01$) shows that all our neighborhood reuse strategies generate neighborhoods with significantly more co-ratings with the target user than UserKNN for $q \in [2; 100]$ rating queries. This indicates that ReuseKNN generates neighborhoods that comprise less, but more reliable neighbors than in case of traditional UserKNN, which can foster recommendation accuracy [2, 40, 67].

However, for Ciao, our neighborhood reuse strategies tend to generate significantly more reliable neighborhoods for only a few rating queries. As in our neighborhood growth experiment, we attribute this to the small user profiles in Ciao, which makes neighborhood reuse less effective due to the scarcity of reusable neighbors.

Moreover, for all datasets, explicit neighborhood reuse strategies generate more reliable neighborhoods than implicit neighborhood reuse strategies. We suppose that this is caused by the small neighborhoods that are generated by explicit
neighborhood reuse strategies (see Section 5.1.1). Specifically, the smaller the neighborhood, the easier it is to identify reliable neighbors with a large rating overlap with the target user.

5.1.3 Accuracy. In Figure 6, we compare ReuseKNN to traditional UserKNN in terms of recommendation accuracy. Here, we find that our neighborhood reuse strategies can generate more accurate recommendations than UserKNN. This shows that reusing neighbors that have already been used in the past for calculating meaningful recommendations, also leads to meaningful (accurate) recommendations in the future.

Specifically, for ML 1M, Douban, and LastFM, a one-tailed Mann-Whitney-U-Test ($\alpha = 0.01$) indicates that all our neighborhood reuse strategies significantly increase recommendation accuracy for a model with $k = 10$ neighbors. Due to personalization, Gain and Gain+Reuse perform best across most datasets.

For LastFM, the unpersonalized neighborhood reuse strategies Expect and Expect+Reuse outperform our personalized neighborhood reuse strategies Gain and Gain+Reuse. We attribute this to LastFM’s small user-to-item ratio as compared to the other datasets (cf. Table 3). This makes it hard to identify neighbors, similar to an item-cold start scenario [72]. Concretely, in the case of personalized neighborhood reuse, selecting reusable neighbors for a specific target user reduces the pool of potential neighbors per item to a personalized subset and leads to a worse performance compared to unpersonalized neighborhood reuse. In contrast, our unpersonalized neighborhood reuse strategies allow using the entire pool of potential neighbors and thus achieve a higher recommendation accuracy for LastFM.

In the case of our least dense datasets Ciao and Goodreads, we observe that our personalized neighborhood reuse strategies Gain and Gain+Reuse can handle these datasets better than our unpersonalized neighborhood reuse strategies Expect and Expect+Reuse. Our personalized neighborhood reuse strategies select neighbors, whose rating data could
Table 4. Percentage of vulnerable users for \( k = 10 \). For most datasets, ReuseKNN’s \textit{Expect} neighborhood reuse strategy leads to fewer vulnerable users than UserKNN (best results, i.e., lowest values, are in \textbf{bold}). For Ciao, neighborhood reuse achieves only minor improvements, as already UserKNN yields a small percentage of vulnerable users.

| Method     | ML 1M | Douban | LastFM | Ciao | Goodreads |
|------------|-------|--------|--------|------|-----------|
| UserKNN    | 80.39%| 96.68% | 99.89% | 8.02%| 65.00%    |
| UserKNN+Reuse | 84.64%| 87.37% | 98.90% | 7.91%| 52.29%    |
| \textit{Expect} | \textbf{24.13%} | \textbf{34.40%} | \textbf{68.20%} | 7.88%| 29.12%    |
| \textit{Expect}+Reuse | 25.11%| 41.59% | 78.85% | 7.77%| \textbf{28.22%} |
| Gain       | 25.09%| 37.43% | 80.28% | 8.19%| 40.51%    |
| Gain+Reuse | 27.61%| 43.92% | 84.27% | 7.98%| 37.22%    |

have been used by the target user in the past (cf. Equation 5). This way, \textit{Gain} and \textit{Gain+Reuse} create a neighborhood for a given target user with sufficient rating data even in sparse datasets.

Plus, we highlight that \textit{Gain} and \textit{Gain+Reuse} significantly increase recommendation accuracy for Goodreads, despite the dataset’s low density. In the case of Ciao, a two-tailed Mann-Whitney-U-Test (\( \alpha = 0.01 \)) reveals no significant differences between our neighborhood reuse strategies and UserKNN for \( k = 10 \), which suggests that all our neighborhood reuse strategies can preserve recommendation accuracy. As shown in Figure 4 before, neighborhood reuse is less effective for Ciao due to the small user profiles. Thus, it makes sense that the recommendation accuracy cannot be improved as effectively as for the remaining datasets.

