Towards Privacy-preserving Machine Learning: Generative Modeling and Discriminative Analysis

Dingfan Chen

A dissertation submitted towards the degree
Doctor of Engineering Science (Dr.-Ing.)
of the Faculty of Mathematics and Computer Science
of Saarland University

Saarbrücken, 2023.

Date of Colloquium:	23.04.2024
Dean of the Faculty:	Prof. Dr. Jürgen Steimle
Chair of the Committee:	Prof. Dr. Isabel Valera
Reviewers:	Prof. Dr. Mario Fritz
	Prof. Dr. Antti Honkela
	Prof. Dr. Catuscia Palamidessi
Academic Assistant:	Dr. Raouf Kerkouche

Dedicated to my family

Abstract

The digital era is characterized by the widespread availability of rich data, which has fueled the growth of machine learning applications across diverse fields such as computer vision, natural language processing, speech recognition, and recommendation systems. Nevertheless, data sharing is often at odds with serious privacy and ethical issues. The sensitive nature of much of this data, which includes personal information on mobile devices, confidential medical treatments, and financial records, demands a cautious approach to data sharing. This caution is not just a matter of ethical responsibility but also a legal mandate, with stringent regulations like the General Data Protection Regulation (GDPR) and the Health Insurance Portability and Accountability Act (HIPAA) establishing barriers that, while protective, can also impede the pace of technological progress. Additionally, the growing trend of using large-scale, web-scraped datasets to build machine learning models raises serious concerns. This approach, often without proper supervision, can unintentionally include private information and copyrighted content not meant for public use, posing risks of privacy violations and legal complications.

This presents a dilemma: the demand for extensive data to power complex machine learning algorithms conflicts with the need to protect personal privacy and intellectual property rights. Addressing this challenge is critical not only to maintain public trust but also to ensure that the progress in machine learning is sustainable, responsible, and aligned with societal values. To this end, this thesis investigates such privacy risks and seeks out viable solutions that permit data sharing within strict privacy constraints. Specifically, this thesis examines three intertwined perspectives within the realm of data privacy in machine learning: (1) **privacy-preserving generative modeling**, which focuses on generating synthetic data while ensuring rigorous privacy guarantees; (2) **privacy attack and defense**, dedicated to assessing and understanding the actual privacy risks inherent in machine learning models; as well as (3) **applications**, which emphasizes the implementation of privacy-preserving training methods on real-world sensitive datasets.

Firstly, we explore privacy-preserving generative modeling, with the goal of creating synthetic data that maintains characteristics of the population distribution relevant for particular tasks, while adhering to rigorous privacy guarantee. Such synthetic data can be utilized and analyzed as if it were the real data, thus enabling progress and facilitating reproducible research in sensitive domains. The foundation of our approach is rooted in differentially private (DP) generative modeling. Our advancements involve the development of sanitization protocols dedicated to generative modeling (Chapter 2), the design of a generation framework that reduces the inherent complexity of DP training (Chapter 3), and offering a novel unified perspective that presents a joint design surface for systematic investigation into future advancements in the field (Chapter 4).

Secondly, we delve into privacy attack and defense mechanisms, particularly focusing on real-world simulations of privacy threats. Our work primarily scrutinizes the membership inference attack, which attempts to determine whether a particular data sample was part of a machine learning model's training set. This type of attack serves as a crucial metric for identifying potential privacy leaks and establishing the lower bound of privacy cost in auditing privacy-preserving algorithms. This thesis presents an in-depth analysis of such attacks against generative models (Chapter 5), and devises effective countermeasures for general discriminative

machine learning models (Chapter 6).

Lastly, we focus on adapting our analytical and design strategies in DP learning mechanisms for practical applications, particularly within the medical field where data privacy is paramount. This research yields findings and insights that guide the development of privacy-centric technologies, tailored for complex real-world data distributions (Chapter 7).

Zusammenfassung

Das digitale Zeitalter ist gekennzeichnet durch die weit verbreitete Verfügbarkeit von umfangreichen Daten, die das Wachstum von Anwendungen des maschinellen Lernens in verschiedenen Bereichen wie Computer Vision, Verarbeitung natürlicher Sprache, Spracherkennung und Empfehlungssystemen angetrieben haben. Dennoch steht das Teilen von Daten oft im Widerspruch zu ernsthaften Datenschutz- und ethischen Fragen. Die sensible Natur vieler dieser Daten, zu denen persönliche Informationen auf mobilen Geräten, vertrauliche medizinische Behandlungen und Finanzaufzeichnungen gehören, erfordert einen vorsichtigen Ansatz beim Datenaustausch. Diese Vorsicht ist nicht nur eine Frage der ethischen Verantwortung, sondern auch ein gesetzliches Mandat, wobei strenge Vorschriften wie die Datenschutz-Grundverordnung (DSGVO) und der Health Insurance Portability and Accountability Act (HIPAA) Barrieren errichten, die zwar schützend sind, aber auch das Tempo des technologischen Fortschritts behindern können. Darüber hinaus wirft der wachsende Trend, großangelegte, aus dem Web extrahierte Datensätze zum Aufbau von Maschinenlernmodellen zu verwenden, ernsthafte Bedenken auf. Dieser Ansatz, oft ohne angemessene Aufsicht, kann unbeabsichtigt private Informationen und urheberrechtlich geschütztes Material enthalten, das nicht für die öffentliche Nutzung bestimmt ist, und birgt Risiken von Datenschutzverletzungen und rechtlichen Komplikationen.

Dies stellt ein Dilemma dar: Die Nachfrage nach umfangreichen Daten zur Speisung komplexer Maschinenlernalgorithmen steht im Konflikt mit dem Bedürfnis, persönliche Privatsphäre und geistige Eigentumsrechte zu schützen. Die Bewältigung dieser Herausforderung ist entscheidend, um nicht nur das öffentliche Vertrauen aufrechtzuerhalten, sondern auch sicherzustellen, dass der Fortschritt im maschinellen Lernen nachhaltig, verantwortungsbewusst und im Einklang mit gesellschaftlichen Werten ist. Zu diesem Zweck untersucht diese Arbeit solche Datenschutzrisiken und sucht nach praktikablen Lösungen, die den Datenaustausch innerhalb strenger Datenschutzbeschränkungen ermöglichen. Insbesondere untersucht diese Arbeit drei miteinander verflochtene Perspektiven im Bereich des Datenschutzes beim maschinellen Lernen: (1) **Datenschutzfreundliche Datenfreigabe**, die sich auf die Erzeugung synthetischer Daten konzentriert und gleichzeitig strenge Datenschutzgarantien gewährleistet; (2) **Datenschutzangriff und -verteidigung**, gewidmet der Bewertung und dem Verständnis der tatsächlichen Datenschutzrisiken, die in Maschinenlernmodellen inhärent sind; sowie (3) **Anwendungen**, die die Implementierung von datenschutzfreundlichen Trainingsmethoden auf realen sensiblen Datensätzen hervorheben.

Erstens erforschen wir die Datenschutz gewahrende Datenfreigabe mit dem Ziel, synthetische Daten zu erstellen, die Eigenschaften der Bevölkerungsverteilung beibehalten, die für bestimmte Aufgaben relevant sind, und dabei strenge Datenschutzgarantien einhalten. Solche synthetischen Daten können genutzt und analysiert werden, als wären sie echte Daten, was Fortschritte ermöglicht und reproduzierbare Forschung in sensiblen Bereichen erleichtert. Die Grundlage unseres Ansatzes ist in der differentiell privaten (DP) generativen Modellierung verankert. Unsere Fortschritte umfassen die Entwicklung von Sanitisierungsprotokollen, die der generativen Modellierung gewidmet sind (Chapter 2), das Design eines Generierungsframeworks, das die inhärente Komplexität des DP-Trainings reduziert (Chapter 3), und bietet eine neuartige einheitliche Perspektive, die eine gemeinsame Designoberfläche für systematische Untersuchungen zukünftiger Fortschritte im Bereich präsentiert (Chapter 4).

Zweitens beschäftigen wir uns mit Angriffs- und Verteidigungsmechanismen für den Datenschutz, insbesondere konzentrieren wir uns auf realitätsnahe Simulationen von Datenschutzbedrohungen. Unsere Arbeit untersucht hauptsächlich den Mitgliedschafts-Inferenzangriff, der zu bestimmen versucht, ob eine bestimmte Datenprobe Teil des Trainingssets eines maschinellen Lernmodells war. Dieser Angriffstyp dient als entscheidendes Maß für die Identifizierung potenzieller Datenschutzlecks und zur Festlegung der Untergrenze der Datenschutzkosten bei der Überprüfung von Datenschutzalgorithmen. Diese Dissertation präsentiert eine eingehende Analyse solcher Angriffe gegen generative Modelle (Chapter 5) und entwickelt wirksame Gegenmaßnahmen für allgemeine maschinelle Lernmodelle (Chapter 6).

Zuletzt konzentrieren wir uns darauf, unsere analytischen und gestalterischen Strategien in DP-Lernmechanismen für praktische Anwendungen anzupassen, insbesondere im medizinischen Bereich, wo Datenschutz von größter Bedeutung ist. Diese Forschung liefert Erkenntnisse und Einsichten, die die Entwicklung von datenschutzorientierten Technologien leiten, die für komplexe reale Datenverteilungen zugeschnitten sind (Chapter 7).

Acknowledgements

First and foremost, I would like to express my deepest gratitude to my supervisor, Prof. Mario Fritz, for his unwavering support, profound insights, and invaluable assistance throughout my Ph.D. journey. He has given me a lot of freedom to choose the research topics that I am interested in. During our meetings, he gave me many insightful suggestions on research, life, and career. He provided me with many opportunities to improve myself, including joining the ProGENEGEN project, attending the ELSA program and top-tier machine learning and security conferences. He also provided me with a lot of guidance and encouragement when I was facing difficulties. All these qualities undeniably establish Mario as an extraordinary advisor.

I am deeply thankful to all the members of my thesis committee, Prof. Antti Honkela, Prof. Catuscia Palamidessi and Prof. Isabel Valera. Their presence on my committee has been a privilege, and their constructive suggestions and feedback during my defense have been invaluable.

I would also like to thank my wonderful colleagues at CISPA. They are all very talented and nice. I particularly want to thank our group members: Raouf, Hui-po, Sahar, Shadi, Hossein, Tobias, Sarath, Tejumade, Zhixiong, Yuxuan. We had in-depth discussions, weekly seminars, reading groups, retreats, lunches, and social events together. Beyond our group, I would like to express special thanks to Dr. Yang Zhang, Dr. Mang Zhao, Yuan Xin, Rui Ye, Xiaowen Jiang, Min Chen as well as all other current and former colleagues. Additionally, I am grateful to the CISPA staff, including those in the front office, travel department, and IT service.

Furthermore, I express profound thanks to the Max Planck Institute, especially to Prof. Bernt Schiele and Connie Balzert, for their substantial support and for granting access to computational resources.

I must also acknowledge my brilliant external collaborators, Dr. Tribhuvanesh Orekondy, Dr. Ning Yu, Dr. Xiaoyi Chen, Vladislav Skripniuk, Derui Zhu, Dr. Lei Ma, Zongxiong Chen, Dr. Qing Li, Dr. Marie Oestreich, Dr. Matthias Becker, and Prof. Jens Grossklags. Their dedication has been crucial to the success of our joint efforts.

Outside CISPA, I would like to express special thanks to my friends in Saarbrücken and other parts of the world — Miaoran Zhang, Xueting Li, Yaoyao Liu, Ning Yu, Wenjia Xu, Di Chen, Fangzhou Zhai, Yue Fan, Dawei Zhu, Xiaoyu Shen, Yongqin Xian, Yang He, Yiting Xia, Yanlong Huang — I owe a debt of gratitude for their support in both my personal and research life. Their friendship has been a cornerstone of my Ph.D. experience.

Lastly, I dedicate this thesis to my family. Their unconditional love and support have been the bedrock of my studies in Germany. It is their backing that has enabled my journey thus far.

Contents

1 Introduction

Contents

The advancement of machine learning (ML) technologies in recent years has been remarkable, predominantly powered by the exponential growth in big data. The vast amounts of rich datasets are pivotal for the training of advanced ML algorithms. These algorithms, especially in the fields of deep learning and reinforcement learning, require extensive data to realize their expressive capabilities and predictive effectiveness, driving forward innovative developments across a variety of industries and practical uses. However, the rapid expansion of ML applications poses significant challenges regarding data privacy and adherence to regulatory standards. The sharing of data, particularly when it involves sensitive personal information, is stringently limited by privacy regulations, such as the General Data Protection Regulation (GDPR) in Europe and the Health Insurance Portability and Accountability Act (HIPAA) in the United States. Such restrictions can largely hinder the pace of ML advancements, especially in sensitive areas.

Moreover, the common practice of constructing machine learning models from large-scale, web-scraped datasets also raises significant concerns. Such methods, often lacking rigorous oversight, can inadvertently incorporate data from entities that do not anticipate or permit such usage. The consequences are significant, potentially leading to severe privacy violations if individual data is used without permission, and risking legal issues by using copyrighted material without the necessary authorization.

Such a landscape poses a paradox — the need for rich data to feed the ever-growing complexity of machine learning algorithms is at odds with the imperative to protect individual privacy and intellectual property. This tension necessitates innovative solutions that can reconcile the hunger for data with the ethical and legal frameworks designed to safeguard society. Addressing this tension requires the development of novel methodologies that can bridge the gap between data requirements for ML and adherence to ethical standards. Potential solutions include differential privacy techniques to sanitize data, federated learning approaches that train models on decentralized data, and synthetic data generation that offers a privacy-compliant alternative for model training.

The goal of this thesis is to address the aforementioned challenges and develop algorithms that enable learning while ensuring privacy guarantees. Our key idea is anchored in the principles of differential privacy (DP), which offers a rigorous mathematical foundation for quantifying and controlling the privacy risks associated with the learning processing. The key contributions are briefly summarized below regarding the main topics of this thesis.

- In Part I, **Privacy-Preserving Generative Modeling**, we examine the generation of synthetic, sanitized data that mimics the real data's distributional characteristics while complying with rigorous DP guarantees.
  - In Chapter 2, we propose a novel gradient sanitization scheme for training deep generative models, which naturally exploits the intrinsic properties of the training pipeline and objective of generative adversarial networks to eliminate information loss during DP training.
  - In Chapter 3, we introduce a generative framework for constructing sanitized data that directly optimizes for downstream analysis objectives to simplify the complexity of DP generation and enhance its utility.
  - In Chapter 4, we present a unified view of DP generative models that provides a joint design surface enabling systematic exploration for the future design of new variants.
- In Part II, **Privacy Attack and Defense**, we investigate the potential privacy leakage by executing practical attacks, with a focus on membership inference attacks, and propose countermeasures against such attacks.
  - In Chapter 5, we present a systematic investigation of membership inference attacks targeting advanced generative models, with a specific focus on diffusion models.
  - In Chapter 6, we propose defense mechanisms that effectively defend against membership inference attacks while maintaining model utility.
- In Part III, **Application**, we study the application of DP generative methods on real-world datasets with complicated distributions and their implications.
  - In Chapter 7, we investigate the application of DP generative methods on gene expression data and establishes a systematic evaluation along several different dimensions.

For the rest of this chapter, we discuss each topic and explain our contributions. Then, we provide an outline of the thesis with relevant publications.

1.1 Privacy-preserving Generative Modeling

The principal aim of this part is to devise practical methodologies for publishing data with stringent privacy guarantees, grounded in the principles of DP data publishing [51, 52, 58], where a sanitized form of the data with rigorous privacy guarantees is publicly released for downstream usage. Such sanitized synthetic data can be leveraged as if it were the real data, analyzed with established toolchains, and can also be shared openly with the research community, facilitating technological advance and reproducible research in sensitive domains.

Traditionally, the sanitization algorithms are designed for capturing low-dimensional statistical characteristics and target at specific downstream tasks (e.g., answering linear queries [179, 73, 19, 223]), which greatly restricts the expressiveness of the released data distribution and fail to generalize to novel tasks unanticipated by the publisher. Instead, inspired by the recent successes of deep generative models in learning high-dimensional representations, the latest works [24, 32, 239, 245, 14] adopt deep generative models as the underlying generation algorithm, achieving promising results in sanitizing high-dimensional samples for general purpose. Unfortunately, existing methods are still struggling to generate high-fidelity sanitized data that is useful for many real-world application scenarios. One major challenge is the shortage of data samples when training the deep generative models under a privacy budget acceptable for real-world deployment. This is especially problematic when considering datasets in sensitive domains (e.g., medical treatment records, genetic data), which typically have a limited amount of data samples but follow complex high-dimensional distributions.

1.1.1 Challenges

Despite significant progress in recent years, privacy-preserving generative modeling still faces many open challenges that significantly impede its practical application. In the following, we outline several key challenges that this thesis aims to address.

- **Hyperparameter Tuning & Training Stability.** Training generative models under DP constraints typically involves a delicate and fragile tuning process. Hyperparameters, such as the gradient clipping bound required by standard DP gradient sanitization method [1], require careful selection. An improper choice can disrupt the entire training process due to the inherent instability of training deep generative models. Furthermore, the straightforward application of vanilla gradient sanitization techniques often results in substantial information loss. This loss occurs due to the vast variance and long-tailed distribution of gradient norms, making it exceedingly difficult to achieve satisfactory performance using existing methods. Addressing this issue is crucial for effectively balancing privacy preservation and model performance.
- **Modeling Complexity.** As a price of the great modeling capacity, deep generative models are data-intensive and generally require a massive amount of diverse data samples to achieve high generation quality. This is even more problematic when the privacy constraint is imposed: given a privacy budget, the randomness required for hiding information about individuals (which is normally implemented as injecting noise into the gradients) scales inversely to the sample size. For a privacy budget that is acceptable for real-world deployment (i.e., ϵ<10 as suggested by [165]), the often insufficient dataset sizes in real-world applications fail to compensate for the distortion caused by DP noise, leading to unstable training and a significant reduction in utility. This is particularly relevant for datasets in sensitive domains (e.g., medical treatment records, genetic data), which typically have limited amount of data samples but follow complex high-dimensional distributions. This characteristic makes the additional randomness from DP a significant hindrance, rendering existing training methods ineffective.
- **Fragmented Research.** Towards designing models that are better compatible with the privacy target, recent works commonly tailor the training objective for the private case [24, 70, 32, 129], all building on top of a generic generator framework. However, research is fragmented as contributions have been made in different domains, different modeling paradigms, different metric/discriminator choices, and different data modalities. Insufficient exploration has been conducted to elucidate the inherent advantages and limitations of distinct methodological categories and to determine their appropriate deployment in scenarios with varying levels of privacy requirements and data characteristics. So far, a unified view of private generators is missing in the literature, albeit that it naturally provides a joint design space to systematically explore novel architecture and leverage the strengths across different methods.

1.1.2 Our Contributions

Towards mitigating the aforementioned key challenges and generally improving privacy-preserving generative modeling, we made the following contributions.

- **In Chapter 2**, we tackle the first challenge, *hyperparameter tuning and training stability*, by proposing a novel gradient sanitization scheme that distorts the information more precisely and obviates the need for delicate hyperparameter tuning of the clipping bound. Instead of distorting all parameter gradients that occur during model training, we propose to sanitize only the gradients with respect to the synthetic samples. This subset is essential for updating the generator and ensures DP guarantees for both the generator and the generated synthetic dataset. Moreover, our new formulation leverages the theoretical property derived from the training objective of Wasserstein GANs, wherein the gradients to be sanitized exhibit a specific norm consistent with the Lipschitz Property of the adopted Wasserstein objective [8, 67]. This property enables us to naturally bypass the search for a clipping norm required for DP training. Experimental evaluations on various datasets demonstrate that our method significantly outperforms state-of-the-art approaches in generation quality and downstream utility. Additionally, our scheme can be seamlessly applied in both centralized and decentralized settings, providing user-level DP guarantees [113] even under an untrusted server in the decentralized scenario.
- **In Chapter 3**, we address the second challenge, *modeling complexity*, by proposing an alternative perspective of DP high-dimensional data generation and a novel DP generation framework. Rather than aiming to fit the entire data distribution as is typically done, our framework instead optimizes a small and representative set of samples. This optimization is guided by discriminative information from downstream tasks. Compared to the default approach, our framework simplifies the task and thus reduces the modeling complexity by leveraging dedicated discriminative information and reducing the dimensionality of the optimization problem. Our framework effectively bridges the gap between utility and generality in private generative and discriminative modeling. Experimental results demonstrate that our work significantly improves sample utility and offers superior practicality for real-world applications.
- **In Chapter 4**, we address the third challenge, *fragmented research*, by presenting a unified view of DP deep generative modeling that categorizes and streamlines existing approaches. This creates a joint design surface that allows for the systematic development of new methods tailored to diverse application scenarios. Within this framework, we engage in a critical analysis of the strengths and weaknesses of various approaches, as well as the intrinsic correlations between them. Our objective is to illuminate key considerations and to inspire future research that will propel the field forward. We then explore possible future directions for DP data generation, aiming to direct the collective efforts of researchers towards significant advances in privacy-preserving machine learning. The chapter concludes by highlighting promising research paths that could enhance the field and aid researchers in navigating the complexities of data privacy.

1.2 Privacy Attacks and Defenses

The primary aim of a privacy attacker is to derive confidential details about individuals from the training datasets used in machine learning models. A key form of this attack is the membership inference attack (MIA), which determines whether a given query data sample was included in the analysis study or the training of a model. The research into privacy attacks, particularly MIAs, is increasingly vital due to the pervasive integration of data-centric technologies into our daily lives. Moreover, such research is essential for assessing and enhancing the privacy protections of systems that handle sensitive data. MIAs also have significant connections to other critical concepts in privacy and data security, such as differential privacy, intellectual property rights protection [194], and the unintended disclosure of training data content [26, 27]. Additionally, the investigation of privacy attacks plays a vital role in auditing the (lower bound of) privacy costs of learning systems, which is closely associated with Part I of our research that aims at providing models with strict upper bound of the privacy costs against potential adversaries.

While stringent DP guarantees serve as a natural defense against privacy attacks, they may not be the optimal approach. Specifically, the protection level afforded by DP can exceed what is necessary to prevent practical attackers, which in turn could compromise the model utility. Therefore, developing practical defenses that optimize the privacy-utility trade-off is of paramount importance.

1.2.1 Challenges

- **Practicability and Effectiveness of Attacks.** The central challenge in research on privacy attacks is twofold: First, it requires determining the data quantities an attacker can access and exploit, which dictates the practicability of the attack. This involves understanding the attack surface across various contexts, particularly those reflecting the real-world use of the targeted model or system. Second, it necessitates identifying the specific data quantities an attacker should leverage to maximize the attack's effectiveness. This process involves pinpointing the system's most susceptible points, where a breach could reveal the most information about the sensitive data processed by the learning system. The intersection of these two aspects, *practicability and effectiveness*, unveils a complex spectrum of challenges that have shaped the trajectory of research in this domain.
- **Privacy-utility Trade-off of Defenses.** Defense mechanisms typically subject to a principled privacy-utility trade-off, where enhanced privacy protection often comes at the cost of reduced model utility. This tendency is observable in existing defenses, which tend to be either less effective in defending against attacks (like regularization techniques), or notably compromise model utility (such as DP mechanisms). Specifically, while adopting DP mechanisms offers a strong defense by providing strict worst-case guarantees against arbitrarily powerful attackers that exceed practical limits, DP methods inevitably sacrifice model utility [170, 87, 74, 34, 97, 86] and meanwhile increase computation burden [64, 41]. In general, developing utility-preserving and computationally efficient defenses that target at practically realizable attacks has become the primary focus of contemporary research in this domain.

1.2.2 Our Contributions

In summary, we made the following contributions towards designing practical and effective attacks and defenses.

- **In Chapter 5**, we pioneer the investigation of practical and effective Membership Inference Attacks (MIAs) against diffusion models, the state-of-the-art technology for AI-based image processing systems with emerging commercial uses in daily life. We begin by elucidating the unique properties and usage patterns of diffusion models, which give rise to new attack surfaces that have not been addressed in previous studies. Instead, we thoroughly examine attack vectors and identify three major attack scenarios that are most representative and prevalent in practice, considering real-world APIs as reference. Moreover, we design novel attacks tailored to diffusion models based on their unique characteristics, consistently surpassing existing solutions by a substantial margin across various settings. Our attacks are based on easily obtainable or estimable quantities and are both straightforward and highly effective, supported by a theoretical basis. Extensive evaluation demonstrates the consistent efficacy of our approach across varied scenarios, yielding key insights into components that impact the vulnerability of real-world models. Lastly, our findings indicate a dual implication: while the common practice of sharing diffusion models presents a markedly high privacy risk, strong attack strategies exist that enable the monitoring of sample usage during model training, which is crucial for intellectual property protection and the early detection of data leakage.

