

A Validated Privacy-Utility Preserving Recommendation System with Local Differential Privacy

Seryne Rahali
*R3S Team/RST Department
 Samovar, Telecom SudParis,
 Institut Polytechnique de Paris
 Evry, France
 Sirine.rahali@supcom.tn*

Maryline Laurent
*Member of the Chair Values and
 Policies of Personal Information
 R3S Team/RST Department
 Samovar, Telecom SudParis,
 Institut Polytechnique de Paris
 Evry, France
 ORCID: 0000-0002-7256-3721*

Souha Masmoudi
*Member of the Chair Values and
 Policies of Personal Information
 R3S Team/RST Department
 Samovar, Telecom SudParis,
 Institut Polytechnique de Paris
 Evry, France
 ORCID: 0000-0002-7194-8240*

Charles Roux
*Telecom SudParis, Institut Polytechnique de Paris
 Evry, France
 Charles.Roux@telecom-sudparis.eu*

Brice Mazeau
*Telecom SudParis, Institut Polytechnique de Paris
 Evry, France
 Brice.Mazeau@telecom-sudparis.eu*

Abstract—This paper proposes a new recommendation system preserving both privacy and utility. It relies on the local differential privacy (LDP) for the browsing user to transmit his noisy preference profile, as perturbed Bloom filters, to the service provider.

The originality of the approach is multifold. First, as far as we know, the approach is the first one including at the user side two perturbation rounds - PRR (Permanent Randomized Response) and IRR (Instantaneous Randomized Response) - over a complete user profile. Second, a full validation experimentation chain is set up, with a machine learning decoding algorithm based on neural network or XGBoost for decoding the perturbed Bloom filters and the clustering Kmeans tool for clustering users. Third, extensive experiments show that our method achieves good utility-privacy trade-off, i.e. a 90% clustering success rate, resp. 80.3% for a value of LDP $\epsilon = 0.8$, resp. $\epsilon = 2$. Fourth, an experimental and theoretical analysis gives concrete results on the resistance of our approach to the plausible deniability and resistance against averaging attacks.

Index Terms—Local differential privacy, recommendation, privacy, RAPPOR, profiles perturbation, Bloom filters, neural networks, XGBoost, Kmeans

I. INTRODUCTION

With the increase of online services, individuals are confronted with a lot of choices when making purchases. For better user experience, service providers rely on recommender systems helping users find items of interest. For this purpose, service providers are massively collecting and analyzing users' data which may threaten users' privacy. Hence, for individuals to get tailor-made services but not at the price of their privacy, as exposed in the survey of 2.000 people [16], and for recommenders to monetize the attention of consumers with targeted advertising, products and services selling, there is a

strong need to elaborate new recommendation systems taking privacy and utility into account.

Contributions This paper proposes a new recommendation system preserving both privacy and utility. The idea is to ensure protection against honest-but-curious entities - service providers and outsiders - while still getting useful recommendations. With that objective in mind, our approach relies on the LDP principle for the user preference profile to be perturbed at the browsing user side through a two-round processing - PRR (Permanent Randomized Response) and IRR (Instantaneous Randomized Response). That perturbation is an adaptation of the LDP-based RAPPOR approach [9] being made suitable for both classification and clustering tasks under local differential privacy. As far as we know, this is the first time that a two-round perturbation processing has been applied to a complete user profile.

With the objective of getting experimental utility vs privacy validation results, a full validation experimentation chain is set up with a recommender being implemented. At the recommender side, two successive mechanisms are performed: a machine learning decoding algorithm based on neural networks or XGBoost for decoding the perturbed Bloom filters and a clustering Kmeans tool for clustering users. It has to be noted that an appropriate user clustering leads straight to relevant recommendations for the user. The idea of the experiment is thus to assess how much users clustering is successful according to their perturbed preferences and a related privacy budget quantifying how much noise is included into their preferences.

Our approach is validated through both extensive experiments and a security analysis. The experiments show that

our method achieves a good utility-privacy trade-off with a 90% clustering success rate, resp. 80.3% for a value of LDP $\epsilon = 0.8$, resp. $\epsilon = 2$. The experimental and theoretical security analysis demonstrates that our approach supports both plausible deniability and resistance against averaging attacks.

Paper organization. Section II gives the useful background about Local Differential Privacy (LDP). Section III surveys existing LDP-related works highlighting their deficiencies. Then our approach is described in Sections IV and V. Section IV introduces the system model with the actors, the utility metrics, the privacy properties and the threat model. Section V details the step-by-step processing phases at both the user side and the recommender side. The three following sections provide a full evaluation of our approach. Section VI studies the utility performance achievements of our decoding and clustering algorithms according to several experimental conditions (privacy budget, Bloom filter parameters). Section VII gives a security analysis of our scheme with regard to the plausible deniability and averaging attacks. Section VIII discusses the utility vs privacy trade-off. Conclusions are given in Section IX.

II. LOCAL DIFFERENTIAL PRIVACY BACKGROUND

Local Differential Privacy (LDP) has its roots in Differential Privacy (DP) works [8], [13]. DP matches the global model where a trusted third party uses DP to produce statistics over a dataset, while withholding information about individuals in the dataset. LDP [18] matches the local model where data can be perturbed right at the source, locally to the user, thus leading to higher privacy guarantees as the trusted third party is no longer necessary.

Definition. A randomized algorithm M satisfies the ϵ local differential privacy [18] where $\epsilon > 0$ if for all pairs of the client's values x and y and for all $S \subseteq \text{range}(M)$:

$$\Pr[M(x) \in S] \leq e^\epsilon \Pr[M(y) \in S] \quad (1)$$

The definition introduces ϵ , known as the privacy budget or the privacy loss. It controls to what extent the output of an algorithm depends on the input, and thus it reflects the desired level of privacy. The smaller is the privacy budget ϵ , ϵ being a positive value, the higher is the privacy level.

III. RELATED WORK

This section makes an overview first on some LDP well-known use cases, and then on some privacy preserving systems including LDP recommender systems.

