Locality Sensitive Hashing with Extended Differential Privacy *

Natasha Fernandes[1,2], Yusuke Kawamoto[3], and Takao Murakami[3]

[1] Macquarie University, Sydney, Australia
[2] Inria, École Polytechnique, IPP, Palaiseau, France
[3] National Institute of Advanced Industrial Science and Technology (AIST), Tokyo, Japan

Abstract. Extended differential privacy, a generalization of standard differential privacy (DP) using a general metric, has been widely studied to provide rigorous privacy guarantees while keeping high utility. However, existing works on extended DP are limited to few metrics, such as the Euclidean metric. Consequently, they have only a small number of applications, such as location-based services and document processing.
In this paper, we propose a couple of mechanisms providing extended DP with a different metric: *angular distance* (or *cosine distance*). Our mechanisms are based on locality sensitive hashing (LSH), which can be applied to the angular distance and work well for personal data in a high-dimensional space. We theoretically analyze the privacy properties of our mechanisms, and prove extended DP for input data by taking into account that LSH preserves the original metric only approximately. We apply our mechanisms to friend matching based on high-dimensional personal data with angular distance in the local model, and evaluate our mechanisms using two real datasets. We show that LDP requires a very large privacy budget and that RAPPOR does not work in this application. Then we show that our mechanisms enable friend matching with high utility and rigorous privacy guarantees based on extended DP.

Keywords: Local differential privacy · locality sensitive hashing · angular distance · extended differential privacy

1 Introduction

Extended differential privacy (extended DP), a.k.a. $d_{\mathcal{X}}$*-privacy* [13], is a privacy notion that provides rigorous privacy guarantees while enabling high utility. Extended DP is a generalization of standard DP [20, 21] in that the adjacency relation (regarded as the Hamming distance) is generalized to a metric. A well-known application is geo-indistinguishability [4, 7, 9], an instance of extended DP

* The authors are ordered alphabetically. This work was supported by the French-Japanese project LOGIS within the Inria Equipes Associées program, by an Australian Government RTP Scholarship (2017278), by ERATO HASUO Metamathematics for Systems Design Project (No. JPMJER1603), JST, and by JSPS KAKENHI Grant Number JP19H04113.

for two-dimensional Euclidean space. Geo-indistinguishability guarantees that a user's location is indistinguishable from any location within a certain radius (e.g., within 5km) in the local model, in which each user obfuscates her own data and sends it to a data collector. It can also be regarded as a *relaxation* of DP in the local model (local DP or LDP [19]) to make two locations within a certain radius indistinguishable (whereas LDP makes arbitrary locations indistinguishable). Consequently, extended DP results in much higher utility than LDP, e.g., for a task of estimating geographic population distributions [4].

Since extended DP is defined using a general metric, it can potentially have a wide range of applications. However, the range of actual applications is limited by the particular metrics for which extended DP mechanisms have been designed. For example, the existing works on locations [4, 7, 9], documents [25], range queries [53], and linear queries [32] are designed for the Euclidean metric, the Earth Mover's metric, the l_1 metric, and the summation of privacy budgets for attributes, respectively. However, there have been no known extended DP mechanisms designed for the angular distance (or cosine distance).

For example, consider friend matching (or friend recommendation) based on personal data (e.g., locations, rating history) [10, 14, 16, 35, 36, 38, 44, 47]. In the case of locations, we can create a vector of visit-counts where each value is the visit-count on the corresponding Point of Interest (POI). Users with similar vectors have a high probability of establishing new friendships [54]. Therefore, we can use the POI vector to recommend a new friend. Similarly, we can recommend a new friend based on the similarity of their item rating vectors, since this identifies users with similar interests [2]. Because the distance between vectors in such applications is usually given by the angular distance (or equivalently, the cosine distance) [2], the angular distance is a natural choice for the utility measure and the metric for extended DP.

In this paper, we focus on friend matching in the local model, and propose two mechanisms providing extended DP with the angular distance. Our mechanisms are based on *locality sensitive hashing* (LSH) [28, 49], which can be applied to a wide range of metrics including the angular distance. Our first mechanism, LapLSH, uses the multivariate Laplace mechanism [25] to generate noisy vectors, and then hashes them into buckets using LSH as post-processing. Our second mechanism, LSHRR, embeds personal data into a binary vector using LSH, and then applies Warner's randomized response [52] for each bit of the binary vector.

The privacy analysis of extended DP is challenging especially for LSHRR. This is because LSH does not precisely preserve the original metric; it *approximates* the original metric via hashing. We theoretically analyze the privacy properties of our mechanisms, showing that they provide extended DP for the input. We also note that much existing work on privacy-preserving LSH [3, 15, 46] fails to provide rigorous guarantees about user privacy. We point out, using a toy example, how the lack of rigorous guarantees can lead to privacy breaches.

We evaluate our mechanisms using two real datasets. We show that LDP requires a very large privacy budget ϵ. This comes from the fact that LDP expresses an upper bound on the privacy guarantee for all inputs. In contrast, extended

DP is a finer-grained notion than LDP in that it describes the privacy guarantee for inputs at various distances. In fact, we show that extended DP enables friend matching with a much smaller privacy budget than LDP for close inputs.

We also explain why RAPPOR [23] and the generalized RAPPOR [51], which are state-of-the-art LDP mechanisms, cannot be applied (either completely lose utility or are computationally infeasible) to friend matching. In short, the Bloom filter used in RAPPOR is not a metric-preserving hashing, and therefore cannot guarantee utility w.r.t. the metric distance between user vectors. This is further elaborated in Sect. 7.4.

Contributions. Our main contributions are as follows:

- We propose two mechanisms providing extended DP with the angular distance: LapLSH and LSHRR. We show that LSH itself does not provide privacy guarantees and could result in complete privacy collapse in some situations. We then prove that our mechanisms provide rigorous guarantees of extended DP. In particular, we show that the distribution of the LSHRR's privacy loss can be characterized as extended notions of concentrated DP [22] and probabilistic DP [39] with input distance. To our knowledge, this work is the first to provide extended DP with the angular distance.
- We apply our mechanisms to friend matching based on rating history and locations. Then we compare LSHRR with LapLSH using two real datasets. We show that LSHRR provides higher (resp. lower) utility than LapLSH for a high-dimensional (resp. low-dimensional) vector. We also show that LDP requires a very large privacy budget ϵ, and RAPPOR does not work for friend matching. Finally, we show that LSHRR provides high utility for a high-dimensional vector (e.g., 1000-dimensional rating/location vector) in the medium privacy regime [1, 55] of extended DP, and therefore enables friend matching with rigorous privacy guarantees and high utility.

All proofs on the technical results can be found in Appendix B.

2 Related Work

2.1 Extended DP

As explained in Sect. 1, there are a number of existing extended DP mechanisms [4, 7, 9, 25, 32, 53] designed for other metrics (e.g., the Euclidean metric, the l_1 metric), which cannot be applied to the angular distance. To our knowledge, our mechanisms are the first to provide extended DP with the angular distance.

In addition, most of the studies on extended DP have studied low-dimensional data such as two-dimensional [4, 7, 9, 32] and six-dimensional [53] data. One exception is the work in [25], which proposed the multivariate Laplace mechanism for 300-dimensional vectors. In this paper, we apply our mechanisms to vectors in 1000-dimensions (much larger than any existing work), and show that our LSHRR provides high utility for such high-dimensional data.

2.2 Privacy-Preserving Friend Matching

A number of studies [10, 14, 16, 35, 36, 38, 44, 47] have been made on algorithms for privacy-preserving friend matching (or friend recommendation). Many of them (e.g., [16, 38, 44, 47]) use cryptographic techniques such as homomorphic encryption and secure multiparty computation. However, such techniques require high computational costs or focus on specific algorithms, and are not suitable for a more complicated calculation of distance such as the angular distance between two rating/location vectors.

The techniques in [10, 14, 35, 36] are based on perturbation. The mechanisms in [10, 35, 36] do not provide DP or its variant, whereas that in [14] provides DP. The technique in [14], however, is based on social graphs and cannot be applied to our setting, where a user's personal data is represented as a rating vector or visit-count vector. Moreover, DP-based friend matching in social graphs can require prohibitive trade-offs between utility and privacy [10, 40].

Similarly, DP mechanisms based on each user's high-dimensional rating/location vector require a very large privacy budget (e.g., $\varepsilon \geq 250$ [37], $\varepsilon \geq 2 \times 10^4$ [42]) to provide high utility. In contrast, our extended DP mechanisms provide meaningful privacy guarantees in high-dimensional spaces with high utility, since extended DP is a finer-grained notion than DP, as explained in Sect. 1.

We also note that a privacy-preserving clustering algorithm in [45] and an item recommendation algorithm in [48] cannot be applied to friend matching.

2.3 Privacy-Preserving LSH

Finally, we note that some studies have proposed privacy-preserving LSH [3, 8, 15, 29, 45, 46, 56]. However, some of them [3, 15, 46] only apply LSH and claim that it protects user privacy because LSH is a kind of non-invertible transformation. In Sect. 4, we show that the lack of rigorous guarantees can lead to privacy breaches. Nissim and Stemmer [45] proposed clustering algorithms based on LSH and the heavy-hitters algorithm. However, their algorithms focus on clustering such as k-means clustering and cannot be applied to friend matching.

Aumüller *et al.* [8] proposed a privacy-preserving LSH algorithm that can be applied to friend matching. Specifically, they focused on a similarity search problem under the Jaccard similarity using up to 2000-dimensional vectors, and proposed an LDP algorithm based on MinHash. After the submission of our paper to a preprint [26], two related papers [29, 56] have been published. Zhang *et al.* [56] proposed an LDP algorithm for rating prediction based on MinHash and knowledge distillation. Hu *et al.* [29] proposed an LDP algorithm based on LSH for federated recommender system.

Our work differs from [8, 29, 56] in the following points. First, [8, 29, 56] only analyzed LDP for hashes, and did not conduct a more challenging analysis of extended DP for inputs. In contrast, our work provides a careful analysis of extended DP, given that LSH preserves the original metric only approximately. We also show that extended DP requires a much smaller privacy budget than LDP. Second, we compared LSHRR with LapLSH in detail, and show that LSHRR (resp. LapLSH) is more suitable for high (resp. low) dimensional data.

3 Preliminaries

In this section, we introduce notations and recall background on locality sensitive hashing (LSH), privacy measures, and privacy protection mechanisms.

Let d_{euc} be the Euclidean distance between real vectors, i.e., $d_{\mathrm{euc}}(\boldsymbol{x}, \boldsymbol{x}') = \|\boldsymbol{x} - \boldsymbol{x}'\|_2$. We write $\mathcal{V}$ for the set of all binary data of length κ, i.e., $\mathcal{V} = \{0,1\}^\kappa$. The *Hamming distance* between $\boldsymbol{v}, \boldsymbol{v}' \in \mathcal{V}$ is: $d_{\mathcal{V}}(\boldsymbol{v}, \boldsymbol{v}') = \sum_{i=1}^{\kappa} |\, v_i - v'_i \,|$.

