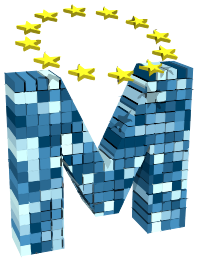


Project title: Multi-Owner data Sharing for Analytics and Integration respecting Confidentiality and OWNer control
Project acronym: MOSAICrOWN
Funding scheme: H2020-ICT-2018-2
Topic: ICT-13-2018-2019
Project duration: January 2019 – December 2021

D5.2
First Report on Privacy Metrics and Data Sanitisation

Editors: Daniel Bernau (SAP SE)
Giovanni Livraga (UNIMI)
Reviewers: Megan Wolf (MC)
Stefano Paraboschi (UNIBG)

Abstract

It is a societal challenge to balance spurring economic growth and preserving personal privacy in the context of (big) data markets. Within this document we lay out the technical challenges in anonymization and present anonymization methods that provide privacy parameters to balance privacy versus utility.

We conclude that the essential concepts for strong anonymization are available. However, the main challenge lies in supporting data owners with their choice of privacy parameters, and data analysts in formulating anonymized, yet utility preserving, data analytics functions.

Type	Identifier	Dissemination	Date
Deliverable	D5.2	Public	2020.06.30

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 825333.

MOSAICrOWN Consortium

1.	Università degli Studi di Milano	UNIMI	Italy
2.	EMC Information Systems International	EISI	Ireland
3.	Mastercard Europe	MC	Belgium
4.	SAP SE	SAP SE	Germany
5.	Università degli Studi di Bergamo	UNIBG	Italy
6.	GEIE ERCIM (Host of the W3C)	W3C	France

Versions

Version	Date	Description
0.1	2020.06.04	Initial Release
0.2	2020.06.25	Second Release
1.0	2020.06.30	Final Release

List of Contributors

This document contains contributions from different MOSAICrOWN partners. Contributors for the chapters of this deliverable are presented in the following table.

Chapter	Author(s)
Executive Summary	Daniel Bernau (SAP SE)
Chapter 1: Introduction	Daniel Bernau, Jonas Böhler (SAP SE)
Chapter 2: Sanitization of Data	Giovanni Livraga (UNIMI), Daniel Bernau (SAP SE)
Chapter 3: Anonymization of Data	Giovanni Livraga (UNIMI), Daniel Bernau (SAP SE)
Chapter 4: Quantifying Privacy for Machine Learning with Membership Inference	Daniel Bernau (SAP SE)
Chapter 5: Conclusions	Daniel Bernau (SAP SE)

Contents

List of Figures

Executive Summary

Ensuring effective protection to a dataset in terms of guaranteeing proper anonymization to its informative content is a complex problem. Whatever the specific disclosure risk that should be counteracted, the scientific community has devoted effort to come up with effective *protection techniques* throughout the past. These techniques led to two interpretations of privacy: *syntactic* and *semantic*. While syntactic privacy demands that, for example, each release of data must be indistinguishably related to no less than a certain number of individuals in the population, semantic privacy aims at satisfying a property of the mechanism chosen for releasing the data. We will discuss a line of approaches for both concepts. For syntactic privacy, k-anonymity, ℓ-diversity and t-closeness will be outlined. For semantic privacy, ε-differential privacy, (ε, δ)-differential privacy and ε-local differential privacy will be outlined. We note that syntactic privacy algorithms have interpretable privacy guarantees and have seen wide adoption for anonymized microdata release. However, they consider specific aspects of the problem and hence can remain vulnerable to possible attacks, and can have limited applicability to high-dimensional data. Algorithms for semantic privacy have seen wide adoption for perturbation of statistical functions. Their mathematically strict privacy guarantees are, however, comparatively hard to interpret. Recent studies have pointed out that both approaches are reasonable, successfully applicable to different scenarios, and there is room for both of them, possibly jointly adopted.

We pay special attention to privacy in the context of machine learning, due to the ubiquity of machine learning in modern software applications, which require massive amounts of training data. We consider machine learning architectures for classification and generative models, and quantify privacy in the context of machine learning via a threat model based on membership inference attacks (aiming to identify who might be contained in the training data).

We see several promising research directions: on the one hand, supporting data owners in choosing privacy parameters. For example, by providing translating and interpreting abstract privacy parameters such as ε into concrete risk probabilities for identity disclosure. On the other hand, mitigating liability risks for data analysts when handling sensitive or personal information by the complementary use of encryption for information security and anonymization for data privacy.

1. Introduction

The online collection of big data, such as people's behaviors, is becoming a major driver of the digital economy. Data and its analysis form the resources of tomorrow's economy as businesses that have collected such massive amounts of data are actively looking for ways of monetizing it. However, while there are great economic opportunities there are also societal risks. Big data collection and analysis may allow sensitive inferences about people's life. For example, genomic or health data which are major drivers of big data may allow inferences about disposition to certain illnesses or personality traits. Future employers could leverage that information to deny access to certain career paths. Even shopping data may reveal such sensitive health-care related information as the case of Target's advertising shows [Hil12]. Hence, it is a societal challenge to balance these objectives of spurring economic growth and preserving personal privacy. Solutions include a variety of approaches from self-controlling and privacy-respecting behavior, to legal regulation and technical protection means. No single solution can work by itself and any technical approach needs to integrate into the legal framework. In particular, the EU data protection regulation includes the categories of personal, pseudonymized and anonymized data. Personal (and pseudonymized) data may only be used for the purpose it has been collected for and if such data are used for other purposes - which may relate to monetization - the data needs to be anonymized. Anonymization means that no re-identification is possible without the original dataset. This proves to be a challenging technical task. Especially, since data may have inherent patterns that remain over time. Hence a small de-anonymized sample may suffice to re-identify entire anonymized datasets. An example attack of this kind is the re-identification of smart meter data [JJR11]. In consequence of these challenges almost no data can be left unmodified for proper anonymization. The idea of sharing sensitive, unmodified data are thus essentially challenged. The goal of MOSAICrOWN is to provide a set of functionalities for the data owners to apply anonymization with measurable and reliable guarantees. As such, the envisioned MOSAICrOWN set of functionalities are anonymization methods that provide a privacy parameter for each method that can be appropriately set balancing privacy versus utility. Hence, the MOSAICrOWN project aims to enable the sharing of personal or sensitive big data sources, i.e., preserving sufficient utility, while also preserving the privacy of the data.

Furthermore, due to the ubiquity of machine learning (ML) in modern software applications, we pay special attention to privacy in the context of ML, which requires large amount of training data. The collection of sufficient training data, to satisfy model generalization and provide meaningful utility, has proven difficult and resulted, in some cases, in privacy violations (e.g., [The17, HMDD19]). We consider two ML architectures that have been investigated for use in data markets: feedforward neural networks for classification and generative models[1] for data generation. For each architecture, we discuss how privacy can be quantified via membership inference

[1] Generative models are ML models that are trained to learn the joint probability distribution $p(X,Y)$ of features X and labels Y of training data.

attacks – which aim to discover who (an individual or group of individuals) is contained in the training data – and mitigation techniques for such threats.

In this document, we will outline the specific technical challenges in anonymization and interpretations of the privacy parameters. Furthermore, we detail privacy threat models for machine learning in the form of membership inference attacks. In the remainder of this section we detail state of the art and innovations produced by MOSAICrOWN. In Chapter 2 we formalize the concept of anonymity (Section 2.1) and related protection techniques (Section 2.2). We then discuss syntactic and semantic privacy interpretations, and introduce technical means for enforcement (e.g., k-anonymity, ε-differential privacy) in Chapter 3. In Chapter 4 we quantify privacy in the context of machine learning by defining threat models and mitigation techniques. We conclude in Section 5.

1.1 State of the Art and MOSAICrOWN Innovation

In this section, we summarize the state of the art for privacy metrics and data sanitization and the innovation produced by MOSAICrOWN.

1.1.1 State of the Art

As studied in the document, many anonymization approaches have been proposed. However, none can provide a one-size-fits-all solution for all privacy protection and utility needs.

Anonymization approaches that follow a *syntactic privacy* interpretation [Sam01, BA05, LDR05, LDR06, DT05b, SDSM14, MKGV07, XT06, CDF+12, DFJ+14, DFJ+15, DFJ+10, LLV07] are typically enforced via *generalization* (e.g., replacing precise values with a coarser ones) and *suppression* (e.g., removing identifiers). Such approaches preserve data truthfulness (i.e., do not directly alter the data) but sacrifice data completeness (due to coarser or removed values).

Approaches following a *semantic privacy* interpretation [Dwo06, MT07, War65, EPK14, ACG+16, Mir17, KLN+08, WBLJ17] typically use *perturbation* (e.g., additive noise). Such methods directly alter the data by modifying its informative content, thus, do not preserve data truthfulness, hence, the modification mechanisms have to be carefully fine-tuned to allow statistical inference on the modified data.

1.1.2 MOSAICrOWN Innovation

The innovation produced by MOSAICrOWN regarding privacy metrics and data sanitization are discussed in this section.

- The first innovation is given in the form of the analysis of the concept of anonymization and related protection techniques in Chapter 2, namely *non-perturbative techniques*, that do not alter the data but remove details (e.g., generalization), and *perturbative techniques*, that alter the data (e.g., by adding noise). Based on this analysis, we discuss syntactic and semantic privacy interpretations and protection techniques in Chapter 3.

- The second innovation consists in the quantification of privacy for machine learning with membership inference in Chapter 4. Machine learning requires large amounts of (sensitive) training data and membership inference attacks aim to infer whether an individual, or a set of individuals, belong to a training dataset. We present and discuss mitigations and privacy

parameter selections with regards to membership inference via semantic techniques during machine learning training.

2. Sanitization of Data

Protecting the privacy of the individuals to whom a dataset refers requires sanitization the dataset, meaning modifications to ensure the inability to link an individual to their information. A possible approach for sanitization is to anonymize the dataset, which has been proven to be a complex task. In this chapter, we illustrate the main issue characterizing anonymization of data (Section 2.1), and present some data protection techniques that can be adopted (Section 2.2).

2.1 The Anonymity Problem

The problem of data anonymization has been heavily investigated in the context of microdata release, where datasets are represented as relational tables with one record for each individual (called *respondent*), and one column for each attribute related to the respondents (e.g., name, date of birth, job, etc.). The first step for protecting the privacy of a dataset requires is removing (e.g., by deleting or encrypting) any identifying attributes, such as names, e-mail addresses, or unique identifiers (such as the social security number). This process, usually referred to as *de-identification*, is unfortunately not enough to ensure *anonymity* to the data. In fact, a de-identified dataset can still include other information, called *quasi-identifiers* (QI), which can be linked to external sources to reduce the uncertainty about the identity of some respondents [DFLS12]. Based on a study performed on the US 2000 Census, Golle discovered that 63% of the entire US population is *uniquely identifiable* by a combination of their gender, ZIP code, and full date of birth [Gol06]. The following example illustrates such re-identification risks. Consider Figure 2.1(a), illustrating a de-identified dataset containing information for a set of hospitalized patients. Figure 2.1(b) illustrates a sample excerpt of a fictitious publicly available voter list of a New York City municipality. It is easy to see that it is possible to exploit attributes `DoB`, `Sex`, and `ZIP` for linking the two datasets, possibly re-identifying (with either full confidence or a certain probability) some of the de-identified respondents in Figure 2.1(a). For instance, the de-identified dataset includes only a female respondent, born in 1958/12/11 and living in the 10180 area (record 11): if this combination of quasi-identifying values is unique in the external world as well, then the voter list can be exploited to uniquely re-identify the eleventh record with respondent *Kathy Doe*, also disclosing the fact that she has been hospitalized for *epilepsy*. Considering that tremendous amounts of data are generated and shared every day, the availability of non de-identified datasets that can be used for linking is a realistic threat. Famous incidents that gained headlines in the news include the Netflix re-identification [NS08], where a de-identified set of Netflix recommendations has been re-identified using the public IMDB recommendations, or the re-identification of credit card data [dMRSP15], where cardholders could be re-identified given a few sample purchases. Unfortunately, unlike direct identifiers, removing QI information might not be a feasible strategy, since QI can represent a large portion of the informative content of a dataset. Therefore, its complete removal risks reducing data utility (e.g., removing also quasi-identifiers from the de-identified dataset in Figure 2.1(a) would leave only a list of diseases, most likely of limited interest to final recipients).

SSN	Name	DoB	Sex	ZIP	Disease
		1960/05/02	F	10041	stroke
		1960/05/20	M	10032	dyspepsia
		1960/05/12	M	10037	achlorhydria
		1960/05/05	F	10044	epilepsy
		1955/09/01	M	10043	helicobacter
		1955/09/02	M	10042	helicobacter
		1955/09/10	F	10039	helicobacter
		1955/09/20	F	10030	helicobacter
		1955/12/07	M	10030	dermatitis
		1955/12/05	M	10031	retinitis
		1958/12/11	*F*	*10180*	epilepsy
		1955/12/25	F	10042	dermatitis
		1955/12/30	F	10045	gastritis
		1960/04/02	F	10036	stroke
		1960/04/05	F	10034	labyrinthitis
		1960/04/10	M	10047	gastritis
		1960/04/30	M	10048	dyspepsia

(a)

Name	Address	City	ZIP	DoB	Sex	Education
...	...	...	...	...	...	...
Kathy Doe	300 Main St.	New York City	*10180*	*58/12/11*	*female*	secondary
...	...	...	...	...	...	...

(b)

Figure 2.1: An example of de-identified medical dataset (a) and of publicly available non de-identified dataset (b)

Given a de-identified dataset, two main kinds of improper disclosure can occur [Fed05].

- *Identity disclosure*, occurring whenever the identity of a respondent can be somehow determined and associated with a record in the de-identified dataset;
- *Attribute disclosure*, occurring when a (sensitive) value can be associated with an individual (without necessarily being able to link the value to a specific record).

There are several factors that can contribute to (or, conversely, reduce) the risks of identity and attribute disclosure [Fed05]. For instance, the existence of high-visibility records, assuming uncommon values for certain attributes (e.g., a very high income, or a rare disease, or a very uncommon job) that can make these records stand out from other ones. Similarly, the more the common attributes between the dataset and the external source of information (and the more external sources as well), the higher the disclosure risks. By contrast, the natural noise characterizing the dataset and the external sources, the presence of data that might not be completely up-to-date or that refer to different temporal intervals, and the use of different formats for representing the information in the dataset and in the external sources can contribute to decreasing the disclosure risks.

2.2 Protection Techniques

Whatever the specific disclosure risk that should be counteracted, the scientific community has devoted major efforts to come up with effective *protection techniques* [CDFS07]. A first distinction

can be made between *masking techniques* and *synthetic data generation techniques*: while these latter ones aim at producing a new, synthetic dataset that maintains some statistical properties of the original data (and can then be safely released or published instead of the original one), the former operate directly on the original data, to sanitize them before releasing. Masking techniques can be classified based on how they operate on the original data, as follows.

- *Non-perturbative techniques* do not directly modify the original data, but remove details from the dataset. Such techniques preserve data truthfulness, sacrificing data completeness by producing imprecise and/or incomplete data. Examples of non-perturbative techniques include *sampling*, *suppression*, *generalization*, *global recoding*, and *bucketization*. Sampling consists in releasing data that are related to a subset of the original population. Protection is provided by the uncertainty about the presence in the dataset of the information about a specific respondent. Suppression selectively removes information from the dataset (for instance, direct identifiers are typically suppressed before release, as discussed above). Generalization selectively replaces some values in the dataset with more general ones: for instance, a complete date of birth can be generalized by releasing only the month and year, or the year of birth. A possible way to enforce generalization is based on the definition of generalization hierarchies, identifying the possible generalized values that can be used to replace more specific ones. Global recoding, which can be seen as a particular kind of generalization, partitions the set of values that can be assumed by an attribute into disjoint intervals, usually of the same width, and associates a label with each interval. Instead of releasing the original values, the labels of the corresponding intervals are published. Two examples of global recoding techniques, specifically designed for numerical attributes, are *top-coding* and *bottom-coding*. Top-coding replaces all values that are above a certain threshold with a given label (e.g., high incomes over 1 million dollars are are replaced by label "$>$1M"). Bottom-coding substitutes the values under a given threshold with a given label (e.g., low incomes less than 50 thousand dollars are replaced by label "$<$50K"). Bucketization operates on sets of attributes whose joint visibility should be prevented (e.g., the name and the disease of a patient), and operates by first partitioning records in buckets and attributes in groups, and then shuffling the partitioned records within buckets so to break their correspondence [DFJ$^+$15, DFJ$^+$10, LLZM12, XT06].

- *Perturbative techniques* distort a dataset by modifying its informative content. Such techniques do not preserve data truthfulness, and hence modifications should be reduced and not compromise the possibility of correctly performing analysis (i.e., the results of the analysis carried out on the perturbed data should not significantly differ from those computed on the original one). Examples of perturbative techniques include *noise addition* and *microaggregation*. Noise addition intuitively adds non-deterministic, controlled noise to the original data collection before release. Protection is provided by the fact that some values (or combinations among them) included in the released table might not correspond to real ones due to perturbation, and by the fact that some values (or combinations among them) included in the original table might not be included in the released one. The degree of noise addition (e.g., sampling from a standard normal distribution vs. sampling from a uniform distribution) is adapted to balance a trade-off between utility and data protection. Microaggregation (originally proposed for continuous numerical data and then extended also to categorical data [Tor04]) selectively replaces original records with new ones. Microaggregation operates by first clustering the records in the original dataset in groups of a certain cardinality in

such a way that records in the same cluster are similar to each other, and then by replacing the records in a cluster with a representative one computed through an aggregation operator (e.g., mean or median).

The protection techniques illustrated above can be adopted to effectively protect a dataset. A straightforward recommendation for a single technique covering all analytical requirements and any data collection is not possible, however, we detail and compare the different techniques in this Deliverable to provide guidance. Given a data collection to be protected and released, some key questions then need to be answered: what technique should be used? Should a joint combination of techniques be preferred to a single one? To which portion of the data (e.g., the entire dataset, a subset of records, a subset of attributes) should the technique be applied? Whatever the answer to these questions, an important observation is that all protection techniques cause an inevitable information loss: non-perturbative techniques produce datasets that are not as complete or precise as the original ones, and perturbative techniques produce datasets that are distorted. For these reasons, it is necessary to define protection approaches that satisfy a privacy requirement via a controlled adoption of some of these protection techniques while limiting information loss, as illustrated in the remainder of this Deliverable.