LDP use cases. LDP concepts are applied in various domains in the literature. For instance, Google proposed RAP-POR [9] so as to collect statistics about malicious URLs from users browsers without causing privacy threats. The proposed solution relies on encoding the values to be sent, by applying two ϵ -LDP perturbation levels. In Microsoft, LDP enables collecting data about the time spent by users in different applications, therefore, identifying its favorite ones and improving the users' experience [7], while still preserving privacy. For reducing possible privacy leakage in deep learning models, [6] proposes LATENT as an intermediate layer designed to satisfy

LDP in deep learning models. The suggested solution enables a data owner to perturb data at the owner's device before the data reach out an untrusted machine learning service.

Privacy preserving recommender systems. The need for privacy preservation in recommender systems triggered research efforts in the last decades. Two main axes based on cryptography and data perturbation are investigated. Kaaniche et al. [11] designed a privacy-preserving framework for recommender systems. They suggested that a user perturbs his profile relying on a collaborative secure computation, that incorporates intermediate nodes between end-users and service providers. Their framework generates additional computation overhead. In [4], [5], Canny proposes cryptographic protocols to preserve privacy in recommender systems. The suggested scheme uses matrix projection and factor analysis. Both techniques result in heavy computational and communication overhead. Aïmeur et al. [1] designed the Alambic system where users private data are shared between the service provider and a semi-trusted third party. The whole public key infrastructure is adopted so as to ensure data protection. Yet, only if the service provider doesn't collude with the semi-trusted third party, the user privacy is protected. Recently, Kim et al. [12] suggested SPIREL, a location based recommender system under LDP. The framework uses the Optimized Randomized Response [18] approach to perturb the transition patterns and refers to the piecewise mechanism [17] in order to perturb gradients. Both suggested algorithms are ϵ -LDP. The solution drawback is its high communication cost, as the transition patterns are encoded in a n^2 -sized bit-array (n denoting the domain of possible locations) by the user before perturbing them with the Optimized Randomized Response. Moreover, the framework takes into account only one location transition from the user's check-in history. Nevertheless, using LDP to protect only one transition doesn't fit well in practice as the user may have many transitions to report. The extension of SPIREL to report many items under LDP is far from obvious as the process relies on the gradient perturbation. While most of the existing works tackle the issue of only one item perturbation, BLIP [2] focuses on perturbing a whole profile using the LDP mechanism. However, the achieved utility in terms of recall does not exceed 0.26 for $\epsilon = 3$. Moreover, the utility computation is performed on perturbed Bloom filters, and does not refer to any decoding algorithm, which is not compatible with an integration into a recommender system. Furthermore, as the similarity computation between profiles is done at user side each time a new node joins the system, the solution can be considered as cost-prohibitive.

IV. SYSTEM MODEL AND OVERVIEW

This section details our full system model, and an overview of our approach. It presents the actors, the metrics for measuring utility, the supported privacy properties along with the considered threat model.

A. Actors

Our system model is composed of the following entities:

- **Client C :** a browsing user submits a query to a Web Search Engine WSE along with his profile made of individual attributes (preferences). C who cares about his privacy, sends to WSE his perturbed preferences according to the LDP method and a privacy budget ϵ .
- **Web Search Engine WSE :** the WSE server responds to the client's query according to the transmitted preference profile. WSE can suggest its own recommendations to C , and can take on the role of a proxy between C and third parties TP by relaying the profile and the TP 's recommendations back to C . Note that the intermediate WSE between C and TP could be any Service Providers.
- **Third Party TP :** TP is an advertising provider or any recommendation provider. Its main goal is to provide users with targeted recommendations.

B. An overview

To avoid leaking his preferences while still getting appropriate recommendations, a user C can decide to send a search query to WSE along with his perturbed preferences. The full processing for perturbing the preferences is detailed in subsection V-A and includes the following steps which are depicted in Figure 1:

- 1) C preferences using a Bloom filter [9]
- 2) Execute the permanent randomized response (PRR) step over the Bloom filter [9]
- 3) Execute the Instantaneous Randomized Response (IRR), as a second perturbation step [9], over the modified Bloom filter
- 4) Send the query along the perturbed preferences to WSE

Upon receiving the request, WSE forwards the perturbed profile to the recommender, either TP or WSE , and next referred for clarity as TP . TP next needs to decode the given preferences using a machine learning algorithm, resulting in a reconstructed approximate user profile (cf. subsection V-B). Then based on the obtained profile and the similarity among profiles, TP executes a clustering algorithm (cf. subsection V-C) and classifies C in one of the group of users, for next delivering targeted recommendations suitable for that group of users to C via WSE .

C. Metrics for Measuring Utility

Our recommendation system should preserve the utility - i.e. the adequacy between C 's expectations and the returned recommendations - for maintaining the user experience. This means that the user clustering being performed by recommenders WSE or TP on the perturbed C profile should be as close as possible to the one achieved with the unperturbed C profile.

To measure the utility, the metrics well known for clustering algorithms - accuracy, precision, recall, and F1 score - rely on the following values:

- **True Positive (TP).** The instance belongs to the class, and is predicted to be in the class.

- **False Positive (FP).** The instance does not belong to the class, but it is predicted as a class member.
- **False Negative (FN).** The instance belongs to a class, but it is predicted as not being a member of that class.

The following metrics are used for measuring utility:

- 1) **Accuracy.** The rate of the correct predictions out of all the predictions. Accuracy is sensitive to class imbalance, and is expressed as follows:

$$\text{Accuracy} = \frac{\text{Number of correct predictions}}{\text{Total number of predictions}} \quad (2)$$

- 2) **Precision.** The proportion of items correctly identified to be within a class i out of all the items identified to belong to that class. A low precision helps to identify where the model made incorrect classification. It is defined as:

$$\text{Precision}_i = \frac{\sum TP_i}{\sum TP_i + FP_i} \quad (3)$$

- 3) **Recall.** The proportion of items that should have been annotated with a given label, and that were actually annotated with that label. This is a measure of completeness and is formally defined as follows:

$$\text{Recall}_i = \frac{\sum TP_i}{\sum TP_i + FN_i} \quad (4)$$

- 4) **F1 score.** A weighted average of recall and precision. Thereby, the metric takes into account both false positives and false negatives, as follows:

$$F1 = 2 * \frac{\text{Recall} * \text{Precision}}{\text{Recall} + \text{Precision}} \quad (5)$$

D. Privacy Properties

This section defines the basic privacy properties supported by our approach.