We denote the *set of all probability distributions* over a set $\mathcal{S}$ by $\mathbb{D}\mathcal{S}$. Let $N(\mu, \sigma^2)$ be the normal distribution with mean μ and variance σ^2. Let $A : \mathcal{X} \to \mathbb{D}\mathcal{Y}$ be a randomized algorithm from a finite set $\mathcal{X}$ to another $\mathcal{Y}$, and $A(x)[y]$ (resp. by $A(x)[S]$) be the probability that A maps x to y (resp. an element of S).

3.1 Locality Sensitive Hashing (LSH)

We denote by $\mathcal{X}$ the set of all possible input data. We introduce the notion of a *(normalized) dissimilarity function* $d_{\mathcal{X}} : \mathcal{X} \times \mathcal{X} \to [0,1]$ over $\mathcal{X}$ such that two inputs $\boldsymbol{x}$ and $\boldsymbol{x}'$ have less dissimilarity $d_{\mathcal{X}}(\boldsymbol{x}, \boldsymbol{x}')$ when they are closer, and that $d_{\mathcal{X}}(\boldsymbol{x}, \boldsymbol{x}') = 0$ when $\boldsymbol{x} = \boldsymbol{x}'$. If $d_{\mathcal{X}}$ is symmetric and subadditive, it is a metric.

A *locality sensitive hashing (LSH)* [28] is a family of functions in which the probability of two inputs $\boldsymbol{x}, \boldsymbol{x}'$ having different 1-bit outputs is proportional to $d_{\mathcal{X}}(\boldsymbol{x}, \boldsymbol{x}')$.

Definition 1 (Locality sensitive hashing). A *locality sensitive hashing (LSH) scheme* w.r.t. a dissimilarity function $d_{\mathcal{X}}$ is a family $\mathcal{H}$ of functions from $\mathcal{X}$ to $\{0,1\}$ coupled with a probability distribution $D_{\mathcal{H}}$ such that for any $\boldsymbol{x}, \boldsymbol{x}' \in \mathcal{X}$,

$$\Pr_{h \sim D_{\mathcal{H}}}[h(\boldsymbol{x}) \neq h(\boldsymbol{x}')] = d_{\mathcal{X}}(\boldsymbol{x}, \boldsymbol{x}'), \tag{1}$$

where h is chosen from $\mathcal{H}$ according to the distribution $D_{\mathcal{H}}$. By using independently chosen functions $h_1, h_2, \ldots, h_\kappa$, the *$\kappa$-bit LSH function* $H : \mathcal{X} \to \mathcal{V}$ is:

$$H(\boldsymbol{x}) = (h_1(\boldsymbol{x}),\, h_2(\boldsymbol{x}),\, \ldots,\, h_\kappa(\boldsymbol{x})). \tag{2}$$

We denote by $H^* : \mathcal{X} \to \mathbb{D}\mathcal{V}$ the randomized algorithm that draws a κ-bit LSH H from the distribution $D_{\mathcal{H}}^\kappa$ and outputs the hash value $H(\boldsymbol{x})$ of a given input $\boldsymbol{x}$.

3.2 Examples of LSHs

There are a variety of LSH families corresponding to useful metrics, such as the angular distance [5, 12], Jaccard metric [11], and l_p metric with $p \in (0, 2]$ [18]. In this work, we focus on LSH families for the angular distance.

A *random-projection-based hashing* is a one-bit hashing with the domain $\mathcal{X} \stackrel{\mathrm{def}}{=} \mathbb{R}^n$ and a random vector $\boldsymbol{r} \in \mathbb{R}^n$ that defines a hyperplane through the origin. Formally, we define a *random-projection-based hashing* $h_{\mathsf{proj}} : \mathbb{R}^n \to \{0,1\}$ by:

$$h_{\mathsf{proj}}(\boldsymbol{x}) = \begin{cases} 0 & (\text{if } \boldsymbol{r}^\top \boldsymbol{x} < 0) \\ 1 & (\text{otherwise}) \end{cases}$$

where each element of $\boldsymbol{r}$ is independently chosen from the standard normal distribution $N(0,1)$. By (2), a κ-bit LSH function H_{proj} is built from one-bit hashes $h_{\mathsf{proj}_1}, \ldots, h_{\mathsf{proj}_\kappa}$ that are generated from independent hyperplanes $\boldsymbol{r}_1, \ldots, \boldsymbol{r}_\kappa$.

The random-projection-based hashing h_{proj} is an LSH w.r.t. the *angular distance* $d_\theta : \mathbb{R}^n \times \mathbb{R}^n \to [0,1]$ defined by:

$$d_\theta(\boldsymbol{x}, \boldsymbol{x}') = \tfrac{1}{\pi} \cos^{-1}\left(\frac{\boldsymbol{x}^\top \boldsymbol{x}'}{\|\boldsymbol{x}\|\|\boldsymbol{x}'\|}\right) \tag{3}$$

For example, $d_\theta(\boldsymbol{x}, \boldsymbol{x}') = 0$ iff $\boldsymbol{x} = \boldsymbol{x}'$, while $d_\theta(\boldsymbol{x}, \boldsymbol{x}') = 1$ iff $\boldsymbol{x} = -\boldsymbol{x}'$. $d_\theta(\boldsymbol{x}, \boldsymbol{x}') = 0.5$ exactly when the two vectors $\boldsymbol{x}$ and $\boldsymbol{x}'$ are orthogonal, namely, $\boldsymbol{x}^\top \boldsymbol{x}' = 0$.

3.3 Approximate Nearest Neighbor Search

We recall the nearest neighbor search (NNS) problem and its utility measures.

Given a dataset $S \subseteq \mathcal{X}$, the *nearest neighbor search (NNS)* for an $x_0 \in S$ is the problem of finding the closest $x \in S$ to x_0 w.r.t. a metric $d_{\mathcal{X}}$ over $\mathcal{X}$. A *k-nearest neighbor search (k-NNS)* is the problem of finding the k closest points.

A naive and exact approach to k-NNS is to perform pairwise comparisons of data points, requiring $O(|S|)$ operations. Approaches to improve this computational inefficiency shift the problem to space inefficiency [6]. An alternative approach [30] is to employ LSH to perform *approximate* NNS efficiently. To evaluate the utility, we use the average distance of returned nearest neighbors from the data point x_0 compared with the average distance of true nearest neighbors.

Definition 2 (Utility loss). Let A be an approximate algorithm that produces approximate k nearest neighbors $N \subseteq S$ for a data point $x_0 \in S$ in terms of a metric $d_{\mathcal{X}}$. The *average utility loss* for N w.r.t. the true nearest neighbors T is given by: $\mathcal{U}_A(S) = {}^1\!/_k \sum_{x \in N} d_{\mathcal{X}}(x_0, x) - {}^1\!/_k \sum_{x \in T} d_{\mathcal{X}}(x_0, x)$.

3.4 Privacy Measures and Privacy Mechanisms

Extended differential privacy [13, 34] guarantees that when two inputs x and x' are closer, their corresponding output distributions are less distinguishable. In this paper, we propose a more generalized definition using a function δ over $\mathcal{X}$ and an arbitrary function ξ over $\mathcal{X}$ instead of a metric. The main reason for this generalization is that LSH preserves the metric over the input only probabilistically and approximately, hence cannot fit to [13]'s standard definition.

Definition 3 (Extended differential privacy). Given two functions $\xi : \mathcal{X} \times \mathcal{X} \to \mathbb{R}_{\geq 0}$ and $\delta : \mathcal{X} \times \mathcal{X} \to [0,1]$, a randomized algorithm $A : \mathcal{X} \to \mathbb{D}\mathcal{Y}$ provides (ξ, δ)-*extended differential privacy (XDP)* if for all $x, x' \in \mathcal{X}$ and for any $S \subseteq \mathcal{Y}$,

$$A(x)[S] \leq e^{\xi(x,x')}\, A(x')[S] + \delta(x, x'),$$

where the probability is taken over the random choices in A.

We abuse notation and write δ when $\delta(x, x')$ is a constant. When $\xi(x, x')$ is also a constant ε, the definition gives the (standard) *differential privacy* (DP). When $\xi(x, x') = d_{\mathcal{X}}(x, x')$ and $\delta(x, x') = 0$, the definition gives $d_{\mathcal{X}}$-privacy in [13]. In later sections, we instantiate the metric $d_{\mathcal{X}}$ with the angular distance d_θ.

Finally, we recall some popular privacy protection mechanisms.

Definition 4 (Laplace mechanism [21]). For an $\varepsilon \in \mathbb{R}_{>0}$ and a metric $d_{\mathcal{X}}$ over $\mathcal{X} \cup \mathcal{Y}$, the $(\varepsilon, d_{\mathcal{X}})$-*Laplace mechanism* is the randomized algorithm $Q_{\mathsf{Lap}} : \mathcal{X} \to \mathbb{D}\mathcal{Y}$ that maps an input x to an output y with probability $\frac{1}{c}\exp(-\varepsilon d_{\mathcal{X}}(x, y))$ where $c = \int_{\mathcal{Y}} \exp(-\varepsilon d_{\mathcal{X}}(x, y))\ dy$.

Examples of the $(\varepsilon, d_{\mathcal{X}})$-Laplace mechanism include the one-dimensional [21] and the multivariate Laplace mechanism [25], both equipped with the Euclidean metric. The $(\varepsilon, d_{\mathcal{X}})$-Laplace mechanism provides $(\varepsilon d_{\mathcal{X}}, 0)$-$\mathsf{XDP}$.

Definition 5 (Randomized response [52]). The ε-*randomized response* (ε-*RR*) is the randomized algorithm $Q_{\mathsf{rr}} : \{0, 1\} \to \mathbb{D}\{0, 1\}$ that maps a bit b to another b' with probability $\frac{e^\varepsilon}{e^\varepsilon+1}$ if $b' = b$, and with probability $\frac{1}{e^\varepsilon+1}$ otherwise.

The ε-RR provides ε-DP. Erlingsson et al. [23] introduce the *RAPPOR*, which first uses a Bloom filter to produce a hash value and then applies the RR to each bit of the hash value. The RAPPOR provides ε-DP in the local model.

4 Privacy Properties of LSH

Several works in the literature make reference to the privacy-preserving properties of LSH [3, 17, 46]. The privacy guarantee attributed to LSH mechanisms hinges on its hash function, which 'protects' an individual's private attributes by revealing only their hash bucket. We now apply a formal analysis to LSH and explain why LSH implementations do not provide strong privacy guarantees, and could, in some situations, result in complete privacy collapse for the individual.

Modeling LSH. We present a simple example to show how privacy breaks down. Consider the set of secret inputs $\mathcal{X}=\{(0, 1), (1, 0), (1, 1)\}$ whose element represents whether an individual rated two movies A and B. Then an LSH is modeled as a probabilistic channel $h^* : \mathcal{X} \to \mathbb{D}\{0, 1\}$ that maps a secret input to a binary observation.