3. Anonymization of Data

As mentioned in the previous chapter, anonymization represents a possible strategy for sanitizing a dataset for protecting the privacy of the individuals to whom the dataset refers. Clearly, depending on the privacy requirement that is to be satisfied by anonymization, different interpretations of privacy can exist. A broad classification can distinguish between *syntactic* and *semantic* interpretations [DFLS12]. Syntactic interpretations capture the protection degree enjoyed by respondents with a numerical value. Related anonymization approaches aim at satisfying a syntactic privacy requirement (e.g., each release of data must be indistinguishably related to no less than a certain number of individuals in the population). On the other hand, semantic interpretations are based on the satisfaction of a semantic privacy requirement, and the related anonymization approaches aim at satisfying a property of the mechanism chosen for releasing the data (e.g., the result of an analysis carried out on a released dataset must be insensitive to the insertion or deletion of a record in the dataset). We illustrate some anonymization approaches that follow a syntactic privacy interpretation in Section 3.1 and semantic privacy interpretation in Section 3.2.

3.1 Syntactic Privacy

We now illustrate some anonymization approaches that build on, and satisfy, a syntactic definition of privacy. We start from the first proposal in this direction (k-anonymity, Section 3.1.1) and discuss some extensions (Sections 3.1.2–3.1.3).

3.1.1 k-Anonymity

The first approach for anonymizing a dataset, originally framed in the context of microdata publishing [DFLS12] and aiming to protect against identity disclosure (see Section 2.1), is represented by k-anonymity [Sam01]. k-Anonymity enforces a protection requirement typically applied by statistical agencies, which demands that any released information be *indistinguishably related* to no less than a certain number k of respondents. Following the assumption that re-identification takes advantage of the quasi-identifying attributes (see Section 2), such general requirement is translated into the k-anonymity requirement: each release of data must be such that every *combination of values of quasi-identifiers* can be indistinctly matched to *at least k respondents* [Sam01]. A dataset satisfies the k-anonymity requirement if and only if each record in the released dataset cannot be related to less than k individuals in the population, and vice-versa (i.e., each individual in the population cannot be related to less than k records in the dataset). These two conditions hold since the original definition of k-anonymity assumes that each respondent be represented by at most one record in the released dataset and vice-versa (i.e., each record includes information related to one respondent only). Verifying the satisfaction of the k-anonymity requirement would require to have knowledge of *all* existing external sources of information that an adversary might use for the linking attack. This assumption is indeed unrealistic in practice, and k-anonymity takes

SSN	Name	DoB	Sex	ZIP	Disease
		1960/05	*	100**	stroke
		1960/05	*	100**	dyspepsia
		1960/05	*	100**	achlorhydria
		1960/05	*	100**	epilepsy
		1955/09	*	100**	helicobacter
		1955/09	*	100**	helicobacter
		1955/09	*	100**	helicobacter
		1955/09	*	100**	helicobacter
		1955/12	*	100**	dermatitis
		1955/12	*	100**	retinitis
		1955/12	*	100**	dermatitis
		1955/12	*	100**	gastritis
		1960/04	*	100**	stroke
		1960/04	*	100**	labyrinthitis
		1960/04	*	100**	gastritis
		1960/04	*	100**	dyspepsia

Figure 3.1: An example of 4-anonymous dataset assuming QI=$\{\texttt{DoB}, \texttt{Sex}, \texttt{ZIP}\}$

therefore a safe approach requiring that each respondent be indistinguishable from at least $k-1$ other respondents in the released dataset. A dataset is then said to be k-anonymous if each combination of values of the quasi-identifier appears in it with either zero or at least k occurrences. Since each combination of quasi-identifying values is shared by at least k different records in the dataset, each respondent cannot be associated with less than k records in the released dataset and vice-versa, thus satisfying the original k-anonymity requirement. Traditional approaches to enforce k-anonymity operate on quasi-identifying attributes by modifying their values in the dataset to be released, while leaving sensitive and non-sensitive attributes as they are (let us recall that direct identifiers are removed as the first step). Among the possible data protection techniques that might be enforced on the quasi-identifier, k-anonymity typically relies on the combined adoption of *generalization* and *suppression*, which have the advantage of preserving data truthfulness when compared to perturbative techniques (e.g., noise addition, see Section 2.2). Let us recall that generalization operates by replacing values with more general ones (e.g., a complete date of birth might be generalized to the year of birth), while suppression operates by selectively removing values. Suppression is used to couple generalization as it can help in reducing the amount of generalization to be enforced for achieving k-anonymity. This way, it is possible to produce more precise (though incomplete) datasets. The intuitive rationale is that if a dataset includes a limited number of outliers (i.e., quasi-identifying values with less than k occurrences) that would force a large amount of generalization to satisfy k-anonymity, then these outliers could be more conveniently removed from the dataset, improving the quality of released data. For instance, consider the dataset in Figure 2.1(a) and assume that the quasi-identifier is composed of attribute ZIP only. Since there is only one person living in 10180 area (11th record), to achieve k-anonymity with $k > 1$ attribute ZIP should be generalized removing the last three digits. However, if the 11th record in the dataset is suppressed, 8-anonymity can be achieved by generalizing the ZIP code removing only the last digit.

Generalization and suppression can be applied at different granularity levels. For instance, with reference to relational tables, generalization can be applied at the cell and attribute levels, and suppression at the cell, attribute, and record levels. The combined use of generalization and suppression at different granularity levels produces different classes of approaches to enforce k-anonymity [CDFS07]. The majority of the approaches available in the literature adopt attribute-level generalization and record-level suppression [BA05, LDR05, Sam01]. Figure 3.1 illustrates

Age	Disease
20	flu
25	gastritis
30	dermatitis
35	stroke
40	dyspepsia
45	asthma

(a)

Age	Disease
25	flu
25	gastritis
25	dermatitis
40	stroke
40	dyspepsia
40	asthma

(b)

Figure 3.2: An example of a dataset (a) and of a 3-anonymous version of it (b) obtained adopting microaggregation and assuming QI={`Age`}

a 4-anonymous version of the dataset in Figure 2.1(a), obtained through attribute-level generalization (`DoB`, `Sex`, and `ZIP` have been generalized by removing the day of birth, sex, and last two digits of the ZIP code, respectively) and record-level suppression (the 11th record related to *Kathy* has been suppressed). Symbol $*$ represents any value in the attribute domain. Cell-level generalization has also been investigated as an approach to produce k-anonymous datasets, as it has been shown to reduce the information loss with respect to attribute-level generalization [LDR06]. These approaches have however the drawback of producing datasets where the values in the cells of the same column may be heterogeneous (e.g., some records report the complete date of birth, while other records only report the year of birth).

Regardless of the different level at which generalization and suppression are applied, they inevitably cause a certain amount of information loss (the original informative content is either reduced in the details or removed), and it is therefore essential to compute a k-anonymous dataset that, while protecting users' privacy, is still useful to the recipients. To this aim, it is necessary to compute an optimal k-anonymization minimizing generalization and suppression, which has been shown to be an NP-hard problem [CDFS07], and both exact and heuristic algorithms have been proposed.

As a last remark on k-anonymity, we note that generalization, while having the advantage of preserving data truthfulness, can face scalability issues especially for high-dimensional data, where generalization might need to cover a high number of dimensions [Agg05]. Some recent approaches have been proposed to obtain k-anonymity through microaggregation instead of generalization (see Section 2.2) [DT05b, SDSM14]. To this aim, the QI undergoes microaggregation, so that each combination of QI values in the original dataset is replaced with a microaggregated version. For instance, consider the dataset in Figure 3.2(a) and suppose that the quasi-identifier includes attribute `Age`, while `Disease` is the sensitive attribute. Figure 3.2(b) illustrates a 3-anonymous version of the dataset in Figure 3.2(a) obtained through microaggregation: the original QI values have been grouped in two clusters (the first three records in one, and the last three records in the other one) according to their similarity (in this example, according to an ordering over them), and the values in each cluster are then replaced with a representative aggregate value (in this example, the mean). Note that, since microaggregation is a perturbative protection technique, k-anonymous datasets computed adopting this approach do not preserve data truthfulness (e.g., all records in Figure 3.2(b) but the second and the fifth are not real records according to the original values in Figure 3.2(a)).

3.1.2 ℓ-Diversity

While k-anonymity represents an effective solution to protect respondent identities, it does not protect against attribute disclosure [Sam01]. A k-anonymous dataset can, in fact, still be vulnerable to attacks allowing a recipient to determine (with either full confidence or non-negligible probability) the sensitive information of a respondent. In particular, two attacks that may cause attribute disclosure in a k-anonymous dataset are the *homogeneity attack* [MKGV07, Sam01] and the *external knowledge attack* [MKGV07], as follows.

- *Homogeneity attack*. The homogeneity attack occurs when all the records in an equivalence class (i.e., the set of records with the same value for the quasi-identifier) in a k-anonymous dataset assume the same value for the sensitive attribute. If a data recipient knows the quasi-identifier value of a target individual, she can identify the equivalence class representing her, and then discover the value of her sensitive attribute. For instance, consider the 4-anonymous dataset in Figure 3.1 and suppose that a recipient knows that *Gloria* is a female living in 10039 area and born on 1955/09/10. Since all the records in the equivalence class with quasi-identifier value equal to $\langle 1955/09, *, 100** \rangle$ assume value *helicobacter* for attribute `Disease`, the recipient can infer that she suffers from a *helicobacter* infection.

- *External knowledge attack*. The external knowledge attack occurs when the data recipient possesses some additional (not included in the k-anonymous dataset) knowledge about the respondent, and can use it to reduce her uncertainty about the value of the sensitive attribute of a target respondent. For instance, consider the 4-anonymous dataset in Figure 3.1 and suppose that a recipient knows that her neighbor *Mina* is a female living in 10045 area and born on 1955/12/30. Observing the 4-anonymous dataset, the recipient can only infer that her neighbor suffers from *dermatitis*, *retinitis*, or *gastritis*. Suppose now that the recipient sees *Mina* tanning without screens at the park every day: due to this external information, the recipient can exclude that *Mina* suffers from *dermatitis* or *retinitis*, discovering that she suffers from *gastritis*.

The original definition of k-anonymity has been extended to ℓ-diversity to the aim of counteracting these two attacks. The idea behind ℓ-diversity is to consider also the values of the sensitive attributes when clustering the original records, so that at least ℓ *well-represented* values for the sensitive attribute be included in each equivalence class [MKGV07]. While several definitions for "well-represented" values have been proposed, the simplest formulation of ℓ-diversity requires that each equivalence class be associated with at least ℓ different values for the sensitive attribute. For instance, consider the 4-anonymous and 3-diverse dataset in Figure 3.3 and suppose that a recipient knows that her neighbor *Mina*, a female living in 10045 area and born on 1955/12/30, tans every day at the park (see example above). The recipient can now only exclude value *dermatitis*, but she cannot be sure about whether *Mina* suffers from *gastritis* or a *helicobacter* infection.

The problem of computing an ℓ-diverse dataset minimizing the loss of information caused by generalization and suppression is computationally hard. However, since ℓ-diversity basically requires to compute a k-anonymous dataset (with additional constraints on the sensitive values), any algorithm proposed to compute a k-anonymous dataset that minimizes loss of information can be adapted to guarantee also ℓ-diversity, simply controlling if the condition on the diversity of the sensitive attribute values be satisfied by all the equivalence classes [MKGV07].

As a last remark on ℓ-diversity, we underline an approach for its enforcement that departs from generalization, adopting instead a bucketization-based approach (see Section 2.2). For in-

SSN	Name	DoB	Sex	ZIP	Disease
		1955	M	100**	helicobacter
		1955	M	100**	helicobacter
		1955	M	100**	dermatitis
		1955	M	100**	retinitis
		1960	F	100**	stroke
		1960	F	100**	epilepsy
		1960	F	100**	stroke
		1960	F	100**	labyrinthitis
		1955	F	100**	helicobacter
		1955	F	100**	helicobacter
		1955	F	100**	dermatitis
		1955	F	100**	gastritis
		1960	M	100**	dyspepsia
		1960	M	100**	achlorhydria
		1960	M	100**	gastritis
		1960	M	100**	dyspepsia

Figure 3.3: An example of 4-anonymous and 3-diverse dataset assuming QI=$\{\texttt{DoB}, \texttt{Sex}, \texttt{ZIP}\}$

DoB	Sex	ZIP	GroupID
1955/09/01	M	94143	G1
1955/09/02	M	94142	G1
1955/12/07	M	94130	G1
1955/12/05	M	94131	G1
1960/05/02	F	94141	G2
1960/05/05	F	94144	G2
1960/04/02	F	94136	G2
1960/04/05	F	94134	G2
1955/09/10	F	94139	G3
1955/09/20	F	94130	G3
1955/12/25	F	94142	G3
1955/12/30	F	94145	G3
1960/05/20	M	94132	G4
1960/05/12	M	94137	G4
1960/04/10	M	94147	G4
1960/04/30	M	94148	G4

GroupID	Disease	Count
G1	helicobacter	2
G1	dermatitis	1
G1	retinitis	1
G2	stroke	2
G2	epilepsy	1
G2	labyrinthitis	1
G3	helicobacter	2
G3	dermatitis	1
G3	gastritis	1
G4	dyspepsia	2
G4	achlorhydria	1
G4	gastritis	1

Figure 3.4: An example of a bucketized 3-diverse relation assuming QI=$\{\texttt{DoB}, \texttt{Sex}, \texttt{ZIP}\}$

stance, Anatomy [XT06] (but also other more general techniques that can handle the ℓ-diversity requirement [CDF$^+$12, DFJ$^+$14, DFJ$^+$15, DFJ$^+$10]) is a bucketization-based approach enforcing ℓ-diversity without relying on generalization. With this approach, the records in the original dataset are first partitioned in groups that satisfy ℓ-diversity. All buckets so created are then labeled with their own group identifier, and the original dataset is split into two fragments, in such a way that one includes the attributes composing the quasi-identifier, and the other includes that sensitive attribute. Each record is associated in each fragment with the identifier of the group to which it belongs, and each group in the fragment storing the sensitive attribute includes a record only for each sensitive value appearing in the group and the frequency with which the value is represented in the group. For instance, Figure 3.4 reports a bucketization-based 3-diverse version of the original dataset in Figure 2.1(a) computed with the Anatomy approach: it is easy to see that the protection guarantees offered by the fragments are the same as those offered by the 3-diverse dataset in Figure 3.3, computed instead through traditional generalization.

3.1.3 t-Closeness

Although ℓ-diversity represents a first step in counteracting attribute disclosure, an ℓ-diverse dataset might still be vulnerable to information leakage caused by *skewness* and *similarity attacks* [LLV07], as follows.

- *Skewness attack*. The skewness attack occurs when a data recipient can observe significant differences in the frequency distribution of the sensitive values within an equivalence class, with respect to the frequency distribution of the same values in the population (or in the whole dataset). The idea is that differences in these distributions signal changes in the probability with which a respondent in the equivalence class is associated with a specific sensitive value. As an example, consider the 3-diverse dataset in Figure 3.3 and suppose that a recipient knows that *Alice* is a female living in 10041 area and is born on 1960/05/02. Since two out of the four records in the equivalence class with quasi-identifier value $\langle 1960, F, 100*$ $*\rangle$ assume value *stroke* for attribute `Disease`, it is possible to infer that *Alice* has 50% probability of having had a stroke, compared to the 12.5% of the whole dataset.

- *Similarity attack*. The similarity attack is caused by the fact that ℓ-diversity only requires that the ℓ values in an equivalence class be sintactically similar, without constraints on their semantics. This attack occurs when, in an ℓ-diverse dataset, the sensitive values of the records in an equivalence class are semantically similar, although (as required by ℓ-diversity) syntactically different. For instance, consider the 3-diverse dataset in Figure 3.3 and suppose that a recipient knows that *Carl* is a male living in 10037 area and born on 1960/05/12. Observing the 3-diverse dataset, the recipient can infer that *Carl* suffers from either *gastritis*, *dyspepsia*, or *achlorhydria* and, therefore, that *Carl* suffers from a stomach-related disease.

The definition of t-closeness [LLV07] has been proposed to counteract these two attacks by requiring that the frequency distribution of the sensitive values in each equivalence class be close (i.e., with distance smaller than a fixed threshold t) to that in the whole dataset. Ensuring t-closeness ensures that the skewness attack has no effect, since the knowledge of the quasi-identifier value for a target respondent does not change the probability for a malicious recipient of correctly guessing the sensitive value associated with the respondent. t-Closeness also reduces the effectiveness of the similarity attack, because the presence of semantically similar values in an equivalence class can only be due to the presence of the same values in the dataset with similar relative frequencies, thus not increasing the knowledge of the recipient.

The enforcement of t-closeness requires to evaluate the distance between the frequency distribution of the sensitive attribute values in the whole dataset and in each equivalence class, and several distance metrics can be used to this aim [LLV07].

3.2 Semantic Privacy

A key insight on which semantic privacy was formulated is the impossibility proof of Dalenius desideratum, which demands *nothing about an individual should be learnable from the database that cannot be learned without access to the database* [Dwo06]. In the presence of auxiliary information, an adversary will always be capable of inferring some information about an individual in a dataset given some function result computed from this dataset. Differential privacy, a semantic privacy definition [Dwo06], provides a metric that quantifies the privacy risk incurred by participating in a dataset. Concretely, differential privacy measures how plausibly an individual can

deny membership in a dataset. In contrast to previous anonymization methods based on generalization, differential privacy achieves anonymization of a dataset $D = \{d_1, \ldots, d_n\}$ by perturbation. Thus, differential privacy does not provide truthfulness in comparison to the previously introduced concepts for generalization.

Differential privacy can be enforced either locally on each entry in D, or centrally on the result of a query function $f(\cdot)$ over D. Within the next subsections we will introduce the foundations of differential privacy, and the mechanisms that achieve it.

3.2.1 ε-Differential Privacy

Differential privacy is frequently enforced in the *central setting*. The central setting comprises three actors: *Data Owner*, *Data Analyst* and *Curator*.

Here, Data Owner possesses a dataset D, from data domain DOM, containing sensitive or personally identifiable information. D shall be shared with Data Analyst such that Data Analyst is able to evaluate a query function $f(D)$. However, Data Analyst shall be prevented from learning the original result of $f(D)$. In the central model, the query function $f(\cdot)$ is thus evaluated and perturbed by Curator (trusted third party server) such that it is no longer possible to confidently determine whether $f(\cdot)$ was evaluated on D, or some neighboring dataset D' differing in one individual. *Mechanisms M* fulfilling Definition 1 are used for perturbation of $f(\cdot)$.