- **In Chapter 6**, we introduce a novel defense mechanism that effectively protects against a wide range of MIAs without compromising the defender's model utility. Our strategy hinges on two pivotal insights: firstly, the efficacy of a Bayesian optimal attack only depends on the sample loss under mild assumptions of the model parameters [180]; secondly, a pronounced disparity between the training and testing loss provably elevates membership privacy risks [244]. By strategically moderating the target training loss to better align with the test loss, our method reduces the loss gap, diminishing the distinctiveness between the training and testing loss distributions, and thereby mitigates various attack vectors. Moreover, our approach allows for a utility-preserving (or even improving) defense, greatly improving upon previous results. As a practical benefit, our approach is easy to implement and can be seamlessly integrated into existing classification models with minimal overhead. Comprehensive evaluations across diverse datasets demonstrate that our method notably surpasses state-of-the-art defenses, offering superior protection against MIAs and an improved balance between privacy and utility. To our knowledge, this is the first approach that comprehensively counters a broad spectrum of attacks while preserving model utility.

1.3 Application

While significant advancements have been achieved in the field of privacy-preserving data generation in the past few years, the application of such techniques to real-world complex high-dimensional data distribution with varied modalities remains an open challenging. Notably, existing DP generation techniques have shown promising outcomes when applied to real-world demographic data[1], which inherently possesses properties conducive to privacy (e.g., large sample sizes, low feature dimensions, highly correlated feature distributions) are present. However, the application of such techniques to sensitive domains such as medical data is largely under-explored. Specifically, the unique and diverse characteristics of medical data call for dedicated efforts towards developing effective representation, modeling, and evaluation techniques that have not been thoroughly investigated in existing literature.

1.3.1 Challenges

- **Data Complexity.** Real-world datasets in sensitive domains typically possess a diverse range of data types and characteristics that complicate the application of DP techniques. For instance, the data may include heterogeneous feature types, from numerical to categorical, requiring complex pre-processing. Making such pre-processing steps DP can be non-trivial to implement and may incur considerable information loss. Furthermore, the often large

[1] https://www.census.gov/data.html

number of features and their intricate inter-feature interactions are challenging for DP methods to capture, potentially necessitating extensive feature engineering or specialized model architectures. Morevoer, the typically limited size of datasets further complicates DP learning and exacerbates the risk of overfitting. Lastly, complex data structures, such as correlated samples, may demand the development of specialized privacy models and adapted privacy notions, adding further complexity to the analysis and implementation of DP techniques.

- **Evaluation of Generation Methods.** The evaluation of generation methods poses significant challenges, particularly when dealing with real-world data that exhibits diverse feature types and value ranges. Unlike the straightforward visual inspection for image data, assessing general type of synthetic data requires complex metrics that capture multiple dimensions. Existing studies have mainly focused on downstream utility, i.e., whether synthetic data can substitute real data for downstream analysis such as training machine learning models. However, this single-dimensional evaluation may be insufficient, often leading to over-optimized assessments. A comprehensive evaluation that reliably determines the effectiveness of generation methods and the quality of synthetic data remains underexplored in current literature, despite its critical importance for practical applications. Moreover, the development of robust and reliable metrics for such evaluations is an active area of research that necessitates further effort.

1.3.2 Our Contributions

In summary, we made the following contributions towards investigating the feasibility of applying DP generation methods on real-world complex data distributions from sensitive domains.

- **In Chapter 7**, we initiate a systematic analysis of the application of existing representative DP generation methods in their natural application scenarios, specifically focusing on real-world gene expression data. Our thorough examination covers five distinct DP methods, from deep generative models to parametric density estimations and marginal-based methods, assessing them across downstream utility, statistical properties, and biological plausibility. Our findings illustrate the unique properties of each method, their strengths and weaknesses, and suggest new avenues for research. Our evaluation presents intriguing results: most methods are capable of achieving seemingly reasonable downstream utility, according to the standard evaluation metrics considered in existing literature, yet none of the DP methods is able to accurately capture the biological characteristics of the real dataset. This observation suggests a potential over-optimistic assessment of current methodologies in this field and underscores a pressing need for future enhancements in model design.

1.4 Outline

In this section, we provide an overview of the thesis by briefly summarizing each chapter and drawing a connection between them. We also note the respective publications and collaborations with other researchers.

Chapter 1, Introduction: This chapter presents an overview of the principal research topics explored in this thesis, outlining the interconnections among them and summarizing the key contributions.

Part I, Privacy-preserving Generative Modeling

Chapter 2: Gradient Sanitized Approach. In this chapter, we address the difficulties of training, including hyperparameter tuning and stability, in differentially private generative modeling. We propose a novel gradient sanitization framework and present GS-WGAN, a DP generative adversarial network. Our approach sanitizes a condensed form of the gradients to reduce the information loss incurred by private learning and exploits the analytical properties of the training objectives to bypass the need for searching for a gradient clipping bound required by gradient-based DP training.

The content of this chapter corresponds to the NeurIPS 2020 publication with the title *"GS-WGAN: A Gradient Sanitized Approach for Learning Differentially Private Generators"* [32]. Dingfan Chen is the first author of this paper, under the supervision of Prof. Mario Fritz and in collaboration with Tribhuvanesh Orekondy from Max Planck Institute for Informatics.

Chapter 3: Private Set Generation. In this chapter, we tackle the issue of modeling complexity in DP generative modeling. By leveraging the principled trade-off between the generality of general-purpose distribution fitting and the utility of task-specific modeling, we offer a new perspective for DP generative modeling. Our approach aims to directly optimize for downstream utility, which simplifies the task and reduces the dimension of the optimization problem.

The content of this chapter corresponds to the NeurIPS 2022 publication with the title *"Private Set Generation with Discriminative Information"* [29]. Dingfan Chen is the first author of this paper, under the supervision of Prof. Mario Fritz and in collaboration with Raouf Kerkouche from CISPA.

Chapter 4: Unified View. In this chapter, we work towards addressing the issue of fragmented research in the field of DP generative modeling by presenting a unified view that systematizes existing DP generative models. We propose a taxonomy that categorizes existing methods based on their supported privacy barriers and associated threat models, and we discuss in detail the inherent advantages and disadvantages of each category.

The content of this chapter corresponds to the pre-print with the title *"A Unified View of Differentially Private Deep Generative Modeling"* [30]. Dingfan Chen is the first author of this paper, under the supervision of Prof. Mario Fritz and in collaboration with Raouf Kerkouche from CISPA.

Part II, Privacy Attacks and Defenses

Chapter 5: Privacy Attack for Advanced Generative Models. In this chapter, we investigate effective privacy attacks pertinent to practical usage scenarios of advanced generative models, with a focus on membership inference attacks on diffusion models. We systematically explore the attack surface and present attack strategies that are tailored to various practical settings.

The content of this chapter corresponds to the pre-print with the title *"Data Forensics in Diffusion Models: A Systematic Analysis of Membership Privacy"* [262]. Dingfan Chen is a co-first author of this paper, under the supervision of Prof. Mario Fritz and in collaboration with Derui Zhu and Prof. Jens Grossklags from Technical University of Munich.

Chapter 6: Defense against Privacy Attacks for General Discriminative Models. In this chapter, we study effective and practical defense mechanisms against privacy attacks in practice, with a focus on membership inference attacks on classification models. We improve the principled privacy-utility trade-off by relaxing the training objective to a more achievable level during the training process of the target model, which reduces overfitting and thereby prevents potential privacy leakage without compromising model utility.

The content of this chapter corresponds to the ICLR 2022 paper with the title *"RelaxLoss: Defending Membership Inference Attacks without Lossing Utility"* [33]. Dingfan Chen is the first author of this paper, supervised by Prof. Mario Fritz and in collaboration with Ning Yu, affiliated with Salesforce Research, University of Maryland, and Max Planck Institute for Informatics

Part III, Application

Chapter 7: Privacy-preserving Generation of Gene Expression Data. In this chapter, we embark on a thorough investigation into the application of DP generation techniques to real-world gene expression data, presenting a comprehensive evaluation framework that encompasses aspects such as downstream utility, statistical fidelity, and biological plausibility, and critically investigating diverse representative methods to highlight their intrinsic advantages and disadvantages.

This chapter corresponds to the PETs 2024 paper with the title *"Towards Biologically Plausible and Private Gene Expression Data Generation"* [31]. Dingfan Chen is a co-first author of this paper, under the supervision of Prof. Mario Fritz and in collaboration with Dr. Marie Oestreich and Matthias Becker from the German Center for Neurodegenerative Diseases, as well as Tejumade Afonja and Raouf Kerkouche from CISPA.

Chapter 8: Conclusion and Future Work. This chapter presents a comprehensive conclusion of the thesis, offering insights into the main findings and contributions. Additionally, it provides a vision for future research directions aimed at facilitating the development of trustworthy and privacy-preserving machine learning systems.

1.5 Publications

The content of this thesis has previously appeared in the following publications, ordered as outlined above:

- **[32] Dingfan Chen**, Tribhuvanesh Orekondy, Mario Fritz. "GS-WGAN: A Gradient-Sanitized Approach for Learning Differentially Private Generators". *In Proceedings of Advances in Neural Information Processing Systems (**NeurIPS**)*, 2020.
- **[29] Dingfan Chen**, Raouf Kerkouche, Mario Fritz. "Private Set Generation with Discriminative Information". *In Proceedings of Advances in Neural Information Processing Systems (**NeurIPS**)*, 2022.
- **[30] Dingfan Chen**, Raouf Kerkouche, Mario Fritz. "A Unified View of Differentially Private Deep Generative Modeling". *Transactions on Machine Learning Research (**TMLR**)*, 2024.
- **[262]** Derui Zhu*, **Dingfan Chen*** (equal contribution), Jens Grossklags, Mario Fritz. "Data forensics in diffusion models: A systematic analysis of membership privacy". ***Arxiv** Pre-print*, 2023.

- [33] **Dingfan Chen**, Ning Yu, Mario Fritz. “RelaxLoss: Defending Membership Inference Attacks without Losing Utility”. *International Conference on Learning Representations (**ICLR**)*, 2022.
- [31] **Dingfan Chen***, Marie Oestreich*, Tejumade Afonja* (equal contribution), Raouf Kerkouche, Matthias Becker, Mario Fritz. “Towards Biologically Plausible and Private Gene Expression Data Generation”. *Proceedings on Privacy Enhancing Technologies (**PoPETs**)*, 2024.

Further contributions were made to the following works not discussed in this thesis:

- [34] **Dingfan Chen**, Ning Yu, Yang Zhang, Mario Fritz. “Gan-leaks: A Taxonomy of Membership Inference Attacks against Generative Models”. *ACM SIGSAC Conference on Computer and Communications Security (**CCS**)*, 2020.
- [153] Marie Oestreich, **Dingfan Chen**, Joachim L Schultze, Mario Fritz, Matthias Becker. “Privacy Considerations for Sharing Genomics Data”. ***EXCLI Journal***, 2021.
- [248] Ning Yu*, Vladislav Skripniuk*, **Dingfan Chen**, Larry Davis, Mario Fritz. “Responsible Disclosure of Generative Models Using Scalable Fingerprinting”. *International Conference on Learning Representations (**ICLR**)*, 2022.
- [4] Tejumade Afonja, **Dingfan Chen**, Mario Fritz. “MargCTGAN: A "Marginally" Better CTGAN for the Low Sample Regime”. *DAGM German Conference on Pattern Recognition (**GCPR**)*, 2023.
- [228] Hui-Po Wang, **Dingfan Chen**, Raouf Kerkouche, Mario Fritz. "Fed-GLOSS-DP: Federated, Global Learning using Synthetic Sets with Record Level Differential Privacy". *Proceedings on Privacy Enhancing Technologies (**PoPETs**)*, 2024.

I

Part 1: Privacy-preserving Generative Modeling

In the first part of the thesis, we concentrate on privacy-preserving generative modeling, delving into strategies to alleviate the substantial complexity that arises when integrating rigorous differential privacy guarantees into the training of generative models. We work towards developing practical algorithms that mitigate or bypass the principled challenge of privacy-utility trade-off, stemming from an enhanced gradient sanitization method (Chapter 2), a refined generative framework (Chapter 3), as well as a more in-depth understanding of the inherent characteristics of the algorithm elements (Chapter 4).

In Chapter 2, we introduce an improved gradient sanitization scheme for training deep generators with DP guarantees, addressing the key challenges of hyperparameter tuning and training stability. Our scheme reduce the sanitization scope to a condensed subset of essential gradients, retaining more information and lessening the negative impacts of DP's inherent randomness. Furthermore, our scheme utilizes the analytical properties of the training objective to directly bound the sensitivity required by DP, obviating the need for extensive hyperparameter search and enhancing training stability.

In Chapter 3, we present a novel framework for DP generative modeling that aims to address the intrinsic complexity issue typically associated with such models. We introduce Private-Set, which directly optimizes a small set of representative samples for downstream utility instead of aiming to fit the complete data distribution for general purposes. Our approach simplifies the modeling problem and reduces the dimensionality of the optimization task, which contributes to notably improved model performance.

In Chapter 4, we propose a unified view of existing DP generation algorithms, effectively bridging disparate strands of research within this domain. We introduce a taxonomy that rigorously classifies different methods according to their respective privacy barriers and associated threat models, complemented by a critical assessment of the strengths and limitations of each category. Our unified view presents a joint design space that facilitates the structured exploration of potential future developments in the field of DP data release.

GS-WGAN: A Gradient-Sanitized Approach for Learning Differentially Private Generators 2

Contents

The wide-spread availability of rich data has fueled the growth of machine learning applications in numerous domains. However, growth in domains with highly-sensitive data (e.g., medical) is largely hindered as the private nature of data prohibits it from being shared. To this end, we propose *Gradient-sanitized Wasserstein Generative Adversarial Networks* (GS-WGAN), which allows releasing a sanitized form of the sensitive data with rigorous privacy guarantees. In contrast to prior work, our approach is able to distort gradient information more precisely, and thereby enabling training deeper models which generate more informative samples. Moreover, our formulation naturally allows for training GANs in both centralized and federated (i.e., decentralized) data scenarios. Through extensive experiments, we find our approach consistently outperforms state-of-the-art approaches across multiple metrics (e.g., sample quality) and datasets.

This chapter is based on [32]: As the first author of [32], Dingfan Chen proposed the project idea, implemented the algorithms, conducted all experiments, and served as the main writer of the paper. This paper was published in NeurIPS 2020 and has received more than 130 citations so far. It has been widely recognized as a baseline framework for many papers, being actively used and extended by following works presented in top-tier conferences, e.g., [227, 24]. The code for this work can be found on GitHub [1].

2.1 Introduction

Releasing statistical and sensory data to a broad community has contributed towards advances in numerous machine learning (ML) techniques e.g., object recognition (ImageNet [45]), language modeling (RCV [110]), recommendation systems (Netflix ratings [16]). However, in many sensitive domains (e.g., medical, financial), similar advances are often held back as the private nature of collected data prohibits release in its original form. Privacy-preserving data publishing [52, 58, 14] provides a reasonable solution, where only a sanitized form of the original data (with rigorous privacy guarantees) is publicly released.

Traditionally, sanitization is performed in a differentially private (DP) framework [51].

[1] https://github.com/DingfanChen/GS-WGAN

The sanitization method employed is often hand-crafted for the given input data [115, 254, 138] and the specific data-dependent task the sanitized data is intended for (e.g., answering linear queries) [54, 179, 73, 19]. As a result, such sanitization techniques greatly restrict the expressiveness of the released data distribution and fail to generalize to novel tasks unanticipated by the publisher. Instead, recent privacy-preserving techniques [239, 257, 245, 14] build on top of successes in generative adversarial network (GANs) [65] literature, to generate synthetic data faithful to the original input distribution. Specifically, GANs are trained using a privacy-preserving algorithm (e.g., using DP-SGD [1]) and demonstrate promising results in modeling a variety of real-world high-dimensional data distributions. Common to most privacy-preserving training algorithms for neural network models is *manipulating* the gradient information generated during backpropagation. Manipulation most commonly involves clipping the gradients (to bound sensitivity) and adding calibrated random noise (to introduce stochasticity). Although recent techniques that employ such an approach demonstrate reasonable success, they are mostly limited to shallow networks and fail to sufficiently capture the sample quality of the original data.

In this paper, towards the goal of a generative model capable of synthesizing high-quality samples in a privacy-preserving manner, we propose a differentially private GAN. We first identify that in such a data-publishing scenario, only a subset of the trained model (specifically the generator) and its parameters need to be publicly-released. This insight allows us to surgically manipulate the gradient information during training, and thereby allowing more meaningful gradient updates. By coupling the approach with a Wasserstein [8] objective with gradient-penalty term [67], we further improve the amount of gradient information flow during training. The Wasserstein objective additionally allows us to precisely estimate the gradient norms and analytically determine the sensitivity values. As an added benefit, we find our approach bypasses an intensive and fragile hyper-parameter search for DP-specific hyperparameters (particularly clipping values).

Contributions. In summary, we make the following contributions:

- We propose a novel gradient-sanitized Wasserstein GAN (GS-WGAN), which is capable of generating high-dimensional data with DP guarantee;
- Our approach naturally extends to both centralized and decentralized datasets. In the case of decentralized scenarios, our work can provide user-level DP guarantee [113] under an untrusted server;
- Extensive evaluations on various datasets demonstrate that our method significantly improves the sample quality of privacy-preserving generative models over state-of-the-art approaches.

2.2 Related Work

We review several differentially private GAN models, as well as their relations to our work.

DP-SGD GAN. Training GANs via DP-SGD [1, 239, 257, 14, 211, 57] has proven effective in generating high-dimensional sanitized data. However, DP-SGD relies on carefully tuning of the clipping bound of gradient norm, i.e., the sensitivity value. Specifically, the optimal clipping bound varies greatly with the model architecture and the training dynamics, making the implementation of DP-SGD difficult. Unlike previous works, we selectively apply sanitization to a necessary and sufficient subset of gradients for preserving privacy, which enables us to exploit the theoretical property of Wasserstein GANs [8, 67] for a precise estimation of the sensitivity value, avoiding the intensive search of hyper-parameters while reducing the clipping bias.

PATE. Private Aggregation of Teacher Ensembles (PATE) is recently adapted to generative models and two main approaches were studied: PATE-GAN [245] and G-PATE [129]. PATE-GAN trained multiple teacher discriminators on disjoint data partitions together with a student discriminator. In contrast, we consider a simplified model without a student discriminator.

G-PATE [129] is similar to our work in the sense that, both works trained the discriminator non-privately while only training the generator with DP guarantee, and both sanitized gradients that the generator received from the discriminator. However, G-PATE suffers from two main limitations: (*i*) gradients need to be discretized by using manually selected bins in order to suit for the PATE framework and (*ii*) high-dimensional gradients in the PATE framework bring high privacy costs and thus dimension reduction techniques are required. Our framework can effectively avoid these two limitations and achieve better sample quality due to the novel gradient sanitation, see our experiments.

Fed-Avg GAN [10]. While many works focus on centralized setting, the decentralized case has rarely been studied. To address this, Federated Average GAN (Fed-Avg GAN) proposed to adapt GAN training by using the DP-Fed-Avg [139] algorithm, providing user-level DP guarantee under trusted server. In comparison with Fed-Avg GAN that merely works on decentralized data, our work can tackle both centralized and decentralized data using a single framework. Note that Fed-Avg sanitized parameter gradients of the discriminator in a similar way to DP-SGD, it also suffers from the difficulty of turning hyper-parameters.

2.3 Background

DP provides rigorous privacy guarantees for algorithms while allowing for quantitative privacy analysis. We below present several definitions and theorems that will be used in this work.

Definition 2.3.1. (Differential Privacy (DP) [51]) A randomized mechanism $\mathcal{M}$ with range $\mathcal{R}$ is (ε, δ)-DP, if

$$Pr[\mathcal{M}(S) \in \mathcal{O}] \leq e^{\varepsilon} \cdot Pr[\mathcal{M}(S') \in \mathcal{O}] + \delta \tag{2.1}$$

holds for any subset of outputs $\mathcal{O} \subseteq \mathcal{R}$ and for any adjacent datasets S and S', where S and S' differ from each other with only one training example. $\mathcal{M}$ is the GAN training algorithm in our case, ε corresponds to the upper bound of privacy loss, and δ is the probability of breaching DP constraints. Intuitively, DP guarantees the difficulty of inferring the presence of an individual in the private dataset by observing $\mathcal{M}(S)$.

Definition 2.3.2. (Rényi Differential Privacy (RDP) [142]) A randomized mechanism $\mathcal{M}$ is (λ, ε)-RDP with order λ, if

$$D_\lambda(\mathcal{M}(S) \| \mathcal{M}(S')) = \frac{1}{\lambda - 1} \log \mathbb{E}_{x \sim \mathcal{M}(S)} \left[\left(\frac{Pr[\mathcal{M}(S) = x]}{Pr[\mathcal{M}(S') = x]} \right)^{\lambda - 1} \right] \leq \varepsilon \tag{2.2}$$

holds for any adjacent datasets S and S', where $D_\lambda(P\|Q) = \frac{1}{\lambda-1} \log \mathbb{E}_{x \sim Q}[(P(x)/Q(x))^\lambda]$ denotes the Rényi divergence. Moreover, a (λ, ε)-RDP mechanism $\mathcal{M}$ is also $(\varepsilon + \frac{\log 1/\delta}{\lambda - 1}, \delta)$-DP.

In contrast to DP, RDP provides convenient composition properties to accumulate privacy cost over a sequence of mechanisms (i.e., multiple gradient descent steps in our case).

Theorem 2.3.1. (Composition) For a sequence of mechanisms $\mathcal{M}_1, ..., \mathcal{M}_k$ s.t. $\mathcal{M}_i$ is (λ, ε_i)-RDP $\forall i$, the composition $\mathcal{M}_1 \circ ... \circ \mathcal{M}_k$ is $(\lambda, \sum_i \varepsilon_i)$-RDP.

Our approach is built on top of the Gaussian mechanism defined as follows.

Definition 2.3.3. (Gaussian Mechanism [53, 142]) Let $f : X \to \mathbb{R}^d$ be an arbitrary d-dimensional function with sensitivity being

$$\Delta_2 f = \max_{S,S'} \|f(S) - f(S')\|_2 \tag{2.3}$$

over all adjacent datasets S and S'. The Gaussian Mechanism $\mathcal{M}_\sigma$, parameterized by σ, adds noise into the output,i.e.,

$$\mathcal{M}_\sigma(x) = f(x) + \mathcal{N}(0, \sigma^2 I). \tag{2.4}$$

$\mathcal{M}$ is $(\lambda, \frac{\lambda \Delta_2 f^2}{2\sigma^2})$-RDP.

To provide DP guarantees of the released generator, we exploit the closedness of DP under post-processing, which is formalized as the following theorem.

Theorem 2.3.2. (Post-processing [53]) If $\mathcal{M}$ satisfies (ε, δ)-DP, $F \circ \mathcal{M}$ will satisfy (ε, δ)-DP for any function F with $\circ$ denoting the composition operator.

2.4 Method

Generative Adversarial Networks (GANs) [65]. Our approach models the underlying (private) data distribution using a generative neural network, building on top of recent successes of GANs. GANs (see Figure 2.1(a)) formulate the task of sample generation as a zero-sum two-player game, between two neural network models: discriminator D and generator G. The discriminator D is rewarded for correctly classifying whether a given sample is 'real' (i.e., from the input data distribution) or 'fake' (generated by the generator). In contrast, the task of the generator G is (given some random noise z) to generate samples which fool the discriminator (i.e., causes misclassifications). After training the models in an adversarial manner, the discriminator is discarded and the generator is used as a proxy to draw samples from the original distribution.