For brevity, we deal with a single random-projection-based hashing h in Sect. 3.2. That is, we randomly choose a vector $\boldsymbol{r}$ representing the normal to a hyperplane, and given an input $\boldsymbol{x} \in \mathcal{X}$, the hash function h outputs 0 if $\boldsymbol{r}^\top \boldsymbol{x} < 0$ and 1 otherwise. For example, if $\boldsymbol{r} = (1, -\frac{1}{2})$ is chosen, h is defined as:

$$
\begin{aligned}
&h : \mathcal{X} \to \{0, 1\} \\
&(0, 1) \mapsto 0 \\
&(1, 0) \mapsto 1 \\
&(1, 1) \mapsto 1
\end{aligned}
$$

In fact, there are 6 possible (deterministic) hash functions for any choice of the vector $\boldsymbol{r}$, corresponding to hyperplanes that separate different pairs of points:

h_1	h_2	h_3
$(0,1) \mapsto 1$	$(0,1) \mapsto 0$	$(0,1) \mapsto 1$
$(1,0) \mapsto 0$	$(1,0) \mapsto 1$	$(1,0) \mapsto 0$
$(1,1) \mapsto 0$	$(1,1) \mapsto 1$	$(1,1) \mapsto 1$
h_4	h_5	h_6
$(0,1) \mapsto 0$	$(0,1) \mapsto 1$	$(0,1) \mapsto 0$
$(1,0) \mapsto 1$	$(1,0) \mapsto 1$	$(1,0) \mapsto 0$
$(1,1) \mapsto 0$	$(1,1) \mapsto 1$	$(1,1) \mapsto 0$

Each of h_1, h_2, h_3, and h_4 occurs with probability $1/8$, while h_5 and h_6 each occur with probability $1/4$. The resulting channel h^*, computed as the probabilistic sum of these deterministic hash functions, turns out to leak no information on the secret input (i.e., all outputs have equal probability conditioned on each input).

This indicates that the channel h^* is perfectly private. However, in practice, LSH may require the release of the choice of the vector $\boldsymbol{r}$ (e.g. [17])[4], that is, the choice of hash function is leaked. Notice that in our example, h_1 to h_4 correspond to deterministic mechanisms which leak exactly 1 bit of the secret, while h_5 and h_6 leak nothing. In other words, with 50% probability, 1 bit of the 2-bit secret is leaked. Furthermore, h_1 and h_2 leak the secret $(0,1)$ exactly, and h_3 and h_4 leak $(1,0)$ exactly. Thus, the release of $\boldsymbol{r}$ destroys the privacy guarantee.

The Guarantee of LSH. In general, for any number of hash functions and any length of input, an LSH which releases its choice of hyperplanes also leaks its choice of deterministic mechanism. This means that it leaks the equivalence classes of the secrets. Such mechanisms belong to the 'k-anonymity'-style of privacy mechanisms which promise privacy by hiding secrets in equivalence classes of size at least k. These have been shown to be unsafe due to their failure to compose well [27, 24, 33]. This failure leads to the potential for linkage or intersection attacks by an adversary armed with auxiliary information. For this reason, we consider compositionality an essential property for a privacy-preserving system. LSH with hyperplane release does not provide such privacy guarantees.

5 LSH-based Privacy Mechanisms

In this section, we propose two privacy protection mechanisms called *LSHRR* and *LapLSH*. The former is an extension of RAPPOR [23] w.r.t. LSH, and the latter is constructed using the Laplace mechanism and LSH.

Construction of LSHRR. We introduce the *LSH-then-RR privacy mechanism* (*LSHRR*) as the randomized algorithm that (i) randomly chooses a κ-bit LSH function H, (ii) computes the κ-bit hash value $H(\boldsymbol{x})$ of a given input $\boldsymbol{x}$, and (iii) applies the randomized response to each bit of $H(\boldsymbol{x})$.

To formalize this, we define the (ε, κ)-*bitwise RR* Q_{brr}, which applies the randomized response Q_{rr} to each bit of the input independently. Formally, $Q_{\mathsf{brr}} : \mathcal{V} \to \mathbb{D}\mathcal{V}$ maps a bitstring $\boldsymbol{v} = (v_1, v_2, \ldots, v_\kappa)$ to another $\boldsymbol{y} = (y_1, y_2, \ldots, y_\kappa)$ with probability $Q_{\mathsf{brr}}(\boldsymbol{v})[\boldsymbol{y}] = \prod_{i=1}^{\kappa} Q_{\mathsf{rr}}(v_i)[y_i]$. Then LSHRR is defined as follows.

[4] In fact, since the channel on its own leaks nothing, there *must* be further information released in order to learn anything useful from this channel.

Definition 6 (LSHRR). The *ε-LSH-then-RR privacy mechanism* (*LSHRR*) instantiated with a κ-bit LSH function $H : \mathcal{X} \to \mathcal{V}$ is the randomized algorithm $Q_H : \mathcal{X} \to \mathbb{D}\mathcal{V}$ defined by $Q_H = Q_{\mathsf{brr}} \circ H$. Given a distribution $D^{\kappa}_{\mathcal{H}}$ of the κ-bit LSH functions, the ε-LSHRR w.r.t. $D^{\kappa}_{\mathcal{H}}$ is defined by $Q_{\mathsf{LSHRR}} = Q_{\mathsf{brr}} \circ H^*$.

LSHRR deals with two kinds of randomness: (a) the randomness in choosing a (deterministic) LSH function H from $D^{\kappa}_{\mathcal{H}}$ (e.g., the random seed $\boldsymbol{r}$ in the random-projection-based hashing h_{proj}), and (b) the random noise added by the bitwise RR Q_{brr}. We can assume that each user of this privacy mechanism selects an input $\boldsymbol{x}$ independently of both kinds of randomness, since they wish to protect their own privacy when publishing $\boldsymbol{x}$.

In practical settings, the same LSH function H is often used to produce hash values of different inputs; namely, multiple hash values are dependent on an identical hash seed (e.g., a service provider would generate a hash seed so that multiple users can share the same H to compare their hash values). Furthermore, the adversary might obtain the LSH function H (or the seed $\boldsymbol{r}$ used to produce H), and might learn a set of possible inputs that produce the same hash value $H(x)$ without knowing the actual input x. Therefore, the hash value $H(x)$ might reveal partial information on the input x (see Sect. 4), and the bitwise RR Q_{brr} is crucial in guaranteeing privacy (see Sect. 6 for our privacy analyses).

On the other hand, Q_{brr} causes errors in the Hamming distance as follows:

Proposition 1 (Error bound) *For any $x, x' \in \mathcal{X}$, the expected error in the Hamming distance satisfies $\mathbb{E}[|d_{\mathcal{V}}(Q_H(x), Q_H(x')) - d_{\mathcal{V}}(H(x), H(x'))|] \leq \frac{2\kappa}{1+e^{\varepsilon}}$ where the expectation is taken over the randomness in the bitwise RR.*

Construction of LapLSH. We also propose the *Laplace-then-LSH privacy mechanism* (LapLSH) as the randomized algorithm that (i) randomly chooses a κ-bit LSH function H, (ii) applies the multivariate Laplace mechanism Q_{Lap} to $\boldsymbol{x}$, and (iii) computes the κ-bit hash value $H(Q_{\mathsf{Lap}}(\boldsymbol{x}))$.

Definition 7 (LapLSH). The *$(\varepsilon, d_{\mathcal{X}})$-Laplace-then-LSH privacy mechanism* (*LapLSH*) with a κ-bit LSH function $H : \mathcal{X} \to \mathcal{V}$ is the randomized algorithm $Q_{\mathsf{Lap}H} : \mathcal{X} \to \mathbb{D}\mathcal{V}$ defined by $Q_{\mathsf{Lap}H} = H \circ Q_{\mathsf{Lap}}$. The $(\varepsilon, d_{\mathcal{X}})$-LapLSH w.r.t. a distribution $D^{\kappa}_{\mathcal{H}}$ of the κ-bit LSH functions is defined by $Q_{\mathsf{LapLSH}} = H^* \circ Q_{\mathsf{Lap}}$.

LapLSH also deals with the two kinds of randomness discussed above, and the Laplace mechanism Q_{Lap} is crucial in guaranteeing privacy. One of the main differences from LSHRR is that LapLSH adds noise directly to the input before applying LSH whereas LSHRR adds noise after applying LSH to the input.

In Sect. 7 we implement the multivariate Laplace mechanism with the input domain $\mathcal{X} = \mathbb{R}^n$ and Euclidean distance d_{euc} described in [25]; namely, we generate additive noise by constructing a unit vector uniformly at random over the n-dimensional unit sphere $\mathbb{S}^n$, scaled by a random value generated from the gamma distribution with shape n and scale $1/\varepsilon$.

6 Privacy Analyses of the Mechanisms

We provide an analysis of the privacy guarantees provided by our mechanisms in two operational scenarios: (i) w.r.t. an already-chosen LSH function (e.g., where it has been generated by a service provider), and (ii) w.r.t. all possible choices of the LSH function (e.g., prior to its instantiation by a particular service provider). Note that our analysis is general in that it does not rely on specific metrics or hashing algorithms for LSH.

6.1 LSHRR's Privacy w.r.t. the Particular LSH Function

We first show the privacy guarantee for LSHRR w.r.t. the particular LSH function used by the service provider. This type of privacy is defined using the Hamming distance $d_{\mathcal{V}}$ between the hash values of given inputs, and the degree of privacy depends on the actual selection of the LSH function H (or the hash seeds $\boldsymbol{r}$), which we assume is available to the adversary. Since LSH preserves the original metric $d_{\mathcal{X}}$ only approximately, we obtain XDP guarantee w.r.t. a pseudo-metric $d_{\varepsilon H}$ that approximates $d_{\mathcal{X}}$ as follows.

Proposition 2 (XDP of Q_H) *Let $H : \mathcal{X} \to \mathcal{V}$ be a κ-bit LSH function, and $d_{\varepsilon H}$ be the pseudometric over $\mathcal{X}$ defined by $d_{\varepsilon H}(\boldsymbol{x}, \boldsymbol{x}') = \varepsilon d_{\mathcal{V}}(H(\boldsymbol{x}), H(\boldsymbol{x}'))$ for $\boldsymbol{x}, \boldsymbol{x}' \in \mathcal{X}$. Then the ε-LSHRR Q_H instantiated with H provides $(d_{\varepsilon H}, 0)$-XDP.*

However, we cannot compute $d_{\varepsilon H}$ or the degree of XDP in Proposition 2 until H has been computed. To overcome this unclear guarantee of privacy, in Sect. 6.2 we show a useful privacy guarantee that can be evaluated without requiring H (or hash seeds) generated by the service provider.

Note that the $\kappa\varepsilon$-DP of LSHRR is obtained as the worst case of Proposition 2, i.e., when the hamming distance between vectors is maximum due to an "unlucky" choice of hash seeds or very large distance $d_{\mathcal{X}}(\boldsymbol{x}, \boldsymbol{x}')$ between the inputs $\boldsymbol{x}, \boldsymbol{x}'$. The following proposition guarantees the privacy independently of the actual choice of H.

Proposition 3 (Worst-case privacy of Q_H) *For a κ-bit LSH function H, the ε-LSHRR Q_H instantiated with H provides $\kappa\epsilon$-DP.*

6.2 LSHRR's Privacy w.r.t. the Distribution of LSH Functions

Next, we show LSHRR's privacy guarantee w.r.t. any possible LSH function that may be generated. This type of privacy guarantee is useful in a variety of scenarios. For example, a privacy analyst could evaluate the expected degree of privacy before the service provider fixes the LSH function or hash seeds. For another example, the seeds may be stored in tamper-resistant hardware privately.