Definition 1 (ε-Differential Privacy [Dwo06]) *A mechanism M gives ε-Differential Privacy if for all D,D' $\subseteq$ DOM differing in at most one element, and all sets S $\subseteq$ Range(M)*

$$\Pr[M(D) \in S] \leq e^{\varepsilon} \cdot \Pr[M(D') \in S],$$

where Range(M) denotes the set of all possibles outputs of mechanism M.

Throughout MOSAICrOWN we will refer to ε as privacy parameter. While the choice and interpretation of ε depends on the concrete dataset D at hand, small ε values tend to result in high privacy, and vice versa.

Perturbation is influenced by sensitivity. According to Dwork and Roth [DR14], sensitivity can be interpreted as the maximum impact that a single individual can have on the return value of query function f. Definition 1 holds for all possible differences $|f(D) - f(D')|$ by adapting to the *global sensitivity* of $f(\cdot)$ per Definition 2. The absolute nature of global sensitivity implies that an individual's impact on the result of a query function will never be greater than Δ_f.

Definition 2 (Global Sensitivity) *Let D and D' be neighboring. The global sensitivity of a function $f(\cdot)$, denoted by Δ_f, is defined as*

$$\Delta_f = max_{D,D'} |f(D) - f(D')|.$$

In the following we will introduce two common mechanisms for adding ε-differentially private noise: the Laplace mechanism for numeric perturbation and the exponential mechanism for the perturbation of numeric and categorical values. The use of a respective mechanism is mostly motivated by the insensitivity and stability of input data in regards to noise and by the enforcement model.

The Laplace mechanism of Theorem 1 is suited for the enforcement of ε-differential privacy on numerical valued queries which provide the analyst with a real valued answer. An example for such an operation is a count query.

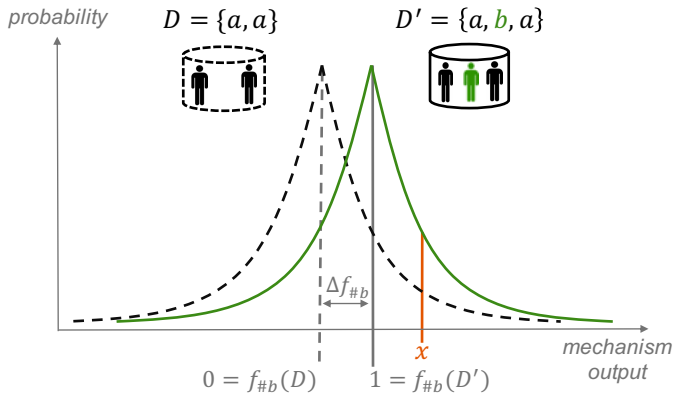


Figure 3.5: Laplace mechanism example: count query

Theorem 1 (Laplace Mechanism [DR14]) *Given a numerical query function* $f : DOM \rightarrow \mathbb{R}^k$, *the Laplace mechanism*

$$M_{Lap}(D, f, \varepsilon) := f(D) + (z_1, ..., z_k)$$

is an ε*-differentially private mechanism when all* z_i *with* $1 \leq i \leq k$ *are independently drawn from* $Z \sim Lap(z, \lambda = \frac{\Delta_f}{\varepsilon}, \mu = 0)$.

For the proof we refer to Dwork et al. [DMNS06, DR14]. As the name already indicates, the Laplace mechanism samples noise from an underlying Laplace distribution. The Laplace distribution is a symmetric exponential distribution centered around mean $\mu = 0$ with scaling factor λ. The Laplace mechanism adds noise with scale $\lambda = \Delta_f / \varepsilon$. Therefore, sensitivity Δ_f is a factor in determining how accurately queries results can be published while preserving a desired level of privacy [DR14]. The scale λ is growing (1) as the sensitivity Δ_f increases for a given privacy parameter ε or (2) as the privacy parameter ε decreases for a given sensitivity Δ_f.

We now want to discuss an example. Assume a count function $f_{\#b}(\cdot)$ for value b, original dataset $D = \{a, a\}$ and a neighboring dataset $D' = \{a, b, a\}$. Without the application of differential privacy we would observe two deterministic outputs:

$$x = \begin{cases} 0 & f_{\#b}(D) \\ 1 & f_{\#b}(D') \end{cases} \tag{3.1}$$

By using the Laplace mechanism we span a Laplace distribution around $f_{\#b}(D)$ and $f_{\#b}(D)$ and thus the function result x becomes non-deterministic as depicted in Figure 3.5.

ε-differential privacy assures that for any function result x the divergence of probability between stemming from D and D' is bounded:

$$\frac{\Pr[M_{Lap}(f_{\#b}(D')) = x]}{\Pr[M_{Lap}(f_{\#b}(D)) = x]} \leq e^{\varepsilon}$$

Besides the additive Laplace mechanism the Exponential Mechanism, provided in Definition 3, is a widely used choice for arbitrary perturbation of categorical and numerical data. This mechanism is useful in situations when adding noise to the result of a query destroys its value [DR14]. Instead, the exponential mechansim samples from a more meaningful, predefined range of possible outputs.

Definition 3 (Exponential Mechanism [LLSY16]) *For any quality function $q : \mathbb{D} \times O \to \mathbb{R}$, where $\mathbb{D}$ is the set of all possible datasets, and a privacy parameter ε, the exponential mechanism $M_q^{\varepsilon}(D)$ outputs $o \in O$ with probability proportional to $\exp\left(\frac{\varepsilon q(D,o)}{2\Delta_q}\right)$, where for all datasets D, D' differing in one record*

$$\Delta_q = \max_{\forall o \in O; D, D'} |q(D,o) - q(D',o)|$$

is the sensitivity of the quality function. That is,

$$\Pr[M_q^{\varepsilon}(D) = o] = \frac{\exp\left(\frac{\varepsilon q(D,o)}{2\Delta_q}\right)}{\sum_{o' \in O} \exp\left(\frac{\varepsilon q(D,o')}{2\Delta_q}\right)}.$$

We refer the interested reader to McSherry and Talwar [MT07] for the initial proof that the exponential mechanism satisfies ε-differential privacy. Note that the exponential mechanism allows to encode a preference towards values close to the true value through the quality function.

We now want to revisit and advance an example initially outlined by Cormode [Cor18] for the use of the exponential mechanism. Assume we want to compute the (lower) median of a sorted array of an even number unique elements:

$$D = \{1,3,5,7,8,11,12,13,17,22\}.$$

Consequently, the rank of each element in D is:

$$rank_D(\cdot) = 0,1,2,3,4,5,6,7,8,9,$$

thus, the true lower median is represented by 8. For the application of the exponential mechanism we first need to define a quality function that assign a high score to elements in D which have a rank close to the true lower median:

$$\begin{aligned} q(\cdot, D) &= \left| -rank_D(\cdot) - \frac{|D|}{2} + 1 \right|, \text{yielding} \\ &= -4,-3,-2,-1,0,-1,-2,-3,-4,-5. \end{aligned}$$

In the exponential mechanism quality function scores become weights:

$$\begin{aligned} weight(\cdot) &= \exp\left(\frac{\varepsilon}{2} q(\cdot, D)\right), \text{yielding} \\ &= e^{-2\varepsilon}, e^{-\frac{3\varepsilon}{2}}, e^{-\varepsilon}, e^{-\frac{\varepsilon}{2}}, 1, e^{-\frac{\varepsilon}{2}}, e^{-\varepsilon}, e^{-\frac{3\varepsilon}{2}}, e^{-2\varepsilon}, e^{-\frac{5\varepsilon}{2}}. \end{aligned}$$

Figure 3.6 illustrates the resulting weight distribution. Note that the distribution is centered around the element with the highest utility (i.e., the original median). The shape of this distribution yields high utility, since probability falls sharply as the quality score decreases [DR14].

The Fundamental Law of Information Recovery states that privacy cannot be guaranteed if overly accurate answers to too many questions are made public [DN03], and this disintegration of privacy must be measured. A beneficial characteristic of differential privacy is the ability to quantify the privacy loss of an individual within a dataset over a series of ε-differentially private mechanism evaluations. This characteristic is referred to as *composition*. The most basic composition theorem is sequential composition as stated in Theorem 2.

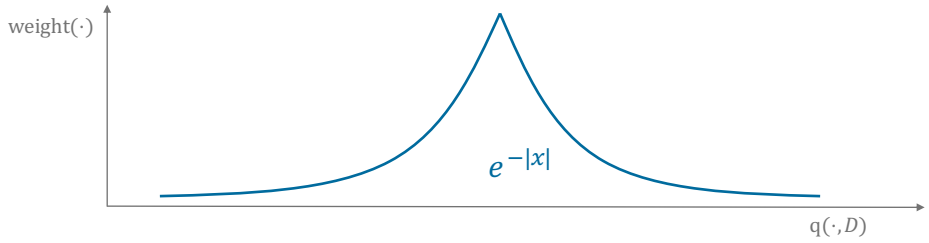


Figure 3.6: Exponential mechanism example: median query

Theorem 2 (Sequential Composition [DR14]) *Let $M_1, M_2, \ldots, M_k$ be k algorithms (that take auxiliary inputs) that satisfy ε_1-DP, ε_2-DP,..., ε_k-DP respectively, with respect to the input dataset D. Publishing*

$$t = < t_1, t_2, \ldots, t_k >, \text{ where } t_1 = M_1(D), t_2 = M_2(t_1, D), \ldots, t_k = (< t_1, \ldots, t_{k-1} >, D)$$

satisfies $(\sum_{i=1}^{k} \varepsilon_i)$-DP.

Sequential composition is a naive pessimistic view, assuming that the privacy loss (i.e., information gain of Data Analyst) is maximum in each invocation of a mechanism. There is a series of results that yield tighter bounds on the privacy loss by analyzing it as a random variable [DRV10, ACG+16, KOV17].

3.2.2 (ε, δ)-Differential Privacy

Definition 1 is very strict with respect to only allowing relative differences in probabilities of all possible outputs between a database D and any neighboring database D'. This strictness leads to the inapplicability of some statistical distributions for ε-differential privacy, and might especially have a severe effect on utility (e.g., trading in a small loss in privacy for a large gain in utility by suppressing low probability elements from the set of possible outputs).

(ε, δ)-differential privacy introduces an additional, additive privacy parameter δ besides the relative privacy parameter ε to mitigate the previous arguments. We formalize (ε, δ)-differential privacy in Definition 4.

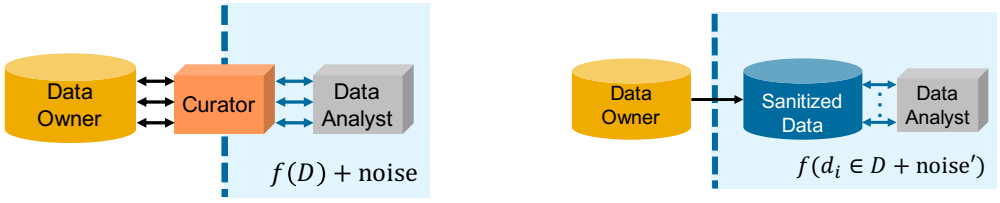
Definition 4 ((ε, δ)-Differential Privacy [DR14]) *A mechanism M gives (ε, δ)-Differential Privacy if for all $D, D' \subseteq DOM$ differing in at most one element, and all outputs $S \subseteq Range(M)$*

$$\Pr[M(D) \in S] \leq e^{\varepsilon} \cdot \Pr[M(D') \in S] + \delta,$$

where Range(M) denotes the set of all possibles outputs of mechanism M.

δ allows a larger difference between the probability of mechanism output values and thus relaxes the strict definition of ε-differential privacy. (ε, δ)-differential privacy offers a relaxed, weaker guarantee compared to ε-differential privacy with the same value of ε. In fact, ε-differential privacy can also be written as $(\varepsilon, \delta = 0)$-differential privacy. Thinking of Definition 4 as providing ε-differential privacy in $1 - \delta$ percent of all function evaluations it is clear to see why literature demands δ to be $\ll \frac{1}{|D|}$.

The Gauss mechanism of Theorem 3 is frequently used to provide (ε, δ)-differential privacy for real valued function. The Gauss mechanism uses global ℓ_2-sensitivity Δ_{f2}, as formalized in Definition 5.



(a) Central setting.

(b) Local setting.

Figure 3.7: Comparison of trust boundary between central and local setting.

Definition 5 (Global ℓ_2-Sensitivity) *Let D and D' be neighboring. The global ℓ_2-sensitivity of a function $f(\cdot)$, denoted by Δ_{f2}, is defined as*

$$\Delta_{f2} = max_{D,D'}|f(D) - f(D')|_2.$$

Theorem 3 (Gauss Mechanism [DR14]) *Given a numerical query function $f : DOM^n \to \mathbb{R}^k$, the Gauss mechanism*

$$M_{Gau}(D, f, \varepsilon, \delta) := f(D) + (z_1, ..., z_k)$$

is an (ε, δ)-differentially private mechanism for any $\varepsilon, \delta \in (0,1)$ when all z_i with $1 \leq i \leq k$ are independently drawn from $Z \sim N(0, \sigma^2)$, with $\sigma \geq c\frac{\Delta_{f_2}}{\varepsilon}$ and $c^2 > 2ln(\frac{1.25}{\delta})$.

In contrast to the Laplace distribution the Gauss distribution does not enjoy the sliding property. Here, the guarantee of an observation being within a factor of e^{ε} between the likelihoods of D and D' no longer holds for large noise values occurring in the distribution tails, bounded by δ.

Theorem 2 can be extended to hold for (ε, δ)-differential privacy by considering $\sum_{i=1}^{k} \delta_i$.

3.2.3 ε-Local Differential Privacy

Until now, we limited our discussion of differential privacy to the central setting where privacy is enforced by Curator, a trusted third party, by the application of differentially private mechanisms. However, given the data market focus of MOSAICrOWN, we also investigate stronger options without this trust assumption (and less accuracy), providing a choice for the market participants. We will refer to the setting that includes only two actors, Data Owner and Analyst, as *local setting*.

Figure 3.7 provides a comparison of the actors and Data Owner trust boundary (dotted, blue line) for the central and local setting. In Figure 3.7(b) we can observe that in the local setting they exchange anonymized data, which is similar to the release of microdata discussed in the previous chapter.

To apply differential privacy in the local setting we introduce ε-Local Differential Privacy in Definition 6. Instead of bounding the difference between probability distributions from which a query answer could have been produced, ε-LDP demands that for any possible input value from a domain *DOM* an ε-LDP mechanism returns any possible value v_2 from *DOM* with non-zero probability.

Definition 6 (ε-Local Differential Privacy) *An algorithm M satisfies ε-Local Differential Privacy (ε-LDP), where $\varepsilon \geq 0$, if and only if for any input v_1 and v_2, we have*

$$\forall S \in Range(M) : \Pr[M(v_1) \in S] \leq \exp(\varepsilon) \cdot \Pr[M(v_2) \in S]$$

where $Range(M)$ denotes the set of all possible outputs of the algorithm M.

We will now illustrate the application of ε-LDP by discussing Randomized Response [War65]. Randomized response is a technique that provides plausible deniability to survey participants. Assume a survey containing sensitive questions that can only be answered *yes* or *no*. The survey participant is requested to throw a coin per question. If the coin shows *heads*, the participant will report his true answer. Else, the participant will throw the coin again and report *yes* in case of heads and otherwise *no*. This algorithm fulfills ε-LDP, since any true input can result in any output from the domain $\{\text{Yes}, \text{No}\}$. To be more specific, for a fair coin the algorithm is $\varepsilon = \ln(3)$ differentially private since:

$$\frac{\Pr[r = \text{Yes}|\ \text{truth} = \text{Yes}]}{\Pr[r = \text{Yes}|\text{truth} = \text{No}]} = \frac{3/4}{1/4} = 3 = e^{\varepsilon}.$$

Based on the noise generation (i.e., fair coin flip) we can reason about the true counts contained in the noisy responses and improve the accuracy as follows. Assuming we want to approximate the true fraction of *yes* answers, which we denote p_y. The expected portion y of *yes* values under a fair coin is $\frac{1}{2} \cdot p_y + \frac{1}{4}$, i.e., the true frequency (p_y) is reported when the flipped coin shows heads (probability $1/2$) and a random yes is reported when the flipped coin shows tails and a second flip heads (probability $1/4$). Given this equation, we can approximate the actual portion p_y of yes answers as $2y + \frac{1}{4}$. Thus, these noisy responses allow us to approximate true answers of a group while providing privacy for each individual.

Protocols extending randomized response have for example seen adoption for small value domains such as reporting user statistics (e.g., Erlingsson et al. [EPK14]). However, adaption is problematic in case of large value domains.

3.2.4 Discussing Utility and Privacy

We recall that semantic privacy is motivated by the impossibility of releasing *useful* statistics about a dataset without disclosing insights about the underlying respondents in a dataset. An extreme example illustrates usefulness and revisits the differentially private median from Section 3.2.1. To provide the technically strongest data protection, we could assign the same weight to every entry o in the set of possible outputs O from the exponential mechanism. This would essentially construct a uniform distribution and prevent any usefulness of the released data. Thus we decided to specify a probability distribution that encodes a preference towards useful outputs (i.e., close to the true median). This preference can be steered by adapting ε: by letting ε become small (≈ 0) we increase the privacy and decrease the usefulness, by letting ε grow we decrease the privacy and increase the usefulness. However, note that the effect of an increase/decrease in ε is relative with respect to the underlying dataset and the evaluated function (cf. Definition 1;[KM11]) and thus there is no absolute guidance for good or bad ε values.