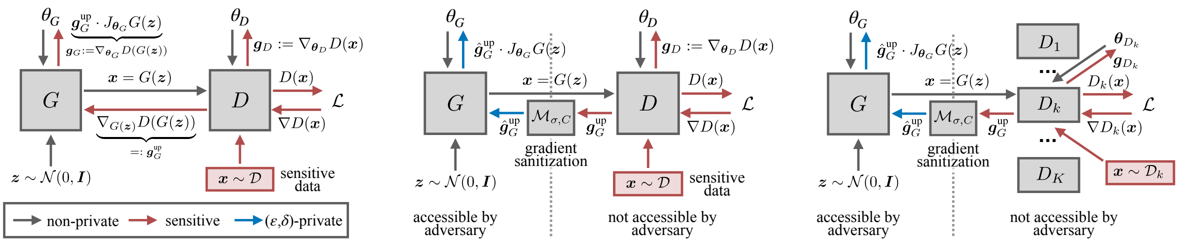
Differentially Private GANs. Releasing the generator as a substitute for the original training data distribution entails privacy risks [34]. Consequently, along the lines of recent work [14, 239, 211, 257], our goal is instead to train the GAN in a privacy-preserving manner, such that any privacy leakage upon disclosing the generator is bounded. A simple approach towards the goal is replacing the typical training procedure (SGD) with a differentially private variant (DP-SGD [1]) and thereby limiting the contribution of a particular training example in the final trained model. DP-SGD enforces the desired privacy requirement by *(i)* clipping the gradients g_t to have an L_2-norm no larger than C at each training step; and *(ii)* sampling random noise and adding it to the gradients, before performing descent on the trained parameters $\boldsymbol{\theta}$:

$$\boldsymbol{g}^{(t)} := \nabla_{\boldsymbol{\theta}} \mathcal{L}(\boldsymbol{\theta}_D, \boldsymbol{\theta}_G) \qquad \text{(gradient)} \tag{2.5}$$

$$\hat{\boldsymbol{g}}^{(t)} := \mathcal{M}_{\sigma,C}(\boldsymbol{g}^{(t)}) = \mathrm{clip}(\boldsymbol{g}^{(t)}, C) + \mathcal{N}(0, \sigma^2 C^2 \boldsymbol{I}) \qquad \text{(sanitization mechanism)} \tag{2.6}$$

$$\boldsymbol{\theta}^{(t+1)} := \boldsymbol{\theta}^{(t)} - \eta \cdot \hat{\boldsymbol{g}}^{(t)} \qquad \text{(gradient descent step)} \tag{2.7}$$

While such an approach provides rigorous privacy guarantees, there are multiple shortcomings: *(i)* the sanitization mechanism $\mathcal{M}_{\sigma,C}$, primarily due to clipping, significantly destroys the



(a) Vanilla GAN
(Without privacy barrier)

(b) GS-WGAN
(Ours, with privacy barrier)

(c) Fed-GS-WGAN
(Ours, in a Federated setup)

Figure 2.1: Approach outline. Our gradient sanitization scheme ensures DP training of the generator.

original gradient information, and thereby affects utility; and *(ii)* finding a reasonable clipping value C in the mechanism to balance utility with privacy is especially challenging. In particular, as the gradient norms exhibit a heavy-tailed distribution, choosing a clipping value requires an exhaustive search. Moreover, since the clipping value is extremely sensitive to many other hyperparameters (e.g., learning rate, architecture), it requires persistent re-tuning. Now, we discuss how we address these shortcomings within our gradient-sanitized approach.

Selectively applying Sanitization Mechanism. We begin by exploiting the fact that after training the GAN, only the generator G is released. Consequently, we can perform gradient steps by selectively applying the sanitization mechanism only to the corresponding subset of parameters $\boldsymbol{\theta}_G$:

$$\boldsymbol{\theta}_D^{(t+1)} := \boldsymbol{\theta}_D^{(t)} - \eta_D \cdot \boldsymbol{g}_D^{(t)} \qquad (\hat{\boldsymbol{g}}_D^{(t)} = \boldsymbol{g}_D^{(t)}; \text{Discriminator}) \tag{2.8}$$

$$\boldsymbol{\theta}_G^{(t+1)} := \boldsymbol{\theta}_G^{(t)} - \eta_G \cdot \hat{\boldsymbol{g}}_G^{(t)} \qquad (\hat{\boldsymbol{g}}_G^{(t)} = \mathcal{M}_{\sigma,C}(\boldsymbol{g}_G^{(t)}); \text{Generator}) \tag{2.9}$$

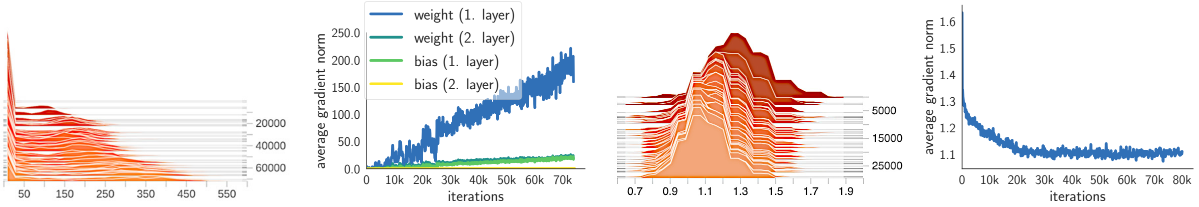
Apart from reducing the number of parameters sanitized, this also provides a benefit of more reliably training a discriminator. In addition, we exploit the chain rule to further narrow the scope of the sanitization mechanism:

$$\boldsymbol{g}_G = \nabla_{\boldsymbol{\theta}_G} \mathcal{L}_G(\boldsymbol{\theta}_G) = \nabla_{G(\boldsymbol{z};\boldsymbol{\theta}_G)} \mathcal{L}_G(\boldsymbol{\theta}_G) \cdot J_{\boldsymbol{\theta}_G} G(\boldsymbol{z}; \boldsymbol{\theta}_G) \tag{2.10}$$

$$\hat{\boldsymbol{g}}_G = \mathcal{M}_{\sigma,C}\underbrace{\left(\nabla_{G(\boldsymbol{z})} \mathcal{L}_G(\boldsymbol{\theta}_G)\right)}_{\boldsymbol{g}_G^{\text{upstream}}} \cdot \underbrace{J_{\boldsymbol{\theta}_G} G(\boldsymbol{z}; \boldsymbol{\theta}_G)}_{J_G^{\text{local}}} \tag{2.11}$$

The above becomes easier to intuit by considering a typical loss function $\mathcal{L}_G(\boldsymbol{\theta}_G) = -D(G(\boldsymbol{z};\boldsymbol{\theta}_G))$. As illustrated in Figure 2.1(b), Equation 2.11 can then be considered as placing the privacy barrier for gradient information backpropagating from the discriminator back to the generator, by applying the sanitization mechanism on $\boldsymbol{g}_G^{\text{upstream}}$. Note that the second term (J_G^{local}) is the local generator jacobian computed independent of training data, and hence does not require sanitization. Consequently, using a more precise application of the sanitization mechanism on the gradient information, our goal here is to maximally preserve the true gradient direction during training.

Bounding sensitivity using Wasserstein distance. To bound the sensitivity of the optimizer on individual training examples, a key step in sanitization mechanisms is to *clip* (Equation 2.6) the gradient vector $\boldsymbol{g}$ (Equation 2.5) before updating parameters (Equation 2.7). Clipping is typically performed in L_2 norm, by replacing the gradient vector $\boldsymbol{g}$ by $\boldsymbol{g}/\max(1, ||\boldsymbol{g}||_2/C)$ to



(a) DP-SGD (b) DP-SGD (c) Ours (d) Ours

Figure 2.2: Gradient norm (before clipping) dynamics during the GAN training process. In the experiment, the clipping bound is chosen to be 1 and 1.1 in 2.2(c) and 2.2(a) respectively.

ensure $||\boldsymbol{g}||_2 \leq C$. However, clipping significantly destroys gradient information, as reasonable choices of C (e.g., 4 [1]) are significantly lower than the gradient-norms observed (12 ± 10 in our case) when training neural networks using standard loss functions. We propose to alleviate the issue by leveraging a more suitable loss function, which generates bounded gradients (with norms close to 1) by construction. Specifically, we use as our loss the Wasserstein-1 metric [8], which measures the statistical distance between the real and generated data distributions. Here, the training process can be interpreted as minimizing integral probability metrics (IPMs) $\sup_{f \in \mathcal{F}} | \int_M f dP - \int_M f dQ|$ between real (P) and generated (Q) data distributions, where $\mathcal{F} = \{f : \|f\|_L \leq 1\}$ (i.e., the discriminator function f is 1-Lipschitz continuous). Theoretically, the optimal discriminator has a gradient norm being 1 almost everywhere under P and Q [67] (i.e., $\|\boldsymbol{g}_G^{\text{upstream}}\|_2 \approx 1$).

We incorporate the norm constraint into our training objective in the form of a gradient penalty term [67]:

$$\mathcal{L}_D = -\mathbb{E}_{\boldsymbol{x}\sim P}[D(\boldsymbol{x})] + \mathbb{E}_{\tilde{\boldsymbol{x}}\sim Q}[D(\tilde{\boldsymbol{x}})] + \lambda \mathbb{E}[(\|\nabla D(\alpha \boldsymbol{x} + (1-\alpha)\tilde{\boldsymbol{x}})\|_2 - 1)^2] \tag{2.12}$$

$$\mathcal{L}_G = -\mathbb{E}_{\boldsymbol{z}\sim P_z}[D(G(\boldsymbol{z}))] \tag{2.13}$$

where $\mathcal{L}_D$ and $\mathcal{L}_G$ represent training objectives for the discriminator and the generator, respectively. λ is the hyper-parameter for weighting the gradient penalty term and P_z denotes the prior distribution for the latent code variable $\boldsymbol{z}$. The variable $\alpha \sim \mathcal{U}[0,1]$, uniformly sampled from $[0,1]$, regulates the interpolation between real and generated samples.

As a natural consequence of the Wasserstein objective, bounding the norms of our target gradient $\boldsymbol{g}_G^{\text{upstream}}$ (Equation 2.11) during training is integrated in our training objective (last term in Equation 2.12). Consequently, we observe significantly lower variance in gradient norms during training (see Figure 2.2(c)-2.2(d)) compared to training using a standard GAN loss (see Figure 2.2(a)-2.2(b)). As a result, bounding the sensitivity (gradient norms) is now largely delegated to our training procedure and clipping using the sanitization mechanism destroys significantly less information. Additionally, we obtain the optimal clipping threshold of $C = 1$, as $\|\boldsymbol{g}_G^{\text{upstream}}\|_2 \approx 1$ based on the theoretical property of Wasserstein GANs. This allows us to derive a fixed and bounded sensitivity, eliminating the need for intensive hyper-parameter search for a proper clipping threshold. Following this clipping strategy, a data-independent privacy cost can be determined by the following theorem, whose proof is provided in Appendix A.

Theorem 2.4.1. Each generator update step satisfies $(\lambda, 2B\lambda/\sigma^2)$-RDP where B is the batch size.

Privacy Amplification by Subsampling. A well-known approach for increasing privacy of a mechanism is to apply the mechanism to a random subsample of the database, rather than on the entire dataset [116, 12, 230]. Intuitively, subsampling decreases the chances of leaking information about a particular individual since nothing about that individual can be leaked once the individual is not included in the subsample. In order to further reduce the privacy cost, we subsample the whole dataset into different subsets and train multiple discriminators independently on each subset. At each training step, the generator randomly queries one discriminator while the selected discriminator updates its parameters on the generated data and its associated subsampled dataset.

Extending to Federated Learning. In addition to improving the privacy guarantee, performing subsampling in our setup also naturally accommodates training a generative model on decentralized datasets (with a discriminator trained on each disjoint data subset). Recently, Augenstein et al. [10] identified such techniques are extremely relevant when training models in a federated setup [140], i.e., when the training data is private and distributed among edge devices. We outline our method to train a differentially private GAN in a federated setup in Figure 2.1(c) and remark some subtle differences between our approach and Fed-Avg GAN [10] here: *(i)* the discriminators are retained at each client in our framework while they are shared between the server and client in Fed-Avg GAN; *(ii)* the gradients are sanitized at each client before sending to the server, with which we provide DP guarantee even under an untrusted server. In contrast, the unprocessed information is accumulated at the server before being sanitized in Fed-Avg GAN; and *(iii)* The gradients w.r.t. the samples are transferred in GS-WGAN, while Fed-Avg GAN transfers the gradients w.r.t. discriminator network parameters.

2.5 Experiment

2.5.1 Experiment Setup

To validate the applicability of our method to high-dimensional data, we conduct experiments on image datasets. In line with previous works, we use MNIST [108] and Fashion-MNIST [238] dataset. We model the joint distribution of images and the corresponding labels, i.e., the label is supplied to both the generator and the discriminator, and the image is generated conditioned on the input. During both training and inference, we use a uniform prior distribution for generating labels, which is independent of the training dataset and thus does not incur additional privacy cost (in contrast, [211] needs to assume the labels are non-private).

Evaluation Metrics. We evaluate along two fronts: *privacy* (determined by ε) and *utility*. For utility, we consider two metrics: (a) *sample quality*: realism of the samples produced – evaluated by **Inception Score (IS)** [184, 112] and **Frechet Inception Distance (FID)** [77] (standard in GAN literature); and (b) *usefulness for downstream tasks*: we train downstream classifiers on 60k privately-generated data points and evaluate the prediction accuracy on real test set. We consider Multi-layer Perceptrons (MLP), Convolutional Neural Networks (CNN) and 11 scikit-learn [162] classifiers (e.g., SVMs, Random Forest). We include the following metrics in the main paper: **MLP Acc** (MLP accuracy), **CNN Acc** (CNN accuracy), **Avg Acc** (Averaged accuracy of all classification models), **Calibrated Acc** (Averaged accuracy of all classification models normalized by the accuracy when trained on real data). The detailed results are presented in Appendix A.

[2] PATE provides data-dependent ε, i.e., publishing ε value will introduce privacy cost. Thus, G-PATE is not directly comparable to other methods and is excluded from our analysis study (Section 2.5.3).

		IS↑	FID ↓	MLP ↑ Acc	CNN ↑ Acc	Avg ↑ Acc	Calibrated ↑ Acc
MNIST	Real	9.80	1.02	0.98	0.99	0.88	100 %
	G-PATE [2]	3.85	177.16	0.25	0.51	0.34	40%
	DP-SGD GAN	4.76	179.16	0.60	0.63	0.52	59%
	DP-Merf	2.91	247.53	0.63	0.63	0.57	66%
	DP-Merf AE	3.06	161.11	0.54	0.68	0.42	47%
	Ours	**9.23**	**61.34**	**0.79**	**0.80**	**0.60**	**69%**
Fashion-MNIST	Real	8.98	1.49	0.88	0.91	0.79	100%
	G-PATE	3.35	205.78	0.30	0.50	0.40	54%
	DP-SGD GAN	3.55	243.80	0.50	0.46	0.43	53%
	DP-Merf	2.32	267.78	0.56	0.62	0.51	65%
	DP-Merf AE	3.68	213.59	0.56	0.62	0.45	55%
	Ours	**5.32**	**131.34**	**0.65**	**0.65**	**0.53**	**67%**

Table 2.1: Quantitative results on MNIST and Fashion-MNIST ($\varepsilon = 10, \delta = 10^{-5}$)

Architecture and Warm-start. We highlight two strategies adopted for improving the sample quality as well as reducing the privacy cost: *(i) Better model architecture*: While previous works are limited to shallow networks and thereby bottle-necking generated sample quality, our framework allows stable training with a complex model architecture (DCGAN [168] architecture for the discriminator, ResNet architecture (adapted from BigGAN [21]) for the generator) to help improve the sample quality; and *(ii) Discriminator warm-starting*: To bootstrap the training process, we pre-train discriminators along with a non-private generator for a few steps, and we subsequently train the private generator using the warm-starting values of the discriminators. Note that our framework allows pre-training on the original private dataset without compromising privacy (in contrast, [257] needs to use external public datasets).

2.5.2 Comparison with Baselines

Baselines. We consider the following state-of-the-art methods designed for DP high-dimensional data generation: **DP-Merf** and **DP-Merf AE** [69], **DP-SGD GAN** [211, 239, 257], and **G-PATE** [129]. While PATE-GAN [245] demonstrates promising results on low-dimensional data, we currently do not consider it as we were unable to extend it to our image datasets (more details in Appendix A) for a fair comparison. For DP-Merf, DP-Merf AE, and G-PATE, we use the source code provided by the authors. For DP-SGD GAN, we adopt the implementation of [211], which is the only work that provides executable code with privacy analysis. For a fair comparison, we evaluate all methods with a privacy budget of (ε, δ)=$(10, 10^{-5})$ (consistently used in previous works) over 60K generated samples.

Results. We present the qualitative results in Figure 2.3 and the quantitative results in Table 2.1. In terms of sample quality, we find (Table 2.1, columns IS and FID) our method consistently provides significant improvements over baselines. For instance, considering inception scores, we find a relative improvement of 94% (9.23 vs. 4.76 of DP-SGD GAN) on MNIST and 45% on Fashion-MNIST (5.32 vs. 3.68 of DP-Merf AE).

Furthermore, our method also generates samples that better capture the statistical properties of the original data and are thereby making aiding performances of downstream tasks. For instance, our approach increases performance of a downstream MLP classifier (Table 2.1,

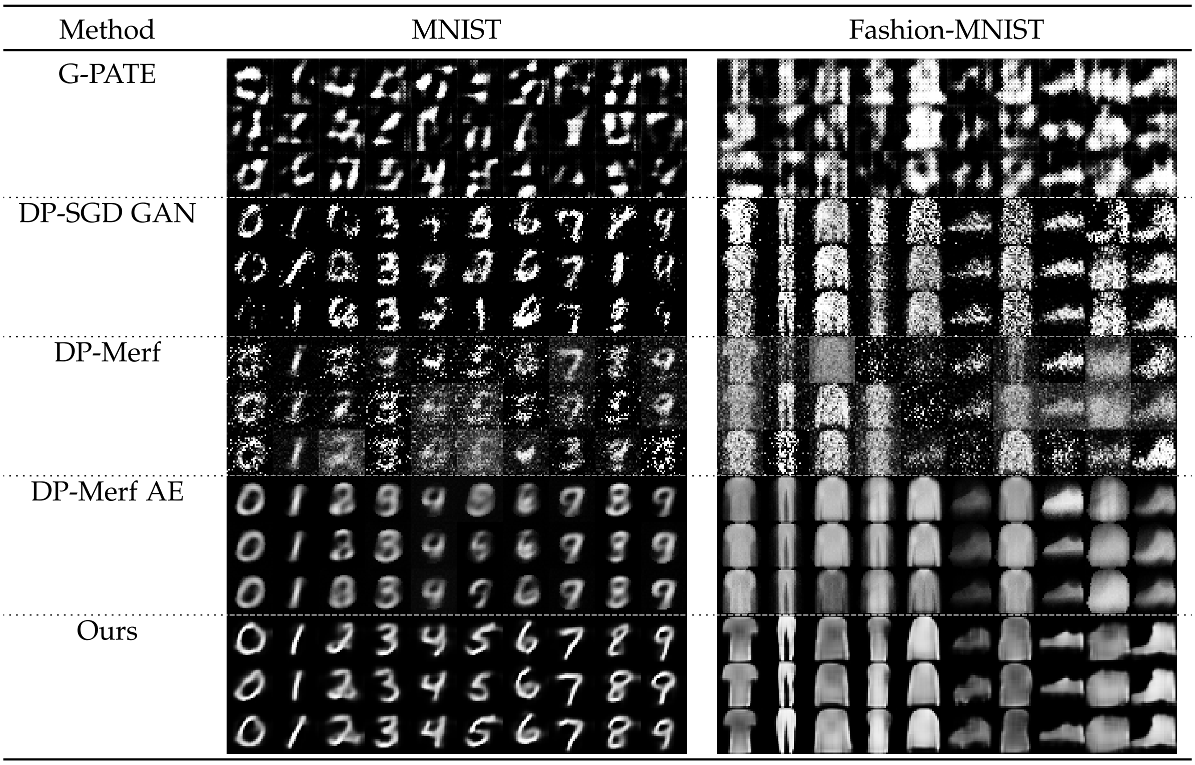
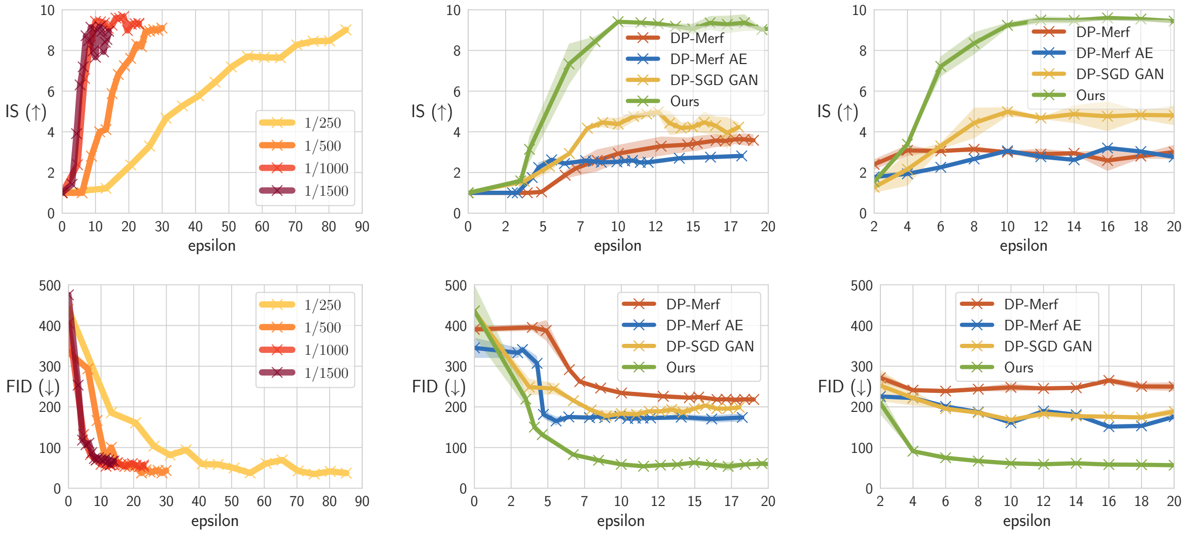


Figure 2.3: Generated samples with $(\varepsilon, \delta) = (10, 10^{-5})$



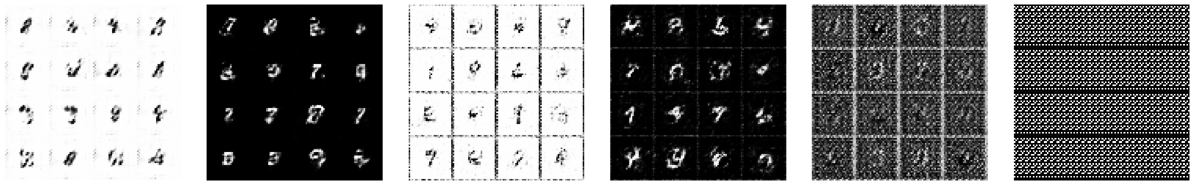
(a) Effects of subsampling rates

(b) Effects of Iterations

(c) Effects of Noise scale

Figure 2.4: Privacy-utility trade-off on MNIST with $\delta = 10^{-5}$. (Top row: IS. Bottom row: FID.)

column MLP Acc) by 25% (0.79 vs. 0.63 of DP-Merf) on MNIST and 16% (0.65 vs. 0.56 of DP-Merf) on Fashion-MNIST. In a word, our approach demonstrates significant improvements across multiple metrics and high-dimensional image datasets.



(a) Ours (with bug) (b) Ours (without bug) (c) Fed-Avg GAN (with bug) (d) Fed-Avg GAN (without bug) noise=0.01 (e) Fed-Avg GAN (without bug) noise =0.1 (f) Fed-Avg GAN (without bug) noise =0.5

Figure 2.5: Qualitative results on federated EMNIST.

2.5.3 Influence of Hyperparameters

The privacy/utility performances of our approach is primarily determined by three factors:*(i)* **subsampling rates** γ, *(ii)* number of training **iterations**, and *(iii)* **noise scale** σ. We now investigate how these factors influence privacy cost ε and utility (sample quality measured by IS and FID), and additionally compare with baselines:

Subsampling rates. We evaluate the sample quality of our method considering multiple choices of subsampling rates ($\gamma \in [1/250, 1/500, 1/1000, 1/1500]$) over the training iterations. The results are presented in Figure 2.4(a), where the x-axis corresponds to the ε value evaluated at different iterations. We observe that the sub-sampling rate should be sufficiently small for achieving a reasonable sample quality while providing a strong privacy guarantee. A value of $1/1000$ yields relatively good privacy-utility trade-off, while further decreasing the sub-sampling rate does not necessarily improve the results.