5.1.4 Percentage of Vulnerable Users. In Table 4, we measure how well our neighborhood reuse strategies can reduce the percentage of users that are vulnerable. For all of our five datasets, all our strategies lead to less vulnerable users than traditional UserKNN. Especially ReuseKNN’s \textit{Except} neighborhood reuse strategy shows the best (i.e., lowest) percentage of vulnerable users. For example, for the ML 1M dataset, UserKNN leads to 80.39% of users that are vulnerable, since their privacy risk exceeds threshold \( \tau = 92.89 \) (cf. Section 4.5), whereas \textit{Expect} leads to only 24.13% vulnerable users and thus, fewer users need to be protected with DP.

For Ciao, our neighborhood reuse strategies achieve only minor improvements over UserKNN. The reason is that UserKNN already yields a small percentage of vulnerable users and as such, ReuseKNN leads to only small improvements. Additionally, our previous findings show that the effect of neighborhood reuse on Ciao is smaller than on the remaining datasets due to the small average user profile size (cf. Table 3). This leads to a lack of reusable neighbors and, thus, also limits the effect neighborhood reuse has on the percentage of vulnerable users.

5.1.5 Summary. Overall, we find that ReuseKNN can significantly reduce the size of target users’ neighborhoods by neighborhood reuse. In particular, explicit neighborhood reuse yields smaller neighborhoods than implicit neighborhood reuse, since it prioritizes neighbors from previous rating queries over new neighbors and as such, limits the size of a target user’s neighborhood more strictly than implicit neighborhood reuse.

Despite the much smaller neighborhoods, ReuseKNN identifies neighbors that have many more co-rated items with the target user than in the case of UserKNN. As related work shows, these reliable neighbors can be crucial for recommendation accuracy [2, 40, 67].

Based on the much smaller but more reliable neighborhoods, ReuseKNN can generate recommendations with significantly higher accuracy than traditional UserKNN. However, for sparse datasets, personalized neighborhood reuse seems to be a better solution than unpersonalized neighborhood reuse.

Manuscript submitted to ACM
Plus, ReuseKNN can substantially reduce the percentage of vulnerable users with high privacy risk. In general, our Except neighborhood reuse method yields the least vulnerable users.

5.2 ReuseKNN$_{DP}$

Next, we present our results on ReuseKNN$_{DP}$, i.e., neighborhood reuse with DP.

5.2.1 Accuracy. For ReuseKNN, we showed that our neighborhood reuse strategies can preserve, and in many cases significantly improve, the recommendation accuracy over traditional UserKNN. However, in the following, we discuss whether these findings also hold for a combination of ReuseKNN and DP, i.e., ReuseKNN$_{DP}$.

In Figure 7, we compare ReuseKNN$_{DP}$’s recommendation accuracy to UserKNN$_{DP}$’s. Similar to the non-DP experiment shown in Figure 6, our neighborhood reuse strategies can generate more accurate recommendations than UserKNN$_{DP}$. For ML 1M and LastFM, a one-tailed Mann-Whitney-U-Test ($\alpha = 0.01$) indicates that all our neighborhood reuse strategies significantly increase recommendation accuracy for a model with $k = 10$ neighbors. Additionally, for ML 1M, Gain$_{DP}$ and Gain+Reuse$_{DP}$ perform better than our non-DP baseline UserKNN.

Moreover, we observe that LastFM is highly sensitive to the incorporation of DP, since the mean absolute error magnitudes differ substantially between our non-DP experiment in Figure 6 and our DP experiment in Figure 7. In line with our previous results on non-DP ReuseKNN, also ReuseKNN$_{DP}$’s unpersonalized neighborhood reuse strategies Except$_{DP}$ and Except+Reuse$_{DP}$ cannot increase recommendation accuracy for Ciao and Goodreads, which are our two sparsest datasets. However, our personalized neighborhood reuse strategies Gain$_{DP}$ and Gain+Reuse$_{DP}$ generate recommendations with significantly higher accuracy for Goodreads. For Ciao, no significant differences are
Fig. 8. Logarithm (base 10) of the privacy risk averaged over all users. All our neighborhood reuse strategies yield lower privacy risk than UserKNN\textsubscript{DP}. This is due to the fact that ReuseKNN\textsubscript{DP} reduces the percentage of users with a privacy risk of \( \tau \) (i.e., vulnerables) and simultaneously, decreases the privacy risk of the remaining users (i.e., secures). Overall, we find that our unpersonalized neighborhood reuse strategies Expect\textsubscript{DP} and Expect+Reuse\textsubscript{DP} achieve the best user privacy.

found according to a two-tailed Mann-Whitney-U-Test (\( \alpha = 0.01 \)). Thus, Gain\textsubscript{DP} and Gain+Reuse\textsubscript{DP} can preserve recommendation accuracy.