Besides the privacy parameter ε, the sensitivity Δ_f significantly affects usefulness of differentially private results. Assume we are provided a dataset D containing employee salaries for SAP SE, and now want to provide the average salary $\frac{\sum_{i=0}^{|D|} d_i}{|D|}$ with $\varepsilon = 2$ differential privacy. We can decompose the above computation into two functions: $f_{\sum}(D)$ with sensitivity $\Delta_f = 10,000,000$[1] and $f_{\#}(D)$ with sensitivity $\Delta_f = 1$. The sequential composition theorem (Definition 2) allows us to to split our budget of $\varepsilon = 2$ and thus we can compute the DP average by applying the Laplace mechanism to $f_{\sum}(D)$ and $f_{\#}(D)$ with $\varepsilon = 1$ each:

[1] The highest individual salary in 2018 was EUR 9.4 Mio.; SAP SE Annual Report on Form 20-F `https://www.sap.com/docs/download/investors/2018/sap-2018-annual-report-form-20f.pdf`

$$\frac{f_{\sum}(D) + Lap\left(\frac{\Delta_f = 10{,}000{,}000}{\varepsilon = 1}\right)}{f_{\#}(D) + Lap\left(\frac{\Delta_f = 1}{\varepsilon = 1}\right)}.$$

We directly observe that the noise scale is significantly larger for $f_{\sum}(D)$ than it is for $f_{\#}(D)$ and thus will provide more noise and significantly impact usefulness. Thus it is of main importance to investigate functions with small Δ_f or bounding Δ_f when applying DP, e.g., by clipping the maximum individual salary in our example.

4. Quantifying Privacy for Machine Learning with Membership Inference

Machine learning is ubiquitous in software applications nowadays and the sharing of trained machine learning models with third parties is relevant for data markets. However, the success of machine learning (ML) depends as much on sophisticated algorithms as it does on the availability of large sets of training data. Gathering sufficient amounts of training data for satisfying model generalization has proven cumbersome especially for sensitive data and, in some cases, resulted in privacy violations due to data misuse (e.g., the inappropriate legal basis for the use of National Health Service (NHS) data in the DeepMind project [The17, HMDD19]). Furthermore, privacy threats are arising since the desire to identify on which data a model was trained gives rise to two attack categories against machine learning models: model inversion, which aims for reconstructing a training dataset with missing parts [FLJ+14, FJR15, TZJ+16], and membership inference (MI) [SSSS17]. The latter attack is striving to identify whether an individual, or a set of individuals, *belong to* a certain training dataset; thus, it is especially relevant for MOSAICrOWN. We argue that membership inference is relevant for two actors: an adversary performing *single* record MI and a regulator performing *set* MI. Single MI is used in previous work to model an adversary who is mainly interested in identifying individuals within a dataset. However, set MI is relevant for regulatory audits since it can be used to prove that a specific set of records was used to train a model. If the practitioner who trained the model was not authorized to use a specific dataset for this purpose, regulators can apply set MI to prove data privacy violations.

A mitigation to privacy violations and privacy threats is offered by anonymization with differential privacy (DP) during machine learning training. Within this section we consider two machine learning architectures that have been investigated for use in data markets throughout the first half of MOSAICrOWN: feedforward neural networks for classification and generative models[1] for data generation. For each architecture we discuss how privacy can be quantified by using membership inference attacks and how privacy can be ensured by using differential privacy.

The results presented in this chapter have been published as conference paper [HHB19] and a preprint [BGRK19]. This chapter is structured as follows. First, we give some notation and preliminaries in Section 4.1. In Section 4.2 we introduce membership inference threat models. In Section 4.3 we suggest to compare privacy guarantees under a single membership inference adversary for feedforward neural networks. In Section 4.4, we introduce and formalize two attacks which are applicable to both single and set membership inference against generative models. To this end, details regarding GANs and VAEs are provided.

[1] Generative models are ML models that are trained to learn the joint probability distribution $p(X,Y)$ of features X and labels Y of training data.

Symbol	Description
X	Set of vectors $\vec{x}_1,\ldots,\vec{x}_j$ where $x_j^1,\ldots,x_j^i$ denote attribute values (*features*) of $\vec{x}_j$.
Y	Set of k target variables $y_1,\ldots,y_k$ (*labels*).
$\mathtt{C}$	$\lvert Y\rvert$.
$\vec{y}$	Vector of target variables (*labels*) where variable $y_j \in \vec{y}$ represents the label for $\vec{x}_j \in X$.
$\hat{y}$	Predicted target variable, i.e., $\hat{y} = h(\vec{x})$.
$p(\vec{x})$	Softmax confidence for $\vec{x}$.
D	$D := (X,\vec{y})$.
d	A record $d \in D$, where $d := (\vec{x},y)$.
n	$\lvert D_{\mathtt{target}}^{\mathtt{train}}\rvert$.

Table 4.1: Notations and context.

4.1 Preliminaries & Notation

The set of notations that is used throughout this chapter is summarized in Table 4.1. In contrast to anonymization methods based on generalization, differential privacy (DP) [Dwo06] anonymizes a dataset $D = \{d_1,\ldots,d_n\}$ by perturbation. DP can be either enforced locally to each entry $d \in D$, or centrally to an aggregation function $f(D)$. In the following, we recall parts of the definition of central and local differential privacy from Sections 3.2.1 and 3.2.3 in the new context of machine learning.

Central Differential Privacy. In the central model the aggregation function $f(\cdot)$ is evaluated and perturbed by a trusted server. Due to perturbation it is no longer possible for an adversary to confidently determine whether $f(\cdot)$ was evaluated on D, or some neighboring dataset D' differing in one element. Thus, assuming that every participant is represented by one element, privacy is provided to participants in D as their impact of presence (absence) on $f(\cdot)$ is limited.

To enforce DP in the central setting (CDP) in deep learning we use differentially private versions of two standard gradient optimizers: SGD and Adam[2]. We refer to these CDP optimizers as DP-SGD and DP-Adam. A differentially private optimizer represents a differentially private training mechanism M_{nn} that updates the weight coefficients θ_t of a neural network per training step $t \in T$ with $\theta_t \leftarrow \theta_{t-1} - \alpha(\tilde{g})$, where $\tilde{g} = M_{nn}(\partial loss/\partial\theta_{t-1})$ denotes a Gaussian perturbed gradient and α is some scaling function on $\tilde{g}$ to compute an update, i.e., learning rate or running moment estimations. Differentially private noise is added by the Gaussian mechanism of Definition 3 as suggested by Abadi et al. [ACG+16]. After T update steps, M_{nn} outputs a differentially private weight matrix θ which is used by the prediction function $h(\cdot)$ of a neural network. A CDP gradient optimizer bounds the sensitivity of the computed gradients by a clipping norm C based on which the gradients get clipped before perturbation.

Since weight updates are performed iteratively during training, a composition of M_{nn} is required until the the training step T is reached and the final private weights θ are obtained. For CDP, we measure privacy decay under composition by tracking the noise levels σ we used to invoke the Gaussian mechanism. After training, we transform and compose σ under Renyi differential privacy [Mir17], and transform the aggregate again to CDP. We chose this accumulation method over other advanced composition schemes (e.g., Advanced Composition or Moments

[2]The Tensorflow privacy package: `https://github.com/tensorflow/privacy`.

Accountant [KOV17, ACG+16]) since it provides tighter bounds for heterogeneous mechanism invocations.

Local Differential Privacy. We refer to the perturbation of entries $d \in D$ as local differential privacy [WBLJ17]. LDP is the standard choice when the server that evaluates a function $f(D)$ is untrusted. We adapt the definitions of Kasiviswanathan et al. [KLN+08] to achieve LDP by using local randomizers LR, i.e., we use Definition 6 with algorithm $M = LR$ in the following.

In the experiments within this section, we use a local randomizer to perturb each record $d \in D$ independently. Since a record may contain multiple correlated features (e.g., pixels in an image, items in a preference vector) a local randomizer must be applied repeatedly which results in a sequentially increasing loss of privacy. A series of local randomizer executions per record composes a local algorithm according to Definition 7. ε-local algorithms are ε-local differentially private [KLN+08], where ε is a summation of all composed local randomizer guarantees.

Definition 7 (Local Algorithm) *An algorithm is ε-local if it accesses the database D via LR with the following restriction: for all $i \in \{1,\ldots,|D|\}$, if $LR_1(i),\ldots,LR_k(i)$ are the algorithms invocations of LR on index i, where each LR_j is an ε_j-local randomizer, then $\varepsilon_1 + \ldots + \varepsilon_k \leq \varepsilon$.*

We perturb low domain data with randomized response [War65], a (composed) local randomizer. According to Equation (4.1) randomized response yields $\varepsilon = \ln(3)$ LDP for a one-time collection of values from binary domains (e.g., $\{\texttt{yes},\texttt{no}\}$) with two fair coins [EPK14]. That is, retention of the original value with probability $\rho = 0.5$ and uniform sampling with probability $(1-\rho)\cdot\rho$.

$$\varepsilon = \ln\left(\frac{\rho + (1-\rho)\cdot\rho}{(1-\rho)^2}\right) = \ln\left(\frac{\Pr[\texttt{yes}|\texttt{yes}]}{\Pr[\texttt{yes}|\texttt{no}]}\right). \tag{4.1}$$

In our evaluation we also look at image data for which we rely on the local randomizer by Fan [Fan18] for LDP image pixelation. The randomizer applies the Laplace mechanism of Definition 1 with scale $\lambda = \frac{255\cdot m}{b^2\cdot\varepsilon}$ to each pixel. Parameter m represents the neighborhood in which LDP is provided. Full neighborhood for an image dataset would require that any picture can become any other picture. As a rule of thumb providing DP within large neighborhoods will require high ε values to retain meaningful image structure, and vice versa. High privacy will result in uniform random black and white images.

Within this section we consider the use of LDP and CDP for deep learning along a generic data science process (e.g., CRISP-DM [WH00]). In such processes, the dataset D of a data owner DO is (i) transformed, and (ii) used to learn a model function $h(\cdot)$ (e.g., classification), which (iii) afterwards is deployed for evaluation by third parties. In the following, $h(\cdot)$ will represent a neural network. DP is applicable at every stage in the data science process. In the form of LDP by perturbing each record $d \in D$, while learning $h(\cdot)$ centrally with a CDP gradient optimizer, or to the evaluation of $h(\cdot)$ by federated learning with CDP voting [PAE+17]. We focus on the data science process without collaboration and keep federated learning for future reference.

When applying DP in the data science process, the privacy-accuracy trade-off is of particular interest. Similar to the evaluation of regularization techniques that apply noise to the training data to foster generalization (e.g., [GBC16, GC95, Mat92]) we judge utility by the test accuracy of $h(\cdot)$. I.e., the accuracy of $h(\cdot)$ on test data after having learned $h(\cdot)$ from training data.

4.2 Threat Models in Deep Learning

In this section, we introduce the threat model called Membership Inference (MI).

4.2.1 Background of Membership Inference

The goal of membership inference (MI) is to gather evidence that a specific record or a set of records belongs to the training dataset of a given machine learning model. MI thus represents an approach for measuring how much a model leaks about individual records of a population beyond what it reveals about an arbitrary member of the population. The success rates of MI attacks against a model are tightly linked to overfitting (i.e., the generalization error [YFJ17]). The poorer a model generalizes the more specificities it contains about individual training data records.

In this section, two kinds of MI are considered: single MI and set MI. The single MI is comparable to common experiment setups for MI [SSSS17, HMDD19]. In the set MI setting a regulator has to recognize which of the two provided sets contains training data records. This section considers two actors corresponding to single and set MI, respectively. The first actor is an honest-but-curious adversary $\mathcal{A}$ and the second actor is a regulatory body $\mathcal{R}$. Each actor focuses on a specific task: the adversary $\mathcal{A}$ is common in MI literature and infers whether a single record was present in the training dataset using *single* membership inference. The regulatory body $\mathcal{R}$ performs *set* membership inference to identify whether a set of records was present in the training dataset. This attack can provide evidence that a certain set of training data was illegally used to train a generative model.

Both actors are assumed to have no access to the underlying training dataset of the generative model, and they refrain from activities that maliciously modify this target model.

4.2.2 Adversarial Actor: Single MI

Single MI has been used by previous work to evaluate attacks against GANs [HMDD19] . In this setting, the honest-but-curious adversary $\mathcal{A}$ has to identify individual records which were used to train the model. To this end M records from the training data and M records from the test dataset $\{x_1, \ldots, x_{2M}\}$ are given.

The Membership Inference attacks against generative adversarial networks and feedforwards that are discussed within this section rely on a function $\hat{f}(x)$ that can be computed for each of the records. The intuition is that this function attains higher values for training data records. Details on how this function is realized are given in the corresponding sections. In the following description of the attack types we use the general notation $\hat{f}(x)$.

For every record x_i, $\mathcal{A}$ has to decide whether it was part of the training data. In general, $\mathcal{A}$ picks the M records with the M greatest values of the function $\hat{f}(x)$.

Attack Type 1 (Single Membership Inference) *Let $\mathcal{A}$ be an adversary who is able to compute the function $\hat{f}(x)$ for every record x.*

1. *Choose records $\{x_1, \ldots, x_M\}$ from the training data.*
2. *Choose records $\{x_{M+1}, \ldots, x_{2M}\}$ from the test data.*
3. *$\mathcal{A}$ is presented the set $\{x_1, \ldots, x_{2M}\}$.*
4. *$\mathcal{A}$ labels the M records with highest values $\hat{f}(x_i)$ as training data.*

We denote the M records chosen by $\mathcal{A}$ as $\{x_1^A, \ldots, x_M^A\}$. We call the proportion of actual training data in this set

$$\frac{1}{M} \cdot \left| \{ i \mid x_i^A \in \{x_1, \ldots, x_M\} \} \right|$$

the accuracy of the attack for single MI.

4.2.3 Regulatory Actor: Set MI

Set MI corresponds to the needs of regulators and auditors aiming to prove data privacy violations in machine learning. One set consisting of M records from the training data $\{x_1, \ldots, x_M\}$ and another set consisting of M records from the test data $\{x_{M+1}, \ldots, x_{2M}\}$ are shown to a regulator $\mathcal{R}$ in either order. The task of $\mathcal{R}$ is to decide which of the two sets is a subset of the original training data. Contrary to single MI, $\mathcal{R}$ knows which records belong to the same data source (training data or test data). However, $\mathcal{R}$ does not know which set is a subset of the original training data.

Similar to single MI $\mathcal{R}$ computes the function $\hat{f}(x)$ for every record and selects the M records with the M highest values $\hat{f}(x)$. For each of the selected records, $\mathcal{R}$ checks to which set it belongs and eventually selects the set from which most of these records stem as subset of the original training data.[3] Note that this is equivalent to taking the set with the higher median. Since we do not have any prior knowledge on the type of distribution of the $\hat{f}$-values this is more robust than considering the mean.

Attack Type 2 (Set Membership Inference) *Let $\mathcal{R}$ be an adversary able to calculate the function $\hat{f}(x)$ for every record x.*

1. *Choose records $\{x_1, \ldots, x_M\}$ from the training data.*
2. *Choose records $\{x_{M+1}, \ldots, x_{2M}\}$ from the test data.*
3. *$\mathcal{R}$ is presented the sets $\{x_1, \ldots, x_M\}$ and $\{x_{M+1}, \ldots, x_{2M}\}$.*
4. *$\mathcal{R}$ identifies the M records with highest values $\hat{f}(x_i)$.*
5. *$\mathcal{R}$ chooses the set from which most of these records stem.*
6. *If both have the same number of representatives $\mathcal{R}$ picks one set randomly.*

The accuracy of an attack of this type is defined as the average success rate of $\mathcal{R}$, i.e., the probability that $\mathcal{R}$ identifies the true subset of the training data.

4.2.4 Relevance for Real-World Use Cases

The formalized MI attack types are an alternative to assessing a single record x by computing $\hat{f}(x)$ and considering the record part of the training data if the value exceeds a threshold. While the single record approach is conceptually similar, the formalized types contributed in this section are closer to real-world use cases. For example, in machine learning as a service (MLaaS) applications access to both test and training data is implicitly given. Hence, the single MI and set MI attack types can be automatically conducted. Increased MI attack accuracies suggest that the model quality is insufficient w.r.t. privacy.

[3] If an equal number of records belong to the first and the second set, $\mathcal{R}$ picks one of the sets with probability 50%.

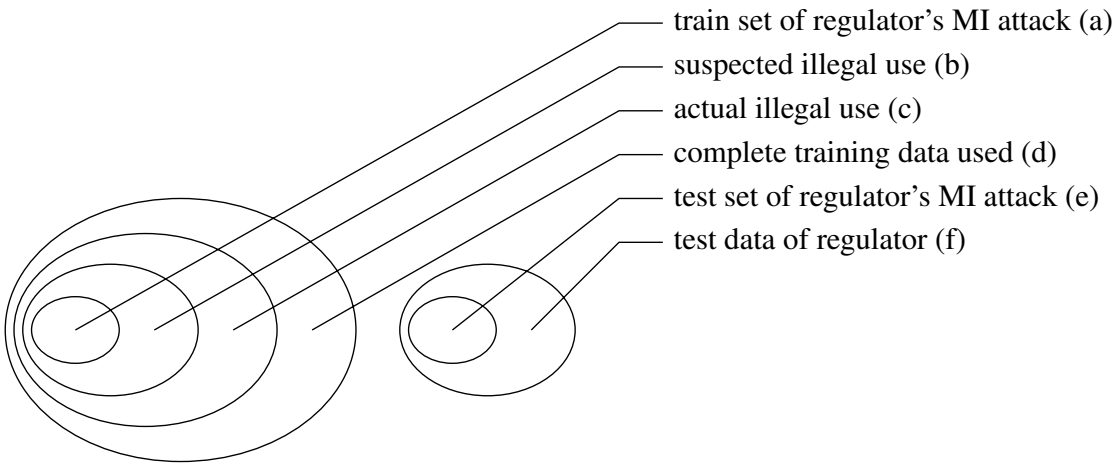


Figure 4.1: Venn diagram of training and test data in the regulatory use case for $\mathcal{R}$.

Figure 4.1 visualizes the regulatory use case. The regulator $\mathcal{R}$ suspects that a certain dataset was illegally used to train a model (b). Actually, even more data was used illegally (c). Moreover, some legally obtained data might have been used. Together with the illegal data, it represents the complete training data (d). $\mathcal{R}$'s set of suspected data is used as train set in the set MI attack (a). $\mathcal{R}$ also needs test data (f) from which a subset (e) is used as test set for the attack. If the attack is successful the illegal use can be proven. Otherwise, the attack does not perform better than random guessing. By repeating the attack for multiple choices of subsets (a) and (f) $\mathcal{R}$ ensures statistical significance. Note that $\mathcal{R}$ does not need to know the entire training data since the MI attacks also work for subsets of the entire training data. The accuracy does not depend on the concrete subset choice as we will show in our experiments in Section 4.4.4.