Iterations. We evaluate all methods during the course of training, where more iterations lead to higher utilities, but at the expense of accumulating a higher privacy cost ε. From Figure 2.4(b), we find our approach yields better sample qualities with fewer iterations (and hence lower ε). Specifically, across the range of iterations, we find IS increases by 10-90%, while the FID decreases by 20-60% compared to baselines.

Noise scale. We calibrate the noise scale of each method to certain privacy budget ε and show the resulting privacy-utility curves in Figure 2.4(c). Similar to the previous case, our method achieves a consistent improvement in both metrics spanning a broad range of noise scale (privacy budget ε).

2.5.4 Federated Setting Evaluation

Our approach allows to perform privacy-preserving training of a GAN in federated setup, where sensitive user dataset is partitioned across K clients (e.g., edge devices). Such a training scheme is useful to privately inspect data for debugging. For evaluation, we consider a real-world debugging task introduced in [10]: to detect the erroneous flipping of pixel intensities, which occurs in a fraction of client devices. Two GAN models are trained: one on client data that are suspected to be erroneous flipped (with bug) and one on the client data that are believed to be normal (without bug). The samples generated by these two GAN models should exhibit different appearance such that the bug can be detected by inspecting the generated samples. To mimic the real-world situation where the server is blind to the

	IS ↑	FID ↓	epsilon ↓	CT (byte) ↓
Fed Avg GAN	10.88	218.24	9.99×10^6	$\sim 3.94 \times 10^7$
Ours	**11.25**	**60.76**	$\mathbf{5.99 \times 10^2}$	$\sim \mathbf{1.50 \times 10^5}$

Table 2.2: Quantitative results on federated EMNIST ($\delta = 1.15 \times 10^{-3}$)

erroneous pre-processing, only a fraction of the suspected users is indeed affected by the bug. This has the realistic property that the client data is non-IID and poses additional difficulties in the GAN training. A detailed description about the data can be found in Appendix A.

We conduct experiments on the Federated EMNIST dataset [23] and compare our GS-WGAN with **Fed-Avg GAN** [10].As shown in Figure 2.5(a) and 2.5(b), the presence of bug is clearly identifiable by inspecting the samples generated by our model. Moreover, as shown in Table 2.2, our GS-WGAN yields better sample quality (0.28× smaller FID) with a significantly lower privacy cost ($10^4\times$ smaller ε) compared to Fed-Avg GAN. Furthermore, our method shows better robustness against large injected noise. This is illustrated in Figure 2.5(e) and 2.5(f): a noise scale larger than 0.1 inevitably leads to failure in training Fed-Avg GAN, whereas our method can tolerate 10 times larger noise scale. In addition, we show in the last column of Table 2.2 the amortized communication cost (CT) required for performing one update step on the generator. Specifically, this corresponds to the total number of transferred bytes (including both server-to-client and client-to-server) averaged over all participating clients. Our GS-WGAN allows each client to retain its discriminator locally and only the gradients w.r.t. generated samples are communicated (which is significantly more compact than gradients w.r.t model parameters, as done by Fed-Avg GAN). We observe that GS-WGAN achieves a magnitude of 10^2 gain in reducing the communication cost.

2.6 Conclusion

In this paper, we presente a differentially-private approach *GS-WGAN* to sanitize sensitive high-dimensional datasets with provable privacy guarantees while simultaneously preserving informativeness of the sanitized samples. Our primary insight is that privacy-preserving training (which sacrifices utility) can be selectively applied only to the generator (which is publicly released) while the discriminator (which is discarded post-training) can be trained optimally. Additionally, introducing a Wasserstein training objective allows us to exploit the Lipschitz property of the discriminator and leads to precise estimates of the sensitivity value without exhaustive hyper-parameters search. Our extensive evaluation presents encouraging results: sensitive datasets can be effectively distilled to sanitized forms which nonetheless preserves informativeness of the data and allows training downstream models.

3 Private Set Generation with Discriminative Information

Contents

Differentially private data generation techniques have become a promising solution to the data privacy challenge — it enables sharing of data while complying with rigorous privacy guarantees, which is essential for scientific progress in sensitive domains. Unfortunately, restricted by the inherent complexity of modeling high-dimensional distributions, existing private generative models are struggling with the utility of synthetic samples. In contrast to existing works that aim at fitting the complete data distribution, we directly optimize for a small set of samples that are representative of the distribution under the supervision of discriminative information from downstream tasks, which is generally an easier task and more suitable for private training. Our work provides an alternative view for differentially private generation of high-dimensional data and introduces a simple yet effective method that greatly improves the sample utility of state-of-the-art approaches.

This chapter is based on [29]: As the first author of [29], Dingfan Chen proposed the project idea, implemented the algorithms, conducted all experiments, and was the main writer of the paper. This paper was published in NeurIPS 2022 and has garnered recognition, being cited in several surveys and benchmarks, e.g., [249, 109, 181, 233]. The source code for this work is available on Github [1].

3.1 Introduction

Data sharing is vital for the growth of machine learning applications in numerous domains. However, in many application scenarios, data sharing is prohibited due to the private nature of data (e.g., individual data stored on mobile devices, medical treatments, and banking records) and the corresponding stringent regulations, which greatly hinders technological progress. Differentially private (DP) data publishing [51, 52, 58] provides a compelling solution to such challenge, where only a sanitized form of the data is publicly released. Such sanitized synthetic data can be leveraged as if it were the real data, analyzed with established toolchains, and can be shared openly to the public, facilitating technological advance and reproducible research in sensitive domains.

[1] https://github.com/DingfanChen/Private-Set

Yet, generation of high-dimensional data with DP guarantees is highly challenging and traditional DP algorithms designed for capturing low-dimensional statistical characteristics are not applicable to this task [179, 73, 19, 223]). Instead, inspired by the great successes of deep generative models in learning high-dimensional representations, recent works [24, 32, 239, 245, 14, 70] adopt deep generative neural networks as the underlying generation backbone and incorporate the privacy constraints into the training procedure, such that any privacy leakage upon disclosing the data generator is bounded.

However, these methods have common shortcomings: *(i)* deep generative models are known to be data-demanding [93], which becomes even harder to train when considering the privacy constraints [32, 24]; *(ii)* they do not guarantee any optimal solution for the downstream task (e.g. classification). In fact, existing models are still struggling to generate sanitized data that is useful for downstream data analysis tasks. For example, when training a convolutional neural network (ConvNet) classifier on the private generated data, the highest test accuracy reported in literature is $< 85\%$ for MNIST dataset with $(\varepsilon, \delta) = (10, 10^{-5})$ [24], which lags far behind the discriminative baseline ($> 98\%$ with $(\varepsilon, \delta) = (1.2, 10^{-5})$ [212]) and makes private generative models less appealing for many practical scenarios with data analysis as the end goal.

In this work, we learn to synthesize informative samples that are privacy-preserving and are optimized to train neural networks for downstream tasks. In contrast to existing approaches, we directly optimize a small set of samples instead of the deep generative models that is notoriously difficult to train in a private manner. Moreover, we exploit discriminative information from downstream tasks to guide the samples towards containing more useful information for downstream analysis. Compared to existing works, we improve the task utility by a large extent (up to 10% downstream test accuracy improvement over state-of-the-art approach), while still preserving the flexibility and generality across varying configurations for downstream analysis. As an added benefit, our formulation naturally distilled the knowledge of original data into a much smaller set, which largely saves the memory and computational consumption for downstream analysis.

Contributions. We summarize our main contributions as follows:

- We present a new perspective of private high-dimensional data generation, with which we aim to bridge the utility and generality gap between the private generative and discriminative models. We believe this alternative view opens up new possibilities in different research directions ranging from private analysis to generation.
- We introduce a simple yet effective method for generating informative samples that are optimized for training downstream neural networks, while maintaining generality as well as reducing the memory and computation consumption as added benefits.
- Experimental results demonstrate that, in comparison to existing works, our work improves the sample utility by a large margin and offers superior practicability for real-world application scenarios.

3.2 Background

We consider the standard central model of DP in this paper. We below present several definitions and theorems that will be used in this work.

Definition 3.2.1. (Differential Privacy (DP) [51]) A randomized mechanism $\mathcal{M}$ with range $\mathcal{R}$ is (ε, δ)-DP, if

$$Pr[\mathcal{M}(\mathcal{D}) \in \mathcal{O}] \leq e^{\varepsilon} \cdot Pr[\mathcal{M}(\mathcal{D}') \in \mathcal{O}] + \delta \tag{3.1}$$

holds for any subset of outputs $\mathcal{O} \subseteq \mathcal{R}$ and for any adjacent datasets $\mathcal{D}$ and $\mathcal{D}'$, where $\mathcal{D}$ and $\mathcal{D}'$ differ from each other with only one training example, i.e., $\mathcal{D}' = \mathcal{D} \cup \{x\}$ for some x (or vice versa). $\mathcal{M}$ corresponds to the set generation algorithm in our case, ε is the upper bound of privacy loss, and δ is the probability of breaching DP constraints. DP guarantees the difficulty of inferring the presence of an individual in the private dataset by observing the generated set of samples $\mathcal{M}(\mathcal{D})$.

Our approach is built on top of the Gaussian mechanism defined as follows.

Definition 3.2.2. (Gaussian Mechanism [53]) Let $f : X \rightarrow \mathbb{R}^d$ be an arbitrary d-dimensional function with sensitivity being

$$\Delta_2 f = \max_{\mathcal{D},\mathcal{D}'} \|f(\mathcal{D}) - f(\mathcal{D}')\|_2 \tag{3.2}$$

over all adjacent datasets $\mathcal{D}$ and $\mathcal{D}'$. The Gaussian Mechanism $\mathcal{M}_\sigma$, parameterized by σ, adds noise into the output, i.e.,

$$\mathcal{M}_\sigma(x) = f(x) + \mathcal{N}(0, \sigma^2 I). \tag{3.3}$$

For $\varepsilon, \delta \in (0,1)$, $\mathcal{M}_\sigma$ is (ε,δ)-DP if $\sigma \geq \sqrt{2\ln(1.25/\delta)}\Delta_2 f/\varepsilon$.

Any privacy cost is bounded upon releasing the private set of generated data due to the closedness of DP under post-processing.

Theorem 3.2.1. (Post-processing [53]) If $\mathcal{M}$ satisfies (ε,δ)-DP, $F \circ \mathcal{M}$ will satisfy (ε,δ)-DP for any data-independent function F with $\circ$ denoting the composition operator.

3.3 Method

We consider a standard classification task where we are given a private dataset $\mathcal{D} = \{(\boldsymbol{x}_i, y_i)\}_{i=1}^N$ with $\boldsymbol{x}_i \in \mathbb{R}^d$ the feature, $y_i \in \{1, ..., L\}$ the class label, N the number of samples, L the number of label classes. Our objective is to synthesize a set of samples $\mathcal{S} = \{(\boldsymbol{x}_i^{\mathcal{S}}, y_i^{\mathcal{S}})\}_{i=1}^M$ such that *(i)* samples in $\mathcal{S}$ have the same form as data in $\mathcal{D}$; *(ii)* a neural network trained on $\mathcal{S}$ should maximally match generalization performance of a deep neural network that is trained on $\mathcal{D}$; *(iii)* the privacy leakage of $\mathcal{D}$ when releasing $\mathcal{S}$ is upper bounded by a pre-defined privacy level (ε, δ).

Let $F(\cdot\,; \boldsymbol{\theta}^{\mathcal{D}})$ and $F(\cdot\,; \boldsymbol{\theta}^{\mathcal{S}})$ be the deep neural networks parameterized by $\boldsymbol{\theta}^{\mathcal{D}}$ and $\boldsymbol{\theta}^{\mathcal{S}}$ that are trained on $\mathcal{D}$ and $\mathcal{S}$ respectively. The objective can be formulated as:

$$\mathbb{E}_{(\boldsymbol{x},y)\sim P_{\mathcal{D}}}[\ell(F(\boldsymbol{x};\boldsymbol{\theta}^{\mathcal{D}}), y)] \simeq \mathbb{E}_{(\boldsymbol{x},y)\sim P_{\mathcal{D}}}[\ell(F(\boldsymbol{x};\boldsymbol{\theta}^{\mathcal{S}}), y)] \tag{3.4}$$

where ℓ denotes the loss function (e.g., cross-entropy for the classification task) and the expectation is taken over the real data distribution $P_{\mathcal{D}}$.

Equation 3.4 can be naturally achieved once $\boldsymbol{\theta}^{\mathcal{S}} \approx \boldsymbol{\theta}^{\mathcal{D}}$. In particular, when given the same initialization $\boldsymbol{\theta}_0^{\mathcal{D}} = \boldsymbol{\theta}_0^{\mathcal{S}}$, solving for $\boldsymbol{\theta}_t^{\mathcal{S}} \approx \boldsymbol{\theta}_t^{\mathcal{D}}$ at each training iteration t leads to $\boldsymbol{\theta}^{\mathcal{S}} \approx \boldsymbol{\theta}^{\mathcal{D}}$ as desired. This can be achieved by optimizing the synthetic set $\mathcal{S}$ such that it yields a similar gradient as if the network is trained on the real dataset at each iteration t:

$$\min_{\mathcal{S}} \mathcal{L}_{\text{dis}}(\nabla_{\boldsymbol{\theta}}\mathcal{L}(\mathcal{S}, \boldsymbol{\theta}_t), \nabla_{\boldsymbol{\theta}}\mathcal{L}(\mathcal{D}, \boldsymbol{\theta}_t)) \tag{3.5}$$

where $\nabla_{\boldsymbol{\theta}}\mathcal{L}(\mathcal{S}, \boldsymbol{\theta}_t))$ corresponds to the gradient of the classification loss on the synthetic set $\mathcal{S}$, $\nabla_{\boldsymbol{\theta}}\mathcal{L}(\mathcal{D}, \boldsymbol{\theta}_t)$ denotes the stochastic gradient on the real data, and $\mathcal{L}_{\text{dis}}$ is a sum of cosine distances between the gradients at each layer [261, 259] (See Appendix B for more details).

Algorithm 1: Private Set Generation (PSG)

```
Input: Dataset $\mathcal{D} = \{(\boldsymbol{x}_i, y_i)\}_{i=1}^N$, learning rate for update network parameters $\tau_{\boldsymbol{\theta}}$ and
        $\tau_{\mathcal{S}}$, sampling probability $\rho$, gradient clipping bound $C$, number of runs $R$, outer
        iterations $T$, inner iterations $J$, batches $K$, desired privacy cost $\varepsilon$ given a
        pre-defined $\delta$
Output: Synthetic set $\mathcal{S}$
Compute the required DP noise scale $\sigma$ numerically [1, 143] so that the privacy cost
  equals $\varepsilon$ after the training; Initialize synthetic set $\mathcal{S}$ (features $\boldsymbol{x}^{\mathcal{S}}$ are from Gaussian
  noise; labels are balanced set depending on the pre-defined number of samples per
  class) ;
for run in $\{1, ..., R\}$ do
    Initialize model parameter $\boldsymbol{\theta}_0 \sim P_{\boldsymbol{\theta}_0}$;
    for outer_iter in $\{1, ..., T\}$ do
        $\boldsymbol{\theta}_{t+1} = \boldsymbol{\theta}_t$
        for batch_index in $\{1, ..., K\}$ do
            Sample a batch $\{(x_i, y_i)\}_{i=1}^{B_k}$, where each $(x_i, y_i)$ from $\mathcal{D}$ is uniformly
              sampled with probability $\rho$;
            for each $(\boldsymbol{x}_i, y_i)$ in the batch do
                // Compute per-example gradients on real data
                     $g_{\boldsymbol{\theta}_t}^{\mathcal{D}}(\boldsymbol{x}_i) = \ell(F(\boldsymbol{x}_i; \boldsymbol{\theta}_t), y_i)$
                // Clip gradients
                     $\widetilde{g_{\boldsymbol{\theta}_t}^{\mathcal{D}}}(\boldsymbol{x}_i) = g_{\boldsymbol{\theta}_t}^{\mathcal{D}}(\boldsymbol{x}_i) \cdot \min(1, C / \|g_{\boldsymbol{\theta}_t}^{\mathcal{D}}(\boldsymbol{x}_i)\|_2)$
            end
            // Add noise to average gradient with Gaussian mechanism
                 $\widetilde{g_{\boldsymbol{\theta}_t}^{\mathcal{D}}} = \frac{1}{B_k} \sum_{i=1}^{B_k} (\widetilde{g_{\boldsymbol{\theta}_t}^{\mathcal{D}}}(\boldsymbol{x}_i) + \mathcal{N}(0, \sigma^2 C^2 I))$
            // Compute parameter gradients on synthetic data and update $\mathcal{S}$
                 $g_{\boldsymbol{\theta}_t}^{\mathcal{S}} = \nabla_{\boldsymbol{\theta}} \mathcal{L}(\mathcal{S}, \boldsymbol{\theta}_t)) = \frac{1}{M} \sum_{i=1}^{M} \ell(F(\boldsymbol{x}_i^{\mathcal{S}}; \boldsymbol{\theta}_t), y_i^{\mathcal{S}})$
                 $\mathcal{S} = \mathcal{S} - \tau_{\mathcal{S}} \cdot \nabla_{\mathcal{S}} \mathcal{L}_{\text{dis}}(g_{\boldsymbol{\theta}_t}^{\mathcal{S}}, \widetilde{g_{\boldsymbol{\theta}_t}^{\mathcal{D}}})$
        end
        for inner_iter in $\{1, ..., J\}$ do
            // Update network parameter using $\mathcal{S}$
                 $\boldsymbol{\theta}_t = \boldsymbol{\theta}_t - \tau_{\boldsymbol{\theta}} \cdot \nabla_{\boldsymbol{\theta}} \mathcal{L}(\mathcal{S}, \boldsymbol{\theta}_t)$
        end
    end
end
return Synthetic set $\mathcal{S}$
```

To mimic the training procedure, $\mathcal{S}$ and the network $F(\cdot; \boldsymbol{\theta})$ are updated jointly in an iterative manner, where in each outer iteration the $\mathcal{S}$ is trained to minimize the gradient matching loss $\mathcal{L}_{\text{dis}}$ and in each inner iterations the network parameters $\boldsymbol{\theta}_t$ are optimized towards minimizing the classification loss on the synthetic set $\mathcal{S}$. Moreover, $\mathcal{S}$ is optimized over multiple initializations of network parameters $\boldsymbol{\theta}_0$ to ensure the generalization ability of $\mathcal{S}$ over different random initialization when training a downstream model. The objective can be summarize as follows [229, 261, 259]:

$$\mathcal{S} = \arg\min_{\mathcal{S}} \mathbb{E}_{\boldsymbol{\theta}_0 \sim P_{\boldsymbol{\theta}_0}} \sum_{t=0}^{T-1} [\mathcal{L}_{\text{dis}}(\nabla_{\boldsymbol{\theta}} \mathcal{L}(\mathcal{S}, \boldsymbol{\theta}_t), \nabla_{\boldsymbol{\theta}} \mathcal{L}(\mathcal{D}, \boldsymbol{\theta}_t))] \tag{3.6}$$

where $P_{\boldsymbol{\theta}_0}$ stands for the distribution over the initialization of network parameters.

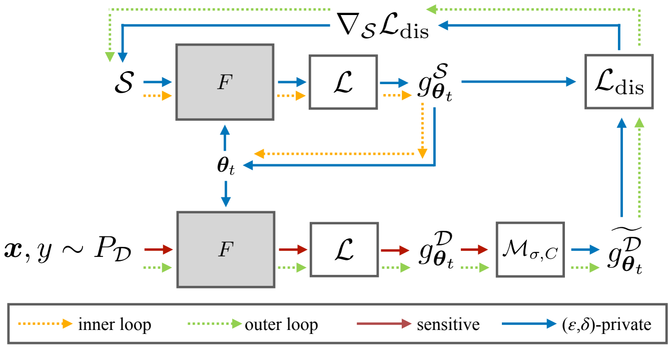

$\nabla_{\mathcal{S}}\mathcal{L}_{\text{dis}}$
$\mathcal{S}$
F
$\mathcal{L}$
$g^{\mathcal{S}}_{\boldsymbol{\theta}_t}$
$\mathcal{L}_{\text{dis}}$
$\boldsymbol{\theta}_t$
$\boldsymbol{x}, y \sim P_{\mathcal{D}}$
F
$\mathcal{L}$
$g^{\mathcal{D}}_{\boldsymbol{\theta}_t}$
$\mathcal{M}_{\sigma,C}$
$\widetilde{g^{\mathcal{D}}_{\boldsymbol{\theta}_t}}$
inner loop
outer loop
sensitive
(ε,δ)-private

Figure 3.1: Illustration for the training pipeline.

We incorporate DP constraints by sanitizing the stochastic gradient on real data $\nabla_{\boldsymbol{\theta}}\mathcal{L}(\mathcal{D}, \boldsymbol{\theta}_t)$ at each outer iteration, while leaving the inner iterations unchanged as their privacy is guaranteed by the post-processing theorem 3.2.1. The final objective can be formulated as follows:

$$\mathcal{S} = \arg\min_{\mathcal{S}} \mathbb{E}_{\boldsymbol{\theta}_0 \sim P_{\boldsymbol{\theta}_0}} \sum_{t=0}^{T-1} [\mathcal{L}_{\text{dis}}(g^{\mathcal{S}}_{\boldsymbol{\theta}_t}, \widetilde{g^{\mathcal{D}}_{\boldsymbol{\theta}_t}})] \tag{3.7}$$

where we use $\widetilde{g^{\mathcal{D}}_{\boldsymbol{\theta}_t}}$ to denote the parameter gradient on $\mathcal{D}$ that is sanitized via Gaussian mechanism 3.2.2, and $g^{\mathcal{S}}_{\boldsymbol{\theta}_t}$ to denote the parameter gradient on $\mathcal{S}$. The whole pipeline is summarized in Algorithm 1 and illustrated in Figure 3.1. We use the subsampled Renyi-DP accountant [1, 143] to compute the overall privacy cost accumulated for iteratively updating $\mathcal{S}$. Note that the training procedure and the privacy computation are approximately as simple as training a classification network with DP-SGD, which in general has lower difficulty than training a DP deep generative models as done in existing works (witnessed by a significant performance gap in terms of the classification accuracy). Moreover, in contrast to previous works, our synthetic set $\mathcal{S}$ is directly optimized for downstream tasks, which naturally leads to superior downstream utility to existing approaches.

3.4 Related Work

Differentially Private Generative Models. Training deep generative models in a private manner has become the default choice for private high-dimensional data generation. Existing methods typically adopt differentially private stochastic gradient descent (DP-SGD) [1, 211, 32, 24] or Private Aggregation of Teacher Ensembles (PATE) [158, 159, 130, 227] to equip the deep generative models with rigorous privacy guarantees. Despite significant progress in mitigating training instabilities and improving generation (visual) quality, existing works are still far from being optimal in terms of the sample utility. This is mainly because existing works are attempting to solve a problem that is inherently hard and almost impossible to be solved accurately under the current private training framework. In contrast, we directly optimize the samples (rather than the deep generative models that are much harder to train in a private setting) and exploit the knowledge from a general class of downstream tasks that can be employed on the samples to further guide the training.