The privacy guarantee without relying on specific LSH functions or hash seeds is modeled as a probability distribution of degrees of XDP over the random choice of seeds. Then this can be characterized as an extension of concentrated DP [22] and probabilistic DP [39] with input distance, yielding the XDP guarantee.

In the privacy analysis, we deal with the situation where multiple users produce hash values by employing the same hash seeds, as seen in typical applications such as approximate NNS. Then we define privacy notions for the mechanisms that share randomness among them.

Formally, we denote by $A_r : \mathcal{X} \to \mathbb{D}\mathcal{Y}$ a randomized algorithm A with a shared input $r \in \mathcal{R}$. Given a distribution λ over a finite set $\mathcal{R}$ of shared input, we denote by $A_\lambda : \mathcal{X} \to \mathbb{D}\mathcal{Y}$ the randomized algorithm that draws a shared input r from λ and behaves as A_r; i.e., $A_\lambda(x)[y] = \sum_{r\in\mathcal{R}} \lambda[r]A_r(x)[y]$. Then we extend the notion of privacy loss [22] with shared randomness as follows.

Definition 8 (Privacy loss). For a randomized algorithm $A_r : \mathcal{X} \to \mathbb{D}\mathcal{Y}$ with a shared input r, the *privacy loss* on $y \in \mathcal{Y}$ w.r.t. $x, x' \in \mathcal{X}$, $r \in \mathcal{R}$ is defined by:

$$\mathcal{L}_{x,x',y,r} = \ln\left(\frac{A_r(x)[y]}{A_r(x')[y]}\right),$$

where the probability is taken over the random choices in A_r. Given a distribution λ over $\mathcal{R}$, the *privacy loss random variable* $\mathcal{L}_{x,x'}$ of x over x' w.r.t. λ is the real-valued random variable representing the privacy loss $\mathcal{L}_{x,x',y,r}$ where a *shared randomness* r is sampled from λ and y is sampled from $A_r(x)$.

To characterize the privacy loss random variable $\mathcal{L}_{x,x'}$ for LSHRR, we introduce an extension of CDP [22] wth input distance $d(x, x')$ as follows.

Definition 9 (CXDP). Let $\mu \in \mathbb{R}_{\geq 0}$, $\tau \in \mathbb{R}_{>0}$, $\lambda \in \mathbb{D}\mathcal{R}$, and $d : \mathcal{X} \times \mathcal{X} \to \mathbb{R}_{\geq 0}$ be a metric. A random variable Z over $\mathbb{R}$ is *τ-subgaussian* if for all $s \in \mathbb{R}$, $\mathbb{E}[\exp(sZ)] \leq \exp(\frac{s^2\tau^2}{2})$. A randomized algorithm $A_\lambda : \mathcal{X} \to \mathbb{D}\mathcal{Y}$ provides *(μ, τ, d)-mean-concentrated extended differential privacy (CXDP)* if for all $x, x' \in \mathcal{X}$, the privacy loss random variable $\mathcal{L}_{x,x'}$ of x over x' w.r.t. λ satisfies that $\mathbb{E}[\mathcal{L}_{x,x'}] \leq \mu d(x, x')$, and that $\mathcal{L}_{x,x'} - \mathbb{E}[\mathcal{L}_{x,x'}]$ is τ-subgaussian.

Then we obtain the following CXDP guarantee for LSHRR.

Proposition 4 (CXDP of Q_{LSHRR}) *The ε-LSHRR provides $(\varepsilon\kappa, \frac{\varepsilon\kappa}{2}, d_{\mathcal{X}})$-CXDP.*

To clarify the implication of CXDP, we introduce an extension of probabilistic DP [39] with input distance, which we call PXDP. Intuitively, (ξ, δ)-PXDP guarantees $(\xi, 0)$-XDP with probability $1 - \delta$.

Definition 10 (PXDP). Let $\lambda \in \mathbb{D}\mathcal{R}$, $\xi : \mathcal{X} \times \mathcal{X} \to \mathbb{R}_{\geq 0}$, and $\delta : \mathcal{X} \times \mathcal{X} \to [0, 1]$. A randomized algorithm $A_\lambda : \mathcal{X} \to \mathbb{D}\mathcal{Y}$ provides *(ξ, δ)-probabilistic extended differential privacy (PXDP)* if for all $x, x' \in \mathcal{X}$, $\Pr[\,\mathcal{L}_{x,x'} > \xi(x, x')\,] \leq \delta(x, x')$. We abuse notation to write δ when $\delta(x, x')$ is constant.

In Appendix B, we show that CXDP implies PXDP and that PXDP implies XDP. Based on these, we show that LSHRR provides PXDP and XDP as follows.

Theorem 1 (PXDP/XDP of Q_{LSHRR}) *Let $\delta \in \mathbb{R}_{>0}$, $\varepsilon' = \varepsilon\sqrt{\frac{-\ln\delta}{2}}$, and $\xi(\boldsymbol{x}, \boldsymbol{x}') = \varepsilon\kappa d_{\mathcal{X}}(\boldsymbol{x}, \boldsymbol{x}') + \varepsilon'\sqrt{\kappa}$. The ε-LSHRR provides (ξ, δ)-PXDP, hence (ξ, δ)-XDP.*

For our experimental evaluation, we show a privacy guarantee that gives tighter bounds but requires the parameters dependent on the inputs $\boldsymbol{x}$ and $\boldsymbol{x}'$.

Proposition 5 (Tighter bound for PXDP/XDP) *For $a, b \in \mathbb{R}_{>0}$, let $D_{\mathrm{KL}}(a\|b) = a \ln \frac{a}{b} + (1-a) \ln \frac{1-a}{1-b}$. For an $\alpha \in \mathbb{R}_{>0}$, we define:*

$$\xi_\alpha(\boldsymbol{x}, \boldsymbol{x}') = \varepsilon\kappa(d_{\mathcal{X}}(\boldsymbol{x}, \boldsymbol{x}') + \alpha)$$
$$\delta_\alpha(\boldsymbol{x}, \boldsymbol{x}') = \exp\big(-\kappa D_{\mathrm{KL}}(d_{\mathcal{X}}(\boldsymbol{x}, \boldsymbol{x}') + \alpha \| d_{\mathcal{X}}(\boldsymbol{x}, \boldsymbol{x}'))\big).$$

Then the ε-LSHRR provides $(\xi_\alpha, \delta_\alpha)$-PXDP, hence $(\xi_\alpha, \delta_\alpha)$-XDP.

6.3 Privacy Guarantee for LapLSH

Finally, we also show that LapLSH provides XDP. This is immediate from the fact that XDP is preserved under the post-processing by an LSH function.

Proposition 6 (XDP of $Q_{\mathsf{Lap}H}$ and Q_{LapLSH}) *The $(\varepsilon, d_{\mathcal{X}})$-LapLSH $Q_{\mathsf{Lap}H}$ with a κ-bit LSH function H provides $(\varepsilon d_{\mathcal{X}}, 0)$-XDP. The $(\varepsilon, d_{\mathcal{X}})$-LapLSH Q_{LapLSH} w.r.t. a distribution $D^{\kappa}_{\mathcal{H}}$ of the κ-bit LSH functions also provides $(\varepsilon d_{\mathcal{X}}, 0)$-XDP.*

7 Experimental Evaluation

We show an experimental evaluation of LSHRR and LapLSH on two real datasets: MovieLens [41] and FourSquare [54]. Our goal is to determine the utility of these mechanisms when compared with a (slow but accurate) true nearest neighbor search. As a baseline, we also show the performance of vanilla (non-private) LSH.

7.1 Datasets and Experimental Setup

Our problem of interest is *privacy-preserving friend matching (or friend recommendation)*. In this scenario, we are given a dataset of users in which each user is represented as a (real-valued) vector of attributes. The data curator's goal is to recommend k friends for each user based on their k-nearest neighbors.

For our experiments, we used the following two datasets:

MovieLens. The MovieLens 25m dataset [41] contains 162000 users with ratings across 62000 movies, with ratings ranging from 1 to 5. We normalized the scores, i.e., to mean 0, and gave unseen movies a score of 0. For each user, we constructed a rating vector that consists of the user's rating for each movie.

Foursquare. The Foursquare dataset (Global-scale Check-in Dataset with User Social Networks) [54] contains 90048627 check-ins by 2733324 users on POIs all over the world. We extracted 107091 POIs in New York and 10000 users who have visited at least one POI in New York. For each user, we constructed a visit-count vector, which consists of a visit-count value for each POI.

For both datasets, we generated input (rating/visit-count) vectors of length $n = 100, 500, 1000$ to evaluate the effectiveness of LSH. Reduced vector lengths

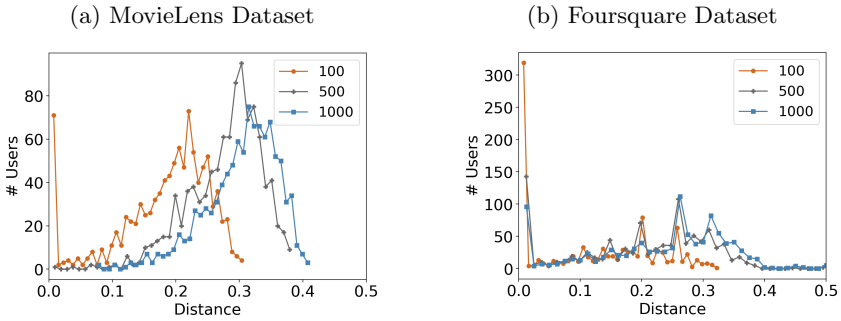

Fig. 1: Distributions of angular distances d_θ to nearest neighbor for $k = 1$ for each user, plotted for vectors with dimensions 100, 500 and 1000. The distance 0.5 represents orthogonal vectors; i.e., having no items in common. The privacy guarantee for users is a function of the distance d_θ to their nearest neighbors.

were used because LSH has poor utility for larger vector lengths and the utility of our mechanisms requires a good baseline utility for LSH.

We computed the k nearest neighbors w.r.t. the angular distance d_θ for 1000 users for $k = 1, 5, 10$ using standard NNS (i.e., pairwise comparisons over all inputs). The distributions of True Nearest Neighbor distances are shown in Fig. 1.

κ-bit LSH (for $\kappa = 10, 20, 50$) was implemented using the *random-projection-based hashing*. For each user, we then computed their k nearest neighbors for $k = 1, 5, 10$ using the Hamming distance on bitstrings. To compute the overall (ξ, δ)-XDP guarantee as per Proposition 5, we fixed $\delta = 0.01$ and $d_\theta = 0.1$ and varied ε to generate ξ values in the range 0.1 to 20.

Fig. 1 shows that about 43% (resp. 16%) of input vectors with 100 (resp. 1000) dimensions are within the distance of 0.1 in the Foursquare dataset. Thus, extended DP with $d_\theta = 0.1$ is useful to hide such input vectors.

