Privacy for Targeted Advertising

Avradip Mandal
Fujitsu Laboratories of America
Sunnyvale, CA, USA
amandal@us.fujitsu.com

John Mitchell
Stanford University
Stanford, CA, USA
jcm@cs.stanford.edu

Hart Montgomery
Fujitsu Laboratories of America
Sunnyvale, CA, USA
hmontgomery@us.fujitsu.com

Arnab Roy
Fujitsu Laboratories of America
Sunnyvale, CA, USA
aroy@us.fujitsu.com

***Abstract*—In the past two decades, targeted online advertising has led to massive data collection, aggregation, and exchange. This infrastructure raises significant privacy concerns. While several prominent theories of data privacy have been proposed over the same period of time, these notions have limited application to advertising ecosystems. Differential privacy, the most robust of them, is inherently inapplicable to queries about particular individuals in the dataset. We therefore formulate a new definition of privacy for accessing information about unknown individuals identified by some form of token that is chosen randomly but correlated with web interaction. Unlike most current privacy definitions, our's takes probabilistic prior information into account and is intended to reflect the use of aggregated web information for targeted advertising.**

We explain how our theory captures the natural expectation of privacy in the advertising setting and avoids the limitations of existing alternatives. However, although we can construct artificial databases which satisfy our notion of privacy together with reasonable utility, we do not have evidence that real world databases can be sanitized to preserve reasonable utility. In fact we offer real world evidence that adherence to our notion of privacy almost completely destroys utility. Our results suggest that a significant theoretical advance or a change in infrastructure is needed in order to obtain rigorous privacy guarantees in the digital advertising ecosystem.

***Index Terms*—Privacy, Utility, Data sharing, Targeted advertisements**

I. Introduction

Targeted online advertising is a massive ongoing business. In this market, which has become increasingly sophisticated over the last two decades, data aggregators collect information on user behavior. This data is then used by advertisers to participate in auctions for web impressions. The infrastructure allows advertisers to pay for ad placement at prices that reflect their estimate of the value of showing the ad to a user with certain characteristics. Data on user web behavior is used to bid on ad placement. The user profile data used in targeted advertising can include attributes such as an individual's web browsing and search history, age, gender, geolocation, and other characteristics and preferences. Vendors can develop user models that significantly characterize the user's propensity to buy a vendor's products and use their advertising budget to reach the most likely receptive audience.

The increasingly voluminous and personal nature of data collected for targeted advertising has led to a host of privacy concerns. Although individual attributes are disclosed without personal identifiers, a significant body of research has established that when combined with publicly available data, purportedly 'anonymized' attributes can lead to significant privacy risks for individuals [22] [7] [21] [23].

Let's consider a concrete example. A vendor such as Walmart may use an ad network such as Rubicon, which handles ad campaigns. Ad networks bid on ad placements which are managed by ad exchanges, such as AdNexus. Ad exchanges manage ad placements offered by content providers, such as CNN. When a user Alice browses a page on CNN, her attributes are shared with AdNexus. AdNexus strips off blatant linkage information like persistent tracking IDs and shares the user attributes with Rubicon. Rubicon then matches the attributes with the preferential model given by Walmart and bids for placement of Walmart's ad on Alice's browser. The correspondence between anonymized user attributes and an actual user is managed using identifiers or profile information that, for simplicity, we will refer to as a super-cookie.

There has been quite a bit of research done on changing the fundamental model of the online advertising ecosystem. Some works, such as Adnostic [31], have proposed alternative models for ad networks. In Adnostic, the user's browser is provisioned with an extension which keeps track of her behavior. No user attributes are sent to a third party. The ad networks propose a certain number of ads to the browser along with a preference model, and the browser chooses which ads to display based on the user attributes and the preference model attached to each ad. Which ad is shown is oblivious to the ad network. The billing problem which needs to know how many times an ad was shown is solved cryptographically.

There are several problems with this alternative model which are undesirable to the vendors. Blocking the transmission of user attributes from the ad exchange to the ad network prevents the ad networks from adaptively applying vendor preference models that may update and may potentially be more complex than those supported by an a priori framework. In addition, the models may be the 'secret sauce' of the vendors - highly proprietary in nature and thus undesirable to send to the ad exchanges.

Perhaps sophisticated cryptographic constructions building on program obfuscation and functional encryption can solve the dual privacy problem - protecting the user's data and the vendor's model - but secure efficient solutions look years away. Additionally, allowing vendors to work adaptively on user data to build their models would be horrendously complicated using cryptography.

There are compelling reasons for the stakeholders to con-

tinue with the existing model, and unless someone can develop a model without many of the drawbacks mentioned above, we must ask the following question:

> In the targeted ad ecosystem that tracks user behavior to increase economic efficiency, what form of user privacy is possible?

As a premise of our work in this model, we assume that ad exchanges are trusted because they possess and are gathering user data anyway (so we cannot assume anything if we do not trust them) and are interested in guaranteeing user privacy while transmitting anonymized data to the ad networks. The ad networks, on the other hand, can be adversarial and potentially attempt to uncover user data, possibly in conjunction with publicly available data about users.

Existing Privacy Notions. We argue that natural adaptations of existing privacy notions have limitations in this setting. We specifically discuss the notions of k-anonymity, ℓ-diversity, t-closeness, differential privacy, and pufferfish.

k-anonymity: The notion of k-anonymity [26], [29] says that an anonymized data set is k-anonymous when an adversary that is attempting to deanonymize the database cannot match any user to a set of less than k rows. However, it says nothing about the contents of those rows or what can be learned about the user. Sensitive information about users might be leaked, and in the worst case all the rows might be exactly the same, revealing all information about the user.

ℓ-diversity: ℓ-diversity [19] addresses the previously mentioned drawback of k-anonymity and ensures each equivalence class has *diverse* entries. However, if we have different prior distributions for different users, ℓ-diversity does not readily imply sufficient residual privacy for all users.

t-closeness: In some scenarios the users do not mind if the leaked information is the same as the majority of the crowd. In those cases, ℓ-diversity might become prohibitively expensive. t-closeness [17] addresses this issue by giving the guarantee that the distribution of sensitive attributes looks more or less the same in every equivalence class. This approach has the drawback that the distinction between sensitive and non-sensitive attributes is highly domain dependent and often user and data dependent as well (for instance, a person with no criminal record might not mind having that information released, but a person with a previous felony might desire privacy).

Differential Privacy: Differential privacy [11] allows leakage of information which does not depend on individual user preferences. In our setting we actually want to release data that corresponds to an individual user, but in a way that prevents learning about other attributes linked to the given real user. Queries of a database that depend solely on single rows are inherently not differentially private. Synthetic databases [4] allow release of "informative enough" data at record level, but the rows do not have any correspondence with the original rows. Thus the ad network setting fundamentally disallows differentially private release.

Pufferfish: Pufferfish [15] is a generalized framework for privacy. Pufferfish also contains notions of priors, and privacy is measured with respect to some secret variables. The authors show that, for some choice of parameters and queries, their notion implies differential privacy. Pufferfish is broad enough to encompass virtually all definitions of privacy, so the primary issue in designing a pufferfish based system is that people would need help from privacy domain experts in order to set up a privacy-preserving ecosystem.