Coreset Selection and Generation. Our work is largely motivated by recent success in distilling a large dataset into a much smaller set of representative samples, i.e., the coreset. For example, samples from a dataset are selected to be representative based on their ability to mimic the gradient signal [145], hardness to fit [209], distance to the cluster centers [234, 173], etc. Instead of selecting samples from the dataset, our work focus on synthesizing informative samples from scratch [229, 261, 259, 127] under DP constraints, and optimizing the sample utility for training downstream neural networks. While recent work [48] has shown promising results in dataset distillation under privacy concerns, obtaining strict privacy guarantees has remained challenging. Our set generation formulation is also similar in spirit to works in the field of private queries release [179, 73, 19, 72] which synthesize a set of pseudo-data (under

(a)

	MNIST		FashionMNIST	
	ε=1	ε=10	ε=1	ε=10
DP-CGAN	-	52.5	-	50.2
G-PATE	58.8	80.9	58.1	69.3
DataLens	71.2	80.7	64.8	70.6
GS-WGAN	-	84.9	-	63.1
DP-Merf	72.7	85.7	61.2	72.4
DP-Sinkhorn	-	83.2	-	71.1
Ours (spc=20)	**80.9**	**95.6**	**70.2**	**77.7**

(b)

	MNIST			FashionMNIST		
	spc=10	spc=20	full	spc=10	spc=20	full
Real	93.6	95.9	99.6	74.4	77.4	93.5
DPSGD	-	-	96.5	-	-	82.9
DP-CGAN	57.4	57.1	52.5	51.4	53.0	50.2
GS-WGAN	83.3	85.5	84.9	58.7	59.5	63.1
DP-Merf	80.2	83.2	85.7	66.6	67.9	72.4
Ours	**94.9**	**95.6**	-	**75.6**	**77.7**	-

Table 3.1: Test accuracy (%) on real data of downstream ConvNet classifiers when training on the synthetic set with $\delta = 10^{-5}$. **(a)** Comparison under different privacy cost $\varepsilon \in \{1, 10\}$. **(b)** Comparison when varying the number of samples per class (spc) for training the downstream ConvNet with $\varepsilon = 10$, while "full" corresponds to 6000 samples per class. We show the results when training on real data non-privately and with DPSGD [1] as reference.

DP guarantees) that is representative of the original data in answering linear queries. However, as neural networks exhibit highly nonlinear properties, methods targeted at linear queries are generally not applicable to our case and are algorithmically distinct from approaches designed for neural nets.

3.5 Experiment

3.5.1 Classification

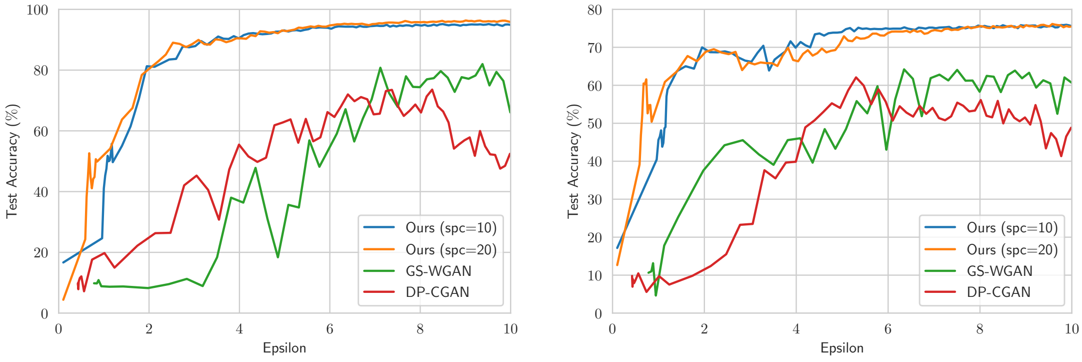
We first compare private set generation (PSG) with existing DP generative models on standard classification benchmarks including MNIST [108] and FashionMNIST [238].

Setup. We use by default a ConvNet with 3 blocks where each block contains one Conv layer with 128 filters, followed by Instance Normalization [218], ReLU activation and AvgPooling modules, and a fully connected (FC) layer as the final output layer. We initialize the network parameters using Kaiming initialization [75] and the synthetic samples using standard Gaussian. We report the averaged results over 3 runs of experiments for all the comparisons. We list below the default hyperparameters used for the main experiments and refer to the Appendix B for more details: Clipping bound $C = 0.1$, R =1000 for $\varepsilon = 10$ (and 200 for $\varepsilon = 1$), number of samples per class (spc) $\in \{10, 20\}$, $K = 10$, $T = 10$ for spc=10 (and =20 for spc=20).

Comparison to state of the art. We show in Table 3.1(a) the results of, to the best of our knowledge, all existing DP high-dimensional data generation methods (whose validity has been justified via peer review at top-tier conferences) that report results on the benchmark datasets we consider. These include DP-CGAN [211], G-PATE [130], DataLens [227], GS-WGAN [32], DP-Merf [70], DP-Sinkhorn [24]. For methods that are not open-sourced, we report the original results from the published paper. As shown in Table 3.1(a), our formulation results in significant improvement in the sample utility (measured by test accuracy on real data) for training downstream classification models. Specifically, the improvement is consistent and significant (around 5-10% increase over different configurations) for both the low privacy budget regime (ε=1) (around 8-9% improvement over SOTA in this case) and a relatively high privacy regime (ε=10) where all the investigated methods achieve convergence (around 10% and 5% increase in test accuracy for MNIST and FashionMNIST, respectively). Note that in contrast to most existing methods that show superiority only for a certain range of privacy

	MNIST						FashionMNIST					
	ConvNet	LeNet	AlexNet	VGG11	ResNet18	MLP	ConvNet	LeNet	AlexNet	VGG11	ResNet18	MLP
Real	99.6	99.2	99.5	99.6	99.7	98.3	93.5	88.9	91.5	93.8	94.5	86.9
DP-CGAN	50.2	52.6	52.1	54.7	51.8	54.3	50.2	52.6	52.1	54.7	51.8	54.3
GS-WGAN	84.9	83.2	80.5	87.9	89.3	74.7	54.7	62.7	55.1	57.3	58.9	65.4
DP-Merf	85.7	87.2	84.4	81.7	81.3	85.0	72.4	67.9	64.9	70.1	66.7	**73.1**
Ours (spc=10)	94.9	91.3	90.3	93.6	**94.3**	86.1	75.6	**68.0**	**66.2**	74.7	**72.1**	62.8
Ours (spc=20)	**95.6**	**93.0**	**92.3**	**94.5**	94.1	**87.1**	**77.7**	**68.0**	59.1	**76.8**	70.8	62.2

Table 3.2: Comparison of generalization ability across different network architecture with $(\varepsilon, \delta) = (10, 10^{-5})$. Our generated set is optimized with *ConvNet*, while the downstream classifiers are of different architecture. The classifiers are trained on the full synthetic set for baseline methods.



(a) MNIST

(b) FashionMNIST

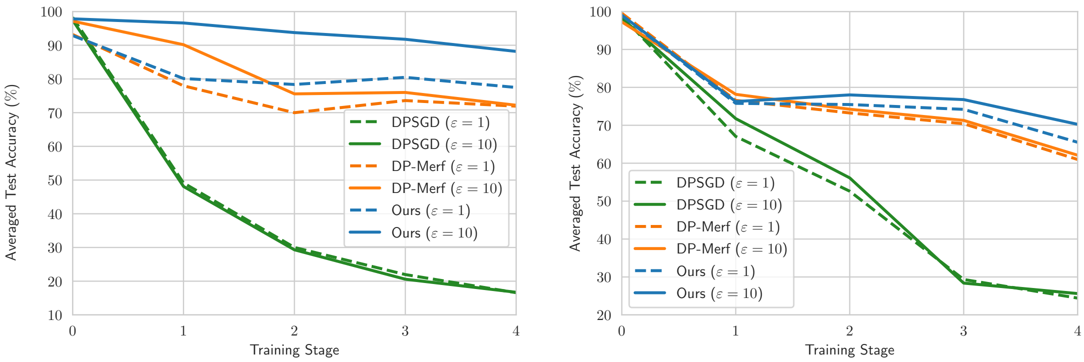
Figure 3.2: Comparison of the convergence rate to existing private generative models with iteratively accumulated privacy cost. X-axis: privacy cost ε, Y-axis: utility (i.e., test accuracy (%)) for training downstream ConvNet classifiers.

levels, our improvement covers a wide range, if not all, of practical scenarios spanning across different privacy levels.

We then focus on the open-sourced methods that are strictly comparable (e.g., G-PATE and DataLens provide data-dependent ε, i.e., publishing ε value will introduce privacy cost and are thus not directly comparable) to ours and conduct a comprehensive investigation through different angles.

Memory and computation cost. We additionally show that our method is the only one that simultaneously shows advantages in reducing the memory and computation consumption of downstream analysis. As shown in Table 3.1(b), training the classifier with full (6000 samples per class) size of samples in most cases yields an upper bound for the test accuracy, while training on randomly subsampled smaller sets will decrease the performance, unless the generated samples are not informative such that they can be harmful to the downstream tasks (e.g., for DP-CGAN). In contrast, we directly optimize to compress the useful information into a small set of samples and naturally save the memory and computation consumption for downstream analysis tasks.

Generalization ability across different architectures. One natural concern of our formulation could be the generalization ability to unseen situations. While we exploit discriminative information to guide the training, we (in principle) inevitably trade the generality off against task-specific utility, leaving no performance guarantees for new models. Interestingly, as shown in Table 3.2, we find that our generated set still provides better utility than all baseline methods



(a) SplitMNIST

(b) SplitFashionMNIST

Figure 3.3: Comparison for private training in the continual learning setting with $\delta = 10^{-5}$ and different ε. X-axis: training stage, Y-axis: averaged test accuracy (over all the stages till the current one). We use a ConvNet classifier in this case and set spc $= 10$ for our method and spc $= 6000$ for DP-Merf as default.

in most cases, even though the models for evaluation have a completely different architecture from the one we used for training. The only case where our generated set does not work well is for training MLPs. We conjecture that it is due to the difference in the network properties that result in distinct gradient signals: for example, layers in MLPs are densely connected while being sparsely connected in ConvNets, and Convolutional layers are translation equivalent while FC layers in MLPs are not. Moreover, we argue that this may not be a bug, but a feature. Note that the reference results on real data also indicate that the MLP is inferior to other architectures while models with ConvNet, VGG, or ResNet architecture perform well in most cases. In this regard, results on our generated set generally align well with the result on real data, which suggests the possibility of conducting model selection with our private generated set.

Convergence rate. For most private (gradient-based) iterative methods, the privacy cost accumulates in each training iteration, and thus faster convergence is highly preferable. We show in Figure 3.2 the training curves where the y-axis denotes the utility and the x-axis corresponds to the privacy. We observe that our method generally has a much faster convergence rate than existing methods that need to accumulate the privacy cost for each iteration. In particular, our method already achieves a decent level of utility with $\varepsilon \leq 2$ which is much lower than the privacy budget used in most previous works (normally $\varepsilon = 10$).

3.5.2 Application: Private Continual Learning

The utility guarantee of our formulation requires that the network architecture is known to the data provider/generator. Fortunately, it is not a rare case in practice. In particular, our method is naturally applicable to cases where *(i)* there are multiple parties involved in training a model and they agree on one common training protocol (i.e., the network architecture is known to all participants); *(ii)* each party has its own data whose privacy need to be protected (i.e., the training need to be DP); *(iii)* data on each party exhibit distinct property and is all informative for the final task (i.e., a synthetic set of representative samples that capture such properties would greatly aid the final task).

One example is continual learning [119, 187] where the training of the classification network is split into stages. Here we consider a setting adjusted to the DP training: to protect the privacy of its data, each party is responsible for a different training stage where it performs DP training of the model on its data, and subsequently delivers the DP model to the party responsible for the next training stage. Note that the raw data would not be transferred as otherwise the privacy would be leaked.

We conduct DP training on the SplitMNIST and SplitFashionMNIST datasets where the data is partitioned into 5 parts (based on the class labels, which corresponds to the class-incremental [173] setup) and we assume each part is held by one party and can not be accessed by others for privacy sake (See Appendix B for more details). We show in Figure 3.3 (green curves) the baseline results of DP training of model under the private class-incremental setting (i.e., each party finetune the model is obtained from the previous stage on its own data using DP-SGD). Apparently, this naive training scheme leads to catastrophic forgetting of information learned in the early stages. Even worse is that the common strategy to cope with this issue requires transferring a small set of real data to other parties such that it can be replayed in the later training stage [173, 15, 166], which is not directly applicable to the private setting as transferring the data breaks privacy. In contrast, private generation methods can be seamlessly applied to this case, where a set of DP synthetic samples is transferred to enable the final model to learn the characteristics of each partition of data. In particular, our formulation is better suitable for this setting than other generation methods as the network architecture is known to all participants and samples can be tailored to the specific network via our formulation. This is verified in Figure 3.3, where our synthetic samples are generally more informative for training the classifier when compared to DP-Merf – the overall best existing works in terms of the downstream utility. Moreover, as our formulation condenses the information into a small set of samples by construction, we also enjoy the advantages when considering the computation, storage, and communication cost.

3.6 Discussion

In this section, we present several key factors that distinguish our approach from existing ones and discuss possible concerns regarding our private set generation formulation.

Trade-off between Visual Quality and Task Utility. Our formulation is designed for optimizing the utility of downstream analysis tasks instead of the visual appearance as done in previous works, thereby leaving no performance guarantee for the visual quality of the synthetic samples. Moreover, the optimization of the private synthetic set is unconstrained and unregulated over the whole data space, with the gradient signal as the only guidance. As the data to gradient mapping is generally non-injective (i.e., different data can results in the same gradient), searching for the correct data given the gradient is an indefinite problem, which inevitably leads to outcomes that are out of the data manifold in practice. This can be seen in the first row of Figure 3.5 where we plot our private synthetic samples trained under the default setting.

Recall that one key difference between our formulation and existing works is that: we directly optimize for a set of samples instead of the deep generative models. We then take a further step and investigate whether this difference is the key factor that determines the samples' visual quality. To do this, we employ a *untrained*

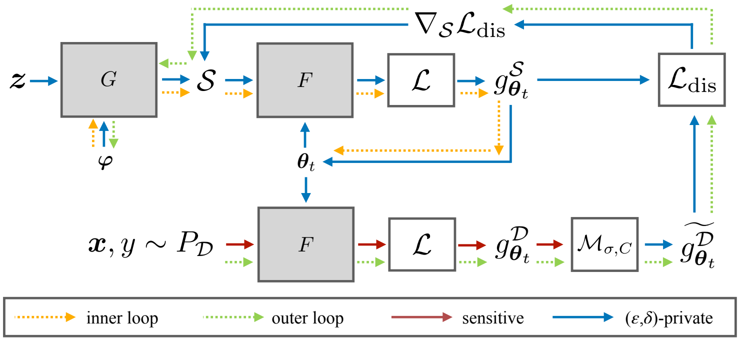
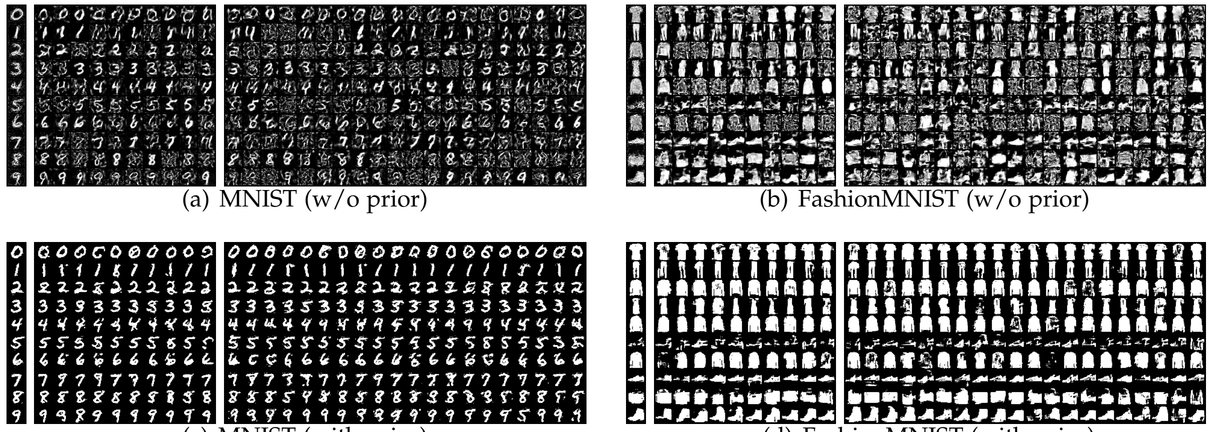

$\nabla_{\mathcal{S}}\mathcal{L}_{\text{dis}}$
z
G
$\mathcal{S}$
F
$\mathcal{L}$
$g^{\mathcal{S}}_{\theta_t}$
$\mathcal{L}_{\text{dis}}$
φ
θ_t
$x, y \sim P_{\mathcal{D}}$
F
$\mathcal{L}$
$g^{\mathcal{D}}_{\theta_t}$
$\mathcal{M}_{\sigma,C}$
$\widetilde{g^{\mathcal{D}}_{\theta_t}}$
inner loop
outer loop
sensitive
(ε,δ)-private

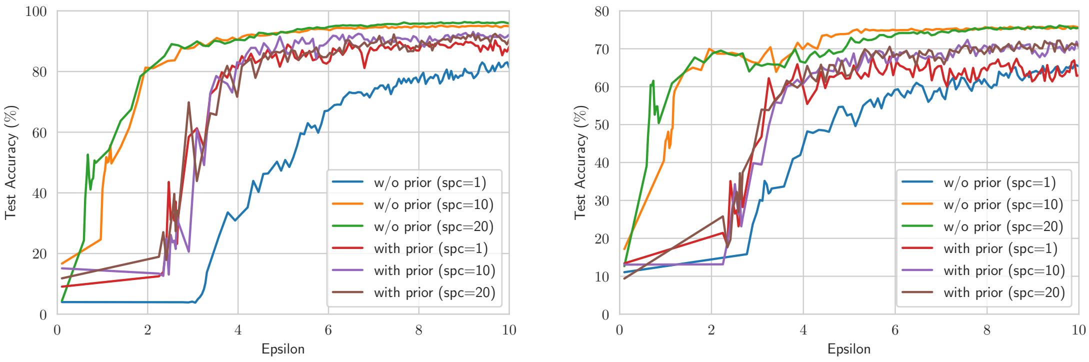
Figure 3.4: Training pipeline (with prior).



(a) MNIST (w/o prior) (b) FashionMNIST (w/o prior)

(c) MNIST (with prior) (d) FashionMNIST (with prior)

Figure 3.5: Our synthetic samples under $(\varepsilon, \delta) = (10, 10^{-5})$ for MNIST and FashionMNIST datasets with or without (w/o) incorporating a freshly initialized DCGAN generator network as image prior.



(a) MNIST (b) FashionMNIST

Figure 3.6: Comparison of the convergence rate when training with or without (w/o) the image prior from the DCGAN architecture. X-axis: privacy cost ε, Y-axis: test accuracy (%) for training downstream ConvNet classifiers.

DCGAN [168] model from [32] as the generator backbone (denoted as G), let $\boldsymbol{x}^{\mathcal{S}}$ to be outputs of G, and then optimize over the network parameter of G using the gradient matching loss as in Equation 3.5 (see Figure 3.4 for a visual illustration). Mathematically, this transforms Equation 3.7 into:

$$\min_{\boldsymbol{\varphi}} \mathbb{E}_{\boldsymbol{\theta}_0 \sim P_{\boldsymbol{\theta}_0}} \sum_{t=0}^{T-1} [\mathcal{L}_{\text{dis}}(g^{\mathcal{S}}_{\boldsymbol{\theta}_t}, \widetilde{g^{\mathcal{D}}_{\boldsymbol{\theta}_t}})] \quad \text{with} \quad \mathcal{S} = \{G(\boldsymbol{z}_i; \boldsymbol{\varphi}), y_i^{\mathcal{S}}\}_{i=1}^{M} \tag{3.8}$$

where $\boldsymbol{\varphi}$ is the parameter of G, $\boldsymbol{z}_i$ is random Gaussian noise (fixed during training). Basically, this formulation restricts the synthetic images to be within the output space of G, and the inductive bias introduced by the convolutional structure serves as deep image prior [219] to regularize the visual appearance of the synthetic images.

We show the synthetic samples in the second row of Figure 3.5, and compare the utilities with our original formulation in Figure 3.6. We observe that the prior from the deep generative model is indeed important for the visual quality. However, interestingly, better visual quality does not mean better utility. Specifically, optimizing over the parameter of generator G exhibits a slower convergence than directly optimizing the samples, while the final performance is also inferior (See quantitative results in Table 3.3). This gives several important indications that help inform future research in this field: *(i)* the goal of achieving better downstream utility may be incompatible with the goal of achieving better sample visual quality, while dedicated efforts towards different goals are necessary; *(ii)* deep generative models may not be the best option for the task of private data generation as they result in suboptimal utility (mainly due to its slow convergence), which questions the current default way of thinking in this field.

Scalability & Transparency. We discuss the possible issues when scaling to more complicated datasets which: *(i)* contains a large number of label classes; *(ii)* are diverse and require a large number of samples to capture the statistical characteristics of the data distribution. For *(i)*, the complexity of our (and all the other) approaches will definitely increase as the number of label classes increases. When considering the number of variables that need to be optimized, the complexity increases linearly for our case, while for all methods (that optimize over the network parameters) the increase is no less than ours. While the application of DP deep learning (of discriminative models) to datasets with >10 label classes is rare, we anticipate that dealing with a much larger number of label classes is too ambitious for DP generative modeling for now. For *(ii)*, we conduct the experiment when varying the number of samples per class and present the results in 3.4, where we indeed observe the training difficulty when the number of samples increases. We conjecture that it is mainly because the gradient signals for updating the synthetic samples get sparser when the number increases, which results in a lower convergence rate and thus worse results especially when the allowed privacy budget is low. However, it is arguable whether this is a shortage as smaller amounts of samples allow more savings in the storage and computation consumption while providing greater transparency of downstream analysis.

	MNIST			FashionMNIST		
	1	10	20	1	10	20
w/o prior	81.4	**94.9**	**95.6**	**66.7**	**75.6**	**77.7**
with prior	**88.2**	92.2	90.6	63.0	70.2	70.7

Table 3.3: Test accuracy (%) on real data of downstream ConvNet classifier with or without (w/o) adopting image prior from DCGAN under $(\varepsilon, \delta) = (10, 10^{-5})$.

MNIST				FashionMNIST			
1	10	20	50	1	10	20	50
81.4	94.9	95.6	94.0	66.7	75.6	77.7	71.3

Table 3.4: Test accuracy (%) on real data of downstream ConvNet classifier when varying the numbers of samples per class (spc) under $(\varepsilon, \delta) = (10, 10^{-5})$.

Generality and Expressiveness. Our formulation focus on the task of training downstream neural networks, and thus have no guarantees for other (and more general) purpose. In contrast, deep generative models are designed for capturing the complete data distribution and, once perfectly trained, can be applied to more general cases. While our formulation seems to be inferior in this regard, we argue that this should not be a major shortcoming that outweighs the advantages: First of all, while deep generative models in principle have much greater expressiveness than a small set of samples, such upper bound is hard, if not impossible, to be achieved in the privacy learning setting. Instead, compromising the upper bound for a more achievable target is worthy and shows great improvement over existing works as demonstrated in Section 3.5.1. Moreover, our formulation generalizes seamlessly to any gradient-based learning methods that a downstream analyst may adopt. While such methods already cover the most part of the possible analysis algorithms that could be adopted for high-dimensional data, we believe that our approach does exhibit a good level of practical applicability.

3.7 Conclusion

We introduce a novel view of private high-dimensional data generation: instead of attempting to train deep generative models in a DP manner, we directly optimize a set of samples under the supervision of discriminative information for downstream utility. We present a simple yet effective method that allows synthesizing a small set of samples that are representative of the original data distribution and informative for training downstream neural networks. We demonstrate via extensive experiments that our formulation leads to great improvement over state-of-the-art approaches in terms of the task utility, without losing the generality for performing analysis tasks in practice. Moreover, our results question the current default way of thinking and provide insights for further pushing the frontier in the field of private data generation.

4 A Unified View of Differentially Private Deep Generative Modeling

Contents

The availability of rich and vast data sources has greatly advanced machine learning applications in various domains. However, data with privacy concerns comes with stringent regulations that frequently prohibit data access and data sharing. Overcoming these obstacles in compliance with privacy considerations is key for technological progress in many real-world application scenarios that involve sensitive data. Differentially private (DP) data publishing provides a compelling solution, where only a sanitized form of the data is publicly released, enabling privacy-preserving downstream analysis and reproducible research in sensitive domains. In recent years, various approaches have been proposed for achieving privacy-preserving high-dimensional data generation by private training on top of deep neural networks. In this paper, we present a novel unified view that systematizes these approaches. Our view provides a joint design space for systematically deriving methods that cater to different use cases. We then discuss the strengths, limitations, and inherent correlations between different approaches, aiming to shed light on crucial aspects and inspire future research. We conclude by presenting potential paths forward for the field of DP data generation, with the aim of steering the community toward making the next important steps in advancing privacy-preserving learning.