For Douban, we observe no significant differences between our neighborhood reuse strategies and UserKNN\textsubscript{DP}. We found empirically that for Douban, UserKNN\textsubscript{DP} and ReuseKNN\textsubscript{DP} utilize more rating data from vulnerable users, than in case of our remaining datasets. Thus, we measure the fraction of rating data, each user contributes to the dataset, i.e., \(|R_u|/|R|\), where \( R \) are all users’ ratings and \( R_u \) are user \( u \)’s ratings. We find that for Douban, the 5% of users with the largest user profiles contribute substantially more ratings to the dataset than for our other datasets, i.e., 0.0008 (ML 1M), 0.0022 (Douban), 0.0012 (LastFM), 0.0009 (Ciao), and 0.0003 (Goodreads). This suggests that in the case of Douban, the recommendation process more often utilizes these users due to their abundance of rating data. This, however, makes these users more vulnerable. Therefore, we suppose that this strong utilization of DP-protected rating data from vulnerable users leads to no significant differences in accuracy between UserKNN\textsubscript{DP} and ReuseKNN\textsubscript{DP}.

We additionally compare ReuseKNN\textsubscript{DP} to Full\textsubscript{DP}. Our results suggest that our personalized reuse strategies Gain\textsubscript{DP} and Gain+Reuse\textsubscript{DP} generate recommendations with significantly higher accuracy, while Except\textsubscript{DP} and Except+Reuse\textsubscript{DP} show no significant differences. Thus, all our neighborhood reuse strategies can preserve recommendation accuracy for this dataset.

5.2.2 Privacy Risk When DP is Applied. In ReuseKNN\textsubscript{DP}, vulnerable users with high privacy risk are protected with DP and as such, their privacy risk is set to the privacy risk threshold \( \tau \) as realized with our PrivacyRisk\textsubscript{DP} metric. Moreover, secure users’ privacy risk is also reduced since they are rarely exploited as neighbors in the recommendation process (cf. Figure 1).
Table 5. Pearson correlation between an item’s popularity and its occurrences in users’ top items for \( k = 10 \). Weakest correlations (mean best) are in **bold**. A z-Test shows, with **(\( \alpha = 0.01 \))** that ReuseKNN\(_{DP} \) can significantly reduce item popularity bias over UserKNN\(_{DP} \). For Ciao, no differences can be found.

| Method               | ML 1M       | Douban     | LastFM     | Ciao        | Goodreads   |
|----------------------|-------------|------------|------------|-------------|-------------|
| UserKNN              | 0.8405      | 0.6780     | 0.7339     | 0.9755      | 0.9318      |
| UserKNN\(_{DP} \)    | 0.8742      | 0.7589     | 0.8625     | 0.9758      | 0.9409      |
| UserKNN+Reuse\(_{DP} \) | 0.8750      | 0.7523     | 0.8779     | 0.9759      | 0.9407      |
| Full\(_{DP} \)       | 0.8800      | 0.7675     | 0.8597     | 0.9778      | 0.9523      |
| Expect\(_{DP} \)     | 0.8686      | **0.7406** | 0.8773     | 0.9767      | **0.9317**  |
| Expect+Reuse\(_{DP} \) | 0.8711      | **0.7429** | 0.8831     | 0.9763      | **0.9313**  |
| Gain\(_{DP} \)       | 0.8725      | **0.7428** | 0.8621     | 0.9769      | 0.9454      |
| Gain+Reuse\(_{DP} \) | 0.8734      | **0.7453** | 0.8741     | 0.9765      | 0.9413      |

In Figure 8, we visualize the privacy risk of our four ReuseKNN\(_{DP} \) variants and our three baselines UserKNN, UserKNN\(_{DP} \), and Full\(_{DP} \). We find that all our neighborhood reuse strategies combined with DP can improve user privacy over UserKNN\(_{DP} \). Specifically, a one-tailed Mann-Whitney-U-Test (\( \alpha = 0.01 \)) reveals that for all our neighborhood reuse strategies on all datasets and for \( k = 10 \), users have significantly less privacy risk than in UserKNN\(_{DP} \).

However, for LastFM, this privacy improvement is smaller than for the other datasets. Due to the large percentage of vulnerable users in LastFM for all approaches (cf. Table 4), most users are set to the maximum privacy risk (\( \tau \)) independent of the approach. Thus, the small percentage of secure users is insufficient to reduce the average privacy risk via neighborhood reuse in the case of LastFM.