Our contributions. In this paper we formalize the online advertising ecosystem setting and define a notion of privacy for the targeted ad ecosystem which we justify to be the correct notion of privacy for this setting. We reason that this notion avoids the limitations of existing notions and closely corresponds to what we expect a privacy preserving mechanism to deliver.

We begin with the observation that without any prior knowledge about an individual existing in the public it does not make sense to quantify her privacy - a de-identified record is essentially unlinkable to any existing identity. It is also realistic to model prior knowledge and public knowledge. So in our model, we assume that a prior distribution on the attributes on the universe of users is already given. A release mechanism is privacy preserving if the uncertainty on the attributes of a given user decreases very little conditioned on the released data.

However attractive this notion may seem, we mostly have dismal conclusions to offer. Although we can construct artificial databases which satisfy our notion of privacy together with reasonable utility, we do not have evidence that real world databases can be sanitized to preserve reasonable utility when our notion of privacy is enforced. We also offer some theoretical results that suggest proving privacy in our setting is difficult or impossible. This potentially disappointing news may be intrinsic to targeted advertising. If so, we must either accept this or change the model to allow rigorous privacy assurance.

Organization. The rest of the paper is organized as follows. In Section II, we formalize the ad ecosystem setting and define our privacy notion. In Section III, we compare this notion to existing privacy notions and justify how it is better suited to the ad ecosystem setting than these notions. Section IV discusses cases where it seems difficult or impossible to achieve our notion of privacy without significantly compromising utility. Section V discusses experimental results with real world data and also gives quantitative intuition of how our privacy notion behaves with respect to some generic scenarios. Finally we conclude with discussion points in Section VI.

II. Definitions

In this section we formalize the entities and processes of interest in an ad ecosystem and define a notion of privacy which essentially captures the intuition that data release reduces uncertainty about the attributes of a given user.

Essentially the ad exchanges are abstracted as data collectors which have an interest in protecting user privacy outside their perimeter. In the model they are assumed to be trusted since they have all of the user's attributes to begin with. We are interested to quantify the privacy loss that any release mechanism that the ad exchanges employ leads to. The ad networks which cater to the vendors act as the privacy adversary. The aim is to protect the privacy of the users, while providing adequate utility to the vendors.

A. Model and Assumptions

We assume a fixed set of users in the universe, of which we have publicly known identifiers. In a given model, we have a secret *database*. We view information on the users as a sequence of column values. Let's call the domain of this sequence X.

For each user, there is an a priori distribution on X. Aggregated over all users, let's call this distribution $\mathcal{P}$. Essentially the prior is defined as a distribution over the database which is intended to be sanitized and released. The construction of such a prior may be a highly subjective matter. It may be the case that actually some other attributes, possibly overlapping, are known publicly about the users. From these attributes and possibly with the aid of statistical inferences or machine learning, a distribution on the database of interest may be computed. Such processes can vary from adversary to adversary and are arguably orthogonal to the notion of privacy. We delineate the subjective process of construction of the prior from the semantics independent definition of privacy that we develop in this section by assuming it to be given to us.

We will measure privacy as the decrease in uncertainty of the adversary, with the database being drawn according to the distribution $\mathcal{P}$.

The operational concept is that there is a mechanism M which takes $\mathcal{P}$ and the real database D and produces a 'sanitized' database D'. After this point, there is a 'supercookie' table, which we model as a random bijective function τ from the domain of users $\mathcal{U}$ to a set of indices, which can simply be $\mathbb{Z}_{|\mathcal{U}|}$. The ad network, whom we model as an adversary in this setting, is finally given the anonymized and sanitized table $\tau(\mathrm{M}(D, \mathcal{P}))$. Once the adversary makes a decision based on this information regarding which indices s_i to target, the provider reverse looks up users $\tau^{-1}(s_i)$ and proxy routes the ads to the indicated users.

For example, given no other information about a user of a retail site, we can assume that the user is female with 50% probability. If we get the additional information that the user usually buys women's jackets, the probability goes up significantly. Each additional piece of information leads to a possibly higher certainty about the X value of that user. In other words, the entropy of X should decrease.

B. Formalization

Let's first define all the domains in the model. There is a set of individuals or 'users' $u_1, u_2, \cdots$ in the universe, that we will denote by $\mathcal{U}$. We will denote by X the domain of the set of columns, which are the attributes of interest of a user. A database is traditionally a sequence of rows, one for a user and its set of attributes. We will formally look at it as a mapping from the set of users to the set of attributes X. Thus the domain $\mathcal{D}$ of all such databases is the set of functions from $\mathcal{U}$ to X. For a user u, let D_u denote the column values for u.

We summarize the above formally in the following definition:

Definition 2.1 (Domains):

$$\begin{aligned}
&\text{Set of users } \mathcal{U} \stackrel{\text{def}}{=} \{u_1, u_2, \cdots\} \\
&\text{Set of databases } \mathcal{D} \stackrel{\text{def}}{=} \mathcal{U} \to X,\ D \in \mathcal{D},\ D_u \stackrel{\text{def}}{=} D(u) \\
&\text{Distribution on databases } \mathcal{P} : \mathcal{D} \to [0,1] \\
&\text{Mechanism } \mathrm{M} : \mathcal{D} \to \mathcal{D}' \\
&\text{Mapping function } \tau : \mathcal{U} \to \mathcal{U}' \\
&\qquad \text{and naturally extended as } \tau : \mathcal{D} \to \mathcal{D}'
\end{aligned}$$

1) Privacy Definition: Now we are in a position to define privacy. In the next definition, we take entropy as the measurement of uncertainty. However, any other measurement, like min-entropy and so on, can also be appropriate in certain situations. We can also use computational entropy when it might be appropriate. What is more important is the notion of taking the difference in uncertainty of the linkage between a user and her data, in case of privacy, before and after getting the transformed database.

Definition 2.2:

$$\begin{aligned}
\text{Privacy}_u &= H_{D \leftarrow \mathcal{P}}(u, D_u) - H_{D \leftarrow \mathcal{P}}(u, D_u \mid \tau(\mathrm{M}(D, \mathcal{P}))) \\
\text{Privacy} &= \max_{u \in \mathcal{U}} [\text{Privacy}_u]
\end{aligned}$$

Informally, privacy is defined to be just the decrease in uncertainty consisting of the user and their data. Thus, privacy loss is almost literally defined as the quantity of information learned about a user, which is exactly what it should be intuitively. Note that we need to define privacy as a whole to be global over all users, as the loss of any one user's privacy means the database is not privacy-preserving.

We also note that the users in the database may be a subset $\mathcal{U}$ of all users in the universe $\mathcal{U}^*$. This may lead to a potentially better privacy given the uncertainty over which users may be in the database. Our definitions could be adapted to that setting, but we avoid it in this paper for simplicity.