7.2 Comparing Privacy and Utility

We use the angular distance d_θ as our utility measure, i.e., to determine similar users for the purposes of recommendations. For utility loss, we use Definition 2 instantiated with the angular distance d_θ. We compare the utility loss of each mechanism w.r.t. a comparable privacy guarantee, namely the overall privacy budget $\varepsilon\, d_{\text{euc}}(\boldsymbol{x}, \boldsymbol{x}')$ for LapLSH and $\xi_\alpha(\boldsymbol{x}, \boldsymbol{x}')$ for LSHRR (Proposition 5). However, as LSHRR's privacy guarantee depends on the angular distance d_θ and LapLSH's depends on d_{euc}, they cannot be compared directly. For comparison using the same metric, we use the relationship between the Euclidean and angular distances for normalized vectors $\boldsymbol{x}, \boldsymbol{x}'$:

$$d_{\text{euc}}(\boldsymbol{x}, \boldsymbol{x}') \;=\; \sqrt{2 - 2\cos(\pi \cdot d_\theta(\boldsymbol{x}, \boldsymbol{x}'))}\;. \tag{4}$$

We normalized input vectors to length 1 (noting that the normalization does not affect the angular distance, hence utility), and transformed $\varepsilon\, d_{\text{euc}}(\boldsymbol{x}, \boldsymbol{x}')$ into

$\xi_\alpha(\boldsymbol{x}, \boldsymbol{x}')$ using (4) (Since $\xi_\alpha(\boldsymbol{x}, \boldsymbol{x}')$ depends on α and $d_\theta(\boldsymbol{x}, \boldsymbol{x}')$, we perform comparisons against various reasonable ranges of these variables).

We note that the trade-off between privacy and utility means that users with similar profiles will be indistinguishable from each other, whereas users with very different profiles can be distinguished. This is an inherent trade-off determined by the correlation between the sensitive and useful information to be released.

7.3 Experimental Results

We compared the performance of LapLSH and LSHRR with that of vanilla LSH in Fig. 2. We observe that LSHRR outperforms LapLSH when the dimension of the input vector is $n = 100$, 500, or 1000. This is because LapLSH needs to add noise for each element of the input vector (even if the vector is sparse and includes many zero elements) and the total amount of noise is very large in high-dimensional data. In contrast, when the vector length is $n = 50$, LapLSH ($\kappa = 50$ bits) outperforms LSHRR ($\kappa = 50$ bits). We conjecture that this is because the total amount of noise used in LapLSH is small for low-dimensional data whereas LSHRR needs to add a large amount of noise for each element of the hash when the hash length κ is large. We expect LapLSH performance to improve further over LSHRR for smaller values of n.

Interestingly, we observe that although the performance of LSH degrades as the hash length κ decreases, the performance of LSHRR and LapLSH both remain relatively stable. This is mainly because when κ is 5 times larger, the amount of information expressed by the hash can be roughly 5 times larger whereas the amount of noise added to each bit is also 5 times larger. When the privacy budget is $\xi = 20$, the performance of LSHRR on larger bit-lengths ($\kappa = 20$ or 50) overtakes the performance of 10-bit LSHRR. This is because the utility loss of LSHRR is bounded below by the utility loss of the corresponding LSH; i.e., LSHRR converges to LSH with the same hash length κ as ξ increases.

Fig. 2 also shows that when the total privacy budget ξ is around 2, LSHRR achieves lower utility loss than a uniformly random hash, i.e., LSHRR when the total privacy budget is 0. LSHRR achieves much lower utility loss when the total privacy budget is around 5. We can interpret the value of the total privacy budget in terms of the flip probability in the RR. For example, when we use the 20-bit hash, the total privacy budget of 5 for $d_\theta = 0.05$ corresponds to the case in which the RR flips each bit of the hash with the probability approx. 0.27. Therefore, we flip around 5-bits on average out of 20-bits in this case.

We also note that the total privacy budget used in our experiments is much smaller than the privacy budget ϵ previously used in the low privacy regime [31]. Specifically, Kairouz *et al.* [31], and subsequent works (e.g., [1, 43, 50, 55]) refer to $\epsilon = \ln |\mathcal{X}|$ as a privacy budget in the low privacy regime. Since our experiments deal with high-dimensional data, a privacy budget of $\ln |\mathcal{X}|$ would be extremely large. For example, when we use the 1000-dimensional rating vector in the MovieLens dataset, the privacy budget in the low privacy regime is: $\epsilon = \ln |\mathcal{X}| = \ln 5^{1000} = 1609$. The total privacy budget in our experiments ($\xi \leq 20$) is much smaller than this value, and falls into the *medium privacy regime* [1, 55].

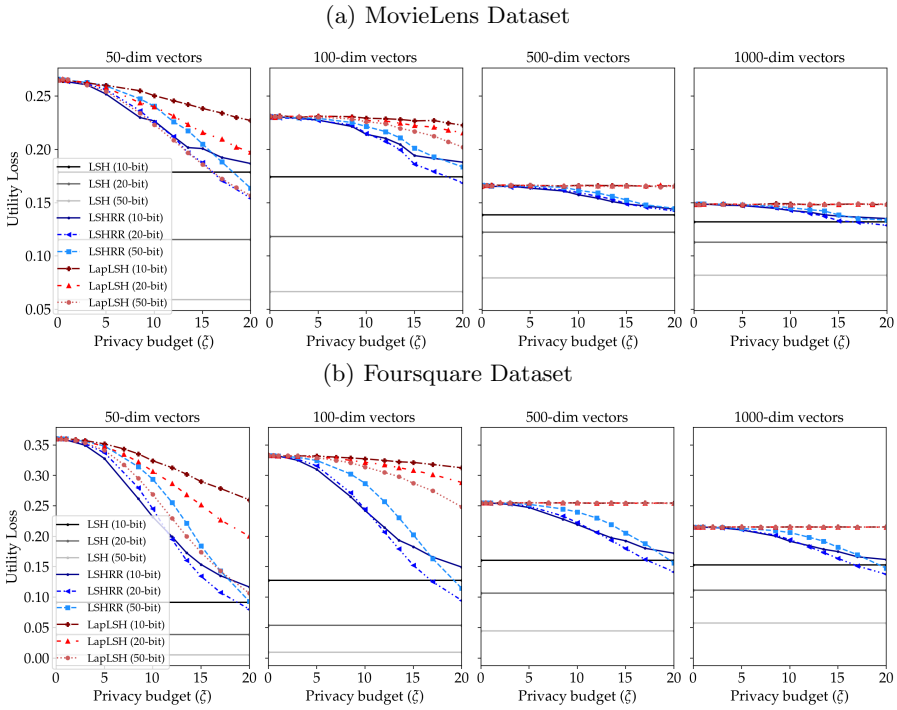

Fig. 2: Utility loss (y-axis) versus privacy budget ξ (x-axis) for LSHRR, LapLSH and LSH on n-dimensional vectors. ξ is computed for various κ, and $d_\theta = 0.1$.

Note that LDP requires a much larger privacy budget than extended DP. For example, by Proposition 5, when $\kappa = 50$ and $d_\theta = 0.05$ (resp. 0.1), the total privacy budget $\xi = 20$ in extended DP corresponds to the total privacy budget 120 (resp. 80) in LDP. More details are shown in Appendix A.

Finally, we compare LSHRR with LapLSH in terms of time complexity and general applicability. For time complexity, LapLSH requires $O(n\kappa)$ operations (construction of n-dimensional noise, then κ-bit hashing). In contrast, LSHRR requires $O(m\kappa)$ operations, where m is the number of non-zero elements in the input vector (κ-bit hashing on m non-zero elements followed by κ-randomized response). Since $m \ll n$ in practice, LSHRR is significantly more efficient.

For general applicability, LSHRR can be used with other metrics such as the Jaccard metric [11], Earth Mover's metric [12], and l_p metric [18] by choosing a suitable LSH function, whereas LapLSH is designed for the Euclidean metric only. Thus, LSHRR has more potential applications than LapLSH.

In summary, we find that LSHRR is better than LapLSH in terms of both time complexity and general applicability, and provides high utility with a reasonable privacy level for a high-dimensional data (100 dimensions or more).

7.4 Inapplicability of the RAPPOR

We finally explain that neither the RAPPOR [23] nor the generalized RAPPOR [51] can be used for friend matching based on high-dimensional personal data. These mechanisms apply a Bloom filter to an input vector before applying the randomized response. Typically, this Bloom filter is a hash function that neither allows for efficiently finding an input from its hash value, nor preserves the metric $d_{\mathcal{X}}$ over the inputs. For instance, [23] uses MD5 to implement the Bloom filter.

Let us consider two approaches to perform the nearest neighbor search using RAPPOR: *comparing two hash values* and *comparing two input vectors*.

In the first approach, the data collector calculates the Hamming distance between obfuscated hash values. Then the utility is completely lost, because the Bloom filter does not preserve the metric $d_{\mathcal{X}}$ over the inputs. Hence we cannot recommend friends based on the proximity of input vector in this approach.

In the second approach, the data collector tries to invert obfuscated hash values to the original input vector, and calculates the angular distance between the input vectors to find nearest neighbors. Since the Bloom filter may not allow for efficiently finding an input from its hash value, the data collectors need to perform exhaustive searches, i.e., to compute the hash values of all possible input data $\mathcal{X}$. However, this is computationally intractable when the input domain $\mathcal{X}$ is very large. In particular, our setting deals with high-dimensional input data (e.g., $|\mathcal{X}| = 5^{1000}$ in the 1000-dimensional MovieLens rating vector), and thus it is computationally infeasible to invert hash values into input vectors.

In summary, the first approach (comparing two hashes) results in a complete loss of utility, and the second approach (comparing two input vectors) is computationally infeasible when the input data are in a high-dimensional space. Therefore, the RAPPOR cannot be applied to our problem of friend matching. The same issue applies to a generalized version of the RAPPOR [51].

In contrast, our mechanisms can be applied to friend matching even when $|\mathcal{X}|$ is very large, because LSH allows us to approximately compare the distance between the input vectors without computing them from their hash values.

8 Conclusion

In this paper, we proposed two extended DP mechanisms LSHRR and LapLSH. We showed that LSH itself does not provide privacy guarantees and could result in complete privacy collapse in some situations. We then proved that LSHRR and LapLSH provide rigorous guarantees of extended DP. To our knowledge, this work is the first to provide extended DP with the angular distance.

By experiments with real datasets, we show that LSHRR outperforms LapLSH on high-dimensional data. We also show that LSHRR provides high utility for a high-dimensional vector, thus enabling friend matching with rigorous privacy guarantees and high utility.