Across all datasets, we observe that our unpersonalized neighborhood reuse strategies Expect\(_{DP} \) and Expect+Reuse\(_{DP} \) yield the best (lowest) privacy risk. This finding is in line with our previous results in Table 4, which show that Expect\(_{DP} \) performs best with respect to minimizing the percentage of vulnerable users. Thus, only a few users have a maximum privacy risk of \( \tau \), and the high number of secure users enables to drastically reduce the average privacy risk. For example, the average privacy risk of secure users for a model with \( k = 10 \) neighbors for Expect\(_{DP} \) is 11.45 for ML 1M, 18.34 for Douban, 49.92 for LastFM, 15.29 for Ciao, and 18.99 for Goodreads compared to the privacy risk of secure users for UserKNN\(_{DP} \) that is 50.83 for ML 1M, 62.13 for Douban, 73.42 for LastFM, 21.76 for Ciao, and 41.13 for Goodreads. Additionally, a one-tailed Mann-Whitney-U-Test (\( \alpha = 0.01 \)) reveals that for ML 1M, Douban, Ciao, and Goodreads, these differences are significant. Thus, for secure users, Expect\(_{DP} \) yields a substantially smaller privacy risk than UserKNN\(_{DP} \).

5.2.3 Positivity-Popularity Correlation. We test for item popularity bias in ReuseKNN\(_{DP} \)’s recommendations by measuring the Pearson correlation between an item’s popularity and its occurrences in users’ sets of top items, i.e., the items with the highest estimated rating score (see Table 5).

For ML 1M, Douban, LastFM, and Ciao, the non-DP baseline UserKNN yields weaker popularity bias than all baselines and neighborhood reuse strategies that use DP. This fits well to related research [27] that argues that popularity bias can arise due to the recommender system’s inability to personalize recommendations when DP is applied to user data. However, for ML 1M, Douban, LastFM, and Goodreads, our neighborhood reuse strategies can generate less popularity-biased recommendations than our DP-baseline UserKNN\(_{DP} \). For ML 1M, Douban, and Goodreads, this difference is also statistically significant according to a one-tailed z-test on the z-transformed correlation coefficient [28, 43, 93] as indicated by **(\( \alpha = 0.01 \))**. We investigate this in more detail and find that the neighbors identified by ReuseKNN\(_{DP} \) have on average a larger user profile size (i.e., more distinct items) than the neighbors identified by...
Table 6. Item coverage of ReuseKNNDP, UserKNNDP, and our baselines (best results are printed in **bold**). For ML 1M, Douban, Ciao, and Goodreads, ExpectDP can estimate positive rating scores for a larger percentage of the item catalog than UserKNNDP. For LastFM, only our FullDP baseline can cover more items than UserKNNDP.

| Method               | ML 1M | Douban | LastFM | Ciao | Goodreads |
|----------------------|-------|--------|--------|------|-----------|
| UserKNN              | 87.94%| 23.50% | 6.11%  | 63.19%| 29.56%    |
| UserKNNDP            | 88.77%| 26.33% | 15.54% | 64.03%| 31.59%    |
| UserKNN+ReuseDP      | 88.37%| 27.67% | 15.46% | 64.26%| 31.74%    |
| FullDP               | **89.53%** | **27.65%** | **15.86%** | **66.72%** | **34.13%** |
| ExpectDP             | 88.83%| 28.75% | 14.32% | 64.38%| 34.89%    |
| Expect+ReuseDP       | 88.24%| 28.49% | 14.67% | 64.62%| 34.22%    |
| GainDP               | 88.07%| 28.61% | 14.77% | 64.01%| 31.46%    |
| Gain+ReuseDP         | 88.05%| 28.45% | 14.84% | 64.18%| 31.83%    |

UserKNNDP. As shown by related work on item popularity bias in recommender systems (e.g., [1, 52]), users with a larger user profile size tend to consume less popular items, which leads to less popularity bias.

However, for Ciao, no notable differences can be observed. We again attribute this to the small number of ratings per user in this dataset (cf. Table 3). With this, neighborhood reuse is less effective and consequently, has only a small impact on the generation of recommendations. Thus, for our Ciao dataset, neighborhood reuse has no effect on item popularity bias.

5.2.4 Item Coverage. In addition, we also evaluate if ReuseKNNDP can estimate high rating scores only for a limited set of items. In Table 6, we present the percentage of items from the entire item catalog that occur within users’ sets of top items.

For ML 1M, Douban, Ciao, and Goodreads, our neighborhood reuse strategies cover larger parts of the item catalog than traditional UserKNNDP. This means that many items can receive high estimated rating scores. Specifically, ExpectDP yields the highest item coverage values across our neighborhood reuse strategies.

In case of LastFM, we observe that all our neighborhood reuse strategies estimate high estimated rating scores for fewer items than UserKNNDP. However, we underline that the item coverage values for LastFM are negatively correlated with our accuracy results in Figure 7. This indicates that for LastFM, there is a trade-off between recommendation accuracy and item coverage, similar to the well-known trade-off between precision and recall [14, 73].

However, it is noteworthy that for most datasets, FullDP yields the highest item coverage and our non-DP baseline UserKNN yields the lowest item coverage. This makes sense, since FullDP protects all rating data with DP and thus, the estimated rating scores are more random than for the remaining approaches. Consequently, this leads to more randomized recommendations, and thus, to high item coverage [31].