2) Information Learned: It is also useful for us to define a quantity that we will call 'Information Learned'. Intuitively, this is exactly the information contained in the released anonymized database that cannot be inferred from the prior distribution. We formally define information learned as follows:

Definition 2.3:

$$\begin{aligned}
\text{InfoLearned}_u = {} & H_{D \leftarrow \mathcal{P}}(\tau(u), D_u \mid \tau(u)) - \\
& H_{D \leftarrow \mathcal{P}}(\tau(u), D_u \mid \tau(\mathrm{M}(D, \mathcal{P})), \tau(u)) \\
\text{InfoLearned} = {} & \text{Exp}_{u \leftarrow \mathcal{U}} [\text{InfoLearned}_u]
\end{aligned}$$

Note that the definitions of privacy and information learned are very similar and the difference is rather subtle. Privacy is essentially defined as the decrease in uncertainty of the pair consisting of the user and her data, whereas information learned is defined as the decrease in uncertainty of just the data, with just its index identified by the supercookie mapping. For the global definition of information learned, we use the average instead of maximum, because for most applications that concern us, we will not be too upset if there are a few outliers where we learn too little information.

Useful Simplification. In most real world cases, the mechanisms we apply to the database in order to preserve privacy will be independent of the real identifier of the users. In other words, the mechanism M commutes with τ, i.e., $\tau(\mathrm{M}(D, \mathcal{P}))$ is identical as a distribution to $\mathrm{M}(\tau(D), \tau(\mathcal{P}))$, where $\tau(\mathcal{P})$ is the prior over the databases permuted with the supercookie mapping. For such mechanisms, the following expression is equivalent to InfoLearned$_u$:

$$H_{D\leftarrow\mathcal{P}}(u, D_u) - H_{D\leftarrow\mathcal{P}}(u, D_u \mid \mathrm{M}(D, \mathcal{P}))$$

Functional Information Learned. In many scenarios, information learned may be defined with respect to a function f on the domain X. In that case, the above definition of Information Learned is modified to:

Definition 2.4:

$$\text{InfoLearned}_{u,f} = H_{D\leftarrow\mathcal{P}}(\tau(u), f(D_u) \mid \tau(u)) - H_{D\leftarrow\mathcal{P}}(\tau(u), f(D_u) \mid \tau(\mathrm{M}(D, \mathcal{P})), \tau(u))$$

'Functional Information Learned' allows us to specify a utility function for more practical purposes. In many databases, some of the information is just irrelevant, and users may not care about certain values. For instance, sexual orientation may not be correlated with beer preference, and giving sexual orientation to an advertiser may be both a privacy risk and offer no benefit to the advertiser in practice. Defining functional information learned allows us to escape the conundrum of learning useless information.

III. Relation to existing privacy notions

In this section we discuss how the previously known privacy notions such as k-anonymity, ℓ-diversity, t-closeness, differential privacy and pufferfish relate to our notion of privacy.

k-anonymity: k-anonymity [26], [29] is the most popular notion for privacy preserving data release. It guarantees that the row corresponding to any user can not be deanonymized to less than k many rows. However, in the worst case all the rows might be exactly the same revealing all of the information about the user. On the other hand if the released dataset is *not* k-anonymous, there exists at least one user who suffers a privacy loss with left over privacy at most $\log_2(k-1)$ bits in our definition.

ℓ-diversity: ℓ-diversity [19] addresses the above mentioned drawback of k-anonymity. It prevents attribute leakage by ensuring each equivalence class has *diverse* entries.

The basic principle of ℓ-diversity is as follows:

Definition 3.1: An equivalence class is ℓ-diverse if there are at least ℓ "well-represented" values for the sensitive attribute. A release dataset follows ℓ diversity principle if every equivalence class in the dataset is ℓ-diverse.

The term "well-represented" can have different meanings:

1) **Distinct ℓ-diversity**: This is the simplest and most straight forward notion, which guarantees there are at least ℓ distinct sensitive values in each equivalence class. However, this does not rule out probabilistic interference attacks, when some sensitive values are more common than the rest.
2) **Entropy ℓ-diversity**: Entropy of an equivalence class E is defined as,
$$H(E) = -\sum_{s\in S} p(E, s) \log p(E, s).$$
Here, S is the domain of sensitive values and $p(E, s)$ denotes fraction of records in equivalence class E with sensitive value s. The release dataset has ℓ-diversity if for every equivalence class E, $H(E) \geq \log \ell$. In order to have entropic ℓ-diversity, the entropy of the sensitive attribute in the entire dataset must be at least $\log \ell$. Sometimes this is too restrictive [19].
3) **Recursive (c, ℓ)-diversity**: Recursive (c, ℓ)-diversity ensures that in any equivalence class the most frequent value does not appear too many times and the least frequent values do not appear too rarely. Let m be the number of sensitive values in any equivalence class and r_i (for $i \in \{1, \cdots, m\}$) be the number of times i^{th} most frequent sensitive value appears in the equivalence class E. Then the equivalence class E is recursive (c, ℓ)-diverse if $r_1 < c(r_\ell + r_{\ell+1} + \cdots + r_m)$. The released dataset has recursive (c, ℓ)-diversity, if all equivalence classes are recursive (c, ℓ)-diverse.

The notion of 'entropy ℓ-diversity' closely matches with our entropy based privacy definition. In fact we can argue that if the released anonymized dataset is not entropy ℓ-diverse then there exists one user who suffers a privacy loss with left over privacy at most $\log_2 \ell$ bits. In the special case where the prior distribution for all users are exactly the same, entropy ℓ-diversity indeed implies that the left over entropy for any user is at least $\log_2 \ell$ bits. However, if we have different prior distributions for different users, ℓ-diversity does not readily imply $\log_2 \ell$ left over privacy for all users.

t-closeness: t-closeness [17] addresses these issues by giving the guarantee that the distribution of sensitive attribute looks more or less the same in every equivalence class. In our setting, we do not really address this concern. Even if the leaked attribute is exactly the same for all users, we consider that as legitimate privacy loss. We chose to keep our framework semantics oblivious for simplicity.

The basic principle of t-closeness is as follows:

Definition 3.2: [17] An equivalence class is said to have t-closeness if the "distance" between a sensitive attribute in this

class and the distribution of the attribute in the whole table is no more than a threshold t. A release dataset has t-closeness if all equivalence classes have t-closeness.

To prevent similarity attacks which depend on the semantics of sensitive attributes, [17] ruled out the usual statistical distance or the entropic Kullback-Leibler (KL) distance [16]. Instead they proposed usage of Earth Mover's distance (EMD) [25] (a variation of Monge-Kantorovich transportation distance [13]) which takes semantic distances into account.

Differential Privacy: Differential privacy [9], [2], [10] only allows leakage of aggregate information. In an online advertising ecosystem, we want to enable advertisers to reach individuals, which requires releasing sanitized individual user preferences. This inherently violates differential privacy, as an attacker immediately learns of the existence of a user in the database. This makes it impossible to apply differential privacy.

It is also worth pointing out that differential privacy has a much more stringent security requirement than our definition of privacy: privacy must hold for all possible choices of priors. Our notion of privacy allows for privacy for some particular choices of priors, which theoretically inspires hope that our notion would allow more useful data release than would have been possible with the prior restrictions of differential privacy.