Note that in both single and set MI we assume that there are exactly as many test records as training records. In the regulatory use case of set MI this is realistic since a sample of the larger of the two sets can be used if they are not of equal size. To make the results of single and set MI comparable, and to be in line with the balanced setting in previous work [SSSS17], we also decided to use this setup in single MI. Note that this is potentially an advantage for $\mathcal{A}$.

4.3 Quantifying Privacy Risks in Feedforward Neural Networks with MI

While several MI attacks have been formulated, this Deliverable solely refers to the black-box single MI attack by Shokri et al [SSSS17] as an exemplary membership inference attack against feedforward neural networks. Section 4.3.1 introduces this MI attack. Throughout this section we will illustrate how the attack is used against a synthetic dataset that is introduced in Section 4.3.2 to assess LDP and CDP privacy parameters. The assessment is done through an experiment in Section 4.3.3.

4.3.1 Black-Box MI Attack

The black-box MI attack assumes an honest-but-curious Adversary $\mathcal{A}$ with access to a trained prediction function $h(\cdot)$ and predictions from $h(\cdot)$ (e.g., softmax confidence values). We refer to the trained ML model against which the MI attack is applied as *target model*. Within three steps

the MI attack exploits that an ML classifier such as a neural network tends to classify a record d from the model's training dataset $D^{\texttt{train}}_{\texttt{target}}$ with differing softmax confidence $p(\vec{x})|h(\vec{x})$ to its true label y in comparison to a record $d \notin D^{\texttt{train}}_{\texttt{target}}$.

First, data owner $\mathcal{DO}$ trains a ML model, *target model*, for some classification task with records from a dataset $D^{\texttt{train}}_{\texttt{target}}$. After training $\mathcal{DO}$ exposes the target model to the adversary $\mathcal{A}$ for inference tasks, e.g., through an API. Second, $\mathcal{A}$ trains copies of the target model w.r.t. structure and hyper-parameters, so called *shadow models*, on data statistically similar to $D^{\texttt{train}}_{\texttt{target}}$. For any $i \neq j$, it applies that

$$\begin{aligned} &|D^{\texttt{train}}_{\texttt{shadow}_\texttt{i}}| = |D^{\texttt{test}}_{\texttt{shadow}_\texttt{i}}| \\ \wedge\ &D^{\texttt{train}}_{\texttt{shadow}_\texttt{i}} \cap D^{\texttt{test}}_{\texttt{shadow}_\texttt{i}} = \varnothing \\ \wedge\ &|D_{\texttt{shadow}_\texttt{i}} \cap D_{\texttt{shadow}_\texttt{j}}| \geq 0. \end{aligned}$$

After training, each shadow model is invoked by $\mathcal{A}$ to classify all respective training and test data, i.e., $p(\vec{x})$, $\forall\ d \in D^{\texttt{train}}_{\texttt{shadow}_\texttt{i}} \cup D^{\texttt{test}}_{\texttt{shadow}_\texttt{i}}$. Since $\mathcal{A}$ has full control over $D^{\texttt{train}}_{\texttt{shadow}_\texttt{i}}$ and $D^{\texttt{test}}_{\texttt{shadow}_\texttt{i}}$, each shadow model's output $(p(\vec{x}), y)$ is appended with a label "`in`" if the corresponding record $d \in D^{\texttt{train}}_{\texttt{shadow}_\texttt{i}}$. Otherwise, its label is "`out`".

Third, a binary classifier, *attack model*, is trained by $\mathcal{A}$ per target variable $y \in Y$ to map $p(\vec{x})$ to the indicator "`in`" or "`out`". The triples $(p(\vec{x}), y, \texttt{in}/\texttt{out})$ serve as attack model training data, i.e., $D^{\texttt{train}}_{\texttt{attack}}$. The attack model thus exploits the imbalance between predictions on $d \in D^{\texttt{train}}_{\texttt{target}}$ and $d \notin D^{\texttt{train}}_{\texttt{target}}$.

Finally, the attack model is evaluated on tuples $(p(\vec{x}), \hat{y})$ $\forall d \in D^{\texttt{train}}_{\texttt{target}}$, which simulates the worst case where $\mathcal{A}$ tests membership for all training records.

We find the black-box MI attack to be especially effective when some classes within the training data comprise a comparatively low number of records and the overall training data distribution is imbalanced. We observed these imbalanced classes especially when learning models from sensitive or personal training data.

Evaluating CDP and LDP under MI

Considering that both DP and MI are tailored to the protection and identification of individual data records, we argue for evaluating DP privacy by MI Attack precision and recall, and focus on two privacy questions: "How many records predicted as `in` are truly contained in the training dataset?" (precision), and "How many truly contained records are predicted as `in`?" (recall). We calculate $\mathcal{A}$'s MI precision and recall as the average score over the instances of all classes to ensure comparability to the results of Shokri et al. [SSSS17]. We illustrated that $\mathcal{DO}$ has two options to apply DP within the data science process. Either LDP by applying a local randomizer on the training data and using the resulting $LR(D^{\texttt{train}}_{\texttt{target}})$ for training, or central DP with a differentially private optimizer on $D^{\texttt{train}}_{\texttt{target}}$. A discussion and comparison limited to the privacy parameter ε likely falls short and potentially leads data scientists to incorrect conclusions. Thus, data scientists give up flexibility w.r.t. applicable learning algorithms and may miss a favorable privacy-accuracy trade-off, if they rule out the use of LDP due to comparatively greater ε and instead solely investigate CDP (e.g., DP-SGD). Instead we suggest to compare LDP and CDP by their concrete effect on an MI attack. While we consider the MI attack of Shokri et al. [SSSS17] our methodology is applicable to other MI attacks as well. Depending on the notion of privacy, the MI attack scheme described earlier in this section changes slightly. When examining CDP, we train both target and shadow models on the same datasets as would be used without any anonymization. However, a

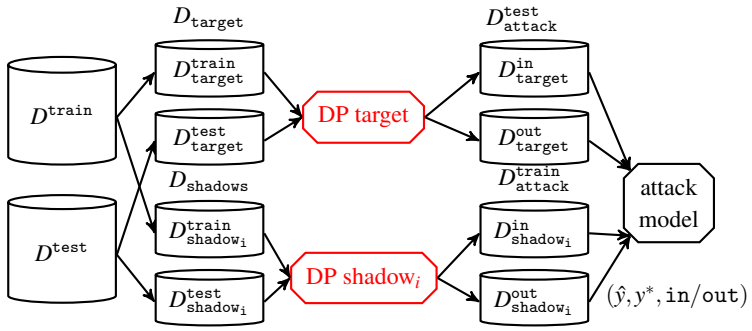


Figure 4.2: MI under central DP with DP gradient optimizer.

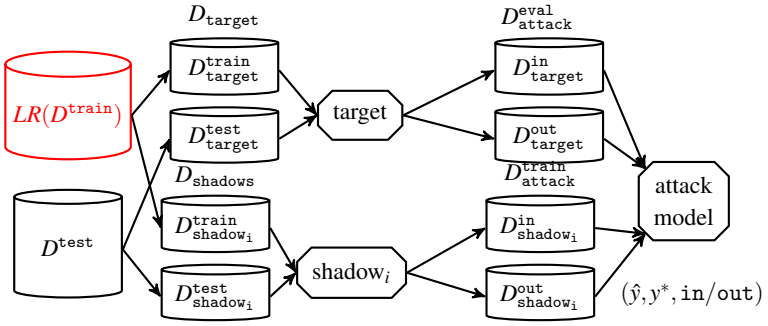


Figure 4.3: MI against LDP in target model training

similar configured CDP optimizer is used during the training phase for target and shadow models. Adversary $\mathcal{A}$ obtains higher MI precision and recall when relying on an equally configured DP optimizer for shadow model training, compared to use of the non-DP optimizer for shadow model training. The LDP setup requires a local randomizer to perturb the training inputs for both target and shadow models. See Figure 4.2 and Figure 4.3 for comparison.

We calculate the relative privacy-accuracy trade-off for LDP and CDP as the relative difference between $\mathcal{DO}$'s change in test accuracy to the change in MI precision and recall, and introduce a measure for quantification in the following: Efficient privacy-accuracy trade-offs in LDP or CDP must reduce $\mathcal{A}$'s susceptibility without sacrificing significant test accuracy for $\mathcal{DO}$. In the following we define our measure w.r.t. MI precision. However, the definition is analogously applicable to MI recall, and will be used for MI precision and recall throughout this work.

Let $prec_{orig}$ be $\mathcal{A}$'s MI precision and acc_{orig} be $\mathcal{DO}$'s original test accuracy, and let $prec_{\varepsilon}$ be $\mathcal{A}$'s MI precision and acc_{ε} be $\mathcal{DO}$'s resulting test accuracy after application of LDP or CDP. Also, let acc_{base} be the minimal test accuracy of $1/\texttt{C}$ and $prec_{base}$ be the minimal MI precision of 0.5 (e.g., random guessing). For the calculation of φ we will clip decreases below the baseline (i.e., set values below the baseline to the baseline) since these indicate an Adversary worse than random guessing. Normalizing yields mitigation efficiency φ as stated below. Equation (4.2) requires $prec_{orig} - prec_{base} \neq 0$ and $acc_{orig} - acc_{base} \neq 0$, i.e., the original model is vulnerable to MI and yields meaningful test accuracy.

$$\varphi = \frac{\left(\frac{\text{maximum decrease accuracy} - \text{actual decrease accuracy}}{\text{maximum decrease accuracy}}\right)}{\left(\frac{\text{maximum decrease precision} - \text{actual decrease precision}}{\text{maximum decrease precision}}\right)}$$

$$= \frac{\frac{(acc_{orig} - acc_{base}) - (acc_{orig} - acc_{\varepsilon})}{acc_{orig} - acc_{base}}}{\frac{(prec_{orig} - prec_{base}) - (prec_{orig} - prec_{\varepsilon})}{prec_{orig} - prec_{base}}} \tag{4.2}$$

φ quantifies the relative loss in accuracy in its numerator and the relative gains in privacy in its denominator for a given privacy parameter ε. Hence, φ presents the relative privacy-accuracy trade-off as a ratio which we seek to maximize. When the relative gain in privacy (lower MI precision or MI recall) exceeds the relative loss in accuracy φ will be > 1. In contrast, if the loss in test accuracy exceeds the gain in privacy φ will be < 1.

4.3.2 Dataset

Skewed Purchases We specifically crafted this dataset to mimic a situation for transfer learning, i.e., the application of a trained model to novel data which is similar to the training data w.r.t. format but following a different distribution. This situation arises in a classification scenario where customer shopping carts shall be classified to customer groups, if for example not enough high-quality shopping cart data for a specific retailer are available yet. Thus, only few high-quality data (e.g., manually crafted examples) can be used for testing and large amounts of low quality data from potentially differing distributions for training (e.g., from other retailers). In effect the distribution between train and test data varies for this dataset. The dataset consists of $200,000$ records with 600 features (e.g., products possibly contained in a shopping cart) and is available in four versions with $\mathtt{C} \in \{10, 20, 50, 100\}$ labels. Each vector x in the training dataset X is generated by using two independent random coins to sample a value from $\{0,1\}$ per position $i = 1, \ldots, 600$. The first coin steers the probability $\Pr[x_i = 1]$ for a fraction of 600 positions per x. We refer to these positions as indicator bits (*ind*) which indicate products frequently purchased together. The second coin steers the probability $\Pr[x_i = 1]$ for a fraction of $600 - (\frac{600}{|\mathtt{C}|})$ positions per x. We refer to these positions as noise bits (*noise*) that introduce scatter in addition to *ind*. We let $\Pr_{ind}[x_i = 1] = 0.8 \wedge \Pr_{noise}[x_i = 1] = 0.2$, $\forall x \in X_{train}$, and $\Pr_{ind}[x_i = 1] = 0.8 \wedge Pr_{noise}[x_i = 1] = 0.5 \wedge x \in X_{test}, 1 \leq i \leq 600$. This dataset has a difference in information entropy between test and train data of ≈ 0.3. The difference would be ≈ 0, if there is no skew. Figure 4.4 depicts the MI precision over the different training dataset distributions and a fixed test distribution, and illustrates that datasets with varying train and test distributions are actually more vulnerable to MI and thus potentially require stronger privacy parameters.

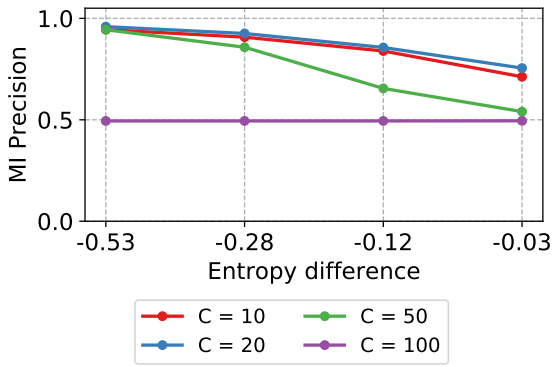


Figure 4.4: Skewed Purchases skew effect on MI precision

Dataset	LDP composed ε	Comment
Skewed Purchases	$4,800 - 60$	$600 \times \varepsilon_i$

Table 4.2: Overview of LDP ε.

		Skewed Purchases			
C		10	20	50	100
Learning rate	Baseline	10^{-3}	10^{-3}	10^{-3}	10^{-3}
	CDP	10^{-3}	10^{-3}	10^{-3}	10^{-3}
	LDP	10^{-3}	10^{-3}	10^{-3}	10^{-3}
Batch Size	Baseline	100	100	100	100
	CDP	100	100	100	100
	LDP	100	100	100	100
Epochs	Baseline	200	200	200	200
	CDP	200	200	200	200
	LDP	200	200	200	200

Table 4.3: Hyperparameters

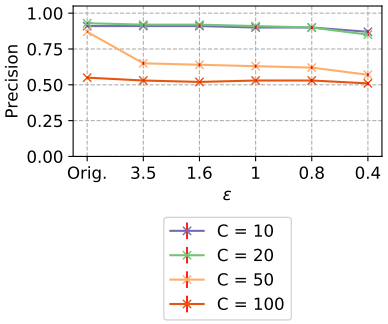
4.3.3 Experiment

We perform a single experiment. The experiment compares LDP and CDP under MI precision and recall instead of privacy parameter ε. The experiment is analyzed through three sets of figures. First, by plotting test accuracy and MI precision and recall over ε, respectively. The three resulting graphs map ε to MI precision and recall, and test accuracy. We present this information for CDP in Figure 4.5 and for LDP in Figure 4.6. Second, by comparing the achievable MI precision and recall over target model test accuracy in a scatterplot to identify strictly better privacy-accuracy trade-offs. We present this information for LDP and CDP in Figure 4.7. Third, by calculating φ (cf. Equation (4.2) in Section 4.3.1) to identify efficient privacy parameters for $\mathcal{DO}$ w.r.t. mitigating MI precision. The corresponding figures are 4.5(d) for CDP and 4.6(d) for LDP.

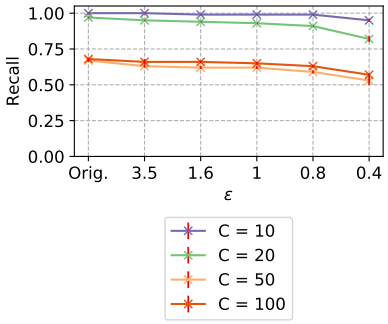
For all executions of the experiment CDP noise is sampled from a Gaussian distribution (cf. Definition 3) with $\sigma =$ *noise multiplier* $z \times$ *clipping norm* C. We evaluate increasing noise regimes by evaluating noise multipliers $z \in \{2,4,6,8,16\}$ until model convergence and calculate the resulting ε at a fixed $\delta = \frac{1}{n}$. We denote the non-private, original MI precision and $\mathcal{DO}$ test accuracy as *original*. For LDP we use the same hyperparameters as in the original training and evaluate randomized response as local randomizer. For each randomizer we state the individual ε_i per invocation (i.e., per anonymized value) and ε per record (i.e., collection of dependent values). We apply randomized response to the dataset with a range of privacy parameter values $\varepsilon_i \in \{0.1,1,3,5,8\}$ that reflect varying retention probabilities.

Skewed Purchases An effect which we observed in transfer learning practice lets us to further analyze differing distributions between train and test data. This effect is, for example, encountered when insufficient high-quality data for training is initially available and reference data that potentially follows a different distribution has to be acquired for first training. CDP results in strong privacy guarantees $\varepsilon \in \{3.5,1.6,1,0.8,0.4\}$.

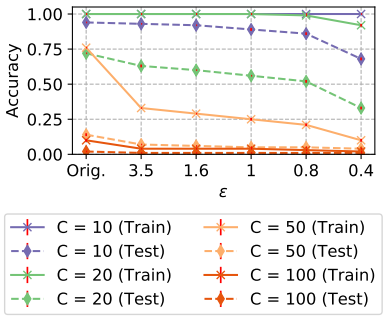
Figure 4.5(a) states the changes in MI precision over privacy guarantees ε. While for the simple classification tasks $\mathtt{C} \in \{10,20\}$ MI precision remains at the original even over all ε, for $\mathtt{C} = 50$ MI precision drops close to the baseline at 0.53 already for $\varepsilon = 3.5$. In contrast, ε has



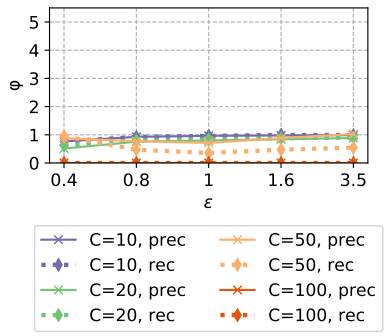
(a) MI precision (CDP)



(b) MI recall (CDP)



(c) Target model accuracy (CDP)



(d) φ Skewed Purchases (CDP)

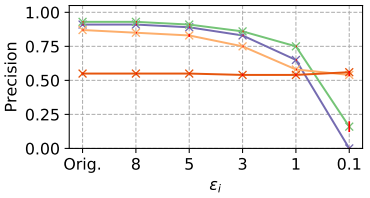
Figure 4.5: $\mathcal{DO}$ accuracy and privacy analysis on Skewed Purchases (error bars lie within most points) for CDP.

only a small effect on MI recall as depicted in Figure 4.5(b). The baseline is solely reached for $\mathtt{C} \in \{50, 100\}$ at $\varepsilon = 0.4$. Figure 4.5(c) presents target model accuracies over ε. The decrease in test accuracy is comparatively stronger for $\mathtt{C} \in \{10, 20\}$ due to the low initial baseline for $\mathtt{C} \in \{10, 20\}$. CDP comes indeed at a heavy cost on this dataset: DP-SGD weakens $\mathcal{A}$'s MI precision while significantly lowering $\mathcal{DO}$'s test accuracy.