5.2.5 Summary. Overall, our results are in line with the previously presented results for our non-DP ReuseKNN. Through neighborhood reuse, and thus, reducing the number of users that need to be protected with DP, recommendation accuracy can be preserved, and in many cases even significantly improved over UserKNNDP.

Also, we underline that all our neighborhood reuse strategies used in ReuseKNNDP lead to significantly smaller privacy risk than UserKNNDP. In particular, our unpersonalized neighborhood reuse method ExceptDP performs best in increasing user privacy. This shows that the combination of neighborhood reuse and DP provides higher privacy than UserKNNDP.
Table 7. We compare all evaluated strategies to our UserKNN\(\text{DP}\) baseline (\(k = 10\)). Also, we perform a one-tailed Mann-Whitney-U-Test (\(\alpha = 0.01\)) and mark significantly lower values than in case of UserKNN\(\text{DP}\) with **. Overall, personalized reuse (Gain\(\text{DP}\) and Gain+Reuse\(\text{DP}\)) yields the best accuracy and unpersonalized reuse (Expect\(\text{DP}\) and Expect+Reuse\(\text{DP}\)) gives the lowest privacy risk. For Douban and LastFM, Expect\(\text{DP}\) is well-suited as it yields the most accurate recommendations and lowest privacy risk. For the remaining datasets, all our neighborhood reuse strategies utilized in ReuseKNN\(\text{DP}\) provide a less serious accuracy-privacy trade-off than UserKNN\(\text{DP}\).

|                | ML 1M          | Douban        | LastFM        | Ciao          | Goodreads     |
|----------------|----------------|---------------|---------------|---------------|---------------|
|                | MAE  | Privacy R.  | MAE  | Privacy R.  | MAE  | Privacy R.  | MAE  | Privacy R.  | MAE  | Privacy R.  |
| UserKNN        | 0.80 | 330.77      | 0.66 | 665.17       | 0.78 | 35.21        | 0.80 | 182.26       |
| UserKNN\(\text{DP}\) | 0.82 | **31.90**   | 0.68 | **69.62**    | **103.80** | **103.77**   | **0.81** | **27.61**   | **0.83** | **75.71**   |
| UserKNN+Reuse\(\text{DP}\) | 0.81 | 87.16       | 0.68 | 87.16        | 118.13 | 103.56       | 0.81 | 26.54        | 0.83 | 68.35        |
| Full\(\text{DP}\)    | 0.83 | 0.00        | 0.69 | 0.00         | 128.41 | 0.00         | 0.87 | 0.00         | 0.85 | 0.00         |
| Expect\(\text{DP}\)    | **0.80** | 31.03        | 0.68 | **47.25**    | **114.78** | **86.81**    | **0.82** | **21.53**   | **0.83** | **40.95**   |
| Expect+Reuse\(\text{DP}\) | **0.79** | **33.50** | 0.68 | **46.57** | **115.31** | **93.95**   | **0.81** | **26.74** | **0.81** | **55.90** |
| Gain\(\text{DP}\)    | **0.79** | **36.60** | 0.68 | **31.55** | **116.04** | **96.07**   | **0.81** | **25.87** | **0.82** | **52.48** |

Besides, we find that for ReuseKNN\(\text{DP}\), high estimated rating scores are weaker correlated to item popularity than in case of UserKNN\(\text{DP}\). Moreover, ReuseKNN\(\text{DP}\) can estimate high rating scores for a larger set of items than UserKNN\(\text{DP}\). This provides evidence that ReuseKNN\(\text{DP}\) does not increase item popularity bias.

5.3 Discussion

We find that ReuseKNN\(\text{DP}\) can preserve recommendation accuracy and, in many cases, even improve recommendation accuracy over UserKNN\(\text{DP}\). The same time, ReuseKNN\(\text{DP}\) leads to significantly smaller privacy risk than UserKNN\(\text{DP}\), due to the small number of neighbors to which DP has to be applied. Besides, ReuseKNN\(\text{DP}\) does not increase item popularity bias over traditional UserKNN\(\text{DP}\).

In Table 7, we illustrate a condensed summary of experimental results for all our evaluated approaches, i.e., UserKNN and ReuseKNN, UserKNN\(\text{DP}\), ReuseKNN\(\text{DP}\), and Full\(\text{DP}\). Specifically, we present the accuracy (i.e., MAE@\(k\)) and average privacy risk (i.e., PrivacyRisk\(\text{DP}@k\)) values for a model with \(k = 10\) neighbors.

Overall, we can observe that non-DP UserKNN results in low MAE, but high privacy risk values. This shows that approaches without DP sacrifice a user’s privacy for recommendation accuracy. However, our neighborhood reuse strategies with DP provide a less serious trade-off between recommendation accuracy and privacy.