Pufferfish: One issue of the t-closeness privacy notion is that it does not consider the attacker's prior knowledge about the secret attributes. Kifer and Machanavajjhala [14], [15] introduced a privacy framework called pufferfish which handles adversarial knowledge in its privacy definition. Pufferfish can also be seen as extension of differential privacy. In the pufferfish framework one needs to specify a set of potential secrets. Privacy in pufferfish guarantees that even with access to the released dataset, potential secrets do not get revealed. The real hardness of deploying a pufferfish based system is that we require the help of domain experts who would define privacy sensitive variables. We can think of our privacy notion as a simpler, practical and intuitive pufferfish instantiation where all the bits of information are equally privacy sensitive. On the other hand, it is also possible to modify our privacy goals by providing more weights to sensitive bits of information.

Related Works in Anonymity: Anonymity is a related notion which describes how individuals are hard to identify within a set of a priori defined individuals, given some event data. Several works [8], [27], [30], [6] provide an information-theoretic framework for measuring anonymity with applications geared towards mixing networks. Our work is mainly focused on privacy and information revealed about a particular user rather than purely anonymity. As mentioned in [32], anonymity and privacy are in some sense dual of each other (anonymity is the hardness of linking a record to a person, privacy is the hardness of linking a user to pieces of a record). So some results mentioned in these papers for the entropic anonymity setting might carry over to our entropic privacy settings as well. For instance, in some scenarios Renyi entropy or min-entropy might be a better choice compared to the plain entropy measure.

Further Reading: [28] studies the problem of linkability of records, which is not directly addressed by our framework, because our analysis assumes every user's data is independent.

[3] takes ideas from stochastic analysis techniques used in the financial industry for determining the future risks of data release. However this also focuses on anonymity and linkability, not privacy.

[12] presents a negative result which shows any optimal (with respect to some utility) ℓ-diverse data release provides more information to the attackerthe optimality of the released data can break the privacy guarantee. Our negative result is not directly related to this–we show for reasonable utility gain we cannot provably show privacy for many reasonable priors.

[18] formalizes the notion of membership privacy which is very similar to differential privacy while taking into account prior beliefs.

[24] defines the meaning of various privacy terms such as anonymity, unlinkabilty, and unobservability, and [5] introduces an application-agnostic unifying framework to accomodate all such notions.

It is also useful to note that our lower bounds may apply to some of these alternate definitions of privacy and anonymity as well.

IV. Lower Bounds

We next discuss cases where it seems difficult or impossible to achieve notions of privacy under our definition of privacy. We believe that these instances are scenarios where it is difficult to have 'privacy' in any sort of real-world sense of the word, so we think that they are useful to show that our definition of privacy mirrors that of what real-world people expect it to mean. We briefly outline our main points before delving into the details.

Graph Isomorphism Lower Bound: We first show that, in some circumstances, it can be computationally hard to determine whether or not releasing a given database is privacy-preserving or not. To do this, we show how to construct a (very artificial-looking) database in a way that, if the database can be deanonymized, an arbitrary graph isomorphism problem can be solved. We then fit the database into our privacy paradigm, and show that, since it is hard to tell whether a given graph isomorphism problem will be easy or hard to solve [1] (we can embed easy problem instances, hard problem instances, or problem instances somewhere in between in terms of difficulty), it will be hard to tell whether or not a database is privacy preserving. This mimics our real-world intuition in that we find it difficult to tell whether or not given databases are privacy preserving or not, even with close inspection.

Netflix-Inspired Database: In one of the most well known and impactful works regarding privacy, Narayanan et al [22] deanonymized data given out during the Netflix challenge using the IMDb database. Here, we aim to show that, under our definition of privacy, databases that resemble the Netflix

challenge data are extremely difficult to make private in the presense of outside data, even when techniques such as adding noise are used. To do this, we will approximate real-world database instances mathematically, and show that our approximations of these database distributions do not lend themselves to privacy preserving applications. We note that the fact that these sorts of databases do not preserve privacy under our definition of privacy mirrors the privacy loss that happened in the real world.

Databases Amenable to Privacy: We finally discuss the implications of our above analysis, and examine what sort of databases we can actually give positive results for privacy. Unfortunately, these databases seem very limited, as, generally speaking, the number of columns in the database must be much, much smaller than the number of rows.

A. Graph Isomorphism Lower Bound

In this section, we–very roughly speaking–show that any adversary that can deanonymize an abitrary database can solve the graph isomorphism problem, which is not known to have worst-case polynomial-time algorithms [1]. We can extend this argument to apply to our definition of privacy as well. Our approach consists of the following: we construct two databases, one 'public' and one 'private', using a given graph isomorphism problem to help generate the problem instance. Roughly speaking, each user in the databases corresponds to a node on the graph, and the edges of the graph are encoded as data in the rows. We add the users' names to the appropriate rows in the public database, and add valuable 'personal data' to the rows in the private database as well in order to make the information loss due to deanonymization catastrophic.

In this way, an adversary that is given both the 'private' and 'public' databases is forced to solve an arbitrary graph isomorphism problem in order to match rows in the two given databases and learn private information about the users. The public database will have public user names, and will contain a graph representation. The private database will be anonymized, but contain both an alternative graph representation and lots of private user data. Thus, deanonymization (and privacy) come down exactly to how well an adversary can solve a given graph isomorphism problem. Obviously the 'databases' in this instance are highly unusual and unlikely to look like anything in the real world, but we think that this does illustrate an important lower bound on arbitrary databases.

Details: While we attempt to avoid spending too much space with traditional formalization, we explain our lower bound in more detail now. Let $\mathcal{G}_1 = \{\mathcal{V}_1, \mathcal{E}_1\}$ and $\mathcal{G}_2 = \{\mathcal{V}_2, \mathcal{E}_2\}$ be the definition of two graphs $\mathcal{G}_1$ and $\mathcal{G}_2$ with vertices and edges $\mathcal{V}_i$ and $\mathcal{E}_i$, respectively. Let $n = ||\mathcal{V}_1|| = ||\mathcal{V}_2||$ and $m = ||\mathcal{E}_1|| = ||\mathcal{E}_2||$. Let $v_{1,i}$ be the ith vertex under some ordering of the vertices in $\mathcal{G}_1$, and let $E_{1,i}$ be the set of vertices adjacent to $v_{1,i}$ (i.e., the edge set) in $\mathcal{G}_1$. We define $v_{2,i}$ and $E_{2,i}$ analagously. We do not explicitly make a choice for how to represent the edge set, but note that it can be done for arbitrary graphs with less than n bits for each node/user.

Suppose we let u_i denote user i's real-world identifying information. Next, suppose we construct two databases D and D' in the following ways: first, let $m : \mathbb{Z} \to \mathbb{Z}$ be a map that maps vertex indices in $\mathcal{G}_1$ to their isomorphic vertex indices in $\mathcal{G}_2$. Let the string P_i denote an (arbitrarily large) string of private information about user i. Next, let the row i of the database D be defined as D_i and let $D_i = u_i||v_{1,i}||E_{1,i}$ and similarly define $D'_i = v_{2,m(i)}||E_{2,m(i)}||P_{m(i)}$.