- **Plausible deniability.** This property is granted through the use of an algorithm that satisfies the LDP definition as an obfuscation scheme. In fact, by observing the output of the algorithm, an adversary can not tell with high confidence whether a particular preference was definitely used as an input of the algorithm.
- **Resistance against averaging attacks.** Any adversary having knowledge of several versions of the perturbed Bloom filters can not find out the original preferences through an averaging attack.

E. Threat Model and Privacy Games

Our threat model considers a honest-but-curious adversary attempting to learn the preferences of client C from the perturbed Bloom filters sent by client C . We next define three privacy games, with regard to two adversaries provided with the following abilities:

- 1) **Basic Adversary BA .** BA is not provided with any Machine Learning algorithms. BA is an outsider, i.e. external to our system.
- 2) **Advanced Adversary AA .** AA is provided with Machine Learning algorithms. AA is an honest-but-curious WSE or an honest-but-curious TP .

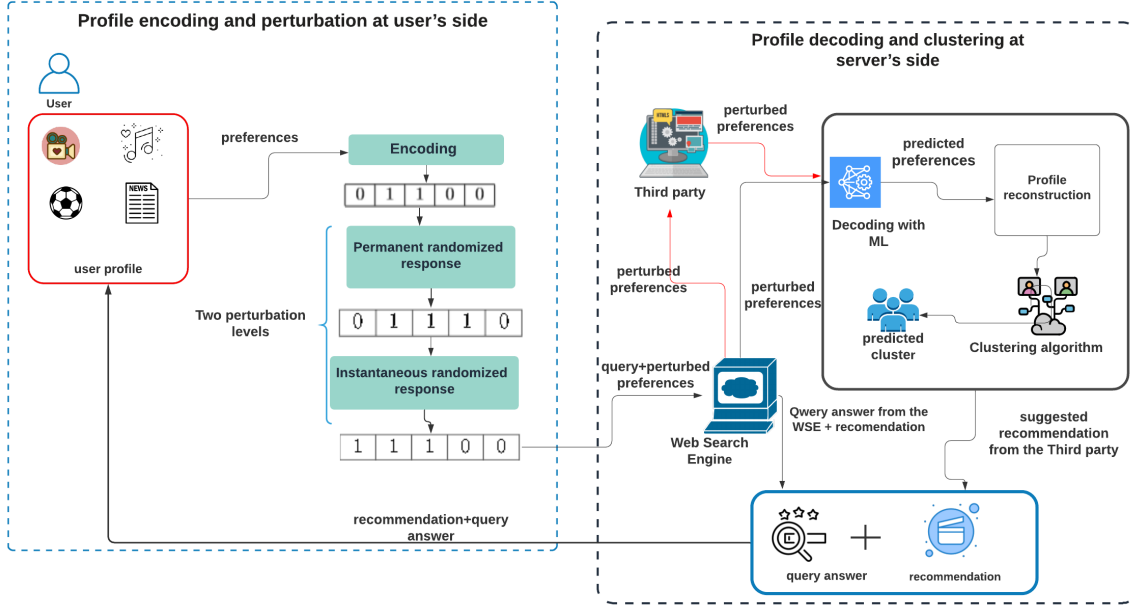


Fig. 1. System overview including a user C and a server (TP or WSE)

1) *The Plausible Deniability Game Conducted over one Preference by a Basic Adversary*: An outsider BA is challenged with the following privacy game:

- **Setup.** The challenger C provides the adversary BA with a set of N preferences, the Bloom filter size M , k hash functions and the privacy budget ϵ .
- **Training phase.** Upon receiving the mentioned parameters, BA is given a polynomial computation time in order to compute some Bloom filters for the transmitted N preferences. Therefore, he can reconstruct a database DBF of perturbed Bloom filters.
- **Challenge phase.** C selects a preference i and sends to C a related perturbed Bloom filter. BA sends back to C his guess about the preference.

BA wins the game if he is able to correctly guess the encoded preference.

If the probability to win the game is negligible, then our approach is said to be resistant against the plausible deniability attack conducted over one preference by a basic adversary.

2) *The Plausible Deniability Game Conducted over multiple preferences by the Advanced Adversary*: The insider AA is challenged with the following privacy game:

- **Setup.** The challenger C provides AA with a set of perturbed preferences along with their original values.
- **Training phase.** AA trains his ML algorithm with provided values.
- **Challenge phase.** C samples a set of preferences from his dataset, and sends to C a related perturbed Bloom filter. AA sends back to C his guess about the set of

preferences, which comes down to correctly decoding the perturbed Bloom filters.

The adversary wins the game if he gets high precision and recall.

If the probability to win the game is negligible, then our approach is said to be resistant against the plausible deniability attack conducted over multiple preferences by an advanced adversary.

3) *Averaging Game*: Any BA or AA adversary is challenged to find out the original preferences from a set of perturbed Bloom filters issued over the same set of C 's preferences.

V. DETAILED PROCESSING PHASES OF OUR APPROACH

In this section, we detail the main three phases of the new recommendation system, an overview of which is described in Section IV-B.

A. Perturbation Phase at the Client

The perturbation is handled at C 's device with the objective to obfuscate C 's preferences. It has to be noted that the set of preferences forms a profile composed of one or many categories where each category is denoted by C_i , $i \in \{1...l\}$ where l denotes the maximum number of supported categories. Each category is composed of groups of interests I_{ij} (i.e. $j \in \{1...g\}$) where g represents the maximum number of groups of interests for category i .

- 1) **Encoding.** As the first step of the perturbation process, the encoding algorithm maps C ' preferences into bits in a Bloom filter.