References

1. Acharya, J., Sun, Z., Zhang, H.: Hadamard response: Estimating distributions privately, efficiently, and with little communication. In: AISTATS. pp. 1120–1129 (2019)
2. Aggarwal, C.C.: Recommender Systems. Springer (2016)
3. Aghasaryan, A., Bouzid, M., Kostadinov, D., Kothari, M., Nandi, A.: On the use of LSH for privacy preserving personalization. In: TrustCom. pp. 362–371 (2013)
4. Alvim, M.S., Chatzikokolakis, K., Palamidessi, C., Pazii, A.: Invited paper: Local differential privacy on metric spaces: Optimizing the trade-off with utility. In: CSF. pp. 262–267 (2018). https://doi.org/10.1109/CSF.2018.00026
5. Andoni, A., Indyk, P., Laarhoven, T., Razenshteyn, I., Schmidt, L.: Practical and optimal LSH for angular distance. In: NIPS. pp. 1–9 (2015)
6. Andoni, A., Indyk, P., Razenshteyn, I.: Approximate nearest neighbor search in high dimensions. In: ICM. pp. 3287–3318. World Scientific (2018)
7. Andrés, M.E., Bordenabe, N.E., Chatzikokolakis, K., Palamidessi, C.: Geo-indistinguishability: differential privacy for location-based systems. In: CCS. pp. 901–914. ACM (2013). https://doi.org/10.1145/2508859.2516735
8. Aumüller, M., Bourgeat, A., Schmurr, J.: Differentially private sketches for Jaccard similarity estimation. CoRR **abs/2008.08134** (2020)
9. Bordenabe, N.E., Chatzikokolakis, K., Palamidessi, C.: Optimal geo-indistinguishable mechanisms for location privacy. In: CCS. pp. 251–262 (2014)
10. Brendel, W., Han, F., Marujo, L., Jie, L., Korolova, A.: Practical privacy-preserving friend recommendations on social networks. In: WWW. pp. 111–112 (2018)
11. Broder, A.Z., Charikar, M., Frieze, A.M., Mitzenmacher, M.: Min-wise independent permutations. Journal of Computer and System Sciences **60**, 630–659 (2000)
12. Charikar, M.S.: Similarity estimation techniques from rounding algorithms. In: STOC. pp. 380–388 (2002)
13. Chatzikokolakis, K., Andrés, M.E., Bordenabe, N.E., Palamidessi, C.: Broadening the scope of Differential Privacy using metrics. In: PETS. pp. 82–102 (2013)
14. Chen, L., Zhu, P.: Preserving the privacy of social recommendation with a differentially private approach. In: SmartCity. pp. 780–785. IEEE (2015)
15. Chen, X., Liu, H., Yang, D.: Improved LSH for privacy-aware and robust recommender system with sparse data in edge environment. EURASIP Journal on Wireless Communications and Networking **171**, 1–11 (2019)
16. Cheng, H., Qian, M., Li, Q., Zhou, Y., Chen, T.: An efficient privacy-preserving friend recommendation scheme for social network. IEEE Access **6**, 56018–56028 (2018)
17. Chow, R., Pathak, M.A., Wang, C.: A practical system for privacy-preserving collaborative filtering. In: ICDM Workshops. pp. 547–554 (2012)
18. Datar, M., Immorlica, N., Indyk, P., Mirrokni, V.S.: Locality-sensitive hashing scheme based on p-stable distributions. In: SCG. pp. 253–262 (2004)
19. Duchi, J.C., Jordan, M.I., Wainwright, M.J.: Local privacy and statistical minimax rates. In: FOCS. pp. 429–438 (2013)
20. Dwork, C.: Differential privacy. In: ICALP. pp. 1–12 (2006)
21. Dwork, C., Mcsherry, F., Nissim, K., Smith, A.: Calibrating noise to sensitivity in private data analysis. In: TCC. pp. 265–284 (2006)
22. Dwork, C., Rothblum, G.N.: Concentrated differential privacy. CoRR **abs/1603.01887** (2016)

23. Úlfar Erlingsson, Pihur, V., Korolova, A.: RAPPOR: Randomized aggregatable privacy-preserving ordinal response. In: CCS. pp. 1054–1067 (2014)
24. Fernandes, N., Dras, M., McIver, A.: Processing text for privacy: an information flow perspective. In: FM. pp. 3–21 (2018)
25. Fernandes, N., Dras, M., McIver, A.: Generalised differential privacy for text document processing. In: POST. pp. 123–148 (2019)
26. Fernandes, N., Kawamoto, Y., Murakami, T.: Locality sensitive hashing with extended differential privacy. CoRR **abs/2010.09393** (2020), https://arxiv.org/abs/2010.09393
27. Ganta, S.R., Kasiviswanathan, S.P., Smith, A.: Composition attacks and auxiliary information in data privacy. In: KDD. pp. 265–273. ACM (2008)
28. Gionis, A., Indyk, P., Motwani, R.: Similarity search in high dimensions via hashing. In: VLDB. pp. 518–529 (1999)
29. Hu, H., Dobbie, G., Salcic, Z., Liu, M., Zhang, J., Lyu, L., Zhang, X.: Differentially private locality sensitive hashing based federated recommender system. Concurrency and Computation Practice and Experience pp. 1–16 (2020)
30. Indyk, P., Motwani, R.: Approximate nearest neighbors: towards removing the curse of dimensionality. In: STOC. pp. 604–613 (1998)
31. Kairouz, P., Bonawitz, K., Ramage, D.: Discrete distribution estimation under local privacy. In: ICML. pp. 2436–2444 (2016)
32. Kamalaruban, P., Perrier, V., Asghar, H.J., Kaafar, M.A.: Not all attributes are created equal: d_x-private mechanisms for linear queries. Proceedings on Privacy Enhancing Technologies (PoPETs) **2020**(1), 103–125 (2020)
33. Kawamoto, Y., Murakami, T.: On the anonymization of differentially private location obfuscation. In: ISITA. pp. 159–163. IEEE (2018)
34. Kawamoto, Y., Murakami, T.: Local obfuscation mechanisms for hiding probability distributions. In: ESORICS. pp. 128–148 (2019)
35. Li, M., Ruan, N., Qian, Q., Zhu, H., Liang, X., Yu, L.: SPFM: Scalable and privacy-preserving friend matching in mobile clouds. IEEE Internet of Things Journal **4**(2), 583–591 (2017)
36. Liu, C., Mittal, P.: LinkMirage: Enabling privacy-preserving analytics on social relationships. In: NDSS (2016)
37. Liu, Z., Wang, Y.X., Smola, A.J.: Fast differentially private matrix factorization. In: RecSys. pp. 171–178 (2015)
38. Ma, X., Ma, J., Li, H., Jiang, Q., Gao, S.: ARMOR: A trust-based privacy-preserving framework for decentralized friend recommendation in online social networks. Future Generation Computer Systems **79**, 82–94 (2018)
39. Machanavajjhala, A., Kifer, D., Abowd, J.M., Gehrke, J., Vilhuber, L.: Privacy: Theory meets practice on the map. In: ICDE. pp. 277–286. IEEE (2008)
40. Machanavajjhala, A., Korolova, A., Sarma, A.D.: Personalized social recommendations - accurate or private? VLDB **4**(7), 440–450 (2020)
41. MovieLens 25m Dataset: https://grouplens.org/datasets/movielens/25m/ (accessed in 2020)
42. Murakami, T., Hamada, K., Kawamoto, Y., Hatano, T.: Privacy-preserving multiple tensor factorization for synthesizing large-scale location traces with cluster-specific features. Proc. Priv. Enhancing Technol. **2021**(2), 5–26 (2021)
43. Murakami, T., Kawamoto, Y.: Utility-optimized local differential privacy mechanisms for distribution estimation. In: USENIX Security. pp. 1877–1894 (2019)
44. Narayanan, A., Thiagarajan, N., Lakhani, M., Hamburg, M., Boneh, D., et al.: Location privacy via private proximity testing. In: NDSS. vol. 11 (2011)

45. Nissim, K., Stemmer, U.: Clustering algorithms for the centralized and local models. In: Algorithmic Learning Theory. pp. 619–653 (2019)
46. Qi, L., Zhang, X., Dou, W., Ni, Q.: A distributed locality-sensitive hashing-based approach for cloud service recommendation from multi-source data. IEEE Journal on Selected Areas in Communications **35**(11), 2616–2624 (2017)
47. Samanthula, B.K., Cen, L., Jiang, W., Si, L.: Privacy-preserving and efficient friend recommendation in online social networks. Trans. Data Privacy **8**(2), 141–171 (2015)
48. Shin, H., Kim, S., Shin, J., Xiao, X.: Privacy enhanced matrix factorization for recommendation with local differential privacy. IEEE Trans. on Knowledge and Data Engineering **30**(9), 1770–1782 (2018)
49. Wang, J., Liu, W., Kumar, S., Chang, S.F.: Learning to hash for indexing big data – a survey. Proceedings of the IEEE **104**(1), 34–57 (2016)
50. Wang, S., Huang, L., Wang, P., Nie, Y., Xu, H., Yang, W., Li, X.Y., Qiao, C.: Mutual information optimally local private discrete distribution estimation. CoRR **abs/1607.08025** (2016), https://arxiv.org/abs/1607.08025
51. Wang, T., Blocki, J., Li, N., Jha, S.: Locally differentially private protocols for frequency estimation. In: USENIX Security. pp. 729–745 (2017)
52. Warner, S.L.: Randomized response: A survey technique for eliminating evasive answer bias. Journal of the American Statistical Association **60**(309), 63–69 (1965)
53. Xiang, Z., Ding, B., He, X., Zhou, J.: Linear and range counting under metric-based local differential privacy. In: ISIT. pp. 908–913 (2020)
54. Yang, D., Qu, B., Yang, J., Cudre-Mauroux, P.: Revisiting user mobility and social relationships in LBSNs: A hypergraph embedding approach. In: WWW. pp. 2147–2157 (2019)
55. Ye, M., Barga, A.: Optimal schemes for discrete distribution estimation under local differential privacy. In: ISIT. pp. 759–763 (2017)
56. Zhang, Y., Gao, N., Chen, J., Tu, C., Wang, J.: PrivRec: User-centric differentially private collaborative filtering using LSH and KD. In: ICONIP. pp. 113–121 (2020)

A Total Privacy Budgets in Extended DP and LDP

Table 1 shows total privacy budgets in extended DP and LDP calculated from Proposition 5 and the fact that the angular distance is 0.5 or smaller.

For example, when $d_\theta = 0.05$ and $\kappa = 10$, 20, and 50, the total privacy budget $\xi = 20$ in extended DP corresponds to the total privacy budget of 55, 79, and 120, respectively, in LDP.

Table 1: Total privacy budgets in extended DP (XDP) and LDP when $d_\theta = 0.05$ or 0.1, $\kappa = 10$, 20, or 50, and $\delta = 0.01$.

(a) $d_\theta = 0.05$

Total privacy budget ξ in XDP	1	5	10	20
Total privacy budget in LDP ($\kappa = 10/20/50$)	3/4/6	14/20/30	28/40/60	55/79/120

(b) $d_\theta = 0.1$

Total privacy budget ξ in XDP	1	5	10	20
Total privacy budget in LDP ($\kappa = 10/20/50$)	2/3/4	10/14/20	21/28/40	42/57/80

B Proofs for the Technical Results

We first recall Chernoff bound, which is used in the proof for Lemma 4.

Lemma 1 (Chernoff bound) *Let Z be a real-valued random variable. Then for all $t \in \mathbb{R}$,*

$$\Pr[Z \geq t] \leq \min_{s \in \mathbb{R}} \frac{\mathbb{E}[\exp(sZ)]}{\exp(st)}.$$

Next we recall Hoeffding's lemma, which is used in the proof for Proposition 4.