Note that the information that an adversary learns about a user in the above setup is entirely determined by the degree to which they can solve the graph isomorphism problem. Every row in D that the adversary can successfully match to a row in D' completely leaks the information of one of the users in the database. Probabilistic analysis works too: if an adversary can determine that a row in D matches up to one of k rows in D', for instance, then the adversary gains information as well.

Finally, we also note that we can use the decisional version of the graph isomorphism problem (deciding if two graphs are isomorphic or not) to create an even stronger (but more artificial) lower bound. This follows by appropriately randomizing the database D' with probability one half and asking the adversary to determine which version of D' they got–the one with real user data or the one with random user data. This stronger bound only is applicable when an adversary does not know an isomorphism exists, which may not be the correct analogy for certain settings in this paper. We omit explaining this result in more detail due to space constraints.

What This Means: This lower bound essentially shows that it can be very hard to determine whether or not a released database is privacy-preserving. While our instance is artificial, it does show that it will be fruitless trying to prove privacy for arbitrary databases. We also note that our public database uses fixed data, rather than distributions. If we only had distributions on users rather than fixed data, the problem would potentially become even harder (since fixed data can just be interpreted as point distributions).

Additionally, note that graph isomorphism is not a problem that has average-case hardness, and it is not known how, in general, to test whether particular cases of graph isomorphism are easy or hard to solve [1]. Thus, suppose some data provider wants to run some basic data privacy checks. Speaking in relative (and asymptotic) terms, there are worst-case databases that will pass any such checks but fail on slightly more complicated checks, which can be utilized by an adversary to gain sensitive information.

We want to conclude this section by pointing out that the databases we use in this lower bound are highly arbitrary and unlikely to resemble any sort of real-world database. However, it does mean that it will be impossible to find any sort of privacy-testing algorithms that work on all databases. This does not rule out algorithms for specific classes of databases, however, that do not admit this graph isomorphism representation. We will focus on these sorts of databases later in the paper.

B. Sparse Datasets

In one of the most well known and impactful works regarding privacy, Narayanan et al [22] deanonymized data given out during the Netflix challenge using the IMDb database. There have been several follow-up works [7], [21], [23] expanding on this deanonymization process, adding both more theory and broader classes of deanonymizable databases to the literature. In this section, we don't really aim to directly improve this line of work. Instead, we focus on the following problem: is there any way to fix these databases so that they are both not deanonymizable (or, ideally, offer privacy according to our definition) and also still useful? Here, we aim to show that, under our definition of privacy, databases that resemble the Netflix challenge data or other similar sparse databases are extremely difficult to make private in the presence of outside data, even when techniques such as adding noise are used, while still providing some utility.

To do this, we show that there exist a large class of databases which resemble the now well-known Netflix/ IMDb databases. We show that basic mitigation techniques on these databases, like adding noise or eliminating columns, only slowly decrease privacy loss, and that, when using these techniques, we eventually reach a point where we have neither privacy nor utility. This implies that these databases are inherently difficult to make private, and either much more sophisticated privacy mechanisms are needed or the databases are just impossible to make private in a useful manner.

1) Database Model: Our model for the Netflix database is going to be a generic sparse database D. Suppose we let n denote the number of rows of D (the number of users) and m denote the number of columns (the list of movies watched in the real-world example). We will select each entry of this database from a Bernoulli distribution $\mathcal{B}_\delta$ for some $\delta \in (0,1)$. This will be the start of our 'anonymized' database.

To create our 'public' database (example: the IMDb database), we will start with an empty database D'. For the sake of simplicity, we will assume D' also has n rows and m columns, although our analysis is essentially the same if this is not the case as long as the number of columns and rows (and their ratio) is large enough. Then, for each nonzero entry in D, we will set the corresponding entry in D' to be one by sampling from some Bernoulli distribution $\mathcal{B}_{\delta'}$ for some $\delta' \in [0,1]$.

To give some margin for error, we can also do another pass and, for each entry in D, set it to one (if it is not already one) if a random sample from some from some Bernoulli distribution $\mathcal{B}_{\delta''}$ for some $\delta'' \in [0,1]$ outputs one. This will allow us to simulate noise, among other things.

Next we introduce some formalism to make the (slight amount of) math easier to follow. Let the set of users $\mathcal{U}$ be defined such that $\mathcal{U} \stackrel{\text{def}}{=} \{u_1, u_2, \cdots, u_n\}$. Let $\mathbf{X} \in \mathbb{Z}_2^{n\times m}$ and, for all $i \in [1,n]$, $\mathbf{x}_i \in \mathbb{Z}_2^m$. We define D to be row-wise constructed as $D_{u_i} = \mathbf{x}_i$.

We circuitously define the prior $\mathcal{P}$ using the following procedure: we set $\mathbf{X}_{i,j} = 1$ if the output of a fresh, independent sample from $\mathcal{B}_\delta$ is one, and set $\mathbf{X}_{i,j} = 0$ otherwise. For our second, transformed database D', which maps users to $\mathbb{Z}_2^m$ according to a matrix $\mathbf{Y}$, where $\mathbf{Y} \in \mathbb{Z}_2^{n\times m}$, we initially set $\mathbf{Y}$ to all zeroes, and, for each nonzero $\mathbf{X}_{i,j}$, set $\mathbf{Y}_{i,j} = 1$ if a sample from $\mathcal{B}_{\delta'}$ results in an output of one. Then we apply a random row permutation to $\mathbf{Y}$, resulting in the public dataset.

Finally, in order to account for real-world effects like noise and database inaccuracy, we modify $\mathbf{X}$ again in the following way: we create a matrix $\mathbf{Z} \in \mathbb{Z}_2^{n\times m}$ where, for each nonzero entry in $\mathbf{X}$, we set $\mathbf{Z}_{i,j} = 1$ if a sample from $\mathcal{B}_{\delta'}$ results in an output of one. While, for certain choices of parameters, $\mathbf{X}$ may represent a 'true' database, we let $\mathbf{Z}$ represent a potentially noisy, more realistic one.

Thus, our question is the following: given $\mathbf{Y}$, $\mathbf{Z}$, and the mapping of users to $\mathbf{Y}$, what can we learn about the mapping of users to $\mathbf{Z}$? In other words, under our definition of privacy, what is the privacy loss? We assume that the rows in these matrices have been randomly permuted, so the order reveals nothing about the user. Note that in some cases (i.e. the case where multiple users' records are identical) it may not be possible to exactly reconstruct the mapping, but it still may be the case that lots of additional information about users (information in $\mathbf{Z}$ but not $\mathbf{Y}$) might be leaked.

2) Basic Calculations: Let's begin by doing a relatively basic calculation. Suppose some user u_i has a k-tuple of nonzero entries in $\mathbf{X}$. This corresponds to a user having three true (i.e. before noise is added) entries in a sparse database. We calculate the probability of the event that the k-tuple exists for the same user and is also unique in $\mathbf{Y}$. If such an event occurs, then clearly the user in question can be deanonymized.