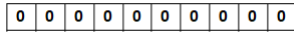


Fig. 2. Empty Bloom filter, $m = 10$

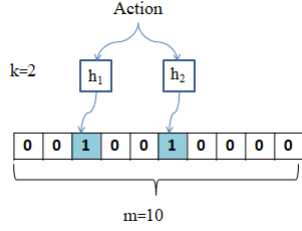


Fig. 3. Feeding the Bloom filter with the preference "Action", $m = 10$, $k = 2$

The first operation is to compute the optimal Bloom filter size m , given the maximum number of preferences encoded into the Bloom filter (i.e. $n = l * g$ if each category includes the same number of groups of interests), and a fixed false positive rate f_p , according to equation 6 [3]:

$$m = -\frac{n \times \ln(f_p)}{(\ln 2)^2} \quad (6)$$

Then, the optimal number of hash functions [14] is computed as given by equation 7.

$$k = \frac{m}{n} \times \ln 2 \quad (7)$$

After selection of m and k appropriate values, the Bloom filter B is initialized with all "0" values, as given in Figure 2. For feeding B with the set of preference values v , C first applies the k hash functions to v , and feeds B with the hash output providing indices. For illustration, let us consider a Bloom filter of size $m = 10$ bits, with $k = 2$ hash functions (h_1, h_2), and two preferences {Action, Fantasy} to be included into the Bloom filter. As given in Figure 3, C first computes the two hashes of the preference *Action*, and gets the results $h_1(\text{Action}) = 3$ and $h_2(\text{Action}) = 6$, thus leading to positioning the 3rd and the 6th bits of the Bloom filter to value 1. The same applies for the preference *Fantasy* as depicted in Figure 4 where $h_1(\text{Fantasy}) = 2$ and $h_2(\text{Fantasy}) = 8$.

- 2) **Permanent Randomized Response (PRR).** This first level perturbation applies over the Bloom Filter B obtained through the encoding phase. This step is executed once over a set of preferences v , whatever the number of search requests done by C to WSE . A noisy bit is derived from each bit of B thus resulting in a perturbed Bloom filter vector B' . The derivation is compliant with the RAPPOR works [9] and considers the following probabilistic processing:

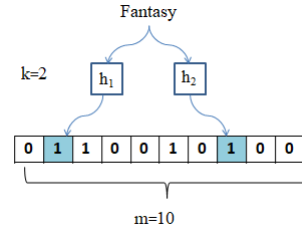


Fig. 4. Feeding the Bloom filter with the preference "Fantasy", $m = 10$, $k = 2$

$$B'[i] = \begin{cases} 1 & \text{with probability } \frac{1}{2}f \\ 0 & \text{with probability } \frac{1}{2}f \\ B[i] & \text{with probability } 1 - f \end{cases} \quad (8)$$

Where f is the privacy level parameter configured by C .

The obtained bit vector B' remains stored and known to C only. This first level perturbation PRR algorithm is ϵ -differential privacy with the following quantified ϵ_1 privacy budget:

$$\epsilon_1 = k \ln \left(\frac{1 - \frac{1}{2}f}{\frac{1}{2}f} \right) \quad (9)$$

- 3) **Instantaneous Randomized Response (IRR).** To guarantee stronger short-term privacy, this second level perturbation [9] is executed for each request done by C to WSE . After getting B' , the user initializes a bit vector S with all zeros and then applies the following probabilistic processing:

$$p(S[i] = 1) = \begin{cases} q & \text{if } B'[i] = 1 \\ p & \text{if } B'[i] = 0 \end{cases} \quad (10)$$

Where p denotes the probability of flipping a bit that equals to 0 into 1 whereas q represents the probability of keeping bits equal to 1.

This second level perturbation IRR algorithm is ϵ -differential privacy with the following quantified ϵ_2 privacy budget [9]:

$$\epsilon_2 = k \ln \left(\frac{q'(1-p')}{p'(1-q')} \right) \quad (11)$$

Where p' , resp. q' , is the probability of observing 1 given that the same Bloom filter bit was set to 0, resp. 1, as defined in the following equations.

$$p' = \frac{1}{2}fq + (1 - \frac{1}{2}f)p \quad (12)$$

$$q' = (1 - \frac{1}{2}f)(1 - q) + \frac{1}{2}f(1 - p) \quad (13)$$

TABLE I
NEURAL NETWORK CONFIGURATION

Parameter	Value
The number of nodes	Input layer: 80 First hidden layer: 60 Second hidden layer: 50
Loss function	Categorical crossentropy
The activation function	Output layer: Softmax Other layers: ReLu
Number of epochs	25
Batch size	70
Optimizer	Adam

TABLE II
XGBOOST CONFIGURATION

Parameter	Preference dataset	Flight dataset
N_estimators	100	100
Max_depth	3	3
Min_child_weight	1	1
Subsample	0, 8	0, 9
Gamma	0, 4	0, 4

B. Decoding Phase by the Recommender

Upon receiving C 's perturbed preferences, TP decodes the received C 's perturbed preferences into some approximate preferences, based on a machine learning algorithm. The problem is modeled as a multiclass classification task, aiming at predicting the classes of perturbed Bloom filters. For instance, given the music category, which contains eight classes (groups of interest): classical, jazz, pop... TP should identify for each perturbed Bloom filter its right label. Two machine learning algorithms - neural network and XGBoost - were selected for their ability to work on perturbed data. Both algorithms are calibrated to fit the specificities of the two following datasets, thus resulting into 2 configurations as detailed below:

- **Preference dataset.** The set of preferences includes 3 categories - movies, music and sports - and 7 classes for movies, 8 for music and 12 for sports.
 - **Flight dataset.** The set of preferences includes 3 categories - destination and flight class type - which are made up 11 classes for destination and 3 for flight class type.
- 1) **Neural network configuration.** The neural network is composed of an input layer which is fed with the perturbed preferences, two hidden layers, and an output layer. Two dropouts are introduced to mitigate the overfitting issue. For both of our datasets, the layers of the algorithm are configured according to the parameters given in Table I.
 - 2) **XGBoost configuration.** XGBoost is a gradient boosting algorithm. Table II gives the parameters calibrated for each dataset to optimize the model's performances. As can be shown, the configuration is slightly the same, except for parameter Subsample.

C. Clustering Tool and Accuracy Measurement

Clustering is done with Kmeans, considering $K = 4$ clusters for a number of profiles that is equal to 80.000 and $K = 5$

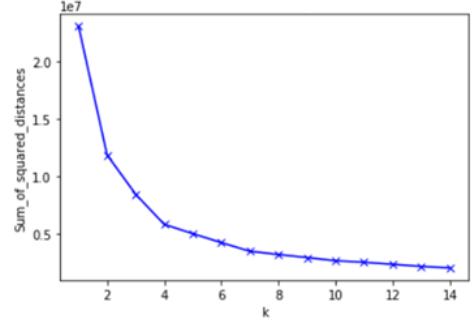


Fig. 5. Searching for the optimal number of clusters

clusters for 90.000 profiles (used later to study the privacy utility trade-off). This choice of K is based on the elbow method [15]. The main idea behind the technique is to run the Kmeans algorithm for different values of clusters and to calculate the Within Cluster Sum of Squares (WCSS). According to this metric, the variability of observations is computed in each cluster. A cluster that has a low value of WCSS is more homogeneous than a cluster with a high WCSS value. Formally, the objective is to minimize the following function.

$$WCSS = \sum_{i \in n} (X_i - Y_i)^2 \quad (14)$$

Where Y_i represents the centroid for observation X_i . Then, for a range of k numbers, the WCSS variation is plotted with respect to k . The optimal value of $k = 4$ was obtained through an experiment we did over 80.000 cleaned traces from a Qwant dataset to the Kmeans, and a range of k varying between 1 and 15. As depicted in Figure 5, there is a sudden huge drop in the WCSS value when increasing the number of clusters from 1 to 2, and a second drop - not as huge as the first one - at the cluster number 4. As the WCSS maintains a minimum value starting from $k = 4$, we deduce an optimal value of $k = 4$.