Thus, in the following, we discuss advantages and disadvantages of our neighborhood reuse strategies for our five datasets. Across all our neighborhood reuse strategies, in general, personalized neighborhood reuse (i.e., Gain\(\text{DP}\) and Gain+Reuse\(\text{DP}\)) provides the best recommendation accuracy. Plus, unpersonalized neighborhood reuse (i.e., Expect\(\text{DP}\) and Expect+Reuse\(\text{DP}\)) yields the lowest privacy risk. For Douban and LastFM, Expect\(\text{DP}\) performs best in both, accuracy and privacy risk. Thus, in this case, Expect\(\text{DP}\) is well suited to provide accurate and private recommendations. For ML 1M, Ciao, and Goodreads, no neighborhood reuse strategy provides the best result in both categories. Thus, it depends on the recommender system service provider to decide what strategy could be utilized.

6 CONCLUSION

In this work, we investigate the efficacy of neighborhood reuse for privacy-aware recommendations in user-based KNN recommender systems. We discuss our approach in a two-stage evaluation procedure: (i) neighborhood reuse only, i.e.,
ReuseKNN, to distill the impact of neighborhood reuse on recommendation accuracy and on the percentage of users that need to be protected with differential privacy, and (ii) neighborhood reuse with differential privacy, i.e., ReuseKNN_{DP}, to investigate the practical benefit of neighborhood reuse for privacy-aware recommendations. We find that ReuseKNN and ReuseKNN_{DP} can substantially reduce the number of users that need to be protected with DP, while outperforming related approaches in terms of accuracy. Also, we highlight that ReuseKNN_{DP} effectively mitigates users’ privacy risk, as most users are rarely exploited in the recommendation process. Our work illustrates how to address privacy risks in recommender systems through neighborhood reuse combined with DP.

Limitations. We recognize the following limitations of our approach: To quantify the privacy risk, we assume that all pieces of data are equally sensitive. In reality, disclosing a particular piece of information could pose a different level of privacy risk than disclosing another piece of information [47, 51, 61]. Also, our experiments quantify how well ReuseKNN_{DP} can predict ratings for target user-item pairs, i.e., rating prediction. A logical next step would be to move from a rating prediction task to a ranking-based recommendation task, i.e., top-n recommendation, i.e., top-n recommendations [77]. Furthermore, we focused on a single family of recommendation algorithms, i.e., user-based KNN, because our main goal was not to beat state-of-the-art recommendation algorithms but to illustrate the benefit of a neighbor reuse concept for differentially private recommendations. Moreover, Dacrema et al. [21] showed that many modern recommendation algorithms, e.g., deep learning, can be outperformed by traditional algorithms such as user-based KNN. This means that this type of algorithm is still very relevant for the recommender systems community.

Future Work. In this work, we have focused on rating prediction and recommendation accuracy. For our future work, we plan to evaluate our approach in a top-n recommendation task, and in terms of beyond-accuracy performance metrics (e.g., diversity). Also, while we evaluated our approach using five datasets of three different recommendation domains (movies, books, and music), future work will consider additional, more sensitive domains, such as, medicine, finance, insurance, or recruiting. Finally, we plan to evaluate our ReuseKNN_{DP} approach for different user groups that differ in their inclination to popular content, e.g., using the dataset provided in our previous research on fairness in music recommender systems [52]. With this, we can study the impact of neighbor reuse and differential privacy on an individual user’s preference towards long-tail items. Hence, our long-term plan is to investigate the interaction between privacy and fairness, two important aspects of trustworthy recommender systems.

ACKNOWLEDGEMENTS & MATERIAL

This research is funded by the H2020 project TRUSTS (GA: 871481) and the “DDAI” COMET Module within the COMET — Competence Centers for Excellent Technologies Programme, funded by the Austrian Federal Ministry for Transport, Innovation and Technology (bmvit), the Austrian Federal Ministry for Digital and Economic Affairs (bmdw), the Austrian Research Promotion Agency (FFG), the province of Styria (SFG) and partners from industry and academia. The COMET Programme is managed by FFG. In addition, this research received support by the Austrian Science Fund (FWF): P33526 and DFH-23; and by the State of Upper Austria and the Federal Ministry of Education, Science, and Research, through grant LIT-2020-9-SEE-113.

The Python-based implementation of our work is publicly available at https://github.com/pmuellner/ReuseKNN. Also, we provide the source code for generating our sample of the Goodreads dataset. All remaining datasets are publicly available as well (see Section 4.3).
References

[29] Dan Frankowski, Dan Cosley, Shilad Sen, Loren Terveen, and John Riedl. 2006. You are what you say: privacy risks of public mentions. In Proceedings of the 29th annual international ACM SIGIR Conference on Research and Development in Information Retrieval. 565–572.

[30] Davide Frey, Rachid Guerraoui, Anne-Marie Kermarrec, and Antoine Rault. 2015. Collaborative filtering under a sybil attack: analysis of a privacy threat. In Proceedings of the Eighth European Workshop on System Security. 1–6.