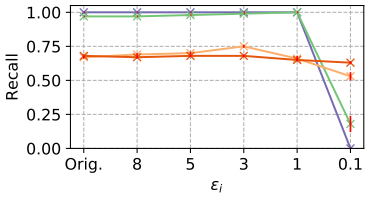
We state MI Precision and recall for over ε_i for LDP in Figures 4.6(a) and 4.6(b). While MI precision is not much affected for $\mathtt{C} \neq 100$ until $\varepsilon_i = 1$, MI recall is solely affected by LDP at a strong $\varepsilon_i = 0.1$ and $\mathtt{C} \in \{10, 20\}$.

Figure 4.6(c) indicates that $\mathcal{DO}$'s test accuracy in LDP is robust to noise under randomized response especially for classification tasks for which n is much larger than the dimensionality l of the training data per class (e.g., $\mathtt{C} \in \{10, 20\}$). For $\mathtt{C} \in \{10, 20\}$ randomized response solely affects $\mathcal{DO}$'s test accuracy under a strong $\varepsilon_i = 0.1$ in contrast to $\mathtt{C} \in \{50, 100\}$ at $\varepsilon_i = 5$. For $\mathtt{C} \in \{10, 20\}$ we observe a regularization effect from randomized response which generalizes $D^{\mathtt{train}}_{\mathtt{target}}$ towards $D^{\mathtt{test}}_{\mathtt{target}}$. Here, the test-train-gap narrows due to increasing test accuracy. Thus, the confidence values also become similar and impede $\mathcal{A}$'s attack model in distinguishing predictions from $D^{\mathtt{train}}_{\mathtt{target}}$ and $D^{\mathtt{test}}_{\mathtt{target}}$ resulting in a decreasing MI recall.

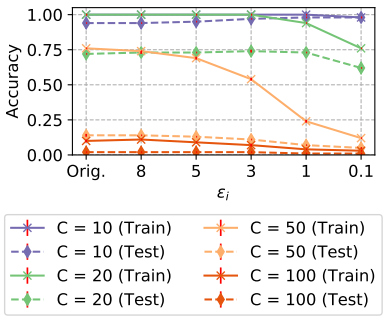
Again we provide scatterplots for $\mathtt{C} = 10$ and 100 in Figure 4.7(a) and 4.7(b). A meaningful relative privacy-accuracy trade-off for MI precision and recall is only achieved under LDP for $\mathtt{C} = 10$ and 20. For $\mathtt{C} = 10$, $\varepsilon_i = 0.1$ LDP trumps $\varepsilon = 3.5$ CDP. Figure 4.5(d) illustrates that almost all $\varphi \leq 1$ for MI recall and precision under CDP. This observation supports our first impression that CDP impacts $\mathcal{DO}$'s test accuracy stronger than $\mathcal{A}$'s MI precision and recall. For LDP we observe high φ in Figure 4.6(d) for $\mathtt{C} \in \{10, 20\}$ and strong $\varepsilon_i \leq 1$. φ is ≈ 50 for MI recall at



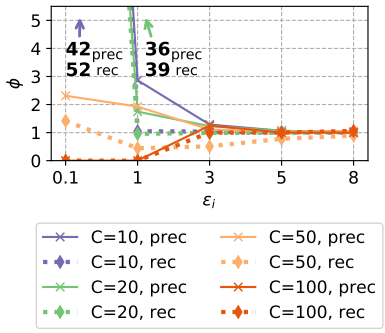
(a) MI precision (LDP) on $D^{\texttt{train}}_{\texttt{target}}$



(b) MI recall (LDP) on $D^{\texttt{train}}_{\texttt{target}}$

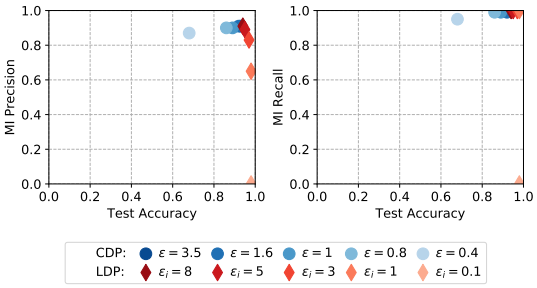


(c) Target model accuracy (LDP)

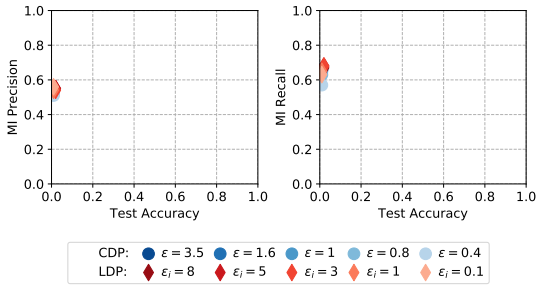


(d) φ Skewed Purchases (LDP)

Figure 4.6: $\mathcal{DO}$ accuracy and privacy analysis on Skewed Purchases (error bars lie within most points) for LDP.



(a) Skewed Purchases $\texttt{C} = 10$



(b) Skewed Purchases $\texttt{C} = 100$

Figure 4.7: $\mathcal{DO}$ accuracy and privacy analysis on Skewed Purchases (error bars lie within most points) for LDP and CDP.

$\varepsilon_i = 0.1$. However, these high efficiency values are mainly due to an increasing test accuracy over ε_i at small decreases in MI recall and precision.

4.4 Quantifying Privacy Risks in Generative Models with MI

In this section we introduce two novel MI attacks that can be used for both single and set MI. Since the attacks details are targeting generative models, we briefly describe VAEs and GANs in Section 4.4.1. The first attack, namely the *Monte Carlo* attack (Section 4.4.2) compares samples drawn from the model to either test or train records. Opposed to existing approaches, only very close samples are considered. Indeed, this distinguishes the attacks from previous approaches like

Attack	Required Access	Applicable	Idea
White-box	Discriminator	GANs	Evaluate Discriminator
Black-box	Samples from Generative Model	Generative Models	Train auxiliary GAN on samples & evaluate Discriminator
Monte Carlo	Samples from Generative Model	Generative Models	Monte Carlo approximation on close samples
Reconstruction Attack	VAE model	VAEs	VAE reconstructs training data more precisely

Table 4.4: Comparison of Attacks

the Euclidean attack [HMDD19] and made the attacks effective. Furthermore, the *Reconstruction* attack (Section 4.4.3) which is optimized for VAEs is presented. A comparison of our attacks and state-of-the-art attacks is given in Table 4.4. An attack is fully specified by the function $\hat{f}(x)$. We evaluate the attacks in Section 4.4.4.

4.4.1 Generative Models

Generative models are ML models that are trained to learn the joint probability distribution $p(X,Y)$ of features X and labels Y of training data. In this work we apply two decoder based models relying on neural networks, namely *Generative Adversarial Networks* (GANs) [GPM+14] and *Variational Autoencoders* (VAEs) [KW13]. Note, however, that our Monte Carlo attack is applicable to all generative models from which one can draw samples. The reconstruction attack specifically targets VAEs.

Generative Adversarial Networks

A GAN consists of two competing models, a *generator* G and a *discriminator* D, which are trained in an adversarial manner (i.e., compete against each other). We describe the approach in detail referring to Figure 4.8.

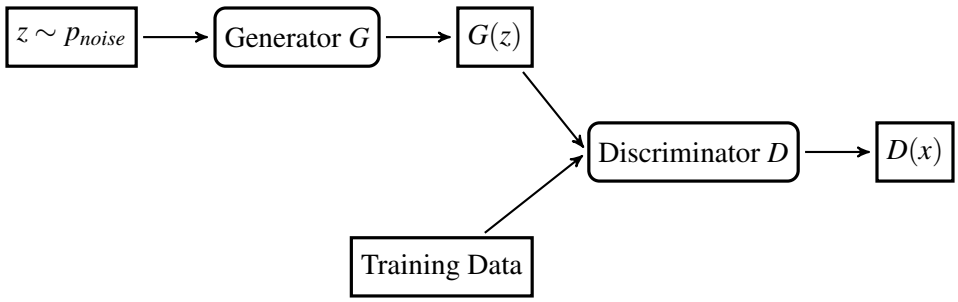


Figure 4.8: Architecture of a Generative Adversarial Network (GAN).

To generate artificial data, a prior z is sampled from a prior distribution p_{noise} (e.g., Gaussian) and fed as input into the generator G. The task of the discriminator D is to output the probability that generated samples stem either from the training data or G. However, G tries to fool D by generating samples that D misclassifies. Hence, the outputs $G(z)$ should look similar to the training data x (i.e. records sampled from p_{data}). This is expressed as a two-player zero-sum game via the following objective function:

$$\min_G \max_D \mathbb{E}_{x\sim p_{data}}[\log D(x)] + \mathbb{E}_{z\sim p_{noise}}[\log(1 - D(G(z)))].$$

Gradients are computed for G and D during training, and usually, after already a few steps of training G produces realistic outputs. A conditional generative model is obtained by providing a condition c (e.g., a class label) as an input both to the generator and the discriminator [GPM+14].

Variational Autoencoders

VAEs [KW13] consist of two networks - an *encoder* E and a *decoder* D. During training each record x is given to the encoder which outputs the mean $E_\mu(x)$ and variance $E_\Sigma(x)$ of a Gaussian distribution. A latent variable z is sampled from this distribution $N(E_\mu(x), E_\Sigma(x))$ and fed into the decoder D. The reconstruction $D(z)$ should be close to the training data record x.

During training two terms need to be minimized. First, the reconstruction error $\|D(z)-x\|$. Second $KL(N(E_\mu(x), E_\Sigma(x))||N(0,1))$, the *Kullback-Leibler divergence* between the distribution of the latent variables z and the unit Gaussian. The second term prevents the network from only memorizing certain latent variables because the distribution should be similar to the unit Gaussian. In practice, both the encoder E and the decoder D are neural networks. Kingma et al. [KW13] provide details on how to train those networks given the training objective with the reparametrization trick. Moreover, they motivate the training objective as a lower bound on the log-likelihood. Sampling from the VAE is achieved by sampling a latent variable $z \sim N(0,1)$ and passing z through the decoder network D. The outputs of the decoder $D(z)$ then serve as samples. Like for GANs, a conditional variant is obtained by providing a condition c as input to the decoder and the encoder.

4.4.2 Monte Carlo Attack

In the following section we introduce the first attack which is applicable to all generative models. The intuition behind the Monte Carlo attack is that the generator G overfits if it tends to output datasets close to the provided training data. Formally, let $U_{\hat{\varepsilon}}(x)$ denote the $\hat{\varepsilon}$-neighborhood of x defined as $U_{\hat{\varepsilon}}(x) = \{x' \,|\, d(x,x') \leq \hat{\varepsilon}\}$ with respect to some distance d. If a sample g of the generative model G is likely to be close to a record x the probability $P(g \in U_{\hat{\varepsilon}}(x))$ is increased. It can be rewritten as

$$P(g \in U_{\hat{\varepsilon}}(x)) = \mathbb{E}_{g \sim p_{generator}}\left(\mathbf{1}_{g \in U_{\hat{\varepsilon}}(x)}\right)$$

and approximated via Monte Carlo integration [Owe13]

$$\hat{f}_{MC-\hat{\varepsilon}}(x) = \frac{1}{n}\sum_{i=1}^{n} \mathbf{1}_{g_i \in U_{\hat{\varepsilon}}(x)}, \tag{4.3}$$

where $g_1, \ldots, g_n$ are samples from $p_{generator}$. Note that samples g_i of the generator G are ignored if their distance to the training data record x is higher than $\hat{\varepsilon}$. In this attack, the estimation $\hat{f}_{MC-\hat{\varepsilon}}(x)$ plays the role of the function $\hat{f}(x)$ attaining higher values for training data records.

An alternative is provided by incorporating the exact distances $d(z_i, x)$ between samples $g_1, \ldots, g_n$ and training data x, and computing

$$\mathbb{E}_{g \sim p_{generator}}\left(-\mathbf{1}_{g \in U_{\hat{\varepsilon}}(x)} \log\left(d(g,x) + \delta\right)\right)$$

where a small δ is chosen to clip off large values ("avoid log(0)") if the distance is zero. The logarithm is to ensure that outliers do not affect the results too much. The Monte Carlo approximation is then given by

$$\hat{f}_{MC-d}(x) = -\frac{1}{n}\sum_{i=1}^{n} \mathbf{1}_{g_i \in U_{\hat{\varepsilon}}(x)} \log d(g_i, x) \quad . \tag{4.4}$$

Here, the estimation $\hat{f}_{MC-d}(x)$ plays the role of the function $\hat{f}(x)$ used to conduct the attack types presented above.

In the case of GANs and VAEs one obtains $g_i \sim p_{generator}$ by sampling from $z_i \sim p_{noise}$ and computing $g_i = G(z_i)$ and $g_i = D(z_i)$, respectively. Note that only a sufficiently large amount of samples has to be provided and no additional information is required. Of course, both attack variants depend on the specification of the distance $d(\cdot,\cdot)$. See below for details.

A further alternative to the attacks discussed could be realized using a Kernel Density Estimator (KDE) [Par62]. In the following we briefly compare the Monte Carlo attack with this metric. An estimation of the likelihood $\hat{f}(x)$ of a data point x using KDE is given by

$$\hat{f}_{KDE}(x) = \frac{1}{nh^d} \sum_{i=1}^{n} K\left(\frac{x-g_i}{h^d}\right), \tag{4.5}$$

where K is typically the Gaussian kernel and h denotes the bandwidth. If this likelihood $\hat{f}_{KDE}(x)$ is significantly higher for training data than for test data the model fails to generalize. Likewise the approximate likelihood values $\hat{f}_{KDE}(x)$ can be used as the function $\hat{f}(x)$ to conduct the single and set MI attack types. However, this attack variation did not perform better than random guessing and is therefore not considered in our evaluation section.

Note that KDE (4.5) can indeed be interpreted as a special case of the proposed distance based method (4.4), where

$$d(x,g_i) = 1/\exp(h^d \cdot K((x-g_i)/h^d)), \text{ and}$$
$$\hat{\varepsilon} = \max_{i=1,\ldots,n} d(x,g_i).$$

As KDE does not perform well for MI against generative models this stresses that choosing the right distance function seems to be key. In contrast to KDE, our attacks exclusively consider samples significantly close to training data x. To fully specify the Monte Carlo attacks concrete distance measures and heuristics for choosing $\hat{\varepsilon}$ are required. We describe our approach for this in the next two subsections.

Distance Measures

Both Monte Carlo (MC) attack variants require a distance function $d(\cdot,\cdot)$ and the distance plays an important role for the success of the MI attack. Therefore, a distance metric suited for the specific data under consideration has to be chosen. For neural networks, image recognition has become a key task and consequently, we formulate distance metrics for image data in the following paragraphs.

Principal Components Analysis. Images are initially represented as a vector of their pixel intensities. A principal component analysis (PCA) is then applied to all vectors in the test dataset. The top 40 components are kept while all other components are discarded. When computing the distance between two new images the PCA transformation is first applied to their vectors of pixel intensities. The Euclidean distance of the two resulting vectors with 40 components each is then defined as the distance of the images.

Histogram of Oriented Gradients. Histogram of Oriented Gradients (HOG) [DT05a] is a computer vision algorithm enabling the computation of feature vectors for images. First, the image is separated into cells. Second, the occurrences of gradient orientations in the cells are counted and a histogram is computed. The histograms are normalized block-wise and concatenated to obtain

a feature vector. Again the Euclidean distance of these vectors is used as image distance. This approach was successfully used by Ebrahimzadeh et al. [EJ14] for an MNIST data classifier.

Color Histogram. According to the intensities in the three color channels, the pixels are sorted into bins. For the pixels of one image, this results in a color histogram (CHIST) which can be represented as a feature vector. The Euclidean distance of these vectors is defined as the image distance.

Heuristics for $\hat{\varepsilon}$

For the attack all pairwise distances $d(x_i, g_j)$ of the records x_i and samples g_j need to be computed. Samples with distances greater than $\hat{\varepsilon}$ to the training data records are ignored. Hence, an appropriate choice of $\hat{\varepsilon}$ is crucial for the success of the attack. We thus formulate two heuristics in the following.

Percentile Heuristic. The first heuristic is to use a fixed percentile of all pairwise distances $d(x_i, g_j)$ as $\hat{\varepsilon}$. By choosing the 0.1% percentile of the distances as $\hat{\varepsilon}$ we can ensure that the corresponding samples in an $\hat{\varepsilon}$-neighborhood are sufficiently close. Note that the MC-$\hat{\varepsilon}$ and MC-d approaches are not necessarily equivalent if this heuristic is employed.

Median Heuristic. The second heuristic avoids the need to choose an additional parameter such as the percentile value. Again, the idea is to exploit the measured distances in the Monte Carlo computation. In this approach, the median of the minimum distance to each record x_i for all the generated samples g_j is chosen:

$$\hat{\varepsilon} = \underset{1 \leq i \leq 2M}{\text{median}} \left(\min_{1 \leq j \leq n} d(x_i, g_j) \right). \tag{4.6}$$

If $\hat{\varepsilon}$ is chosen according to the median heuristic (4.6) the results of MC-$\hat{\varepsilon}$ and MC-d are equivalent in both the single and set MI types as there are always exactly M records with $\hat{f}_{MC-\hat{\varepsilon}}(x_i) > 0$ and $\hat{f}_{MC-d}(x_i) > 0$. A comparison of the MC attack variants is provided in the evaluation in Section 4.4.4.

4.4.3 Reconstruction Attack

The reconstruction attack is solely applicable to VAEs. During training, reconstructions $D(z)$ close to the current training data record x are rewarded. Hence, for training data more precise reconstructions of the VAE can be expected. However, the outputs $D(z)$ are not deterministic. They depend on the latent variable z which is sampled from the distribution $N(E_\mu(x), E_\Sigma(x))$ whose parameters are the output of the encoder network E. Hence, we repeat this process n times and set

$$\hat{f}_{\text{rec}}(x) = -\frac{1}{n} \sum_{i=1}^{n} \|D(z_i) - x\| \tag{4.7}$$

where z_i $(i = 1, \ldots, n)$ are samples from the distribution $N(E_\mu(x), E_\Sigma(x))$. This term is frequently used in practice as part of the loss function of VAEs. One of the contributions of this work is to apply this loss to the problem of membership inference. Specifically, the function $\hat{f}_{\text{rec}}(x)$ is applied in the attack types as the discriminating function $\hat{f}(x)$. This induces the Reconstruction attack. Note that this attack considers a strong adversary $\mathcal{A}$ with access to the VAE model.