Lemma 2 (Hoeffding) *Let $a, b \in \mathbb{R}$, and Z be a real-valued random variable such that $\mathbb{E}[Z] = \mu$ and that $a \leq Z \leq b$. Then for all $t \in \mathbb{R}$,*

$$\mathbb{E}[\exp(tZ)] \leq \exp\big(t\mu + \tfrac{t^2}{8}\big(b-a\big)^2\big).$$

Note that Lemma 2 implies that $\mathbb{E}[\exp(t(Z - \mathbb{E}[Z]))] \leq \exp\big(\frac{t^2}{8}\big(b-a\big)^2\big)$.

Then we recall Chernoff-Hoeffding Theorem, which is used in the proof for Theorem 1. Recall that the Kullback-Leibler divergence $D_{\mathrm{KL}}(a\|b)$ between Bernoulli distributed random variables with parameters a and b is defined by:

$$D_{\mathrm{KL}}(a\|b) = a \ln \tfrac{a}{b} + (1-a) \ln \tfrac{1-a}{1-b}.$$

Lemma 3 (Chernoff-Hoeffding) *Let $Z \sim Binomial(k, p)$ be a binomially distributed random variable where k is the total number of experiments and p is the probability that an experiment yields a successful outcome. Then for any $\alpha \in \mathbb{R}_{>0}$,*

$$\Pr[Z \geq k(p+\alpha)] \leq \exp\big(-kD_{\mathrm{KL}}(p+\alpha\|p)\big).$$

By relaxing this, we have a simpler bound:

$$\Pr[Z \geq k(p+\alpha)] \leq \exp\big(-2k\alpha^2\big).$$

We show the proofs for technical results as follows.

Proposition 1 (Error bound) *For any $x, x' \in \mathcal{X}$, the expected error in the Hamming distance satisfies $\mathbb{E}[|d_{\mathcal{V}}(Q_H(x), Q_H(x')) - d_{\mathcal{V}}(H(x), H(x'))|] \leq \frac{2\kappa}{1+e^{\varepsilon}}$ where the expectation is taken over the randomness in the bitwise RR.*

Proof. By the triangle inequality and $Q_H = Q_{\mathsf{brr}} \circ H$, we have:

$$d_{\mathcal{V}}(Q_H(x), Q_H(x')) \leq d_{\mathcal{V}}(Q_{\mathsf{brr}} \circ H(x), H(x)) + d_{\mathcal{V}}(H(x), H(x')) + d_{\mathcal{V}}(H(x'), Q_{\mathsf{brr}} \circ H(x')).$$

It follows from the definition of the bitwise RR Q_{rr} that for any κ-bit string $v \in \mathcal{V}$, the expected Hamming distance is $\mathbb{E}[d_{\mathcal{V}}(v, Q_{\mathsf{brr}}(v))] = \frac{\kappa}{1+e^{\varepsilon}}$. Thus $\mathbb{E}[d_{\mathcal{V}}(Q_{\mathsf{brr}} \circ H(x), H(x)) + d_{\mathcal{V}}(H(x'), Q_{\mathsf{brr}} \circ H(x'))] = \frac{2\kappa}{1+e^{\varepsilon}}$. Hence we obtain the proposition. □

We present LSHRR's privacy guarantee for hash values, which relies on the XDP of the bitwise RR Q_{brr} w.r.t. the Hamming distance $d_{\mathcal{V}}$ as follows.

Proposition 7 (XDP of BRR) *The (ε, κ)-bitwise RR provides $(\epsilon d_{\mathcal{V}}, 0)$-XDP.*

Proof. Recall the definition of the ε-RR Q_{rr} in Definition 5. Let $r = \frac{1}{e^{\varepsilon}+1}$, $\boldsymbol{v} = (v_1, v_2, \ldots, v_\kappa) \in \mathcal{V}$, $\boldsymbol{v}' = (v'_1, v'_2, \ldots, v'_\kappa) \in \mathcal{V}$, and $\boldsymbol{y} = (y_1, y_2, \ldots, y_\kappa) \in \mathcal{V}$. By definition we obtain:

$$Q_{\mathsf{brr}}(\boldsymbol{v})[\boldsymbol{y}] = \textstyle\prod_{i=1}^{\kappa} r^{|y_i - v_i|}(1-r)^{1-|y_i-v_i|}$$
$$Q_{\mathsf{brr}}(\boldsymbol{v}')[\boldsymbol{y}] = \textstyle\prod_{i=1}^{\kappa} r^{|y_i - v'_i|}(1-r)^{1-|y_i-v'_i|}.$$

By $Q_{\mathsf{brr}}(\boldsymbol{v}')[\boldsymbol{y}] > 0$ and the triangle inequality, we have:

$$\ln \tfrac{Q_{\mathsf{brr}}(\boldsymbol{v})[\boldsymbol{y}]}{Q_{\mathsf{brr}}(\boldsymbol{v}')[\boldsymbol{y}]} \le \ln \prod_{i=1}^{\kappa} \big(\tfrac{1-r}{r}\big)^{|v_i - v'_i|} = \ln \big(\tfrac{1-r}{r}\big)^{d_{\mathcal{V}}(\boldsymbol{v},\boldsymbol{v}')} = \varepsilon d_{\mathcal{V}}(\boldsymbol{v}, \boldsymbol{v}').$$

Therefore Q_{brr} provides $(\epsilon d_{\mathcal{V}}, 0)$-XDP. □

Proposition 2 (XDP of Q_H) *Let $H : \mathcal{X} \to \mathcal{V}$ be a κ-bit LSH function, and $d_{\varepsilon H}$ be the pseudometric over $\mathcal{X}$ defined by $d_{\varepsilon H}(\boldsymbol{x}, \boldsymbol{x}') = \varepsilon d_{\mathcal{V}}(H(\boldsymbol{x}), H(\boldsymbol{x}'))$ for $\boldsymbol{x}, \boldsymbol{x}' \in \mathcal{X}$. Then the ε-LSHRR Q_H instantiated with H provides $(d_{\varepsilon H}, 0)$-XDP.*

Proof. Let $\boldsymbol{x}, \boldsymbol{x}' \in \mathcal{X}$ and $\boldsymbol{y} \in \mathcal{V}$.

$$\begin{aligned} Q_H(\boldsymbol{x})[\boldsymbol{y}] &= Q_{\mathsf{brr}}(H(\boldsymbol{x}))[\boldsymbol{y}] \\ &\le \varepsilon d_{\mathcal{V}}(H(\boldsymbol{x}), H(\boldsymbol{x}')) Q_{\mathsf{brr}}(H(\boldsymbol{x}'))[\boldsymbol{y}] && \text{(by Proposition 7)} \\ &= \varepsilon d_H(\boldsymbol{x}, \boldsymbol{x}') Q_H(\boldsymbol{x}')[\boldsymbol{y}] && \text{(by the def. of } d_{\varepsilon H}) \end{aligned}$$

Hence Q_H provides $(d_{\varepsilon H}, 0)$-XDP. □

Proposition 3 (Worst-case privacy of Q_H) *For a κ-bit LSH function H, the ε-LSHRR Q_H instantiated with H provides $\kappa\epsilon$-DP.*

Proof. Since $d_H(\boldsymbol{x}, \boldsymbol{x}') \le \kappa$ holds for all $\boldsymbol{x}, \boldsymbol{x}'$, this proposition follows from Proposition 2.

Lemma 4 (CXDP ⇒ PXDP) *Let $\mu \in \mathbb{R}_{\ge 0}$, $\tau \in \mathbb{R}_{>0}$, $\lambda \in \mathbb{D}\mathcal{R}$, $A_\lambda : \mathcal{X} \to \mathbb{D}\mathcal{Y}$, and d be a metric over $\mathcal{X}$. Let $\delta \in (0, 1]$, $\varepsilon = \tau\sqrt{-2\ln\delta}$, and $\xi(x, x') = \mu d(x, x') + \varepsilon$. If A_λ provides (μ, τ, d)-CXDP, then it provides (ξ, δ)-PXDP.*

Proof. Assume that A_λ provides (μ, τ, d)-CXDP. Let $x, x' \in \mathcal{X}$. Then we will show $\Pr[\,\mathcal{L}_{x,x'} > \mu d(x, x') + \varepsilon\,] \le \delta$ as follows.

Let $Z = \mathcal{L}_{x,x'} - \mathbb{E}[\mathcal{L}_{x,x'}]$. By the definition of CXDP, we have:

$$\mathbb{E}[\mathcal{L}_{x,x'}] \le \mu d(x, x'), \tag{5}$$

and Z is τ-subgaussian. Let $t = \tau\sqrt{-2\ln\delta}$ and $a = \tau^2$. By the definition of subgaussian variables, $\mathbb{E}[\exp(sZ)] \leq \exp(\frac{as^2}{2})$ holds for any $s \in \mathbb{R}$. Thus we obtain:

$$\begin{aligned}
\Pr[Z \geq t] &\leq \min_{s\in\mathbb{R}} \frac{\mathbb{E}[\exp(sZ)]}{\exp(st)} \\
&\qquad\qquad \text{(by the Chernoff bound in Lemma 1)} \\
&\leq \min_{s\in\mathbb{R}} \exp\big(\tfrac{as^2}{2} - st\big) \qquad \big(\text{by } \mathbb{E}[\exp(sZ)] \leq \exp(\tfrac{as^2}{2})\big) \\
&= \min_{s\in\mathbb{R}} \exp\big(\tfrac{a}{2}\big(s - \tfrac{t}{a}\big)^2 - \tfrac{t^2}{2a}\big) \\
&= \exp\big(-\tfrac{t^2}{2a}\big) \qquad \big(\text{when } s = \tfrac{t}{a}\big) \qquad (6)
\end{aligned}$$

Recall that $\varepsilon = \tau\sqrt{-2\ln\delta}$ by definition. We obtain:

$$\begin{aligned}
&\Pr[\mathcal{L}_{x,x'} > \mu d(x,x') + \varepsilon] \\
&\leq \Pr[\mathcal{L}_{x,x'} > \mathbb{E}[\mathcal{L}_{x,x'}] + \tau\sqrt{-2\ln\delta}] && \text{(by (5) and the def. of } \varepsilon) \\
&= \Pr[Z > \tau\sqrt{-2\ln\delta}] && \text{(by the def. of } Z) \\
&\leq \exp\big(-\tfrac{(\tau\sqrt{-2\ln\delta})^2}{2a}\big) && \text{(by (6) and } t = \tau\sqrt{-2\ln\delta}) \\
&= \delta && \text{(by } a = \tau^2)
\end{aligned}$$

Therefore the randomized algorithm A_λ provides (ξ,δ)-PXDP. □

Lemma 5 (PXDP ⇒ XDP) *Let $\lambda \in \mathbb{D}\mathcal{R}$, $A_\lambda : \mathcal{X} \to \mathbb{D}\mathcal{Y}$, $\xi : \mathcal{X}\times\mathcal{X} \to \mathbb{R}_{\geq 0}$, and $\delta : \mathcal{X}\times\mathcal{X} \to [0,1]$. If A_λ provides (ξ,δ)-PXDP, it provides (ξ,δ)-XDP.*