[31] Jill Freyne and Shlomo Berkovsky. 2013. Evaluating recommender systems for supportive technologies. In User Modeling and Adaptation for Daily Routines. Springer, 195–217.

[32] Arik Friedman, Shlomo Berkovsky, and Mohamed Ali Kaafar. 2016. A differential privacy framework for matrix factorization recommender systems. User Modeling and User-Adapted Interaction 26, 5 (2016), 425–458.

[33] Arik Friedman, Bart P Knuienenburg, Kris Vanhecke, Luc Martens, and Shlomo Berkovsky. 2015. Privacy aspects of recommender systems. In Recommender systems handbook. Springer, 649–688.

[34] Chen Gao, Chao Huang, Dongsheng Lin, Depeng Jin, and Yong Li. 2020. DPLCF: Differentially Private Local Collaborative Filtering. In Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval. 961–970.

[35] Craig Gentry et al. 2009. A fully homomorphic encryption scheme. Stanford university Stanford.

[36] Bernard G Greenberg, Abdel-Latif A Abul-Ela, Walt R Simmons, and Daniel G Horvitz. 1969. The unrelated question randomized response model: Theoretical framework. J. Amer. Statist. Assoc. 64, 326 (1969), 520–539.

[37] Guibing Guo, Jie Zhang, Daniel Thalmann, and Neil Yorke-Smith. 2014. Etaf: An extended trust antecedents framework for trust prediction. In International Conference on Advances in Social Network Analysis and Mining.

[38] Jialiang Han, Yun Ma, Qiaozhu Mei, and Xuanzhi Liu. 2021. DeepRec: On-device Deep Learning for Privacy-Preserving Sequential Recommendation in Mobile Commerce. In Proceedings of the Web Conference 2021. 900–911.

[39] F Maxwell Harper and Joseph A Konstan. 2015. The movielens datasets: History and context. ACM Transactions on Interactive Intelligent Systems (TiiS) 5, 4 (2015), 1–19.

[40] Jon Herlocker, Joseph A Konstan, and John Riedl. 2002. An empirical analysis of design choices in neighborhood-based collaborative filtering algorithms. Information Retrieval (2002), 287–310.

[41] Jonathan L Herlocker, Joseph A Konstan, Al Borchers, and John Riedl. 1999. An algorithmic framework for performing collaborative filtering. In Proceedings of the 22nd annual international ACM SIGIR Conference on Research and Development in Information Retrieval. 230–237.

[42] Jonathan L. Herlocker, Joseph A. Konstan, Loren G. Terveen, and John T. Riedl. 2004. Evaluating Collaborative Filtering Recommender Systems. ACM Trans. Inf. Syst. 22, 1 (jan 2004), 5–53. https://doi.org/10.1145/963770.963772

[43] Dennis E Hinkle, William Wiersma, and Stephen G Jurs. 2003. Applied statistics for the behavioral sciences. Vol. 663. Houghton Mifflin College Division.

[44] Longke Hu, Aixin Sun, and Yong Liu. 2014. Your neighbors affect your ratings: on geographical neighborhood influence to rating prediction. In Proceedings of the 37nd annual international ACM SIGIR Conference on Research and Development in Information Retrieval.

[45] Jingyu Hua, Chang Xia, and Sheng Zhong. 2015. Differentially Private Matrix Factorization. In International Joint Conference on Artifical Intelligence.

[46] Arjan JP Jeckmans, Michael Beye, Zekeriya Erkin, Pieter Hartel, Reginald L Lagendijk, and Qiang Tang. 2013. Privacy in recommender systems. In Transactions on Interactive Intelligent Systems (TiiS).

[47] Davide Frey, Rachid Guerraoui, Anne-Marie Kermarrec, and Antoine Rault. 2015. Collaborative filtering under a sybil attack: analysis of a privacy threat. In Proceedings of the Eighth European Workshop on System Security. 1–6.

[48] Jon Herlocker, Joseph A Konstan, Al Borchers, and John Riedl. 1999. An algorithmic framework for performing collaborative filtering. In Proceedings of the 22nd annual international ACM SIGIR Conference on Research and Development in Information Retrieval. 230–237.

[49] Toshihiro Kamishima, Shotaro Akaho, Hideki Asoh, and Jun Sakuma. 2014. Correcting Popularity Bias by Enhancing Recommendation Neutrality. In RecSys Posters.

[50] Rezvan Joshaghani, Michael D Ekstrand, Bart Knijnenburg, and Hoda Mehrpouyan. 2018. Do Different Groups Have Comparable Privacy Tradeoffs?. In Workshop on Moving beyond a ‘One-Size Fits All’ approach: Exploring Individual Differences in Privacy, in conjunction with the ACM CHI Conference on Human Factors in Computing Systems, CHI 2018.