4.4.4 Evaluation

The two MI attacks formulated in this work are evaluated in comparison to the white and black-box MI attacks of Hayes et al. [HMDD19] against generative models trained on MNIST in Section 4.4.4.

The *white box attack* is solely applicable to GANs and requires access to the discriminator D. The intuition behind this attack is that D tends to attain higher outputs $D(x)$ if the record x was part of the training data due to the indirect reward during training. Specifically, the discriminator D plays the role of the function $\hat{f}(x)$ in this attack. The *black box attack* overcomes the limitation of the white box attack in that it requires no access to D. It is therefore not solely applicable to GANs. For the black box attack, an auxiliary GAN is trained with samples $g_1, \ldots, g_n$ from the target model and the discriminator D' of this newly trained model is used in a white box manner. Consequently, records with increased $D'(x)$ are considered part of the training data. Hence, the discriminator D' serves as the function $\hat{f}(x)$ attaining higher values for training than for test data. Though not explicitly tested in the original paper, the black-box attack is also applicable to other generative models such as VAEs since it only requires access to samples $g_1, \ldots, g_n$. In experiments, the white box attack performed significantly better than the black box attack [HMDD19].

In general, our MC attacks outperformed state of the art, i.e. the white box attack of Hayes [HMDD19], for MNIST which is are considered a very hard dataset to attack with membership inference due to its simplicity. Since it is an upper bound for the accuracy, also the black box attack is outperformed. Since several parameters have to be chosen before the attacks are applied a study of the effect of these parameters is presented in Section 4.4.4. Moreover, additional experiments on VAEs trained on the MNIST dataset are provided in Sections 4.4.4 and 4.4.4.

Setup and Dataset

We evaluated the attacks of Hayes et al. [HMDD19], the Monte Carlo and the Reconstruction attacks for differing 10% subsets of the MNIST dataset. The simple nature of MNIST has proven to result in low MI precision in previous work. The remaining 90% are used as test set in both the single and set MI attack type. To ensure a fair comparison we executed all experiments repeatedly and report standard deviations. Neural networks are implemented with tensorflow [ABC+16], and for the HOG and PCA computations, the python libraries scikit-image and scikit-learn [PVG+11] are used. Experiments were run on Amazon Web Services p2.xlarge (GAN) and c5.2xlarge (VAE) instances.

We first describe the datasets and models used before analyzing the parameters of the attacks.

MNIST MNIST is a standard dataset in machine learning and computer vision consisting of $70,000$ labeled handwritten digits which are separated into $60,000$ training and $10,000$ test records.[4] Each digit is a 28×28 grayscale image. In all subsequent datasets only a 10% subset of the training images is used for training to provoke overfitting. The remaining 90% of the training data are used as test data to compute the accuracies of the attacks. The actual MNIST test data are only used to define the PCA transformation for the PCA based distance. This ensures that the distance is not influenced by the specific choice of the training data or the remaining 90%. Attacks are performed against two state of the art generative models, namely GANs (cf. Section 4.4.1) and VAEs (cf. Section 4.4.1). For the GAN we employ the widely used deep convolutional genera-

[4] http://yann.lecun.com/exdb/mnist/

Heuristic/Percentile	HOG-based distance			
	GAN Monte Carlo-d	GAN Monte Carlo-$\hat{\varepsilon}$	VAE Monte Carlo-d	VAE Monte Carlo-$\hat{\varepsilon}$
Median	63.76±3.83	63.76±3.83	83.50±2.43	83.50±2.43
0.01%	63.76±3.68	66.11±3.70	81.00±2.59	82.25±2.50
0.10%	63.76±3.71	62.08±3.65	74.50±2.90	71.75±2.98
1.00%	60.07±3.84	59.73±3.86	59.50±3.24	54.00±3.29

Table 4.5: Set accuracies for $\mathcal{R}$ with HOG-based distance depending on $\hat{\varepsilon}$ values

Heuristic/Percentile	PCA-based distance			
	GAN Monte Carlo-d	GAN Monte Carlo-$\hat{\varepsilon}$	VAE Monte Carlo-d	VAE Monte Carlo-$\hat{\varepsilon}$
Median	74.84±3.25	74.84±3.25	99.75±0.25	99.75±0.25
0.01%	74.84±3.31	71.94±3.40	95.50±1.34	91.75±1.80
0.10%	64.84±3.69	59.68±3.78	94.75±1.52	95.50±1.43
1.00%	47.42±3.77	51.61±3.76	60.75±3.21	58.50±3.29

Table 4.6: Set accuracies for $\mathcal{R}$ with PCA-based distance depending on $\hat{\varepsilon}$ values

tive adversarial network (DCGAN) [RMC15] architecture which aims to improve both stability and quality of GANs for image generation. This network relies on convolutional neural networks (CNN) which are state of the art for many computer vision tasks. We trained the DCGAN for 500 epochs (i.e., until convergence) with a mini batch size of 128.[5] For the VAE we apply a standard architecture[6] with 90% Dropout and a mini batch size of 128. Due to the different convergence behavior, the VAE is only trained for 300 epochs. For both models, GAN and VAE, we utilize the conditional variant s.t. we can control which digit is generated.

Attack Parameters

The effects of the attack parameters are analyzed in the following. Specifically, for the MC attacks the effect of the heuristic for setting $\hat{\varepsilon}$ and the number of samples n for the Monte Carlo integration are studied. We expect these to be similar for both GANs and VAEs. Hence, the analysis is restricted to the case of VAEs. For the Reconstruction attack, we study how the number of samples n for the reconstruction error estimation affects the accuracy.

Monte Carlo Attack

The set MI accuracies against VAEs trained on MNIST for different choices of $\hat{\varepsilon}$ are reported in Table 4.5 and 4.6 for $\mathcal{R}$. Note that the results of the MC-$\hat{\varepsilon}$ and MC-d attacks do not differ significantly. This suggests that the main contribution is the introduction of $\hat{\varepsilon}$ effectively ignoring samples which are further than $\hat{\varepsilon}$ away from the training records. In the case of the median heuristic, the two MC attack variants yield equivalent performances as expected. However, the median heuristic outperforms the percentile heuristic.

Besides the heuristic for $\hat{\varepsilon}$, a sample size for the Monte Carlo approximation has to be chosen. Hence, we also analyze the performance of the MC-$\hat{\varepsilon}$ attack depending on the sample size. Again, the MC-$\hat{\varepsilon}$ attack is equivalent to the MC-d attack in the case of the median heuristic. The single and set accuracies are stated in Figure 4.9 for $\mathcal{A}$ and $\mathcal{R}$, respectively. In general, higher percentile values ignore fewer samples since $\hat{\varepsilon}$ is increased. A smaller sample size is required to achieve op-

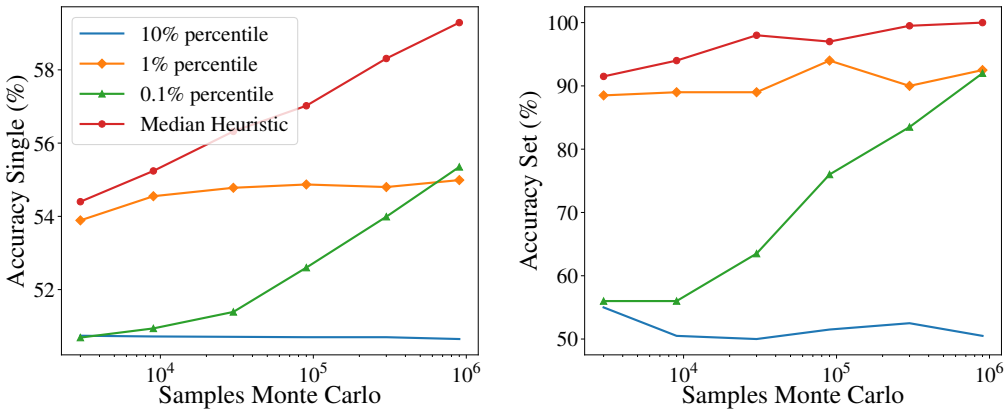
[5] We used https://github.com/yihui-he/GAN-MNIST as a starting point.

[6] We used https://github.com/hwalsuklee/tensorflow-mnist-VAE as a starting point.

timal accuracy for these percentiles. However, the accuracy of higher percentile values is inferior to the ones of lower percentile values.

For example, the 10% percentile attack already reaches its optimum in the minimal case of 3,000 samples and the 1% percentile saturates at 10^4 samples. The 0.1% percentile approach is gaining higher accuracies and does not level off at 10^6 samples. It is noticeable that the median heuristic always outperforms the other heuristics. We conjecture this heuristic to level off at a higher sample size. However, in practice there is a trade-off between computational effort and accuracy of the attack. To study the effect 20 experiments for the median heuristic with 10^7 samples each are conducted, achieving a single record MI accuracy of $59.80 \pm 3.50\%$ for $\mathcal{A}$ and a set MI accuracy of $100.00 \pm 0.00\%$ for $\mathcal{R}$. In the subsequent experiments, we always use 10^6 samples for the Monte Carlo simulations.

The median heuristic is superior to the percentile heuristic for all sample sizes. Moreover, no parameter like the percentile is required. Thus, in all subsequent experiments we apply the median heuristic for which the MC-$\hat{\varepsilon}$ and MC-d attacks are equivalent. We refer to these equivalent approaches simply as *MC attack*.



(a) Adversarial actor: Single MI

(b) Regulartory actor: Set MI

Figure 4.9: MC attack accuracy (differing scales) on MNIST with PCA based distance against VAEs depending on sample size.

Reconstruction Attack

We also study the effect of the sample size n to approximate the reconstruction error

$$\hat{f}_{\text{rec}}(x) = -\frac{1}{n}\sum_{i=1}^{n} \|D(z_i) - x\|. \tag{4.8}$$

In preliminary experiments even small sample sizes of $n = 300$ yielded good accuracies. This suggests that the estimator $\hat{f}_{\text{rec}}(x)$ is accurate enough for small n values. To ensure optimal results we conduct the subsequent experiments with $n = 10^6$ for the reconstruction attack against a VAE trained on MNIST.

Results on MNIST

Following the description of the Monte Carlo estimators of Section 4.4.2, we computed the distance of every record x_i to each of the samples $g_1, \ldots, g_n$. However, the label per record is known

and as we used the conditional variants we could control which digit was generated by the target model under consideration. Thus, we only used samples representing the same digit as the current record x_i.

For the MC $\hat{\varepsilon}$-values we either used a certain percentile of all measured distances or a dynamic $\hat{\varepsilon}$ based on the median heuristic (cf. Section 4.4.2).

(a) GAN after 500 epochs

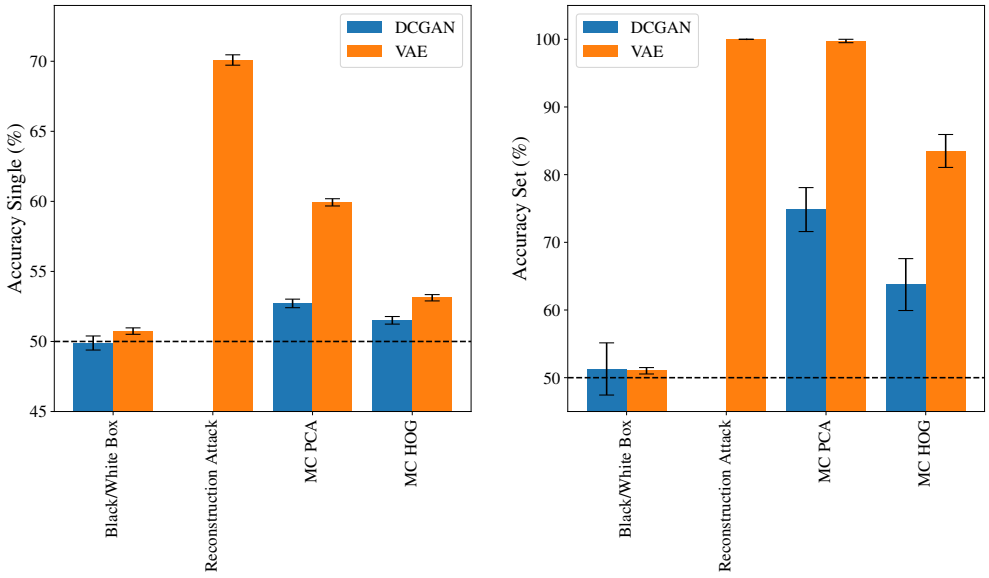
(b) VAE after 300 epochs

Figure 4.10: Generated digits of the GAN and VAE after training on the MNIST dataset.

With a relatively simple dataset such as MNIST it is very hard to find the subtle replications of the training data if the model overfits. In more feature-rich image datasets we assume it should be easier to identify overfitting as even a visual inspection could suffice to recognize the replication of training images. In Figure 4.10 generated digits of both models trained with a random 10% subset of MNIST can be seen. It appears to be nearly impossible to infer membership of training data by visual inspection as there are no remarkable replicated characteristics such as specific colors, elements in fore- or background etc. Note that the samples of both the GAN and the VAE are visually appealing.

Having analyzed the parameters of our proposed attacks, we now compare their accuracies with the recent white-box and black-box attacks of [HMDD19]. To stabilize the results 10 different 10% subsets of the MNIST data are chosen as training data for the GAN and VAE models. For every subset 10 single and set MI attacks are conducted with $M = 100$. While we apply the white-box attack against the GAN, we are limited to the black-box attack in case of the VAE as the latter model does not feature a discriminator. In order to test the black-box attack, a new GAN is trained with 10^6 samples from the target VAE.

For the Monte Carlo estimator $\hat{f}_{MC}$ we use the PCA and HOG based distances introduced in Section 4.4.2. The CHIST distance is not applicable since MNIST solely consists of grayscale images. As described in the previous section, we use $n = 10^6$ samples and the median heuristic. The resulting accuracies are depicted in Figure 4.11. The dotted horizontal baseline at 50% is the average success rate of random guessing. In general, the accuracies of single MI for $\mathcal{A}$ are significantly lower than those of set MI for $\mathcal{R}$. Furthermore, all attacks are much more successful if applied against VAEs instead of GANs. This suggests that in general there is less overfitting in GANs. This observation is consistent with the Annealed Importance Sampling measurements by Wu et al. [WBSG16].



(a) Single Membership Inference

(b) Set Membership Inference

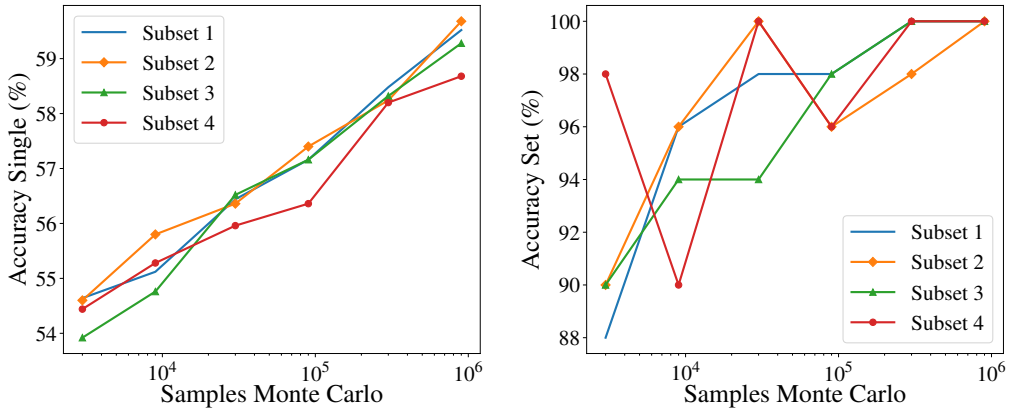
Figure 4.11: Average accuracy (differing scales) of the attacks on MNIST in the single and set experiments with standard deviation.

The black-box and white-box attack do not perform significantly better than the baseline in both experiments. The MC attack clearly outperforms these attacks. When used with PCA distance our MC attack can even infer set membership with nearly 100% accuracy against a VAE. For the GAN the accuracy is still about 75%. In general, accuracies are inferior if the HOG distance is used. As a side fact, the Monte Carlo based attacks with PCA distance take ≈ 7 minutes each on a p2.xlarge instance on AWS. Currently, at the cost of 0.90 US $ per hour, the attacks only cause minor costs. The specialized Reconstruction attack is superior to the MC attack in the case of the VAE yielding $\approx 70\%$ and 100% in the single and set MI attack, respectively. The high accuracies of the attacks we proposed make them especially attractive for the regulatory use case defined in Section 4.2.3.

Effect of Subset Choice

It is unclear how the specific choice of the MNIST 10% subset influences the accuracy of the MC attack. In Figure 4.12 the average MC attack performance with PCA distance against VAEs trained on different subsets are plotted. Attack performances seem independent of the specific subset. We also conduct an F-test to evaluate whether the single accuracy means of the four VAEs are different at 10^6 samples resulting in a p-value ≈ 0.64. Hence, the hypothesis that the means are equal can be accepted with high probability, i.e. the choice of the subset does not significantly influence the attack results. We conclude that the accuracy depends on the size of the training data rather than its specific members.

We remark that in the experiment setups $M = 100$ samples of the 10% subset of the training data and 100 samples of the remaining 90% training data are chosen. The set MI experiments yield high accuracies. Therefore, if a regulator suspects that some dataset was used for training a model this can be recognized with the novel attacks even though other data might have been part of the training data as well. This is an analogous case to the experiment described. Though of course



(a) Adversarial actor: Single MI

(b) Regulartory actor: Set MI

Figure 4.12: MC attack accuracy (differing scales) on MNIST with PCA distance depending on sample size for four different training subsets.

Size	Monte Carlo (PCA dist.)		Reconstruction attack	
	Single	Set	Single	Set
40%	50.79±0.27	57.50±3.24	57.35±0.37	98.50±1.11
20%	57.05±0.32	94.75±1.39	62.23±0.38	100.00±0.00
10%	59.93±0.26	99.75±0.25	70.09±0.37	100.00±0.00

Table 4.7: Accuracies depending on MNIST training data size

more training data was used, we focus on 100 samples. It is very likely that the inappropriately used data are not the only data used to train the model. Hence, the practicability of the MC attack is increased since the regulator does not need to know all the training data to prove that a certain subset was used.