This chapter is based on [30]: As the first author of [30], Dingfan Chen spearheaded its creation, from proposing the central idea and conducting the literature review to conceptualizing a structured tug literature presentation approach, and serving as the main writer. This paper was published in Transactions on Machine Learning Research and received a survey certificate.

4.1 Introduction

Data sharing is crucial for the growth of machine learning applications across various domains. However, in many application scenarios, data sharing is prohibited due to the private nature of data (e.g., individual data from mobile devices, medical treatments, and banking records) and associated stringent regulations, such as the General Data Protection Regulation (GDPR) and the American Data Privacy Protection Act (ADPPA), which largely hinders technological progress in sensitive areas. Fortunately, differentially private (DP) data publishing [51, 52, 58] provides a compelling solution, where only a sanitized form of the data, with rigorous privacy guarantees, is made publicly available. Such sanitized synthetic data can be leveraged as a surrogate for real data, enabling downstream statistical analysis using established analytic tools, and can be shared openly with the research community, promoting reproducible research and technological advancement in sensitive domains.

Traditionally, the sanitization algorithms are designed for capturing low-dimensional statistical characteristics and target at specific downstream tasks (e.g., answering linear queries [179, 73, 19, 223]), which are hardly generalizable to unanticipated tasks involving high-dimensional data with complex distributions. On the other hand, the latest research, inspired by the recent successes of deep generative models in learning high-dimensional representations, applies deep generative models as the foundation of the generation algorithm. This line of approaches, as demonstrated in recent studies [24, 32, 239, 245, 14, 62], have shown promising results in sanitizing high-dimensional samples for general purposes.

Towards designing models that are better compatible with the privacy target, recent research typically customizes the training objective for privacy-centric scenarios [24, 70, 32, 129], all building on top of a foundational generic generator framework. However, research is fragmented as contributions have been made in different domains, different modeling paradigms, different metric and discriminator choices, and different data modalities. So far, a unified view of private generative models is notably missing in the literature, despite its potential to consolidate the design space for systematic exploration of innovative architectures and leveraging strengths across diverse modeling frameworks.

In this paper, we pioneer in providing a comprehensive framework and a unified perspective on existing approaches for differentially private deep generative modeling. Our innovative framework, complemented by an insightful taxonomy, effectively encapsulates approaches from existing literature, categorizing them according to the intrinsic differences in their underlying privacy barriers. We thoroughly assess each category's characteristics, emphasizing crucial points relevant for privacy analysis, and discuss their inherent strengths and weaknesses, with the aim of laying a foundation that supports seamless transition into potential future research.

Moreover, we present a thorough introduction to the key concepts of DP and generative modeling. We highlight the key considerations that should be accounted for when developing DP generative models to ensure results comparable, error-free results. Furthermore, we introduce a taxonomy of existing representative types of deep generative models, classifying them based on the distinctive privacy challenges present during DP training. This introduction aims to equip researchers and practitioners with a systematic approach for the design and implementation of future privacy-preserving data generation techniques.

Lastly, we discuss open issues and potential future directions in the broader field of developing DP generation methods. Our objective is not limited to reviewing existing techniques, but also aims to equip readers with a systematic perspective for devising new approaches or refining existing ones. This work is thoughtfully written to serve diverse audiences, with an effort of providing practitioners with a comprehensive overview of the recent advancements, while aiding experts in reassessing existing strategies and designing innovative solutions for privacy-preserving generative modeling.

4.2 Preliminaries of Differential Privacy

Setting. In this paper, we focus on the standard *central* model of DP, which is commonly agreed upon by all the approaches referenced herein. In this model, a trusted party or server is responsible for managing all data points, executing DP algorithms, and producing sanitized data that conforms to privacy constraints. This sanitized data, generated from the implemented DP algorithms, can be later shared with untrusted parties or released to the public while ensuring strict privacy guarantees. It is noteworthy that although approaches based on local DP may seem to generate a form of synthetic data—where users typically modify their own data due to distrust in the central server and a desire to conceal private information—these methods are fundamentally distinct from the ones explored in this work due to differing threat models and the resulting privacy implications.

Definition 4.2.1 ((ε, δ)-DP [51])**.** A randomized mechanism $\mathcal{M}$ with range $\mathcal{R}$ is (ε, δ)-DP, if

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

holds for any subset of outputs $\mathcal{O} \subseteq \mathcal{R}$ and for any adjacent datasets $\mathcal{D}$ and $\mathcal{D}'$, where $\mathcal{D}$ and $\mathcal{D}'$ differ from each other with only one training example. ε is the upper bound of privacy loss, and δ is the probability of breaching DP constraints. Smaller values of both ε and δ translate to stronger DP guarantees and better privacy protection. Typically, $\mathcal{M}$ refers to the training algorithm of a generative model. DP ensures that inferring the presence of an individual in the private dataset—by observing the trained generative models $\mathcal{M}(\mathcal{D})$—is challenging, with $\mathcal{D}$ being the original private dataset. This same level of guarantee also holds when the attacker observes the samples generated by the trained generative models (i.e., the sanitized dataset) due to the post-processing theorem (Theorem 4.2.1).

Privacy notion. There are two widely used definitions for adjacent datasets in existing works of DP data generation, which result in different DP notions: the "replace-one" and the "add-or-remove one" notions:

- **Replace-one**: adjacent datasets are formed by *replacing* one data sample, i.e., $\mathcal{D}' \cup \{x'\} = \mathcal{D} \cup \{x\}$ for some x and x'. This is sometimes referred to *bounded-DP* in literature.
- **Add-or-remove-one**: adjacent datasets are constructed by *adding* or *removing* one data sample, i.e., $\mathcal{D}' = \mathcal{D} \cup \{x\}$ for some x (or vice versa).

It is crucial to understand that different notions of DP may not provide equivalent privacy guarantees even under identical (ε, δ) values, potentially leading to slight differences in comparisons when algorithms are developed under varying privacy notions, a sentiment also noted in [165]. Specifically, the "replacement" operation in the bounded-DP notion can be understood as executing two edits: removing one data point x and adding another x'. This suggests that the *replace-one* notion may be nested within the *add-or-remove-one* notion, and a naive transformation would result in a (2ε, 0)-DP algorithm under the *replace-one* notion from an algorithm that was (ε, 0)-DP under the *add-or-remove-one* notion. To minimize potential confusion and promote fair comparisons, we emphasize that future researchers should clearly specify the chosen notion in their work. Moreover, we encourage future research to include a privacy analysis for both notions, if technically feasible.

Privacy-preserving data generation is building on top of the closedness of DP under post-processing: if a generative model is trained under a (ε, δ)-DP mechanism, releasing a sanitized dataset generated by the model (for conducting downstream analysis tasks) will also be privacy-preserving, with the privacy cost bounded by ε (and δ).

Theorem 4.2.1 (Post-processing [53]). If $\mathcal{M}$ satisfies (ε, δ)-DP, $F \circ \mathcal{M}$ will satisfy (ε, δ)-DP for any data-independent function F with $\circ$ denoting the composition operator.

While (ε, δ)-DP provides an intuitive understanding of the mechanism's overall privacy guarantee, dealing with composition is more convenient under the notion of Rényi Differential Privacy (RDP). Existing approaches typically use RDP to aggregate privacy costs across a series of mechanisms (such as multiple DP gradient descent steps during generative model training) and then convert to the (ε, δ)-DP notion at the end (Appendix C.3). The formal definitions and the corresponding theorems are listed below.

Definition 4.2.2 (Rényi Differential Privacy (RDP) [142]). A randomized mechanism $\mathcal{M}$ is (α, ρ)-RDP with order α, if

$$D_\alpha(\mathcal{M}(\mathcal{D}) \| \mathcal{M}(\mathcal{D}')) = \frac{1}{\alpha - 1} \log \mathbb{E}_{t \sim \mathcal{M}(\mathcal{D})} \left[\left(\frac{\Pr[\mathcal{M}(\mathcal{D}) = t]}{\Pr[\mathcal{M}(\mathcal{D}') = t]} \right)^\alpha \right] \leq \rho$$

holds for any adjacent datasets $\mathcal{D}$ and $\mathcal{D}'$, where $D_\alpha(P \| Q) = \frac{1}{\alpha - 1} \log \mathbb{E}_{t \sim Q}[(P(t)/Q(t))^\alpha]$ denotes the Rényi divergence.

Theorem 4.2.2 (Composition [142]). For a sequence of mechanisms $\mathcal{M}_1, ..., \mathcal{M}_k$ s.t. $\mathcal{M}_i$ is (α, ρ_i)-RDP $\forall i$, the composition $\mathcal{M}_1 \circ ... \circ \mathcal{M}_k$ is $(\alpha, \sum_i \rho_i)$-RDP.

Theorem 4.2.3 (From RDP to (ε, δ)-DP [13]). If a randomized mechanism $\mathcal{M}$ is (α, ρ)-RDP, then $\mathcal{M}$ is also $\Big(\rho + \log((\alpha - 1)/\alpha) - (\log \delta + \log \alpha)/(\alpha - 1), \delta\Big)$-DP for any $0 < \delta < 1$.

In literature, achieving DP typically involves adding calibrated random noise, with scale proportional to the sensitivity value (Definition 4.2.3), to the private dataset's associated quantity to conceal individual influence. A notable instance of this practice can be formularized as the Gaussian Mechanism, as defined below.

Definition 4.2.3 (Sensitivity). The (global) ℓ_p-sensitivity for a function $f : X \to \mathbb{R}^d$ that outputs d-dimensional vectors is defined as:

$$\Delta_f^p = \max_{\mathcal{D}, \mathcal{D}'} \| f(\mathcal{D}) - f(\mathcal{D}') \|_p \tag{4.1}$$

over all adjacent datasets $\mathcal{D}$ and $\mathcal{D}'$. The sensitivity characterizes the maximum influence (measured by ℓ_p norm) of one individual datapoint on the function's output. When dealing with matrix and tensor outputs, the ℓ_p norm is computed over the vectors that result from flattening the matrices and tensors into vectors.

Definition 4.2.4 (Gaussian Mechanism [53]). Let $f : X \to \mathbb{R}^d$ be an arbitrary d-dimensional function with L_2-sensitivity Δ_f^2. The Gaussian Mechanism $\mathcal{M}_\sigma$, parameterized by σ, adds noise into the output, i.e.,

$$\mathcal{M}_\sigma(\boldsymbol{x}) = f(\boldsymbol{x}) + \mathcal{N}(0, \sigma^2 \boldsymbol{I}). \tag{4.2}$$

For $\varepsilon, \delta \in (0, 1)$, $\mathcal{M}_\sigma$ is (ε, δ)-DP if $\sigma \geq \sqrt{2 \ln(1.25/\delta)} \Delta_2 f / \varepsilon$ and $(\alpha, \frac{\alpha(\Delta_f^2)^2}{2\sigma^2})$-RDP.

4.2.1 Training Deep Learning Models with DP

Additionally, we present the most prominent frameworks for training deep learning models with DP guarantees: Differentially Private Stochastic Gradient Descent (DP-SGD) in Section 4.2.1.1 and Private Aggregation of Teacher Ensembles (PATE) in Section 4.2.1.2.

4.2.1.1 *Differenetially Private Stochastic Gradient Descent (DP-SGD)*

DP-SGD [1] is an adaptation of the standard SGD algorithm that injects calibrated random Gaussian noise into the gradients during the optimization process, which ensures DP due to the Gaussian mechanism. The algorithm consists of the following steps:

1. Compute the per-example gradients for a mini-batch of training examples.
2. Clip the gradients to bound their L_2-norm (i.e., L_2-sensitivity) to ensure that the influence of any individual training example is limited.
3. Add Gaussian noise to the aggregated clipped gradients to introduce the required randomness for DP guarantees.
4. Update the model parameters using the noisy gradients.

The privacy guarantees provided by DP-SGD are determined by the choice of noise multiplier (which defines the standard deviation of the Gaussian noise by multiplying it with the sensitivity), the mini-batch sampling ratio, and the total number of optimization steps. The overall privacy guarantee can be calculated using the composition rule, which accounts for the cumulative privacy loss over multiple iterations of the algorithm. By default, DP-SGD adopts the *add-or-remove-one* notion, leading to a sensitivity value equal to the gradient clipping bound (see Appendix C.2).

4.2.1.2 *Private Aggregation of Teacher Ensembles (PATE)*

The PATE framework [158, 159] consists of two main components: an ensemble of teacher models and a student model. The training process begins with the partitioning of sensitive data into multiple disjoint subsets. Each subset is then used to train a teacher model independently (and non-privately), limiting the effect of each individual training sample to influence only one teacher model. To train a DP student model, a public dataset with similar characteristics to the sensitive data is used. During the training process, the student model queries the ensemble of teacher models for predictions on the public dataset. The teacher models' predictions are then aggregated using a DP voting mechanism, which adds noise to the aggregated votes to ensure privacy. The student model subsequently learns from the noisy aggregated predictions, leveraging the collective knowledge of the teacher models while preserving the privacy of the original training data.

The sensitivity of PATE is measured as the maximum change in label counts for teacher models' predictions between neighboring datasets. Given m teacher models, c label classes, the counts for class j is defined by the number of teachers that assign class j to a query input $\bar{x}$, i.e., $n_j(\bar{x}) = |i : i \in [m], f_i(\bar{x}) = j|$ for $j \in [c]$, where f_i denotes the i-th teacher model. Changing a single data point (whether by replacing, adding, or removing) will at most affect one data partition and, consequently, the prediction for one teacher trained on the altered partition, increasing the counts by 1 for one class and decreasing the counts by 1 for another class. This results in a global sensitivity equal to $\Delta^2_{(n_1,\dots,n_c)} = \sqrt{2}$ for both the *replace-one* and *add-or-remove-one* notion (see Appendix C.2). To reduce privacy consumption, PATE is associated with a data-dependent privacy accountant method to exploit the fact that when teachers have a large agreement, the privacy cost is usually much smaller than the data-independent bound would suggest. Moreover, Papernot et al. [159] suggest private threshold checking for queries to only use teacher predictions with high consensus for training the student model. Notably, to obtain comparable results to approaches with data-independent privacy costs, extra sanitization via smooth sensitivity analysis is required.

4.2.2 Important Notes for Deploying DP Models

The development of DP models necessitate a thorough examination to ensure their correctness for providing a fair comparison of research progress and maintaining public trust in DP methodologies. We present below a series of critical questions that serve as fundamental sanity checks when developing DP models. This enables researchers to rapidly identify and rule out approaches that are incompatible with DP, thereby optimizing their research efforts towards innovation in this domain.

- **What will be released to the public and accessible to potential adversaries?** The most critical question is to determine which components (e.g., model modules, data statistics, intermediate results, etc.) will be made public and, as a result, could be accessible to potential adversaries. This corresponds to the assumed ***threat model*** and establishes the essential concept of a ***privacy barrier***, which separates components accessible to potential attackers from those that are not.

  All components within the attacker-accessible domain must be provided with DP guarantees. One common oversight is neglecting certain data-related intermediate statistics utilized during the model's training phase. These statistics might constitute only a minor aspect of the entire process, or their existence might be implicit, given that they are incorporated into other quantities. Nevertheless, failing to implement DP sanitization for these aspects can undermine the intended DP protection for the outcomes, e.g., the trained model may no longer adhere to DP standards.

  For instance, when pre-processing is required for the usage of a DP model, an additional privacy budget should be allocated for exposing related statistics such as the dataset's mean and standard deviation [212]. From a research standpoint, innovations may involve carefully designing DP mechanisms that apply DP constraints only to components accessible by attackers, while other components can be trained or computed non-privately to maintain high utility. A concrete example includes training a discriminator non-privately and withholding it by the model owner in deploying DP generative adversarial networks (see Section 4.4.3) while only privatizing the generator's training and releasing it to the public with a dedicated DP mechanism.

- **What is the adopted privacy notion and granularity?** While DP asserts that an algorithm's output remains largely unchanged when a single database entry is modified, the definition of a "single entry" can vary considerably (reflecting the concept of ***granularity***), and the way to modify the single entry can also be different (embodying the ***privacy notion***). Thus, the claims of DP necessitate an unambiguous declaration of the sense and level at which privacy is being promised. As discussed in the previous section, the distinction in privacy notion is universally crucial in the design of DP mechanisms. On the other hand, the granularity becomes particularly relevant when handling data modalities that exhibit relatively less structural representations, such as graphs and text. For instance, training DP (generative) language models that provide guarantees at different levels (tokens, sentences, or documents) will lead to substantial differences in the complexity and the application scenarios.

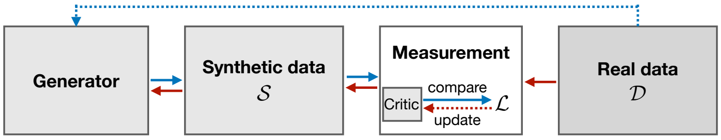


Figure 4.1: Overview of training pipeline of generative models. (**Blue arrow**: forward pass; **Red arrow**: backward pass; Dashed arrows indicate optional processes that may not be present in all generative models.)

- **What constitutes the sensitivity analysis?** Sensitivity analysis demands rigorous attention, focusing on two primary aspects. The first consideration calls for a clear statement of the ***sensitivity type*** in use, e.g., global, local, and smooth sensitivity. Notably, techniques predicated on local and smooth sensitivity are generally not directly comparable to those depending on the global sensitivity. Second, determining the sensitivity bound during the training of a generative model that consists of more than one trainable module may be challenging, as discussed in Section 4.4.4, which necessitates a meticulous analysis to ensure the correctness of the privacy cost computation.

4.3 Preliminaries of Generative Models

In this section, we present a comprehensive overview of representative generative models, with the aim to develop a clear understanding of the essential operations required to achieve DP across different types of generative models, as well as to demonstrate the fundamental differences in their compatibility with private training.

4.3.1 Overview & Taxonomy

Given real data samples $\boldsymbol{x}$ from a dataset of interest, the goal of a generative model is to learn and capture the characteristics of its true underlying distribution $p(\boldsymbol{x})$ and subsequently allows the model to generate new samples from the learned distribution. At a high-level of abstraction, the training pipeline of generative models can be depicted as the diagram in Figure 4.1. The "Measurement" block in the diagram summarizes the general process of comparing the synthetic and real data distributions using a "critic", which yields a loss term $\mathcal{L}$ that quantifies the similarity between the two. This loss term then acts as the training objective for the generator, with the update signal computed and then backpropagated to adjust the generator's parameters and improve its ability to generate realistic samples.

Furthermore, the diagram outlines two optional processes (indicated by dashed arrows), that are involved in some generative models but not all. The first optional process involves guiding the training of the generator by feeding (quantities derived from) real data as inputs, which enables the explicit maximum likelihood computation and categorizes the models into two types: *implicit density* and *explicit density*. The second optional process involves updating the critic to better capture the underlying structure of the data and more accurately reflect the similarity between the distributions. This distinction highlights the usage of either *static (data-independent)* or *learnable (data-dependent)* features for the critic function within implicit density models.

We present a taxonomy of existing representative types of generative models whose private training has been realized in literature in Figure 4.2. We examine the following tiers in the taxonomy trees that exert significant influence on the application scenarios and the design of corresponding private training algorithms:

- *Explicit* vs. *Implicit* Density Models
- *Learnable* vs. *Static* Critics
- *Distribution-wise* vs. *Point-wise* Optimization
- *Tractable* vs. *Approximate* Density

***Explicit* vs. *Implicit* Density Models.** Existing generative models can be divided into two main categories: *explicit density models* define an explicit density function $p_{\text{model}}(\boldsymbol{x}; \boldsymbol{\theta})$, while *implicit density models* learn a mapping that generates samples by transforming an easy-to-sample random variable, without explicitly defining a density function.

These distinctions in modeling design result in different paradigms during the training phase, particularly in how real data samples are used (or accessed) in the process. *Explicit* density models typically use real data samples as inputs to the generator and also for measurement (as demonstrated in Figure 4.1), thereby enabling the tractable computation or approximation of the data likelihood objective. In contrast, *implicit* density models necessitate real data samples solely for the purpose of distribution comparison measurements.

This distinction demarcates potential privacy barriers for these two types of models during DP model training. In the context of implicit models, it is sufficient to privatize the single access point to the real data (Section 4.4.1-4.4.3). However, when dealing with training private *explicit* density models, it becomes essential to apply DP mechanisms that take both access points into account.

***Learnable* vs. *Static* Critics.** The training of generative models necessitates a "critic" to assess the distance between the real and generated distributions, which then builds up the training objectives for optimizing the generator. Specifically for *implicit* density models, the use of different types of critics could potentially influence the placement of privacy barrier when training DP models (Section 4.4.1- 4.4.2).

Within this framework, the critics may exist in two primary forms, namely *learnable* and *static* (data-independent) variants. The distinction between the two lies in whether the critic itself is a parameterized function that undergoes updates during the training of generative models (*learnable*), or a data-independent function that remains static during the training process (*static*).

We do not further differentiate for explicit density models as they typically employ simple, data-independent critic such as L_1 and L_2 losses. Meanwhile, in contrast to implicit models, varying the critics in explicit models typically does not alter the privacy barrier in DP training. This is due to the constraint imposed by multiple access to real data in training of explicit models, which restricts the flexibility in positioning the privacy barriers.

***Distribution-wise* vs. *Point-wise* Optimization.** Generative models are designed to be stochastic and capable of producing a distribution of data. This is achieved by supplying the generator with random inputs (i.e., latent variables), stochastically drawn from a simple distribution, such as the standard Gaussian. The optimization process generally proceeds through mini-batches, essentially serving as point-wise approximations. Through substantial number of update steps that involve various random latent variable inputs, the model is trained to generalize over new random variables during the generation phase, enabling a smooth transition from a *point-wise* approximation to the *distribution-wise* objective.

However, certain contexts may not necessitate the stochasticity nature in these models.

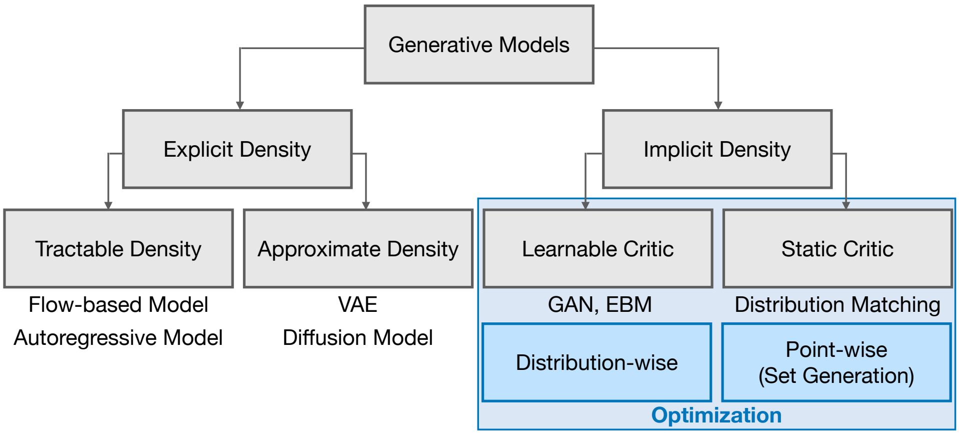


Figure 4.2: Overview of different deep generative models.

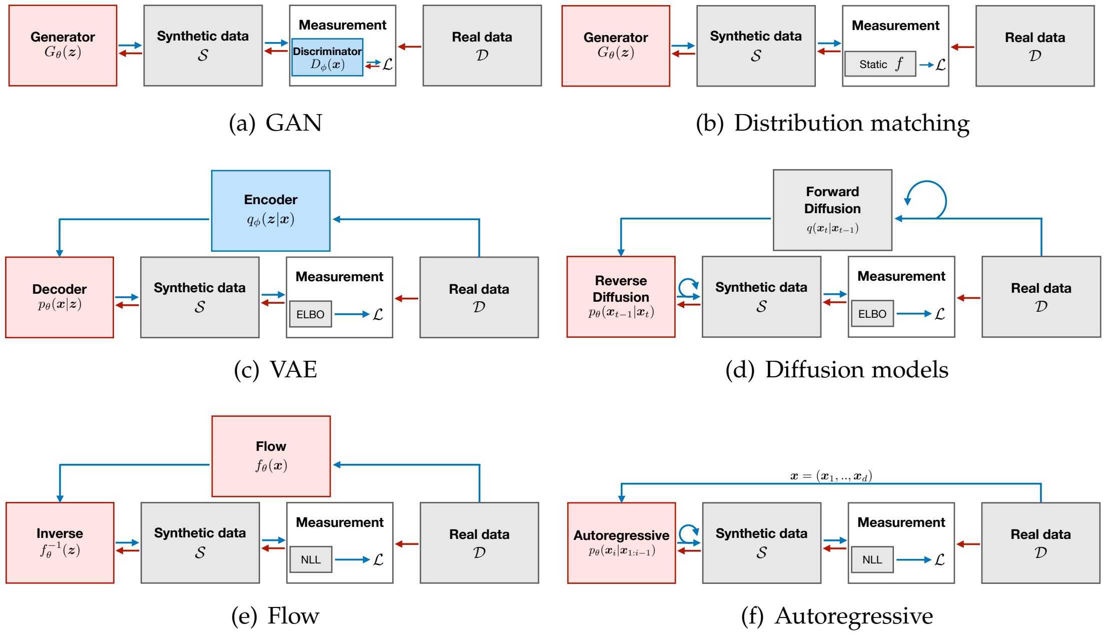
Instead, there might be an intentional focus on generating a small set of representative samples, a notion that resonates with the "coreset" concept. This could involve optimizing the model over a limited, fixed set of random inputs rather than the entire domain. We label this as *point-wise optimization* to distinguish it from the default *distribution-wise optimization* used in training conventional generative models.