Unlikely to usual recommendation systems, our LDP-recommendation system works on perturbed profiles instead of the true users profiles. It is therefore necessary to adapt the accuracy measurement for evaluating the ability of the algorithm to group same-profile users into the same cluster. In our case, in a first experience, Kmeans is applied over a set of original (unperturbed) profiles, thus resulting in some cluster labels, as classically done. In a second experience, Kmeans is fed with profiles which have been first perturbed and then decoded. The clustering results are recorded and then used in a final step for comparing the clustering results with/without perturbation and for measuring accuracy. The more matches we get, the higher our accuracy.

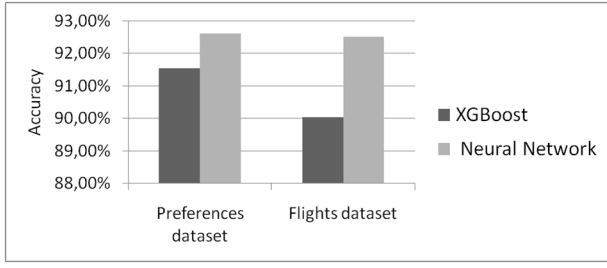


Fig. 6. Accuracy of XGBoost and neural network, $\epsilon = 0.8$, $K = 4$

VI. PERFORMANCE ANALYSIS OF THE DECODING AND CLUSTERING ALGORITHMS

A. Decoding Algorithms Evaluation

As shown in Table III, neural network outperforms XGBoost for both datasets with regard to the measured accuracy (for $K = 4$ clusters). This result is confirmed in Table III for low perturbation level ($\epsilon = 2$) as well as high perturbation level ($\epsilon = 0.80$). It gives higher precision, recall and f1-score and thus higher clustering success rates.

B. Clustering Evaluation

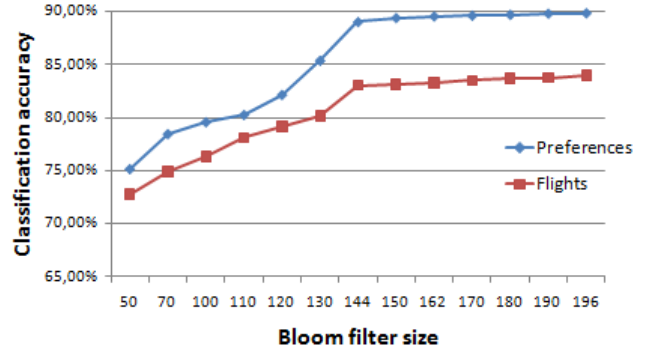
This subsection analyses the influence of different parameters on the clustering results, including the privacy budget value ϵ , the Bloom filter size M and the number of hash functions k .

- **Bloom filter size.** As expected in Figure 7, larger M results in higher classification accuracy, as the Bloom filter collision rate decreases. Yet, this comes at a cost in memory since increasing M leads to larger Bloom filter sizes. As shown, the benefit for high values of M diminishes when M exceeds 144 for this experimental setup. This observation is due to the hash collisions starting to vanish. Next observations are thus considering $M = 144$.
- **Number of hash functions.** Our experiment considers a minimum value of hash functions of 3, which corresponds to the optimal number of hash functions for $M = 144$, according to the equation 7. As depicted in the figure, the classification accuracy decreases when the number of hash function increases. This stems from an increasing number of hash collisions.
- **Privacy budget.** As expected in Figure 7, the clustering accuracy is an increasing function of the privacy budget. Indeed, the higher the privacy budget, the lower the perturbation level, and the higher the accuracy. The preference dataset achieves better clustering results. This might be due to the dataset characteristics, the number of categories, the number of classes per category...

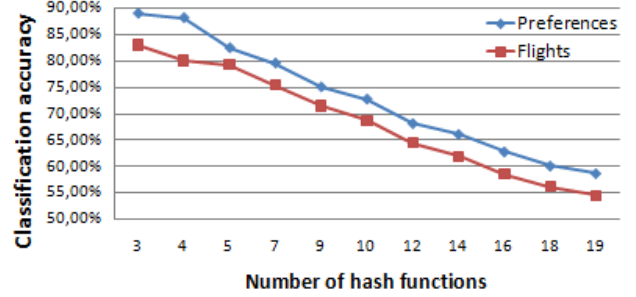
VII. SECURITY ANALYSIS

This section is dedicated to the security analysis with regard to the threat model defined in subsection IV-E.

[Bloom filter size for $\epsilon = 0.85$, $f_p = 0.1$ and $k = 3$]



[Hash functions for $M = 144$, $\epsilon = 0.85$ and $f_p = 0.1$]



[Privacy budget, $M = 144$, $f_p = 0.1$, $k=4$]

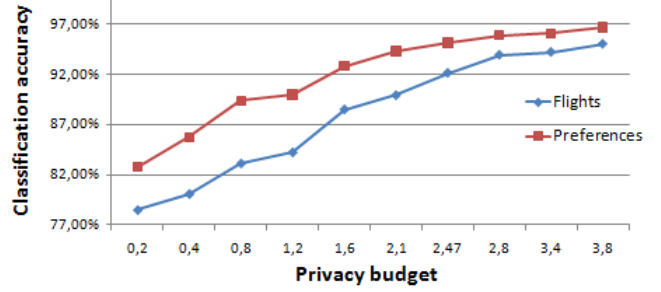


Fig. 7. Impact of several parameters on the clustering results, $K = 4$

A. Plausible Deniability over one Preference

Referring to the game described in Section IV-E, given the perturbed Bloom filter y sent by the challenger C and the preference universe D , the strategy for the basic adversary BA is to compute:

$$\begin{aligned}
 \text{guess}(y) &= \arg \max_{v \in D} Pr[v|y] \\
 &= \arg \max_{v \in D} \frac{\pi(v) \cdot Pr[\phi_{RAPPOR}(v) = y]}{\sum_{a \in D} \pi(a) Pr[\phi_{RAPPOR}(a) = y]} \quad (15)
 \end{aligned}$$

Where ϕ_{RAPPOR} denotes the perturbation mechanism, D is the universe of the preferences to enter into a Bloom filter, $\pi(v)$ is the prior probability of v and is equal to $\frac{1}{|D|}$ for all $v \in D$.