Effect of Training Data Size and Regularization — Mitigations

We also investigate how the size of the training dataset influences the success of the attacks for the MNIST dataset. For this, five VAEs are trained with 20 experiments each since the effect should be similar for GANs. The results for the MC attack and Reconstruction attack are depicted in Table 4.7. When using 40% of the training data instead of the usual 10% the accuracy shrinks from 60% to 51% for single MI and from nearly 100% to only about 58% for set MI in the case of the MC attack. As expected, for 20% the effects are less significant. Clearly, more training data would further reduce the effectiveness of the attacks. However, in the case of the Reconstruction attack, the effects are less significant. Even if 40% are used the set accuracy is still about 100% meaning that the Reconstruction attack is more robust.

In general, the performance declines suggest that generative models make use of the additional information provided by additional training data. Similar effects were observed before in the case of the white-box attack [HMDD19].

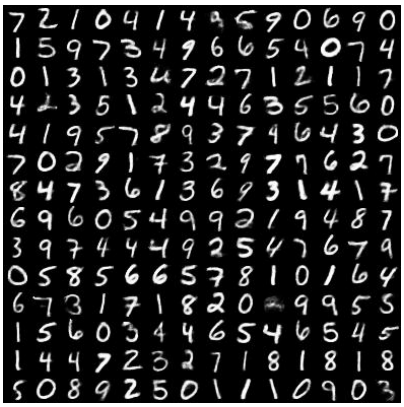
However, often in practice the amount of training data is a bottleneck for training generative models. In consequence, one could use regularization methods to improve the generalization such as *dropout* [SHK+14]. In the case of dropout, certain neurons are switched off during training with given probability to increase the resistance of the network. In the standard case we already

Rate	Monte Carlo (PCA dist.)		Reconstruction attack	
	Single	Set	Single	Set
50%	51.45±0.26	64.75±3.19	53.77±0.34	86.00±3.18
70%	53.17±0.29	78.50±2.71	58.31±0.40	97.00±1.56
90%	59.93±0.26	99.75±0.25	70.09±0.37	100.00±0.00

Table 4.8: Accuracies depending on MNIST Dropout Keep Rates

use dropout with a keep probability of 90% both in the encoder and decoder of the VAE. We also conduct experiments for the MC and Reconstruction attack at lower keep rates of 70% and 50%. The accuracy in the set MI type decreases to 79% at a keep probability of 70% and to 65% at an even reduced keep probability of 50% for the MC attack. Again, the effects are less significant for the Reconstruction attack still yielding $\approx$ 86% set MI accuracy for a 50% keep rate. Detailed results are reported in Table 4.8. The results indicate that dropout can indeed be used in practice to mitigate the proposed MI attacks. This can also be observed in the case of the white-box attack [HMDD19]. However, a lower keep probability also causes the generated images to get increasingly blurry as depicted in Figure 4.13. Hence, there is an inherent trade-off between high image quality and low MI attack accuracies.

(a) GAN on MNIST after 500 epochs



(b) VAE on MNIST after 300 epochs

(c) VAE, 90% Keep Probability

(d) VAE, 70% Keep Probability

(e) VAE, 50% Keep Probability

Figure 4.13: Generated samples of the trained models.

5. Conclusions

We have observed that anonymization can be a valuable tool to mitigate disclosure risks for personal or sensitive data. However, proper anonymization is hard due to a number of inherent challenges. Within MOSAICrOWN we provide and advance anonymization methods with quantitative privacy parameters that allow to balance privacy and utility. These methods provide different trade-offs, as detailed in this Deliverable, and can be used in combination for data collections (e.g., by applying them on different parts of the data).

The first line of such methods is represented by the *generalization based approaches* (k-anonymity, ℓ-diversity, t-closeness) for syntactic privacy. While these approaches have interpretable privacy guarantees and have seen wide adoption for anonymized microdata release, they consider specific aspects of the problem and remain vulnerable to some attacks, and their applicability to high-dimensional data can be limited.

The second line of anonymization methods is represented by *perturbation based algorithms* (ε-differential privacy/CDP, (ε, δ)-differential privacy, ε-local differential privacy/LDP) for semantic privacy. While these comparatively young algorithms have seen wide adoption for perturbation of statistical functions, their mathematically strict privacy guarantees are comparatively hard to interpret, which we aim to alleviate.

We paid special attention to privacy in the context of machine learning. To support data owners this document used membership inference threat models to quantify privacy violations and privacy threats in deep learning with the goal of identifying privacy interpretation techniques for data marketplaces. We outlined two general techniques for data sanitization in deep learning: local differential privacy (LDP) and (central) differential privacy (CDP). LDP is suited for anonymization of microdata training records for deep learning. In contrast, CDP allows to anonymize the deep learning optimization function during training (i.e., anonymized macrodata release). Our initial experiment shows that data scientists should compare the privacy-accuracy trade-off for LDP and CDP per dataset, and consider the relative privacy-accuracy trade-off for LDP and CDP as the ratio of losses in accuracy and privacy over privacy parameters ε. The choice of either differential privacy technique depends on whether the party which is training the machine learning model is trusted (CDP) or not (LDP) and what accuracy one wants to achieve (i.e., CDP for high accuracy with low ε). The scope of MOSAICrOWN does not rule out trusted parties (or hybrid models where cryptographic tools replace such a party), and thus supports both LDP and CDP. Furthermore, we formulated and evaluated attacks (Monte Carlo attack, Reconstruction attack) to evaluate both overfitting and information leakage of generative models.

Bibliography

[ABC+16] M. Abadi, P. Barham, J. Chen, Z. Chen, A. Davis, J. Dean, M. Devin, S. Ghemawat, G. Irving, M. Isard, and M. Kudlur. TensorFlow: A system for large-scale machine learning. In *Proc. of the 12th USENIX Conference on Operating Systems Design and Implementation*, Berkeley, CA, USA, 2016. USENIX Assoc.

[ACG+16] M. Abadi, A. Chu, I. Goodfellow, H.B. McMahan, I. Mironov, K. Talwar, and L. Zhang. Deep Learning with Differential Privacy. In *Proc. of Conference on Computer and Communications Security*, CCS, 2016.

[Agg05] C. C. Aggarwal. On *k*-anonymity and the curse of dimensionality. In *Proc. of the International Conference on Very Large Data Bases*, VLDB, 2005.

[BA05] R. J. Bayardo and R. Agrawal. Data privacy through optimal *k*-anonymization. In *Proc. of ICDE 2005*, Tokyo, Japan, April 2005.

[BGRK19] D. Bernau, P.-W. Grassal, J. Robl, and F. Kerschbaum. Assessing Differential Privacy in Deep Learning with Membership Inference. *arXiv preprint arXiv:1912.11328*, 2019.

[CDF+12] V. Ciriani, S. De Capitani di Vimercati, S. Foresti, G. Livraga, and P. Samarati. An OBDD approach to enforce confidentiality and visibility constraints in data publishing. *Journal of Computer Security*, 2012.

[CDFS07] V. Ciriani, S. De Capitani di Vimercati, S. Foresti, and P. Samarati. *k*-Anonymity. In T. Yu and S. Jajodia, editors, *Secure Data Management in Decentralized Systems*. Springer-Verlag, 2007.

[Cor18] G. Cormode. Building blocks of privacy: Differential privacy mechanisms. `http://dimacs.rutgers.edu/~graham/pubs/slides/privdb-tutorial.pdf`, 2018.

[DFJ+10] S. De Capitani di Vimercati, S. Foresti, S. Jajodia, S. Paraboschi, and P. Samarati. Fragments and loose associations: Respecting privacy in data publishing. *Proc. of the VLDB Endowment*, September 2010.

[DFJ+14] S. De Capitani di Vimercati, S. Foresti, S. Jajodia, G. Livraga, S. Paraboschi, and P. Samarati. Fragmentation in presence of data dependencies. *IEEE Transactions on Dependable and Secure Computing*, November/December 2014.

[DFJ+15] S. De Capitani di Vimercati, S. Foresti, S. Jajodia, G. Livraga, S. Paraboschi, and P. Samarati. Loose associations to increase utility in data publishing. *Journal of Computer Security*, 2015.

[DFLS12] S. De Capitani di Vimercati, S. Foresti, G. Livraga, and P. Samarati. Data privacy: Definitions and techniques. *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems*, December 2012.

[DMNS06] C. Dwork, F. Mcsherry, K. Nissim, and A. Smith. Calibrating noise to sensitivity in private data analysis. In *Proc. of TCC 2006*, New York, NY, USA, March 2006.

[dMRSP15] Y.-A. de Montjoye, L. Radaelli, V. Kumar Singh, and A. "Sandy" Pentland. Unique in the shopping mall: On the reidentifiability of credit card metadata. *Science*, 2015.

[DN03] I. Dinur and K. Nissim. Revealing information while preserving privacy. In *Proceedings of the 22nd Symposium on Principles of Database Systems*, PODS, 2003.

[DR14] C. Dwork and A. Roth. The Algorithmic Foundations of Differential Privacy. *Foundations and Trends in Theoretical Computer Science*, 2014.

[DRV10] C. Dwork, G.N. Rothblum, and S. Vadhan. Boosting and Differential Privacy. In *IEEE 51st Annual Symposium on Foundations of Computer Science*, 2010.

[DT05a] N. Dalal and B. Triggs. Histograms of oriented gradients for human detection. In *Proc. of the 2005 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Piscataway, NJ, USA, 2005. IEEE.

[DT05b] J. Domingo-Ferrer and V. Torra. Ordinal, continuous and heterogeneous k-anonymity through microaggregation. *Data Mining and Knowledge Discovery*, 2005.

[Dwo06] C. Dwork. Differential privacy. In *Proceedings of the International Colloquium on Automata, Languages and Programming*, ICALP, 2006.

[EJ14] R. Ebrahimzadeh and M. Jampour. Efficient handwritten digit recognition based on histogram of oriented gradients and svm. *International Journal of Computer Applications*, 2014.

[EPK14] Ú. Erlingsson, V. Pihur, and A. Korolova. RAPPOR: Randomized Aggregatable Privacy-Preserving Ordinal Response. In *Proceedings of the 21st Conference on Computer and Communications Security (CCS)*. ACM Press, 2014.

[Fan18] L. Fan. Image pixelization with differential privacy. In *Proc. of Conference on Data and Applications Security and Privacy (DBSEC)*, 2018.

[Fed05] Federal Committee on Statistical Methodology. *Statistical policy working paper 22 (Second Version)*. USA, December 2005. Report on Statistical Disclosure Limitation Methodology.

[FJR15] M. Fredrikson, S. Jha, and T. Ristenpart. Model Inversion Attacks that Exploit Confidence Information and Basic Countermeasures. In *Proc. of Conference on Computer and Communications Security (CCS)*, 2015.

[FLJ+14] M. Fredrikson, E. Lantz, S. Jha, S. Lin, D. Page, and T. Ristenpart. Privacy in Pharmacogenetics: An End-to-End Case Study of Personalized Warfarin Dosing. In *Proc. of USENIX Security Symposium*, 2014.

[GBC16] I. Goodfellow, Y. Bengio, and A. Courville. *Deep Learning*. MIT Press, 2016. `http://www.deeplearningbook.org`.

[GC95] Y. Grandvalet and S. Canu. Comments on "Noise injection into inputs in back propagation learning". *IEEE Transactions on Systems, Man, and Cybernetics*, 1995.

[Gol06] P. Golle. Revisiting the uniqueness of simple demographics in the US population. In *Proc. of WPES 2006*, Alexandria, VA, USA, October 2006.

[GPM+14] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, Da. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio. Generative adversarial nets. In *Proc. of Advances in Neural Information Processing Systems 27 (NIPS)*. NIPS Foundation, 2014.

[HHB19] B. Hilprecht, M. Härterich, and D. Bernau. Monte Carlo and Reconstruction Membership Inference Attacks against Generative Models. *Proc. on Privacy Enhancing Technologies (PoPETs)*, 2019.

[Hil12] K. Hill. How target figured out a teen girl was pregnant before her father did. `http://www.forbes.com/sites/kashmirhill/2012/02/16/how-target-figured-out-a-teen-girl-was-pregnant-before-her-father-did/`, 2012.

[HMDD19] J. Hayes, L. Melis, G. Danezis, and E. De Cristofaro. LOGAN: Membership Inference Attacks Against Generative Models. *Proc. on Privacy Enhancing Technologies (PoPETs)*, 2019.

[JJR11] M. Jawurek, M. Johns, and K. Rieck. Smart metering de-pseudonymization. In *Proceedings of the Annual Computer Security Applications Conference*, ACSAC, New York, NY, USA, 2011. ACM.

[KLN+08] S.P. Kasiviswanathan, H.K. Lee, K. Nissim, S. Raskhodnikova, and A. Smith. What can we learn privately? *SIAM Journal on Computing*, 2008.

[KM11] D. Kifer and A. Machanavajjhala. No free lunch in data privacy. In *Proc. of SIGMOD 2011*, Athens, Greece, June 2011.

[KOV17] P. Kairouz, S. Oh, and P. Viswanath. The Composition Theorem for Differential Privacy. *IEEE Transactions on Information Theory*, 2017.

[KW13] D.P. Kingma and M. Welling. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*, 2013.

[LDR05] K. LeFevre, D.J. DeWitt, and R. Ramakrishnan. Incognito: Efficient full-domain *k*-anonymity. In *Proc. of SIGMOD 2005*, Baltimore, MD, USA, June 2005.

[LDR06] K. LeFevre, D.J. DeWitt, and R. Ramakrishnan. Mondrian multidimensional *k*-anonymity. In *Proc. of ICDE 2006*, Atlanta, GA, USA, April 2006.

[LLSY16] N. Li, M. Lyu, D. Su, and W. Yang. Differential Privacy: From Theory to Practice. In *Synthesis Lectures on Information Security, Privacy, and Trust*. Morgan Claypool, 2016.

[LLV07] N. Li, T. Li, and S. Venkatasubramanian. t-Closeness: Privacy beyond k-anonymity and ℓ-diversity. In *Proc. of ICDE 2007*, Istanbul, Turkey, 2007.

[LLZM12] T. Li, N. Li, J. Zhang, and I. Molloy. Slicing: A new approach for privacy preserving data publishing. *IEEE Transactions on Knowledge and Data Engineering*, 2012.

[Mat92] K. Matsuoka. Noise injection into inputs in back-propagation learning. *IEEE Transactions on Systems, Man, and Cybernetics*, 1992.

[Mir17] I. Mironov. Renyi differential privacy. In *Proc. of Computer Security Foundations Symposium (CSF)*, 2017.

[MKGV07] A. Machanavajjhala, D. Kifer, J. Gehrke, and M. Venkitasubramaniam. ℓ-Diversity: Privacy beyond k-anonymity. *ACM Transactions on Knowledge Discovery from Data*, March 2007.

[MT07] F. McSherry and K. Talwar. Mechanism Design via Differential Privacy. In *Annual IEEE Symposium on Foundations of Computer Science (FOCS)*. IEEE, 2007.

[NS08] A. Narayanan and V. Shmatikov. Robust de-anonymization of large sparse datasets. *Proceedings of the IEEE Symposium on Research in Security and Privacy*, 2008.

[Owe13] A.B. Owen. *Monte Carlo theory, methods and examples*. 2013.

[PAE+17] N. Papernot, M. Abadi, Ú. Erlingsson, I. Goodfellow, and K. Talwar. Semi-supervised Knowledge Transfer for Deep Learning from Private Training Data. In *Proc. of Conference on Learning Representations (ICLR)*, 2017.

[Par62] E. Parzen. On estimation of a probability density function and mode. *The annals of mathematical statistics*, 1962.

[PVG+11] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 2011.

[RMC15] A. Radford, L. Metz, and S. Chintala. Unsupervised representation learning with deep convolutional generative adversarial networks. *arXiv preprint arXiv:1511.06434*, 2015.

[Sam01] P. Samarati. Protecting respondents' identities in microdata release. *IEEE Transactions on Knowledge and Data Engineering*, November/December 2001.

[SDSM14] J. Soria-Comas, J. Domingo-Ferrer, D. Sánchez, and S. Martínez. Enhancing data utility in differential privacy via microaggregation-based k-anonymity. *The VLDB Journal*, 2014.

[SHK+14] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov. Dropout: A simple way to prevent neural networks from overfitting. *The Journal of Machine Learning Research*, 2014.

[SSSS17] R. Shokri, M. Stronati, C. Song, and V. Shmatikov. Membership inference attacks against machine learning models. In *Proc. of Symposium on Security and Privacy*, 2017.

[The17] The Guardian Online. The Guardian view on Google's NHS grab: legally inappropriate. `https://www.theguardian.com/commentisfree/2017/may/17/the-guardian-view-on-googles-nhs-grab-legally-inappropriate`, 2017.

[Tor04] V. Torra. Microaggregation for categorical variables: a median based approach. In *Proc. of PSD 2004*, Barcelona, Spain, June 2004.

[TZJ+16] F. Tramèr, F. Zhang, A. Juels, M.K. Reiter, and T. Ristenpart. Stealing machine learning models via prediction apis. In *Proc. of USENIX Security Symposium*, 2016.

[War65] S.L. Warner. Randomized response: a survey technique for eliminating evasive answer bias. *Journal of the American Statistical Association*, 1965.

[WBLJ17] T. Wang, J. Blocki, N. Li, and S. Jha. Locally Differentially Private Protocols for Frequency Estimation. In *Proc. of USENIX Security Symposium*, 2017.

[WBSG16] Y. Wu, Y. Burda, R. Salakhutdinov, and R. Grosse. On the quantitative analysis of decoder-based generative models. *arXiv preprint arXiv:1611.04273*, 2016.

[WH00] R. Wirth and J. Hipp. Crisp-dm: Towards a standard process model for data mining. In *Proc. of Conference on practical applications of knowledge discovery and data mining*, 2000.

[XT06] X. Xiao and Y. Tao. Anatomy: Simple and effective privacy preservation. In *Proc. of VLDB 2006*, Seoul, Korea, September 2006.

[YFJ17] S. Yeom, M. Fredrikson, and S. Jha. The unintended consequences of overfitting: Training data inference attacks. *arXiv preprint arXiv:1709.01604*, 2017.