So, while the actual probability is slightly complicated to compute, we can use union bounds and find an upper bound on the probability that interest us (and still derive a meaningful result). The probability that no other user has the same k-tuple of ones in $\mathbf{Y}$ is $\geq \left(1-(\delta+\delta'')^k\right)^{n-1}$. This is only an upper bound on the probability–again, we are using union bounds here to keep things simple. The probability that the tuple does exist (for the proper user) in $\mathbf{Y}$ is at least $(1-\delta')^k$. Thus, our overall probability of this event occurring is at least $\left(1-(\delta+\delta'')^k\right)^{n-1}(1-\delta')^k$.

This basic calculation will be sufficient for most of our arguments. While it's not remotely close to the best or most efficient deanonymization attacks, it will be illustrative for our purposes.

3) Parameter Comments: In the previous paragraphs, we did a very simple calculation and showed that the probability of a deanonymization event happening was at least $\left(1-(\delta+\delta'')^k\right)^{n-1}(1-\delta')^k$. Assuming that δ' is not too small (which is a reasonable assumption in practice), the dominating term becomes $\left(1-(\delta+\delta'')^k\right)^{n-1}$. Note that if $(\delta+\delta'')^k \approx \frac{1}{n}$, then a deanonymization event occurs with a constant probability.

More concretely, if $\delta + \delta'' \approx \frac{1}{n^k}$ or less (but not too close to zero), then this deanonymization event occurs. For sparse enough databases, this happens with reasonable probability (it is unlikely, for instance, that any normal user has watched a more than $\left(\frac{1}{\#movies}\right)^k$-fraction of all movies on Netflix). Like the literature suggests we can do, we have outlined how to construct a basic deanonymization attack on a class of sparse databases.

4) Mitigation Techniques: What most of the literature on deanonymization does not consider is how to possibly mitigate such deanonymization attacks. Due to the way we have set up our model, we can explain why some of the most basic techniques to mitigate the above (very simple) deanonymization attacks do not work. We mention some of these, and why they fail in many cases, below. Unlike most of the literature, we include a rough estimate of database utility in our arguments, which allows us to make more powerful statements. If a database is privacy preserving but contains almost no useful information, why bother? These mitigation techniques can also sometimes result in databases which have both no privacy and no utility, which is evidence towards impossibility of privacy for these types of mitigation.

Adding Noise: Note that noise is incorporated in our model in the form of the δ'' parameter. Additionally, the δ' parameter can be implicitly used to simulate noise in the public database, although typically it is not safe to assume that 'public' databases will be released in a noisy manner. Adding noise in a straightforward manner still does not defeat our primitive attack, although it does mean the database has to be sparser (or the rows longer).

What is most interesting about this situation is that we can actually reach situations where a database has neither privacy nor any sort of useful utility. Suppose we set $\delta = \delta'' = \frac{1}{2\sqrt{m}}$. 2-tuples in the anonymized database have a constant chance of being present and unique in the public database, so the database is clearly not privacy preserving. However, a full half of the data in the released database is just noise, meaning that is essentially useless for any practical applications. This means that adding noise in a straightforward manner is not a good mechanism to try to privatize a sparse database. There may be more complicated ways of adding noise that work, but we could not think of any simple ones.

Eliminating Columns: Reducing the number of the columns of the database may work to reduce privacy loss, but this also incurs a fairly dramatic utility penalty. In order to make the deanonymization attack impossible, the number of columns (m) must be chosen so that unique k-tuples are unlikely to exist, assuming all other parameters stay the same. This means that, in the expectation δkm must be less than one (for an attack for a particular k to work) which is pretty ridiculous, given the number of nonzero entries in the database would be around kn. Thus, naively compressing the database doesn't seem to work.

'Masking' Columns: A more clever method of making the anonymized database harder to deanonymize might include filling in entries to make the database less sparse using some sort of machine learning. This would obviously make the database not sparse and defeat our stated (and most known) deanonymization attacks. While the required amount of data storage would go up dramatically, accuracy of predictions (and overall utility of the database) would presumably not be affected.

While this might seem like a good idea in theory, in practice we expect dimensionality reduction attacks to work quite well on this sort of data (since none of the rows, when represented as vectors, will presumably be close to each other in the high-dimensional vector space). Unfortunately fully exploring this is outside of the scope of this paper, but a rigorous analysis of these techniques would probably be useful, even if we think it might be unlikely to succeed.

5) What It Means: Our results seem to indicate that transforming sparse databases into privacy-preserving databases while still maintaining some form of utility seems difficult. Obviously better, more complicated techniques might provide solutions where we have failed, but the problem seems very difficult. We have exhausted all of the simple things that we could think of to try to fix these sorts of databases, but it is possible that there still could be solutions.

C. Databases Amenable to Privacy

It is useful for us to briefly consider the sorts of databases that are not covered by our lower bounds, since arbitrary databases with enough columns tend to either be deanonymizable or impossible to classify as deanonymizable or not. We discuss some examples of hypothetical databases where privacy might hold in the paragraphs below. Unfortunately, these databases do not seem to be representative of real-world databases that might exist for online advertising.

Minimal or Nonexistent Priors: In the case where there is no (or a very small amount of) public data or the public data is identical for each user, privacy from our definition follows fairly easily. An attacker only learns the values of the anonymized rows of the database, and nothing that could be used to link back these rows to any real-world users. This follows from the fact that each user is equally likely to correspond to every row in the anonymized database, so deanonymizing the databases is impossible.

Thus, the entropy for each user remaining after the release of the anonymized database is roughly proportional to $\frac{1}{n}$ of the entropy in each row, and this is an upper bound on privacy loss assuming somewhat of a uniform distribution on the rows. The only way this results in a drastic privacy loss is if all of the rows are unexpectedly similar, which we would expect to happen with very low probability.

Very Small Data Release: If the anonymized database contains columns with information approximately proportional to the log of the number of users, and the distribution of the anonymized database isn't too pathological, then we can

also potentially prove privacy under our definition. This just follows using rote, brute-force computation. We can compute the likelihood of every user corresponding to each row, and how much entropy remains for each user in every case.

Assuming that the distribution of the anonymized database isn't too pathological and, if an output appears once, it appears many times, then it can be shown that a user's privacy loss is not large. We defer a more thorough analysis of this to the next section.

V. Experimental Results

In this section, we explore how we can estimate privacy loss for simple anonymization mechanisms. We can show that when priors are exactly same for all users we can precisely measure the privacy loss quite easily. This is the case when only aggregate/population statistics are known about users.

In particular we show that we can publish an anonymized version of the training data available in the Million Songs Dataset [20] with high information gain and low privacy loss. However, we need to assume the only available prior is the aggregate popularity of the songs, which is not realistic - with social networking and readily available online data it is easy to glean something about most users' musical taste. We finally discuss why estimating the privacy loss is hard in case of non-identical priors for different users.