TABLE III
DECODING THE MOVIES CATEGORY WITH XGBOOST AND NEURAL NETWORKS. VALIDATION SET=40.000, $fp = 0.10$, $K = 4$

Metrics	Precision				Recall				F1-score				Support	
	Neural Network		XGBoost		Neural Network		XGBoost		Neural Network		XGBoost		Neural Network	XGBoost
	$\epsilon = 0.80$	$\epsilon = 2$	$\epsilon = 0.80$	$\epsilon = 2$	$\epsilon = 0.80$	$\epsilon = 2$	$\epsilon = 0.80$	$\epsilon = 2$	$\epsilon = 0.80$	$\epsilon = 2$	$\epsilon = 0.80$	$\epsilon = 2$	$\epsilon = 0.80$ and $\epsilon = 2$	$\epsilon = 0.80$ and $\epsilon = 2$
Action	0.95	0.94	0.82	0.90	0.93	0.99	0.81	0.88	0.94	0.96	0.81	0.89	8550	5751
Comedy	0.92	0.96	0.81	0.86	0.95	0.96	0.81	0.86	0.93	0.96	0.81	0.86	5693	5693
Drama	0.87	0.92	0.80	0.89	0.81	0.92	0.92	0.93	0.84	0.92	0.86	0.91	5691	8550
Fantasy	0.92	0.95	0.83	0.85	0.91	0.96	0.79	0.85	0.92	0.95	0.81	0.85	5751	5690
Horror	0.82	0.89	0.82	0.83	0.88	0.97	0.80	0.86	0.85	0.93	0.81	0.84	5721	5691
Romance	0.90	0.98	0.82	0.86	0.93	0.90	0.80	0.85	0.91	0.94	0.81	0.86	5690	5721
Thriller	0.94	0.96	0.89	0.87	0.91	0.81	0.71	0.74	0.92	0.88	0.79	0.80	2904	2904

For computing for each value v the probability that the perturbation mechanism outputs y , BA can refer to the following probability expressed in Equation 16 [10].

$$Pr \left[B'[i] = \phi_{RAPPOR}(B[i]) = 1 \right] = \begin{cases} \frac{e^{\frac{\epsilon}{2\Delta}}}{e^{\frac{\epsilon}{2\Delta}} + 1} & \text{if } B[i] = 1 \\ \frac{1}{e^{\frac{\epsilon}{2\Delta}} + 1} & \text{if } B[i] = 0 \end{cases} \quad (16)$$

Where 2Δ denotes how many positions can change in neighboring vectors at most. In Bloom filter encoding, Δ is equal to k the size of the hash functions domain. B' represents the perturbed version of B .

Experimental results for quantifying the chance of BA of winning the game. The success rate of BA for winning the game can directly be derived from the probability of Equation (16) as the probability of getting $B[i]=1$. This probability can be experimentally measured on our preference dataset, according to ϵ and k values and gives the results depicted in Figure 8. As expected, the lower ϵ and the higher k , the lower the probability for winning the game is. For ϵ ranging from $[0.1, 0.85]$ value, the probability that the adversary wins the game is in average below 0.22. Higher the k values, more difficult it is for BA to win the game as the number of hash collisions increases, thus leading to higher wrong guess.

As a conclusion, the success rate of BA can be low according to the selected ϵ and k values. As such, a suitable trade-off utility vs privacy, as detailed in Section VIII, needs to be found.

B. Plausible Deniability over Multiple Preferences

The attack is managed by an advanced adversary AA as presented in Subsection IV-E. This section provides experimental results, and an in-depth discussion, about the ϵ impact on the success rate of the adversary. The experiment is conducted over 10.000 samples and 20.000 samples given to AA for training. Results are provided in two figures, Figure 9 for the confusion matrices, and Table IV for the classification result statistics which enables to evaluate the decoding ability of the adversary. Table V gives the success rate of an AA adversary to win the game.

Figure 9.a shows that for low privacy budget ($\epsilon = 0.1$), the adversary has mistaken the majority of the classes as the values outside the diagonal are relatively high. His overall

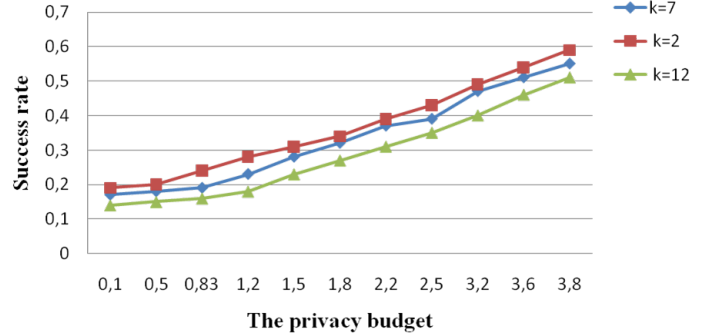


Fig. 8. Success rate of the plausible deniability attack over one preference by a Basic Adversary, the preference dataset

precision and recall are below 29 % as demonstrated by the classification report in Table IV.

An increase of the privacy budget from $\epsilon = 0.1$ to $\epsilon = 1.2$, resp. $\epsilon = 2.4$, improves the classification ability of the adversary. Indeed, the precision and recall reach only 31 %, resp. 49 % in average as depicted in Table IV. Thus the success rate for winning the game by AA $\epsilon = 1.2$, resp. $\epsilon = 2.4$, is equal to 33 %, resp. 52 % (see Table V). One can also notice that for $\epsilon = 2.4$, values inside the diagonal of the confusion matrix in Figure 9.c are higher than for matrices in Figures 9.a and 9.b.