[51] Marius Kaminskas and Derek Bridge. 2016. Diversity, serendipity, novelty, and coverage: a survey and empirical analysis of beyond-accuracy objectives in recommender systems. ACM Transactions on Interactive Intelligent Systems (TiiS) 7, 1 (2016), 1–42.

[52] Toshihito Kamishima, Shota Takami, Ayumu Shirahata, and Hiroshi Kitagawa. 2014. Correcting Popularity Bias by Enhancing Recommendation Neutrality. In RecSys Posters.

[53] Rezvan Joshaghani, Michael D Ekstrand, Bart Knijnenburg, and Hoda Mehrpouyan. 2018. Do Different Groups Have Comparable Privacy Tradeoffs?. In Workshop on Moving beyond a ‘One-Size Fits All’ approach: Exploring Individual Differences in Privacy, in conjunction with the ACM CHI Conference on Human Factors in Computing Systems, CHI 2018.

[54] Marius Kaminskas and Derek Bridge. 2016. Diversity, serendipity, novelty, and coverage: a survey and empirical analysis of beyond-accuracy objectives in recommender systems. ACM Transactions on Interactive Intelligent Systems (TiiS) 7, 1 (2016), 1–42.

[55] Toshihito Kamishima, Shota Takami, Ayumu Shirahata, and Hiroshi Kitagawa. 2014. Correcting Popularity Bias by Enhancing Recommendation Neutrality. In RecSys Posters.

[56] Rezvan Joshaghani, Michael D Ekstrand, Bart Knijnenburg, and Hoda Mehrpouyan. 2018. Do Different Groups Have Comparable Privacy Tradeoffs?. In Workshop on Moving beyond a ‘One-Size Fits All’ approach: Exploring Individual Differences in Privacy, in conjunction with the ACM CHI Conference on Human Factors in Computing Systems, CHI 2018.

[57] Marius Kaminskas and Derek Bridge. 2016. Diversity, serendipity, novelty, and coverage: a survey and empirical analysis of beyond-accuracy objectives in recommender systems. ACM Transactions on Interactive Intelligent Systems (TiiS) 7, 1 (2016), 1–42.

[58] Toshihito Kamishima, Shota Takami, Ayumu Shirahata, and Hiroshi Kitagawa. 2014. Correcting Popularity Bias by Enhancing Recommendation Neutrality. In RecSys Posters.

[59] Rezvan Joshaghani, Michael D Ekstrand, Bart Knijnenburg, and Hoda Mehrpouyan. 2018. Do Different Groups Have Comparable Privacy Tradeoffs?. In Workshop on Moving beyond a ‘One-Size Fits All’ approach: Exploring Individual Differences in Privacy, in conjunction with the ACM CHI Conference on Human Factors in Computing Systems, CHI 2018.
Cort J Willmott and Kenji Matsuura. 2005. Advantages of the mean absolute error (MAE) over the root mean square error (RMSE) in assessing average model performance. Climate Research 30, 1 (2005), 79–82.

Yu Xin and Tommi Jaakkola. 2014. Controlling Privacy in Recommender Systems. In Proceedings of the 27th International Conference on Neural Information Processing Systems - Volume 2 (Montreal, Canada) (NIPS’14). MIT Press, Cambridge, MA, USA, 2618–2626.

Rich Zemel, Yu Wu, Kevin Swersky, Toni Pitassi, and Cynthia Dwork. 2013. Learning fair representations. In International conference on machine learning. PMLR, 325–333.

Bo Zhang and S Shyam Sundar. 2019. Proactive vs. reactive personalization: Can customization of privacy enhance user experience? International Journal of Human-Computer Studies 128 (2019), 86–99.

Mingwu Zhang, Yu Chen, and Jingqiang Lin. 2021. A Privacy-Preserving Optimization of neighbourhood-Based Recommendation for Medical-Aided Diagnosis and Treatment. IEEE Internet of Things Journal (2021).

Minxing Zhang, Zhaochun Ren, Zihan Wang, Pengjie Ren, Zhumin Chen, Pengfei Hu, and Yang Zhang. 2021. Membership Inference Attacks Against Recommender Systems. In Proceedings of the 2021 ACM SIGSAC Conference on Computer and Communications Security. 864–879.

Tianqing Zhu, Gang Li, Yongli Ren, Wanlei Zhou, and Ping Xiong. 2013. Differential privacy for neighborhood-based collaborative filtering. In Proceedings of the 2013 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining. 752–759.

Tianqing Zhu, Yongli Ren, Wanlei Zhou, Jia Rong, and Ping Xiong. 2014. An effective privacy preserving algorithm for neighborhood-based collaborative filtering. Future Generation Computer Systems 36 (2014), 142–155.

Guang Yong Zou. 2007. Toward using confidence intervals to compare correlations. Psychological Methods 12, 4 (2007), 399.