Suppose we have a dataset D, consisting of the listening history of t popular songs $(s_1, \cdots, s_t)$ among n users $(u_1, \cdots, u_n)$. Furthermore, the listening history h_{ij} is only a binary bit value indicating whether user u_i has listened to song j or not. Suppose our prior information is only the global popularity of the songs, i.e., probabilities $(p_1 \cdots, p_t)$ where p_i is the probability whether 'any' user has listened to song s_i or not. Now if we release the de-identified version of the dataset D, the privacy loss for any user can be formulated as:

$$-\sum_{i=1}^{t}(p_i \log p_i + (1-p_i)\log(1-p_i)) + \sum_{j=1}^{2^t}(\tfrac{n_j}{n} \log \tfrac{n_j}{n}).$$

Here n_i is the frequency of the rating vector corresponding to binary representation of i. Moreover, the information learned about any user is

$$-\sum_{i=1}^{t}(p_i \log p_i + (1-p_i)\log(1-p_i)).$$

Million Songs Data Set: The Million Songs Dataset (MSD) challenge was a machine learning challenge hosted in Kaggle. The task was to suggest a set of songs to a user given half of their listening history and the complete listening history of another 1 million people. Here we take the training dataset, containing the listening history of about 1 million users ($1,019,318$ to be precise) and $385,546$ total songs. We noticed, even though there are over quarter million songs, the total entropy of any user is only about 540 bits given the aggregate popularity of the songs.

Figure 1 shows information learned and privacy loss for any user when we release the anonymized listening history of the t most popular songs (for $t \in \{1, \cdots, 20\}$). We can see that in this case it is possible to safely release the anonymized dataset with high information learned and little privacy loss. This is not really surprising because here we have assumed our prior information is the same for all users. Obviously this is not a practical assumption, but if we do not make such assumptions it is difficult or impossible for us to prove privacy.

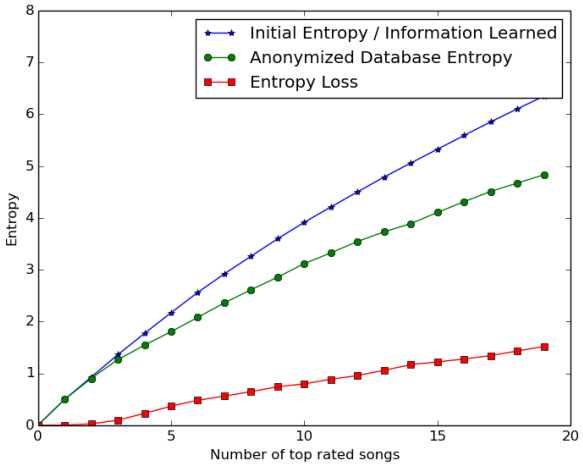

Fig. 1: Privacy in the anonymized million songs dataset assuming average song ratings are the only available prior

Realistic Prior - not identical for all users: Finding a closed form expression and precise evaluation of user entropy given a anonymized database release is a hard task in general, especially when different users have different priors. At first we consider a simple use case: we have n-users, who we denote as $u_1, \cdots, u_n$. The released database contains a single bit s_i for each user $i \in \{1, \cdots, n\}$. Let $\tilde{p}_i$ be the probability that the bit s_i is 1.

Suppose the released database contains t many 1's, i.e. $|\{i \in \{1, \cdots, n\} : s_i = 1\}| = t$. Now, if we have identical priors for all users, i.e. $\tilde{p}_i = p$ the left over entropy can simply be evaluated as

$$(t/n)\log(t/n) + (1 - t/n)\log(1 - t/n).$$

In a more complex case, where users belong to two groups (of size k and $n-k$) with identical priors within each group, the leftover entropy can be evaluated as follows. Suppose $\tilde{p}_i = p_0$ if $i \in A \subseteq \{1, \cdots, n\}$ and $\tilde{p}_i = p_1$ otherwise. We also have $|A| = k$. For any user u_i in the first group, i.e. $i \in A$, the leftover entropy is $q \log q + (1-q)\log(1-q)$ where

$$q = \frac{p_0 \left(\sum_{t'=1}^{t} \binom{k}{t'-1} p_0^{t'-1}(1-p_0)^{k-t'+1} + \binom{n-k}{t-t'} p_1^{t-t'}(1-p_1)^{n-k-t+t'} \right)}{\sum_{t'=0}^{t} \binom{k}{t'} p_0^{t'}(1-p_0)^{k-t'} \binom{n-k}{t-t'} p_1^{t-t'}(1-p_1)^{n-k-t+t'}}$$

The above expression can be evaluated efficiently in polynomial time. In the most general case when $\tilde{p}_i$ is different for each user, leftover entropy for user u_i is $q_i \log q_i + (1 - q_i) \log q_i$ where

$$q_i = \frac{\tilde{p}_i \Big(\sum_{\substack{S \subseteq \{1,\cdots,n\} \setminus \{i\} \\ |S|=t-1}} \prod_{j \in S} \tilde{p}_j \prod_{j \in \{1,\ldots,n\} \setminus S \setminus \{i\}} (1 - \tilde{p}_j) \Big)}{\sum_{\substack{S \subseteq \{1,\cdots,n\} \\ |S|=t}} \prod_{j \in S} \tilde{p}_j \prod_{j \in \{1,\ldots,n\} \setminus S} (1 - \tilde{p}_j)}$$

The naive computation of the above term is not possible in polynomial time. The numerator itself requires $\binom{n}{t}$ many term computations for arbitrary sets, although it can be done efficiently using a recursion. Moreover, the above expression holds only for the case when the anonymized dataset contains a single bit.

For a more general and practical use case where we will be releasing multiple bits, the problem quickly becomes intractable. As we have discussed in Section IV, this problem is infeasible in general. However, if the number of released bits is small (constant or logarithmic in terms of number of users), we might be able to prove that the privacy loss is small. An interesting open problem is finding an efficient approximation of the left over entropy for some realistic set of priors which is of practical importance.

VI. Conclusion

We had two goals when we began writing this paper: we wanted to first formalize the notion of privacy in a modern online advertising ecosystem, and then we wanted to develop provably secure privacy-preserving mechanisms for such an environment.

For the first goal, we discussed why existing notions fall short of what we would hope to achieve in terms of expectations of privacy in an online ad ecosystem. We then developed our own definition of privacy for such a model, and justified how our notion enjoys a better alignment with our intuition about what privacy should actually mean.

Disappointingly, we observed that real world data does not easily render itself to sanitization mechanisms that preserve our notion of privacy while at the same time retain useful information. We showed some theoretical bounds that indicated privacy in real world situations might be difficult or impossible to obtain. We provided a number of characteristics of real world data and desirable utilities that conspire to undermine the practicality of our definition. We also described how hypothetical datasets could be constructed which provide a positive case.

At this point we are at a crossroads. Our approach to solving privacy in a targeted advertising environment seems to have bogged down, but the problem itself is important and is not going away. Targeted ads, user data collection, and everything else that comes with the ecosystem will not go away. If there are no privacy-preserving techniques developed, then privacy will be lost. As such, we think this is an important topic to research even if it is quite difficult to find solutions.

We have several potential thoughts and conclusions that could be drawn with more research. First, traditional privacy might be impossible for what we perceive 'privacy' to be. Second, there might be a better trust model which we have not considered. Third, perhaps our problems could be solved by using client side computation and potentially 'heavy' cryptography. Finally, it might be the case that better machine learning (or column compression) can go a long way towards solving our problems. We discuss these ideas in more detail in the full version of the paper, and encourage readers to think over these themselves and develop next steps for attacking this difficult problem.