C. Averaging Attack

As described in Section IV-E, the adversary is provided with a number of perturbed Bloom filters corresponding to the same set of C 's preferences. The adversary is only able to compute the first PRR output value B' which is the first-level perturbed Bloom filter. As the same B' value is used from one session to another (cf. Equation 10), the adversary is unable to find out the original Bloom filter B . As such, he is not able to win the averaging attack game.

VIII. PRIVACY VS UTILITY TRADE-OFF

This analysis measures the utility in terms of classification accuracy, which is defined as the ability of the recommender to perform correct classification of users, despite the perturbation scheme. Privacy is measured as the success rate of an advanced adversary for decoding the preferences.

TABLE IV
THE CLASSIFICATION REPORTS FOR DIFFERENT PRIVACY BUDGETS. TRAINING SET=20.000, TEST SET=10.000 SAMPLES

Metrics	Precision			Recall			F1-score			Support
	$\epsilon = 0.10$	$\epsilon = 1.2$	$\epsilon = 2.4$	$\epsilon = 0.10$	$\epsilon = 1.2$	$\epsilon = 2.4$	$\epsilon = 0.10$	$\epsilon = 1.2$	$\epsilon = 2.4$	$\epsilon = 0.10, 1.2, 2.4$
Classical	0.24	0.32	0.51	0.29	0.33	0.54	0.26	0.33	0.52	1460
Country	0.27	0.24	0.45	0.08	0.14	0.31	0.12	0.18	0.37	716
Electro	0.25	0.32	0.51	0.34	0.37	0.57	0.29	0.35	0.54	1410
Jazz	0.26	0.34	0.55	0.29	0.41	0.50	0.27	0.37	0.52	1424
Pop	0.24	0.32	0.51	0.26	0.35	0.56	0.25	0.34	0.53	1396
Rap	0.23	0.29	0.43	0.04	0.14	0.32	0.07	0.19	0.37	717
Rock	0.26	0.36	0.49	0.28	0.37	0.52	0.27	0.36	0.50	1432
Techno	0.29	0.33	0.53	0.28	0.33	0.56	0.28	0.33	0.55	1445

TABLE V
SUCCESS RATE OF THE ATTACK

Privacy budget	Success rate
0.1	29%
1.2	33 %
2.4	52 %

The experimental setup considers $k = 15$, 90.000 different profiles, and $K = 5$ clusters. Figures 10.a and 10.b illustrate the achieved trade-off for both datasets with specific parameters $k = 15$ and $K = 5$. As expected, the privacy is a decreasing function of the privacy budget and the utility is a rising function of the privacy budget. There is a privacy vs utility trade-off (curves intersection) happening for 84% of privacy level and 80% of utility for both datasets for a privacy budget equal to 0.58. This point is a kind of optimum when both utility and privacy are equally important.

IX. CONCLUSION

This paper proposes a privacy-preserving recommendation system, which lets the user decide on the amount of preference data he wants to communicate to a recommender, and the amount of LDP noise he wants to introduce into his data. Thus the user can decide how much privacy vs user experience he wants to keep. Through our specific experiment conducted over two datasets, our solution exhibits good performances in terms of privacy and utility, i.e. a 90% clustering success rate, resp. 80.3% for a value of LDP $\epsilon = 0.8$, resp. $\epsilon = 2$, and it shows that a privacy vs utility trade-off can be found for $\epsilon = 0.58$, with 84% privacy level and 80% utility.

X. ACKNOWLEDGEMENTS

This paper is supported in part by Institut Mines-Telecom chair VP-IP for Values and Policies of Personal Information and in part by the European Union's Horizon 2020 research and innovation program under grant agreement No 830892, SPARTA project. Authors are also thankful to Qwant for providing a dataset of cleaned traces which helped to achieve results closer to the ground.

REFERENCES

- [1] Esma Aïmeur, Gilles Brassard, José M Fernandez, and Flavien Serge Mani Onana. A lambic: a privacy-preserving recommender system for electronic commerce. *International Journal of Information Security*, pages 307–334, 2008.
- [2] Mohammad Alaggar, Sébastien Gambs, and Anne-Marie Kermarrec. Blip: non-interactive differentially-private similarity computation on bloom filters. *Symposium on Self-Stabilizing Systems*, pages 202–216, 2012.
- [3] Burton H Bloom. Space/time trade-offs in hash coding with allowable errors. *Communications of the ACM*, 1970.
- [4] John Canny. Collaborative filtering with privacy. *Proceedings 2002 IEEE Symposium on Security and Privacy*, pages 45–57, 2002.
- [5] John Canny. Collaborative filtering with privacy via factor analysis. *Proceedings of the 25th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 238–245, 2002.
- [6] MAP Chamikara, P Bertok, I Khalil, D Liu, and S Camtepe. Local differential privacy for deep learning. *arXiv preprint arXiv:1908.02997*, 2019.
- [7] Bolin Ding, Janardhan Kulkarni, and Sergey Yekhanin. Collecting telemetry data privately. *Advances in Neural Information Processing Systems*, pages 3571–3580, 2017.
- [8] Cynthia Dwork, Frank McSherry, Kobbi Nissim, and Adam Smith. Calibrating noise to sensitivity in private data analysis. *Journal of Privacy and Confidentiality*, pages 17–51, 2016.
- [9] Ulfar Erlingsson, Vasyl Pihur, and Aleksandra Korolova. Rappor: Randomized aggregatable privacy-preserving ordinal response. *Proceedings of the 2014 ACM SIGSAC conference on computer and communications security*, 2014.
- [10] Mehmet Emre Gursay, Acar Tamersoy, Stacey Truex, Wenqi Wei, and Ling Liu. Secure and utility-aware data collection with condensed local differential privacy. Gursay, Mehmet Emre and Tamersoy, Acar and Truex, Stacey and Wei, Wenqi and Liu, Ling, 2019.
- [11] Nesrine Kaaniche, Souha Masmoudi, Souha Znina, Maryline Laurent, and Levent Demir. Privacy preserving cooperative computation for personalized web search applications. In *Proceedings of the 35th Annual ACM Symposium on Applied Computing*, page 250–258, 2020.
- [12] Jong Seon Kim, Jong Wook Kim, and Yon Dohn Chung. Successive point-of-interest recommendation with local differential privacy. *arXiv preprint arXiv:1908.09485*, 2019.
- [13] Spiros Papadimitriou, Feifei Li, George Kollios, and Philip S Yu. The algorithmic foundations of differential privacy. Dwork, Cynthia and Roth, Aaron and others, 2014.
- [14] Sasu Tarkoma, Christian Esteve Rothenberg, and Eemil Lagerspetz. Theory and practice of bloom filters for distributed systems. *IEEE Communications Surveys & Tutorials*, 14(1):131–155, 2011.
- [15] Sasu Tarkoma, Christian Esteve Rothenberg, and Eemil Lagerspetz. Syakur, ma and khotimah, bk and rochman, ems and satoto, bd. *IOP Conference Series: Materials Science and Engineering*, 336(1), 2018.
- [16] Patrick Waelbroeck, Claire Levallois-Barth, Maryline Laurent, and Ivan Meseguer. Personal data and trust: how have post GDPR perceptions and uses evolved?, 2019 (accessed March 8, 2021).