Proof. Assume that A_λ provides (ξ,δ)-PXDP. Let $x, x' \in \mathcal{X}$. By the definition of (ξ,δ)-PXDP, we have $\Pr[\,\mathcal{L}_{x,x'} > \xi(x,x')\,] \leq \delta(x,x')$. Let $S \subseteq \mathcal{Y}$. For each $r \in \mathcal{R}$, let $S'_r = \{y \in S \mid \mathcal{L}_{x,x',y,r} > \xi(x,x')\}$. Then $\sum_r \lambda[r]A_r(x)[S'_r] \leq \delta(x,x')$ and for each $r \in \mathcal{R}$,

$$A_r(x)[S \setminus S'_r] \leq \exp(\xi(x,x')) \cdot A_r(x')[S \setminus S'_r].$$

Hence:

$$\begin{aligned}
A_\lambda(x)[S] &= \textstyle\sum_r \lambda[r]A_r(x)[S] \\
&= \textstyle\sum_r \lambda[r]A_r(x)[S \setminus S'_r] + \sum_r \lambda[r]A_r(x)[S'_r] \\
&\leq \Big(\textstyle\sum_r \lambda[r]\exp(\xi(x,x')) \cdot A_r(x')[S \setminus S'_r]\Big) + \delta(x,x') \\
&\leq \exp(\xi(x,x')) \cdot \Big(\textstyle\sum_r \lambda[r]A_r(x')[S]\Big) + \delta(x,x') \\
&\leq \exp(\xi(x,x')) \cdot A_\lambda(x')[S] + \delta(x,x').
\end{aligned}$$

Therefore A_λ provides (ξ,δ)-XDP. □

To prove the CXDP of LSHRR, we show that the Hamming distance between hash values follows a binomial distribution.

Lemma 6 (Distribution of the Hamming distance of LSH) *Let $\mathcal{H}$ be an LSH scheme w.r.t. a metric $d_{\mathcal{X}}$ over $\mathcal{X}$ coupled with a distribution $D_{\mathcal{H}}$. Let $\boldsymbol{x}, \boldsymbol{x}' \in \mathcal{X}$ be any two inputs, and Z be the random variable of the Hamming distance between their κ-bit hash values, i.e., $Z = d_{\mathcal{V}}(H(\boldsymbol{x}), H(\boldsymbol{x}'))$ where a κ-bit LSH function H is drawn from the distribution $D_{\mathcal{H}}^{\kappa}$. Then Z follows the binomial distribution with mean $\kappa d_{\mathcal{X}}(\boldsymbol{x}, \boldsymbol{x}')$ and variance $\kappa d_{\mathcal{X}}(\boldsymbol{x}, \boldsymbol{x}')(1 - d_{\mathcal{X}}(\boldsymbol{x}, \boldsymbol{x}'))$.*

Proof. By the definition of the Hamming distance $d_{\mathcal{V}}$ and the construction of the LSH-based κ-bit function H, we have $d_{\mathcal{V}}(H(\boldsymbol{x}), H(\boldsymbol{x}')) = \sum_{i=1}^{\kappa} | h_i(\boldsymbol{x}) - h_i(\boldsymbol{x}') |$. Since $\sum_{i=1}^{\kappa} | h_i(\boldsymbol{x}) - h_i(\boldsymbol{x}') |$ represents the number of non-collisions between hash values of $\boldsymbol{x}$ and $\boldsymbol{x}'$, it follows the binomial distribution with mean $\kappa d_{\mathcal{X}}(\boldsymbol{x}, \boldsymbol{x}')$ and variance $\kappa d_{\mathcal{X}}(\boldsymbol{x}, \boldsymbol{x}')(1 - d_{\mathcal{X}}(\boldsymbol{x}, \boldsymbol{x}'))$. □

Proposition 4 (CXDP of Q_{LSHRR}) *The ε-LSHRR provides $(\varepsilon\kappa, \frac{\varepsilon\kappa}{2}, d_{\mathcal{X}})$-CXDP.*

Proof. For a κ-bit LSH function $H \in \mathcal{H}^{\kappa}$,

$$\begin{aligned} Q_H(\boldsymbol{x})[y] &= Q_{\mathsf{brr}}(H(\boldsymbol{x}))[y] \\ &\leq e^{\varepsilon d_{\mathcal{V}}(H(\boldsymbol{x}), H(\boldsymbol{x}'))} Q_{\mathsf{brr}}(H(\boldsymbol{x}'))[y] \quad \text{(by Proposition 7)} \\ &= e^{\varepsilon d_{\mathcal{V}}(H(\boldsymbol{x}), H(\boldsymbol{x}'))} Q_H(\boldsymbol{x}')[y]. \end{aligned}$$

Let Z be the random variable defined by $Z \stackrel{\text{def}}{=} d_{\mathcal{V}}(H(\boldsymbol{x}), H(\boldsymbol{x}'))$ where $H = (h_1, h_2, \ldots, h_{\kappa})$ is distributed over $\mathcal{H}^{\kappa}$, namely, the seeds of these LSH functions are chosen randomly. Then $0 \leq Z \leq \kappa$. By Lemma 6, Z follows the binomial distribution with mean $\mathbb{E}[Z] = \kappa d_{\mathcal{X}}(\boldsymbol{x}, \boldsymbol{x}')$. Then the random variable $\varepsilon Z - \mathbb{E}[\varepsilon Z]$ is centered, i.e., $\mathbb{E}[\varepsilon Z - \mathbb{E}[\varepsilon Z]] = 0$, and ranges over $[-\varepsilon\kappa d_{\mathcal{X}}(\boldsymbol{x}, \boldsymbol{x}'), \varepsilon\kappa(1 - d_{\mathcal{X}}(\boldsymbol{x}, \boldsymbol{x}'))]$. Hence it follows from Hoeffding's lemma (Lemma 2) that:

$$\mathbb{E}[\exp(t(\varepsilon Z - \mathbb{E}[\varepsilon Z]))] \leq \exp\Big(\frac{t^2}{8}\big(\varepsilon\kappa\big)^2\Big) = \exp\Big(\frac{t^2}{2}\big(\frac{\varepsilon\kappa}{2}\big)^2\Big).$$

Hence by definition, $\varepsilon Z - \mathbb{E}[\varepsilon Z]$ is $\frac{\varepsilon\kappa}{2}$-subgaussian. Therefore, the LSH-based mechanism Q_{LSHRR} provides $(\varepsilon\kappa, \frac{\varepsilon\kappa}{2}, d_{\mathcal{X}})$-CXDP. □

Theorem 1 (PXDP/XDP of Q_{LSHRR}) *Let $\delta \in \mathbb{R}_{>0}$, $\varepsilon' = \varepsilon\sqrt{\frac{-\ln\delta}{2}}$, and $\xi(\boldsymbol{x}, \boldsymbol{x}') = \varepsilon\kappa d_{\mathcal{X}}(\boldsymbol{x}, \boldsymbol{x}') + \varepsilon'\sqrt{\kappa}$. The ε-LSHRR provides (ξ, δ)-PXDP, hence (ξ, δ)-XDP.*

Proof. Let $\alpha = \sqrt{\frac{-\ln\delta}{2\kappa}}$. Let Z be the random variable defined by $Z \stackrel{\text{def}}{=} d_{\mathcal{V}}(H(\boldsymbol{x}), H(\boldsymbol{x}'))$ where $H = (h_1, h_2, \ldots, h_{\kappa})$ is distributed over $\mathcal{H}^{\kappa}$. By Lemma 6, Z follows the binomial distribution with mean $\mathbb{E}[Z] = \kappa d_{\mathcal{X}}(\boldsymbol{x}, \boldsymbol{x}')$. Hence it follows from Chernoff-Hoeffding theorem (Lemma 3) that:

$$\Pr[Z \geq \kappa(d_{\mathcal{X}}(\boldsymbol{x}, \boldsymbol{x}') + \alpha)] \leq \exp\big(-2\kappa\alpha^2\big) = \delta.$$

Hence $\Pr[\varepsilon Z \geq \varepsilon\kappa d_{\mathcal{X}}(\boldsymbol{x}, \boldsymbol{x}') + \varepsilon'\sqrt{\kappa}] \leq \delta$. Therefore Q_{LSHRR} provides (ξ, δ)-PXDP. By Lemma 5, Q_{LSHRR} provides (ξ, δ)-XDP. □

Proposition 5 (Tighter bound for PXDP/XDP) *For $a, b \in \mathbb{R}_{>0}$, let $D_{\mathrm{KL}}(a\|b) = a \ln \frac{a}{b} + (1-a) \ln \frac{1-a}{1-b}$. For an $\alpha \in \mathbb{R}_{>0}$, we define:*

$$\xi_\alpha(\boldsymbol{x}, \boldsymbol{x}') = \varepsilon\kappa(d_{\mathcal{X}}(\boldsymbol{x}, \boldsymbol{x}') + \alpha)$$
$$\delta_\alpha(\boldsymbol{x}, \boldsymbol{x}') = \exp\big(-\kappa D_{\mathrm{KL}}(d_{\mathcal{X}}(\boldsymbol{x}, \boldsymbol{x}') + \alpha \| d_{\mathcal{X}}(\boldsymbol{x}, \boldsymbol{x}'))\big).$$

Then the ε-LSHRR provides $(\xi_\alpha, \delta_\alpha)$-PXDP, hence $(\xi_\alpha, \delta_\alpha)$-XDP.

Proof. Let Z be the random variable defined by $Z \stackrel{\mathrm{def}}{=} d_{\mathcal{V}}(H(\boldsymbol{x}), H(\boldsymbol{x}'))$ where $H = (h_1, h_2, \ldots, h_\kappa)$ is distributed over $\mathcal{H}^\kappa$. By Chernoff-Hoeffding theorem (Lemma 3),

$$\Pr[Z \geq \kappa(d_{\mathcal{X}}(\boldsymbol{x}, \boldsymbol{x}') + \alpha)] \leq \delta_\alpha(\boldsymbol{x}, \boldsymbol{x}').$$

Then $\Pr[\varepsilon Z \geq \xi_\alpha(\boldsymbol{x}, \boldsymbol{x}')] \leq \delta_\alpha(\boldsymbol{x}, \boldsymbol{x}')$. Therefore Q_{LSHRR} provides $(\xi_\alpha, \delta_\alpha)$-PXDP. By Lemma 5, Q_{LSHRR} provides $(\xi_\alpha, \delta_\alpha)$-XDP. □

Proposition 6 (XDP of $Q_{\mathsf{Lap}H}$ and Q_{LapLSH}) *The $(\varepsilon, d_{\mathcal{X}})$-LapLSH $Q_{\mathsf{Lap}H}$ with a κ-bit LSH function H provides $(\varepsilon d_{\mathcal{X}}, 0)$-XDP. The $(\varepsilon, d_{\mathcal{X}})$-LapLSH Q_{LapLSH} w.r.t. a distribution $D^\kappa_{\mathcal{H}}$ of the κ-bit LSH functions also provides $(\varepsilon d_{\mathcal{X}}, 0)$-XDP.*

Proof. Since the application of an LSH function is post-processing, the proposition follows from the XDP of the Laplace mechanism. □