References

[1] L. Babai. Graph isomorphism in quasipolynomial time [extended abstract]. In *Proceedings of the 48th Annual ACM SIGACT Symposium on Theory of Computing, STOC 2016, Cambridge, MA, USA, June 18-21, 2016*, pages 684–697, 2016.

[2] B. Barak, K. Chaudhuri, C. Dwork, S. Kale, F. McSherry, and K. Talwar. Privacy, accuracy, and consistency too: a holistic solution to contingency table release. In *Proceedings of the twenty-sixth ACM SIGMOD-SIGACT-SIGART symposium on Principles of database systems*, pages 273–282. ACM, 2007.

[3] S. Berthold and R. Böhme. Valuating privacy with option pricing theory. In *WEIS*, 2009.

[4] A. Blum, K. Ligett, and A. Roth. A learning theory approach to noninteractive database privacy. *Journal of the ACM (JACM)*, 60(2):12, 2013.

[5] J. Bohli and A. Pashalidis. Relations among privacy notions. *ACM Trans. Inf. Syst. Secur.*, 14(1):4:1–4:24, 2011.

[6] S. Clauß and S. Schiffner. Structuring anonymity metrics. In *Digital Identity Management*, pages 55–62. ACM, 2006.

[7] A. Datta, D. Sharma, and A. Sinha. Provable de-anonymization of large datasets with sparse dimensions. In *International Conference on Principles of Security and Trust*, pages 229–248. Springer, 2012.

[8] C. Díaz, S. Seys, J. Claessens, and B. Preneel. Towards measuring anonymity. In *Privacy Enhancing Technologies, Second International Workshop, PET 2002, San Francisco, CA, USA, April 14-15, 2002, Revised Papers*, pages 54–68, 2002.

[9] C. Dwork. Differential privacy. In *Automata, languages and programming*, pages 1–12. Springer, 2006.

[10] C. Dwork, M. Naor, O. Reingold, G. N. Rothblum, and S. Vadhan. On the complexity of differentially private data release: efficient algorithms and hardness results. In *Proceedings of the 41st annual ACM symposium on Theory of computing*, pages 381–390. ACM, 2009.

[11] C. Dwork and K. Nissim. Privacy-preserving datamining on vertically partitioned databases. In *Advances in Cryptology–CRYPTO 2004*, pages 528–544. Springer, 2004.

[12] C. Fang and E. Chang. Information leakage in optimal anonymized and diversified data. In *Information Hiding*, volume 5284 of *Lecture Notes in Computer Science*, pages 30–44. Springer, 2008.

[13] C. R. Givens, R. M. Shortt, et al. A class of wasserstein metrics for probability distributions. *The Michigan Mathematical Journal*, 31(2):231–240, 1984.

[14] D. Kifer and A. Machanavajjhala. A rigorous and customizable framework for privacy. In *Proceedings of the 31st ACM SIGMOD-SIGACT-SIGAI symposium on Principles of Database Systems*, pages 77–88. ACM, 2012.

[15] D. Kifer and A. Machanavajjhala. Pufferfish: A framework for mathematical privacy definitions. *ACM Trans. Database Syst.*, 39(1):3, 2014.

[16] S. Kullback and R. A. Leibler. On information and sufficiency. *The annals of mathematical statistics*, 22(1):79–86, 1951.

[17] N. Li, T. Li, and S. Venkatasubramanian. t-closeness: Privacy beyond k-anonymity and l-diversity. In *2007 IEEE 23rd International Conference on Data Engineering*, pages 106–115. IEEE, 2007.

[18] N. Li, W. H. Qardaji, D. Su, Y. Wu, and W. Yang. Membership privacy: a unifying framework for privacy definitions. In *ACM Conference on Computer and Communications Security*, pages 889–900. ACM, 2013.

[19] A. Machanavajjhala, D. Kifer, J. Gehrke, and M. Venkitasubramaniam. l-diversity: Privacy beyond k-anonymity. *ACM Transactions on Knowledge Discovery from Data (TKDD)*, 1(1):3, 2007.

[20] B. McFee, T. Bertin-Mahieux, D. P. Ellis, and G. R. Lanckriet. The million song dataset challenge. In *Proceedings of the 21st International Conference on World Wide Web*, pages 909–916. ACM, 2012.

[21] A. Narayanan, E. Shi, and B. I. P. Rubinstein. Link prediction by de-anonymization: How we won the kaggle social network challenge. In *The 2011 International Joint Conference on Neural Networks, IJCNN 2011, San Jose, California, USA, July 31 - August 5, 2011*, pages 1825–1834, 2011.

[22] A. Narayanan and V. Shmatikov. Robust de-anonymization of large sparse datasets. In *2008 IEEE Symposium on Security and Privacy (sp 2008)*, pages 111–125. IEEE, 2008.

[23] P. Pedarsani and M. Grossglauser. On the privacy of anonymized networks. In *Proceedings of the 17th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, San Diego, CA, USA, August 21-24, 2011*, pages 1235–1243, 2011.

[24] A. Pfitzmann and M. Köhntopp. Anonymity, unobservability, and pseudonymity - A proposal for terminology. In *Designing Privacy Enhancing Technologies, International Workshop on Design Issues in Anonymity and Unobservability, Berkeley, CA, USA, July 25-26, 2000, Proceedings*, pages 1–9, 2000.

[25] Y. Rubner, C. Tomasi, and L. J. Guibas. The earth mover's distance as a metric for image retrieval. *International journal of computer vision*, 40(2):99–121, 2000.

[26] P. Samarati and L. Sweeney. Protecting privacy when disclosing information: k-anonymity and its enforcement through generalization and suppression. Technical report, Technical report, SRI International, 1998.

[27] A. Serjantov and G. Danezis. Towards an information theoretic metric for anonymity. In *Privacy Enhancing Technologies, Second International Workshop, PET 2002, San Francisco, CA, USA, April 14-15, 2002, Revised Papers*, pages 41–53, 2002.

[28] S. Steinbrecher and S. Köpsell. Modelling unlinkability. In *Privacy Enhancing Technologies*, volume 2760 of *Lecture Notes in Computer Science*, pages 32–47. Springer, 2003.

[29] L. Sweeney. k-anonymity: A model for protecting privacy. *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems*, 10(05):557–570, 2002.

[30] G. Tóth, Z. Hornák, and F. Vajda. Measuring anonymity revisited. In *Proceedings of the Ninth Nordic Workshop on Secure IT Systems*, pages 85–90, 2004.

[31] V. Toubiana, A. Narayanan, D. Boneh, H. Nissenbaum, and S. Barocas. Adnostic: Privacy preserving targeted advertising. In *Proceedings of the Network and Distributed System Security Symposium, NDSS 2010, San Diego, California, USA, 28th February - 3rd March 2010*, 2010.

[32] Y. Tsukada, K. Mano, H. Sakurada, and Y. Kawabe. Anonymity, privacy, onymity, and identity: A modal logic approach. *Trans. Data Privacy*, 3(3):177–198, 2010.