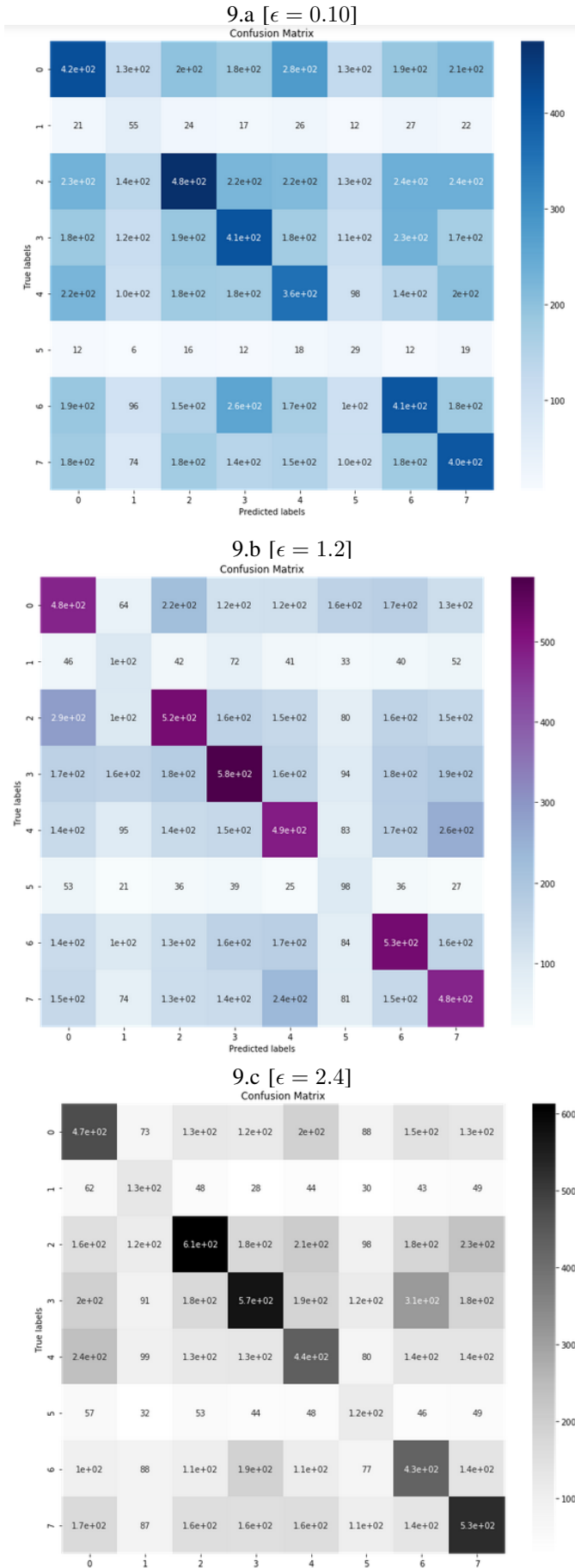


Fig. 9. Decoding ability of an adversary for various privacy budget on Music Category. Test set=10.000 samples

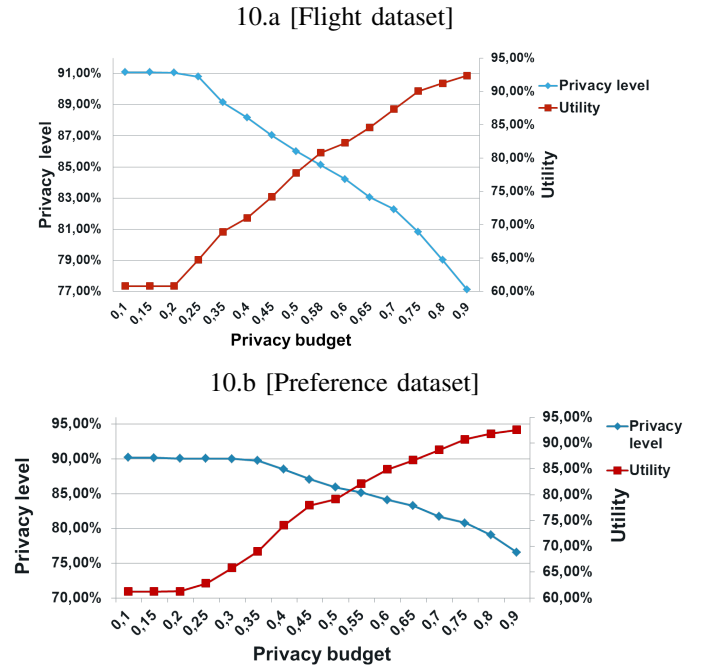


Fig. 10. Privacy vs Utility plots for 90.000 profiles, $k = 15$, $K = 5$

- [17] Ning Wang, Xiaokui Xiao, Yin Yang, Jun Zhao, Siu Cheung Hui, Hyejin Shin, Junbum Shin, and Ge Yu. Collecting and analyzing multidimensional data with local differential privacy. *019 IEEE 35th International Conference on Data Engineering (ICDE)*, pages 638–649, 2019.
- [18] Tianhao Wang, Jeremiah Blocki, Ninghui Li, and Somesh Jha. Locally differentially private protocols for frequency estimation. *26th USENIX Security Symposium Security 17*, 2017.