Recent studies have revealed intriguing advantages of merging insights from both these strategies, particularly in the realm of private learning. For instance, the point-wise optimization method exhibits remarkable compatibility with private learning primarily arises from the fact that *point-wise* optimization is generally less challenging in comparison to the *distribution-wise* training that requires generalization, which generally improves model convergence, and consequently enhances privacy. However, this point-wise approach has its limitations. Unlike distribution-wise training, it does not inherently support generalization over new latent code inputs. This may restrict the stochastic sampling of new synthetic samples during inference. As a result, there is a trade-off between the flexibility of use in downstream applications and improved privacy guarantees.

We do not expressly differentiate between potential optimization strategies for explicit density models within our taxonomy in Figure 4.2, as such distinction is not obvious in the context of explicit density models. In these models, the latent space is typically formulated through a transformation of the distribution within the data space. This transformation process in turn complicates the control of stochasticity throughout the training phase and diminishes the applicability of point-wise optimization.

***Tractable* vs. *Approximate* Density.** For models defining explicit density, a key distinguishing factor that shows practical relevance pertains to whether they allow exact likelihood computations. These models can broadly be categorized into two types: *tractable* density and *approximate* density models. The classification primarily stems from the model structural designs, which either enable tractable density inference or fall within the realm of approximate density.

Existing studies have demonstrated encouraging results when conducting DP training on both types of models. Intriguingly, the DP training mechanisms appear to exhibit minor distinctions when applied to these two different categories. On an optimistic note, such results implies that it might be feasible to attain tractable likelihood computations with a DP guarantee without considerable effort. However, it remains unclear as to whether the difference in model designs will systematically influence their compatibility with DP training.



(a) GAN

(b) Distribution matching

(c) VAE

(d) Diffusion models

(e) Flow

(f) Autoregressive

Figure 4.3: Diagram illustrating training process in generative models. **Blue arrow**: forward pass; **Red arrow**: backward pass.

4.3.2 Representative Models

We provide an illustration of the operational flow of representative generative models in Figure 4.3. As demonstrated, existing representative generative models can be effectively encapsulated within our unified framework shown in Figure 4.1. We proceed to briefly discuss the key characteristics of each type of generative models and their relation to potential implementations for DP training in this subsection.

4.3.2.1 *Implicit Density Models*

As a canonical example of implicit density model, *Generative Adversarial Network (GAN)* [65] employs a generator, $G_{\boldsymbol{\theta}}$ (parametrized by $\boldsymbol{\theta}$), to learn the data distribution with the aid of a discriminator $D_{\boldsymbol{\phi}}$ (parametrized by $\boldsymbol{\phi}$) trained jointly in an adversarial manner, obviating the need for explicit density definition. The generator's functionality is enabled by inputting random latent variables, $\boldsymbol{z}$, drawn from simple distributions such as a standard Gaussian, and mapping these random inputs to the data space. Concurrently, the discriminator is provided with both synthetic and real samples and its training objective is to differentiate between the two. Throughout the training process, the generator and the discriminator compete and evolve, enabling the generator to create realistic samples that can deceive the discriminator, while the discriminator enhances its ability to distinguish between real and fake samples. The original GAN training objective can be interpreted as optimizing the generator to produce synthetic data that minimizes the Jensen-Shannon (JS) divergence between the synthetic and real data distributions. This idea has been expanded in various GAN training objective extensions explored in the literature. For instance, variants have been proposed based on generalizations to any f-divergence [151], Wasserstein distance [8, 67], maximum mean discrepancy (MMD) [18, 111], and Sinkhorn distance [61].

Of particular interest to DP training is the observation that many of these divergence metrics can be approximated without requiring the training of a discriminator network. This has led to recent research in private generative models, which use a static function as the critic instead of a discriminator network. While such approaches might fall short in standard (non-private) generative modeling due to a lower expressive power compared to using learnable critic (that is adaptable to large data with diverse properties), they are highly competitive in DP training, as a static critic can effectively speed up convergence, thereby improving privacy guarantees.

In the case of implicit density models, the generator's interaction with the private dataset is typically indirect (only via the backward pass), meaning that there exists no direct link between the data source, as illustrated in the accompanying diagrams (Figure 4.3). This configuration presents an opportunity to strategically position the privacy barrier anywhere along the backpropagation path where the generator retrieves signals from the real data, facilitating an improved signal-to-noise ratio or simplified implementation. A more comprehensive understanding is presented in Section 4.4.1-4.4.3.

4.3.2.2 *Explicit Density Models*

Several prominent explicit density models have been developed in literature, each with distinct characteristics:

- The *Variational Autoencoder (VAE)* [100] is trained to maximize the Evidence Lower Bound (ELBO), a lower bound of the log-likelihood, which typically simplifies to ℓ_1/ℓ_2 losses on the data sample and its reconstruction under standard Laplacian/Gaussian noise modeling assumptions. The model comprises trainable encoder and decoder modules. Encoding is conducted through the encoder $q_{\boldsymbol{\phi}}$, which maps observed data to its corresponding latent variables, denoted as $\boldsymbol{x} \xrightarrow{q_{\boldsymbol{\phi}}} \boldsymbol{z}$. The dimensions of these latent variables are typically smaller than the data dimension d, embodying the concept of an information bottleneck [208, 189]. The decoder module is responsible for data reconstruction or generation, i.e., $\boldsymbol{z} \xrightarrow{p_{\boldsymbol{\theta}}} \boldsymbol{x}$. Additionally, VAE imposes regularization on the latent distributions to match the pre-defined prior, thereby enabling the generation of valid novel samples during inference.
- *Diffusion models* [191, 196, 80] operate similarly to VAEs in terms of maximizing the ELBO. However, instead of using a trainable encoder to map data to latent variables, diffusion models transform the data iteratively through a linear Gaussian operation, represented as $\boldsymbol{x} \xrightarrow{q} \ldots \xrightarrow{q} \boldsymbol{x}_{t-1} \xrightarrow{q} \boldsymbol{x}_t \xrightarrow{q} \boldsymbol{x}_T$. This procedure causes the latent variables at the final step $\boldsymbol{x}_T$ to form a standard Gaussian distribution and maintain the same dimensionality as the data. The generation process is executed by reversing the diffusion operation, which means iteratively applying $p_{\boldsymbol{\theta}}(\boldsymbol{x}_{t-1}|\boldsymbol{x}_t)$ for all time steps $t \in [T]$. The trainable component of diffusion models resides in the reverse diffusion process, while the forward process is pre-defined and does not require training.
- *Flow-based* models [175, 102], in contrast, minimize the Negative Log-Likelihood (NLL) directly. Uniquely, flow-based models employ the same invertible model for both encoding ($\boldsymbol{x} \xrightarrow{f_{\boldsymbol{\theta}}} \boldsymbol{z}$) and generation ($\boldsymbol{z} \xrightarrow{f_{\boldsymbol{\theta}}^{-1}} \boldsymbol{x}$), by executing either the flow or its inverse. Due to the invertibility demanded by the model construction, the dimensions of the latent variables $\boldsymbol{z}$ are identical to those of the data.
- *Autoregressive models* [107, 155, 221, 220], as another instance of model with tractable density, are also designed to minimize the NLL. Unlike some other models, they accomplish this without the need for explicit latent variables or an encoding mechanism. Instead, these models utilize partially observed data, denoted as $\boldsymbol{x}_{1:i-1}$, where each sample is regarded as a high-dimensional vector with observations up to the $(i-1)^{\text{th}}$ element. The model is

then trained to predict potential values for the subsequent element, x_i. Data generation is conducted through an iterative autoregressive process, where elements of each data vector is predicted one-by-one, starting from initial seeds. This can be represented as $x_0 \overset{p_\theta}{\to} ... \overset{p_\theta}{\to} x_{1:i-1} \overset{p_\theta}{\to} x_{1:i} \overset{p_\theta}{\to} ... \overset{p_\theta}{\to} x_{1:d}$. The component subject to training is the autoregressive model itself. Its parameters, denoted by $\boldsymbol{\theta}$, are optimized to best predict the next elements in the sequence based on previously observed values.

As illustrated in Figure 4.3, all these models require real data or derived quantities (such as latent variables) as inputs to the generator during the training phase. This necessitates a significant difference in the DP training of these models compared to implicit density models, which only need indirect data access through the backward pass. In the context of typical explicit density models, DP constraints must be accounted for, given the access to real data in both the forward and backward pass. This typically results in privacy barriers being directly integrated into the update process of the generator module, as further discussed in Section 4.4.4.

4.3.2.3 *Extensions*

Our diagram has been consciously designed to encompass future developments, including potential hybrid variants of generative models. It facilitates systematic analysis of the modifications required to transition the original training pipeline to a privacy preserving one. Specifically, to train a DP variant of such a model, one could follow the following steps: (1) Illustrate the model components and information flows using diagrams analogous to those shown in Figure 4.3. (2) Determine the component(s) that will be provided with DP guarantees, taking into account practical use requirements and a feasible privacy-utility trade-off. (3) Establish the privacy barrier to ensure the privacy of the targeted component, which will later be made accessible for potential threat exposure. This step should consider all access paths between the target component and the data source. (4) Calculate and bound the sensitivity. (5) Implement the DP mechanism and calculate the accumulated privacy cost of the entire training process.

4.4 Taxonomy

Accompanied by a comprehensive diagram encapsulating the complete spectrum of potential design choices for deep generative models, we put forth a classification system for current DP generative methods. This system is predicated on the positioning of the privacy barrier within the diagram (Figure 4.1). Specifically, for explanatory purposes, we consider the key components within our diagram (the **Generator**, **Synthetic data**, **Measurement**, and **Real data**), resulting in following options for positioning the privacy barrier:

- **B1**: Between **Real data** and **Measurement**
- **B2**: Within **Measurement**
- **B3**: Between **Measurement** and **Synthetic data**
- **B4**: Within **Generator**

B1 through **B4** are introduced sequentially, demonstrating the systematic transition of the privacy barrier from the real data source towards the generator end. The data-processing theorem 4.2.1 ensures that the DP guarantee is upheld as long as the data is "sanitized" through a DP mechanism prior to exposure to potential adversaries. In this context, if a DP training algorithm safeguards against threats introduced by **B1**, then it also provides the same protective guarantee against attackers defined by **B2** through **B4**.

Approach	Privacy barrier	Privacy notion	Sensitivity type	Generative framework	DP framework	Code
DP-Merf [70]	**B1**	Replace-one	Global	Distribution matching	Gaussian	[1]
DP-SWD [171]	**B1**	Replace-one	Smooth	Distribution matching	Gaussian	[2]
PEARL [120]	**B1**	Replace-one	Global	Distribution matching	Gaussian	[3]
DP-HP [224]	**B1**	Replace-one	Global	Distribution matching	Gaussian	[4]
DP-GEN [36]	$\sim$**B1**	–	–	Energy-based model	–	[5]
[71]	**B1**	Replace-one	Global	Distribution matching	Gaussian	[6]
DP-NTK [243]	**B1**	Replace-one	Global	Distribution matching	Gaussian	[7]
DPSDA [122]	**B1**	Add-or-remove-one	Global	Diffusion	Gaussian	[8]
SPRINT-gan [14]	**B2**	Add-or-remove-one	Global	GAN	DP-SGD	[9]
dp-GAN [257]	**B2**	Add-or-remove-one	Global	GAN	DP-SGD	[10]
DPGAN [239]	**B2**	Add-or-remove-one	Global	GAN	DP-SGD	[11]
[215]	**B2**	Add-or-remove-one	–	GAN	empirical DP	–
PATE-GAN [245]	**B2**	Both	Local	GAN	PATE	[12]
[6]	**B2**	Add-or-remove-one	Global	GAN	DP-SGD	[13]
[242]	**B2**	Add-or-remove-one	Global	GAN	DP-SGD	–
DP-CGAN [211]	**B2**	Add-or-remove-one	Global	GAN	DP-SGD	[14]
[57]	**B2**	Add-or-remove-one	Global	GAN	DP-SGD	[15]
DPMI [35]	**B2**	Add-or-remove-one	Global	GAN	DP-SGD	–
DPautoGAN [206]	**B2**	Add-or-remove-one	Global	GAN	DP-SGD	[16]
Private-Set [29]	**B2**	Add-or-remove-one	Global	Distribution matching	DP-SGD	[17]
[17]	**B2**	Add-or-remove-one	Global	GAN	DP-SGD	–
GS-WGAN [32]	**B3**	Both	Global	GAN	DP-SGD	[18]
G-PATE [129]	**B3**	Both	Local	GAN	PATE	[19]
DataLens [227]	**B3**	Both	Global/Local	GAN	PATE	[20]
DP-Sinkhorn [24]	**B3**	Both	Global	Distribution matching	DP-SGD	[21]
DP-GM [3]	**B4**	Add-or-remove-one	Global	VAE	DP-SGD	–
DP-VaeGM [37]	**B4**	Add-or-remove-one	Global	VAE , AE$^+$	DP-SGD	–
DP-SYN [2]	**B4**	Add-or-remove-one	Global	AE$^+$	DP-SGD	–
P3GM [204]	**B4**	Add-or-remove-one	Global	VAE	DP-SGD	[22]
DP-NF [226]	**B4**	Add-or-remove-one	Global	Flow	DP-SGD	[23]
DP^2-VAE [89]	**B4**	Add-or-remove-one	Global	VAE	DP-SGD	–
DPDM [47]	**B4**	Add-or-remove-one	Global	Diffusion	DP-SGD	[24]
[62]	**B4**	Add-or-remove-one	Global	Diffusion	DP-SGD	–
DP-LDM [133]	**B4**	Add-or-remove-one	Global	Diffusion	DP-SGD	[25]
DP-LFlow [88]	**B4**	Add-or-remove-one	Global	Flow	DP-SGD	[26]

Table 4.1: Table summary of existing works. The shaded area corresponds to approaches that require public data features.

[1] https://github.com/ParkLabML/DP-MERF

[2] https://github.com/arakotom/dp_swd

[3] https://github.com/spliew/pearl

[4] https://github.com/parklabml/dp-hp

[5] https://github.com/chiamuyu/DPGEN

[6] https://github.com/ParkLabML/DP-MERF

[7] https://github.com/FreddieNeverLeft/DP-NTK

[8] https://github.com/microsoft/DPSDA

[9] https://github.com/greenelab/SPRINT_gan

[10] https://github.com/alps-lab/dpgan

[11] https://github.com/illidanlab/dpgan

[12] https://github.com/vanderschaarlab/mlforhealthlabpub/tree/main/alg/pategan

[13] https://github.com/nesl/nist_differential_privacy_synthetic_data_challenge/

[14] https://github.com/reihaneh-torkzadehmahani/DP-CGAN

[15] https://github.com/SAP-samples/security-research-differentially-private-generative-models

[16] https://github.com/DPautoGAN/DPautoGAN

[17] https://github.com/DingfanChen/Private-Set

[18] https://github.com/DingfanChen/GS-WGAN

[19] https://github.com/AI-secure/G-PATE

[20] https://github.com/AI-secure/DataLens

[21] https://github.com/nv-tlabs/DP-Sinkhorn_code

[22] https://github.com/tkgsn/P3GM

[23] https://github.com/ChrisWaites/jax-flows/tree/master/research/dp-flows

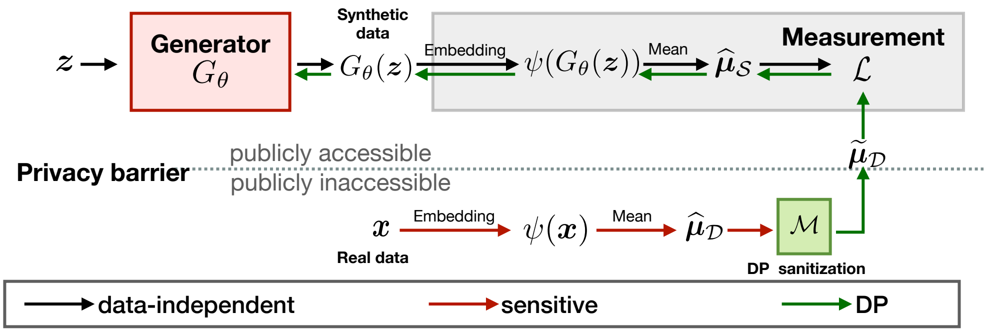


Figure 4.4: Diagram illustrating the general training procedure of methods under **B1**.

The generator end typically represents the smallest unit necessary for preserving the full functionality of the model, implying that the privacy barrier cannot be shifted further without compromising the operational capabilities of the generative model. Moreover, we reserve a more detailed discussion on the threat model (privacy barrier) integrated within the adopted DP mechanism (not specifically relevant to generative models) for later sections, where individual approaches will be introduced. Table 4.1 presents a summary of existing works classified under our taxonomy.

4.4.1 B1: Between Real Data and Measurement

Threat Model. Establishing a privacy barrier between the **Real data** and the **Measurement** entails using a DP mechanism to directly sanitize the data (features), thereby obtaining statistics that characterize the real data distribution for subsequent operations like computing the loss $\mathcal{L}$ as a **Measurement** that serves as the training objective for the generator. This approach provides protection against attackers who might gain access to the sanitized data features or any resultant statistics derived from the sanitized features, such as the loss measured on the sanitized data, any gradient vectors for updating the generator, and the generator's model parameters.

General Formulation. Methods within this category typically adopt the distribution matching framework (illustrated in Figure 4.3(b)), which aims to minimize the statistical distance between real and synthetic data distributions [70, 171, 224]. This distance is assessed with a static, unlearnable function, typically applying a data-independent feature extraction function ψ to project the data samples into a lower-dimensional embedding space and subsequently calculating the (Euclidean) distance between the resulting embeddings of real and synthetic data. The generator is optimized to reduce the disparity between the mean embeddings of synthetic and real data, which can be interpreted as minimizing the maximum mean discrepancy (MMD) between the real and synthetic data distributions [18, 111].

During DP training of these models, data points x_i or feature vectors $\psi(x_i)$ are first clipped or normalized (by norm) to ensure bounded sensitivity. Subsequently, random noise is injected into the mean features derived from the real samples, e.g., via Gaussian mechanism

[24] https://github.com/nv-tlabs/DPDM
[25] https://github.com/SaiyueLyu/DP-LDM
[26] https://github.com/dihjiang/DP-LFlow

(Definition 4.2.4). The objectives can be formulated as follows:

$$\text{Non-private: } \min_{\boldsymbol{\theta}} \left\| \frac{1}{|\mathcal{D}|} \sum_{i=1}^{|\mathcal{D}|} \psi(\boldsymbol{x}_i) - \frac{1}{|\mathcal{S}|} \sum_{i=1}^{|\mathcal{S}|} \psi(G_{\boldsymbol{\theta}}(\boldsymbol{z}_i)) \right\|_2^2 = \min_{\boldsymbol{\theta}} \left\| \widehat{\boldsymbol{\mu}}_{\mathcal{D}} - \widehat{\boldsymbol{\mu}}_{\mathcal{S}} \right\|_2^2 \tag{4.3}$$

$$\text{DP: } \min_{\boldsymbol{\theta}} \left\| \widetilde{\boldsymbol{\mu}}_{\mathcal{D}} - \widehat{\boldsymbol{\mu}}_{\mathcal{S}} \right\|_2^2 \quad \text{with} \quad \widetilde{\boldsymbol{\mu}}_{\mathcal{D}} = \widehat{\boldsymbol{\mu}}_{\mathcal{D}} + \mathcal{N}(0, \Delta^2_{\widehat{\boldsymbol{\mu}}_{\mathcal{D}}} \sigma^2 \boldsymbol{I}) \tag{4.4}$$

with $\widehat{\boldsymbol{\mu}}_{\mathcal{D}} = \frac{1}{|\mathcal{D}|} \sum_{i=1}^{|\mathcal{D}|} \psi(\boldsymbol{x}_i)$ and $\widehat{\boldsymbol{\mu}}_{\mathcal{S}} = \frac{1}{|\mathcal{S}|} \sum_{i=1}^{|\mathcal{S}|} \psi(G_{\boldsymbol{\theta}}(\boldsymbol{z}_i))$ representing the mean features of the real and synthetic data, respectively. Meanwhile, $\widetilde{\boldsymbol{\mu}}_{\mathcal{D}}$ denotes the DP-sanitized mean real data embedding with $\Delta^2_{\widehat{\boldsymbol{\mu}}_{\mathcal{D}}}$ being the sensitivity value that characterizes the influence of each real data point on the mean embedding. A visual illustration can be found in Figure 4.4.

Representative Methods. While all methods in this category adhere to the same general formulation, they primarily diverge in their construction of the feature extraction function ψ and the objective function that forms the training loss $\mathcal{L}$ for the generator. **DP-Merf** [70] employs the MMD minimization approach, optimizing a generator to minimize the difference between synthetic and real data embeddings, using random Fourier features [169] for the embedding function ψ. **DP-SWD** [171] instead employs random projections $\boldsymbol{u} \in \mathbb{S}^{d-1}$ for feature extraction. Specifically, DP-SWD uniformly samples k random directions for data projection, thereby enabling tractable computation of one-dimensional Wasserstein distances along each projection direction. The Sliced Wasserstein Distance (SWD) [167, 20], which is determined as the mean of one-dimensional Wasserstein distances over DP-sanitized projections, serves as the training objective for the generator. Similar to DP-Merf, **PEARL** [120] employs the Fourier transform as the feature extraction function while offering an alternative interpretation of describing the data distribution using the characteristic function with the characteristic function distance as the objective. Furthermore, PEARL proposes learning a re-weighting function for the embedding features, placing greater emphasis on the discriminative features, in order to enhance the expressiveness of the plain Fourier features employed in the DP-Merf approach.

Recent research efforts have primarily focused on identifying informative features that can efficiently capture the underlying characteristics of the data distribution. Specifically, **DP-HP** [224] employs Hermite polynomials as the feature embedding function. This choice of embedding function reduces the required feature dimension, which consequently decreases the effective sensitivity of the data mean embedding and leads to an improved signal-to-noise ratio in the DP training. **Harder et al. [71]** further propose utilizing feature extraction layers from pre-trained classification networks that capture general concepts learned on large-scale public datasets. Additionally, **DP-NTK** [243] introduces the use of the Neural Tangent Kernel (NTK) to represent data, resulting in the gradient of the neural network function serving as the feature map, i.e., $\psi(\boldsymbol{x}) = \nabla_{\boldsymbol{\theta}} f(\boldsymbol{x}; \boldsymbol{\theta})$.

Privacy Analysis. The privacy analysis for methods in this category involves computing the sensitivity and applying the privacy analysis of associated noise mechanisms, such as the Gaussian mechanism (Definition 4.2.4). The sensitivity represents the maximum effect of an individual data point on the mean embedding:

$$\Delta^2 = \max_{\mathcal{D},\mathcal{D}'} \|\widehat{\boldsymbol{\mu}}_{\mathcal{D}} - \widehat{\boldsymbol{\mu}}_{\mathcal{D}'}\|_2 = \left\| \frac{1}{|\mathcal{D}|} \sum_{i=1}^{|\mathcal{D}|} \psi(\boldsymbol{x}_i) - \frac{1}{|\mathcal{D}'|} \sum_{i=1}^{|\mathcal{D}'|} \psi(\boldsymbol{x}'_i) \right\|_2 \tag{4.5}$$

In existing literature, the replace-one privacy notion is commonly used to compute the sensitivity value Δ^2, resulting in an upper bound of $\frac{2}{|\mathcal{D}|}$ when the feature vector by construction has a norm equal to 1 or is normalized with a maximum norm of 1, i.e., $\|\psi(\boldsymbol{x})\|_2 \leq 1$. Deriving the sensitivity value for the add-or-remove-one notion is slightly more technically involved, but applying existing techniques used for the replace-one notion leads to a conservative bound of $\frac{2}{|\mathcal{D}|+1}$ (See Appendix). This implies two things: first, the sensitivity value decreases inversely proportional to the size of the dataset, showing the beneficial effect of the "mean" operation over large datasets, which smooths out individual effects through population aggregation. Second, there is a minor difference in the computed sensitivity between the two privacy notions: $\frac{2}{|\mathcal{D}|+1}$ versus $\frac{2}{|\mathcal{D}|}$. This means that the current comparison results hold with negligible effect when the dataset size is sufficiently large. While achieving a tighter bound for the sensitivity value is possible with the add-or-remove-one privacy notion, it may require additional assumptions.

In contrast to other studies that compute the (worst-case) global sensitivity (Definition 4.2.3), the sensitivity in DP-SWD represents a form of expected value, accompanied by a sufficiently small failure probability. This efficiently harnesses the characteristics of random projections to achieve a tight sensitivity bound, but requires careful comparison to other methods. When combining this sensitivity definition with mechanisms that offer (ε, δ)-DP (i.e., the relaxed DP notion), the final privacy guarantee will be weaker than (ε, δ), due to the additional failure probability derived from the sensitivity itself.

Analysis, Insights, Implications. Methods under this category present several strengths. Firstly, the "mean" operation adopted during the extraction of descriptive feature embeddings significantly reduces the impact of each individual. This leads to a lower sensitivity value that scales in inverse proportion to the number of data points being aggregated through the "mean" operation. As a result, a strong privacy guarantee can be ensured with less randomness required from the DP mechanism. Moreover, they are straightforward to implement, typically necessitating just one instance of sanitization on the computed mean feature (known as "one-shot sanitization") throughout the training process, which further saves the privacy consumption in comparison to iterative methods. These methods also converge quickly and can yield acceptable results even under a low privacy budget, given the ease of fitting the static target, i.e., the noisy mean.

Nevertheless, they come with certain drawbacks. The static feature might not be sufficiently discriminative or informative, lacking the expressiveness found in methods that employ trainable models as critics. Furthermore, the "mean" operation could potentially induce unintended mode collapse in the generated distributions, trading off generation diversity for privacy protection. This situation warrants attention in future works, particularly in optimizing the trade-off between the expressiveness of the feature extraction method in the critic and the privacy cost of achieving such expressiveness. A promising direction could be to exploit knowledge from public non-sensitive data and/or pre-trained models that better describe data without compromising the privacy of the sensitive data.

4.4.2 B2: Within Measurement

Threat Model. The previous category focuses on a static, sanitized statistical summary, derived from a data-independent function, as a replacement for real data when training generative models. However, learnable functions that are able to adapt to diverse data distribution may offer superior expressive power. In this regard, a logical strategy is to incorporate DP into the measurement process, particularly by training a DP critic. This privacy barrier sits "within **Measurement**" and safeguards against adversaries with access to the critic and subsequent quantities, including information flows to the generator. If gradient sanitization techniques like DP-SGD are employed for updating the critic, the DP mechanism further protects against attacks targeting all intermediate gradients w.r.t. the critic's parameters during the training phase.

General Formulation. Methods in this category follow two main principles: Firstly, they use a learnable critic (feature extraction function) that dynamically adapts to the private dataset, necessitating a boundary on the potential privacy leakage of such critic. Secondly, the generator is prohibited from accessing private real data directly, its access limited to indirect interaction through the backward pass. This ensures the generator's update signals are fully derived from the learnable critic. As such, developing a DP critic is sufficient to assure DP for the generator module (and the entire model) for privacy-preserving generation. GAN models (depicted in Figure 4.3(a)) meet these criteria and serve as a foundational framework that most existing DP methods in this category generally conform to.

Representative Methods. The implementation of the privacy barrier within the **Measurement** block is exemplified in **DP-GAN** [257, 239] and concurrent studies [14, 215, 6, 242, 211, 57]. In this context, the discriminator, acting as the learnable critic model, is trained via DP-SGD (Section 4.2.1.1). The privacy of the generator is ensured by the post-processing theorem. As per the public timestamp of paper releases, this approach can be traced back to [14], who proposed training an ACGAN (Auxiliary Classifier GAN) [152] in a DP manner to conditionally generate samples for downstream analysis tasks on medical data. The training pipeline can be formalized as follows, with the illustration shown in Figure 4.5:

$$g_D^{(t)} = \nabla_{\boldsymbol{\phi}} \mathcal{L}(G_{\boldsymbol{\theta}}, D_{\boldsymbol{\phi}}) \qquad \text{(Discriminator gradient)} \tag{4.6}$$

$$g_G^{(t)} = \nabla_{\boldsymbol{\theta}} \mathcal{L}(G_{\boldsymbol{\theta}}, D_{\boldsymbol{\phi}}) \qquad \text{(Generator gradient)} \tag{4.7}$$

$$\tilde{g}_D^{(t)} = \mathcal{M}_{\sigma,C}(g_D^{(t)}) = \text{clip}(g_D^{(t)}, C) + \mathcal{N}(0, \sigma^2 C^2 \boldsymbol{I}) \qquad \text{(Apply DP sanitization)} \tag{4.8}$$

$$\boldsymbol{\phi}^{(t+1)} = \boldsymbol{\phi}^{(t)} - \eta_D \cdot \tilde{g}_D^{(t)} \qquad \text{(Discriminator update)} \tag{4.9}$$

$$\boldsymbol{\theta}^{(t+1)} = \boldsymbol{\theta}^{(t)} - \eta_G \cdot g_G^{(t)} \qquad \text{(Generator update)} \tag{4.10}$$

The generator $G_{\boldsymbol{\theta}}$ and discriminator $D_{\boldsymbol{\phi}}$ are parameterized by $\boldsymbol{\theta}$ and $\boldsymbol{\phi}$, respectively, with η_G and η_D denoting their learning rates. $\mathcal{M}_{\sigma,C}$ refers to the Gaussian mechanism in DP-SGD, with σ representing the noise scale and C indicating the gradient clipping bound. Although we have omitted the sample index in the above equations for the sake of brevity, it should be noted that the clipping function in Equation 4.8 is expected to take per-example gradients as inputs, adhering to the standard procedure of DP-SGD (Section 4.2.1.1). Specifically, it suffices to apply the sanitization only to the gradients that depend on the real data samples, including indirect usage of real samples, such as through gradient penalty terms [67].

Unlike DP-GAN that employs DP-SGD for training the DP discriminator, **PATE-GAN** [245] leverages the PATE framework (Section 4.2.1.2) to train its DP (student) discriminator. PATE-GAN comprises three main components that are jointly trained throughout the process: multiple (non-private) teacher discriminators, a DP student discriminator, and a DP generator. Similar to the original PATE framework, PATE-GAN starts by partitioning the real dataset into disjoint subsets, which subsequently serve to train the teacher discriminators independently. In each training iteration, PATE-GAN follows a sequence of steps: (1) independently updating the teacher discriminators using mini-batch samples from real data partitions and synthetic samples drawn from the generator; (2) querying the teacher discriminators with a set of synthetic samples; (3) the teacher discriminators then engage in a voting process on the real/fake predictions for the synthetic samples they have received, and apply DP noise to the

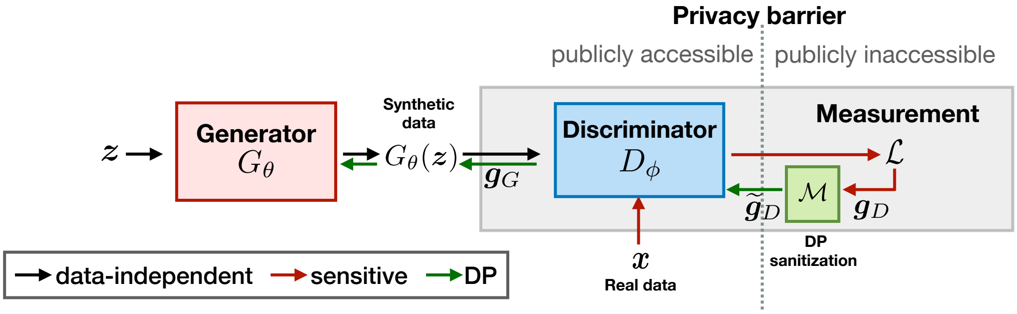


Figure 4.5: Diagram illustrating the training pipeline of **DP-GAN** with a (vertical) privacy barrier of type **B2** as shown.

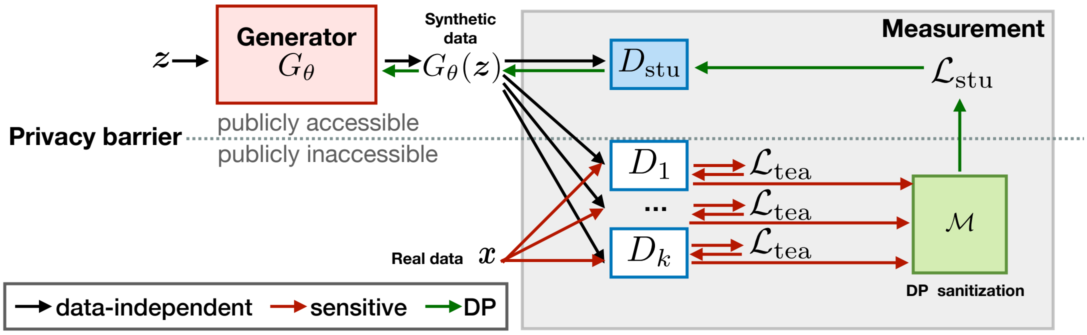


Figure 4.6: Diagram illustrating the training pipeline of **PATE-GAN** with a (horizontal) privacy barrier of type **B2** as shown. $\mathcal{L}_{\text{stu}}$ and $\mathcal{L}_{\text{tea}}$ denote the student and teacher training losses respectively, while D_{stu} is the student discriminator and $D_1, ..., D_k$ represent the teacher discriminators.

results of the vote; (4) training the student discriminator with the query synthetic samples as input and the DP aggregation of teacher predictions as the label; (5) finally, jointly updating the generator and the student discriminator, with the generator querying the student discriminator with new synthetic samples and obtaining update gradient signals from the DP student discriminator. A visual illustration is presented in Figure 4.6.

While the discriminator in the GAN framework aims to distinguish between two distributions, recent research uncovered intriguing results when the learnable critic is designed to target specific downstream tasks, such as classification. Specifically, **Private-Set** [29] employs a classification network as a learnable feature extractor, which is trained with DP-SGD. This learnable feature extractor, combined with the alignment in the gradients serving as the critic, encourages the synthetic data to emulate the training trajectories of the real data during the training process within a classification network, making the synthetic data useful for training downstream classifiers and safe for public release due to the DP guarantees embedded within the measurement process.

Privacy Analysis. Methods in this category inherit the privacy notion and sensitivity computation from their respective framework for training the DP critic (See Section 4.2.1.1-Section 4.2.1.2), while also inheriting the need for careful consideration regarding the application of data-dependent privacy analysis or adherence to privacy notion constraints to ensure comparable results. For methods grounded by DP-SGD, this results in a noticeable disparity between the replace-one and add-or-remove-one DP notions, as illustrated by the doubled sensitivity value when transitioning from the default add-or-remove-one to the replace-one notion, i.e., C versus $2C$ with C denoting the gradient clipping bound. Consequently, a doubled noise scale is required to achieve an ostensibly identical privacy guarantee, inevitably resulting in utility degradation and unfavorable comparison outcomes under the replace-one notion.

Analysis, Insights, Implications. While this training paradigm enjoys several advantages, such as ease of implementation and representative features for characterizing the difference between distributions, several challenges persist when applying such a paradigm in practice. Firstly, the joint training of a generator alongside a critic, which typically necessitates an adversarial approach, is inherently unstable due to the difficulty in maintaining equilibrium between these two components. This instability can be further amplified by the incorporation of gradient clipping and noise addition operations introduced by DP-SGD, or the additional fitting process involved in transferring knowledge from the teacher discriminators to the student one through the PATE framework. Moreover, the DP training of the critic often impedes its convergence, resulting in a sub-optimal critic that may not effectively guide the generator.

Recent studies have investigated various strategies to alleviate these challenges, particularly in the context of GANs. These include warm-starting the GAN discriminator by pre-training on public data [257], dynamically adjusting the gradient clipping bounds during the training process [257], re-balancing the discriminator and generator updates to restore parity to a discriminator weakened by DP noise [17], and exploiting public pre-trained GANs while restricting private modeling to the latent space [35]. In the Private-Set [29] framework that optimizes for downstream classification task, it is reported that optimizing the generator in a point-wise manner (as discussed in Section 4.3) or directly optimizing the synthetic set instead of the generator model can empirically lead to faster convergence and preferable when strong privacy guarantee is required. In this regard, we anticipate promising outcomes from the future development of new variants of DP-compliant training pipelines and objectives that offer improved convergence and, consequently, enhanced privacy guarantees.

4.4.3 B3: Between Measurement and Synthetic Data

Threat Model. In response to challenges associated with training the DP critic (Section 4.4.2), recent studies have proposed shifting the privacy focus from the **Measurement** to the sanitization of the intermediate signal that backpropagates to update the generator, i.e., between **Measurement** and **Synthetic data**. The goal is to preserve the critic's training stability and its utility for accurately comparing synthetic and real data, thereby guiding the generator's training effectively. This strategy ensures privacy when revealing sanitized intermediate gradients exchanged between the generator and the critic during the backward pass, as well as guarantees DP for the generator, which is updated with sanitized gradients. However, this scheme does not provide privacy guarantees for the release of the critics, since their training is conducted non-privately.

General Formulation. Similar to the case outlined in Section 4.4.2, the backbone generative models for this category are typically implicit density models. This restriction is in place as these models do not invoke direct interaction between the real data and the generator during the forward pass, which means that sanitizing the intermediate signals transmitted between the **Measurement** and **Synthetic data** is sufficient for ensuring privacy protection. Methods in this category adhere to the gradient sanitization scheme, which introduces a DP perturbation into the gradients communicated between the critic and generator during the backward pass. This

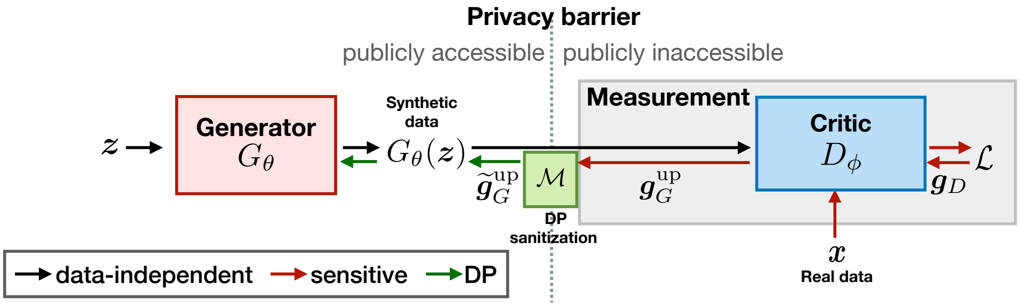


Figure 4.7: The diagram illustrates the general training process of methods incorporating the privacy barrier **B3**. In the figure, g_G^{up} and $\widetilde{g}_G^{\text{up}}$ denote the upstream gradient (referenced as g_G^{upstream} in Equation 4.12) and its sanitized variant, respectively. Note that variations exist in the formulation of critics and their corresponding training paradigms.

can be formulated as follows:

$$g_G^{(t)} = \nabla_\theta \mathcal{L}_G(\theta^{(t)}) = \nabla_{G_\theta(z)} \mathcal{L}_G(\theta^{(t)}) \cdot \boldsymbol{J}_\theta G_\theta(z) \tag{4.11}$$

$$\widetilde{g}_G^{(t)} = \mathcal{M}\big(\underbrace{\nabla_{G_\theta(z)} \mathcal{L}_G(\theta^{(t)})}_{g_G^{\text{upstream}}}\big) \cdot \underbrace{\boldsymbol{J}_\theta G_\theta(z)}_{\boldsymbol{J}_G^{\text{local}}} \tag{4.12}$$

Here, $\mathcal{L}_G$ represents the generator's loss (originating from a critic), and $\mathcal{M}$ denotes a potential DP sanitization mechanism on g_G^{upstream}—the gradient information backpropagating from the critic to the generator. This can be considered as the gradient of the objective with respect to the current synthetic samples. It is important to note that the second term ($\boldsymbol{J}_G^{\text{local}}$), i.e., the local generator Jacobian, is computed independently of training data and thus does not require sanitization. The generator is subsequently updated with the DP sanitized gradient, i.e., $\theta^{(t+1)} = \theta^{(t)} - \eta_G \cdot \widetilde{g}_G^{(t)}$. Meanwhile, the critic, if learnable, is updated normally (non-privately). A visual illustration is presented in Figure 4.7.

Representative Methods. Existing methods explored various choices for the critic and different DP mechanisms to sanitize the upstream gradients g_G^{upstream}. **GS-WGAN** [32] adopts the Gaussian mechanism for sanitization and capitalizes on the inherent bounding of the gradient norm. This follows from the Lipschitz property when employing the Wasserstein distance with gradient penalty [8, 67] as the objective when training a GAN. In contrast, **G-PATE** [129] incorporates the PATE framework as its sanitization mechanism. This approach discretizes the gradients and allows multiple teacher discriminator models to vote on these discretized gradient values. The DP noisy argmax is then transferred to the generator. **DataLens** [227] further improves the signal-to-noise ratio in the PATE sanitization by employing top-K dimension compression.

In a different vein, **DP-Sinkhorn** [24] presents compelling results using a nonparametric critic. Specifically, DP-Sinkhorn estimates the Sinkhorn divergence grounded on L_1 and L_2 losses in the data space, adhering to the distribution matching generative framework as depicted in Figure 4.3(b). This use of a data-independent critic contributes stability to the training process and capitalizes on the privacy enhancement brought by subsampling.

Privacy Analysis. The privacy analysis for this method category largely aligns with the established unit sanitization mechanisms, denoted as $\mathcal{M}$, which function on upstream gradients g_G^{upstream}. Nevertheless, specific attention is necessary given that these intermediate gradients do not directly originate from real data samples. This scenario noticeably influences the sensitivity computation, defined formally by:

$$\Delta^2 = \max_{\mathcal{D},\mathcal{D}'} \| f(\mathbf{g}_G^{\text{upstream}}) - f(\mathbf{g}'^{\text{upstream}}_G) \|_2 \tag{4.13}$$

In this equation, f encapsulates the operations required to set bounds on the sensitivity and to render the associated sanitization mechanism applicable. $\mathbf{g}_G{}^{\text{upstream}}$ and $\mathbf{g}'_G{}^{\text{upstream}}$ symbolize the intermediate upstream gradients originating from neighboring datasets $\mathcal{D}$ and $\mathcal{D}'$ respectively. Specifically, f performs distinct roles according to the method employed: For GS-WGAN and DP-Sinkhorn, f signifies the operation of norm clipping; In G-PATE, f encompasses the processes of dimension reduction and gradient discretization, and the computation of teacher voting histograms based on these discretized gradients; In the context of DataLens, rather than employing random projection and discretization as in G-PATE, f adopts a top-k stochastic sign quantization of the gradients. Subsequent to this operation, the teacher voting histograms are also calculated.

A direct application of the triangle inequality reveals that Δ^2 equals $2C$ (with C representing the gradient clipping bound) in both GS-WGAN and DP-Sinkhorn for both the replace-one and add-or-remove-one notions, while C is further guaranteed to be 1 in GS-WGAN by the nature of the adopted Wasserstein objective. This is notably different from the substantial disparity between the two privacy notions in the standard DP-SGD framework. In G-PATE, the voting histogram diverges by a maximum of 2 entries for each gradient dimension, which are processed independently via DP aggregation. As for the DataLens approch, the change of one data point will at most reverse all the signs of the top-k elements of gradients originated from one teacher model, leading to $\Delta^2 = 2\sqrt{k}$ (See Appendix for details).

Typically, the total privacy cost is calculated based on the RDP accountant (Theorem 4.2.2). Notably, each synthetic sample in a mini-batch constitutes one execution of the sanitization mechanism for the DP-SGD framework, or one query in the PATE framework. In other words, performing an update step with a mini-batch of synthetic samples on the generator can be regarded as a composition of *batch size* times its unit sanitization mechanism.

Analysis, Insights, Implications. Compared to previous categories (Section 4.4.1-4.4.2), shifting the privacy barrier away from the **Measurement** process itself offers several benefits. These include: (1) the flexibility to employ a powerful critic, thereby effectively guiding the generator towards capturing the characteristics of the data distribution; (2) seamless support for different privacy notions (as discussed in privacy analysis above); (3) practically simpler to properly bound the sensitivity. This can be achieved by exploiting the intrinsic properties of the objective [32], or through the usage of the PATE framework [129, 227]. This is particularly beneficial when compared to the previous scenario of learnable critics that typically necessitate a laborious and fragile hyperparameter search for a reasonable gradient clipping bound.

However, the increased expressive capacity comes with the trade-off of relatively high privacy consumption. The accumulation of privacy cost across iterations is notably faster in this scenario than in standard DP-SGD training of a single model: each DP update on the generator in this category equates to a *batch size* number of calls to the Gaussian mechanism, possibly without the advantage of subsampling, as detailed in the preceding privacy analysis section. This markedly contrasts with the standard DP-SGD training on a single discriminator, as mentioned in the previous category (refer to Section 4.4.2), where each individual DP gradient update equates to a *single* execution of the (subsampled) Gaussian mechanism.

Fortunately, this drawback has been partially mitigated through the use of data-dependent privacy analysis (as demonstrated in PATE-based methods like G-PATE and DataLens) that provides analytically tighter results that lead to stronger DP guarantees, or a data-independent critic (as in DP-Sinkhorn) that offers smooth compatibility with subsampling and better convergence. Looking forward, we anticipate further developments from refining this training paradigm, particularly through the utilization of strong backbone discriminators (and generators) trained on external non-private data, thereby optimizing privacy consumption.

4.4.4 B4: Within Generator

Threat Model. DP can be directly integrated into the training or deployment of a generator, the minimal unit within the generative models pipeline essential for maintaining the full generation functionality for future use. Generally, the privacy barrier safeguards against attackers who have access to the trained generator model while a more fine-grained distinguishment between the type of access (e.g., white-box or black-box) may be required depending on the application scenarios and the adopted DP mechanism. If the gradient sanitization scheme is adopted, it can protect against adversaries who can access the white-box generator (and possibly other trainable components subject to DP sanitization) and the intermediate sanitized gradients during the whole training process.

General Formulation. In this context, the training pipeline can be generally simplified to the standard process of training DP classification models. This process, as exemplified by the commonly used DP-SGD framework, entails bounding sensitivity through gradient clipping and subsequently injecting randomness into the generator's gradients. In contrast to category **B3**, where the upstream gradient g_G^{upstream} undergoes sanitization, in this case, it is the final generator gradient $g_G^{(t)}$ (refer to Equation 4.11) that is being sanitized. This results in a difference equivalent to the multiplicator of the local generator Jacobian (refer to Equation 4.12). Special attention should be paid when implementing DP-SGD here, as additional model components (e.g., the encoder in a VAE) alongside the generator could compromise the transparency of the privacy analysis. It is crucial to ensure that the gradient clipping operation is executed accurately to effectively limit each individual real sample's influence on the generator. The presence of an additional model component may disperse individual effects across multiple gradients within a mini-batch, rendering standard per-example gradient clipping inadequate (refer to the discussion in the privacy analysis below). Moreover, to optimize model utility, it is necessary to precisely define the scope of gradient clipping and perturbation to ensure that the implementation does not introduce unnecessary noise exceeding the desired privacy guarantee.

Representative Methods. Existing works have realized such privacy barrier for various types of generative models, particularly those within the explicit density category. Examples include **DP Normalizing Flow** [226, 88], **DP VAE** [37, 3, 2, 204, 164], **DP Diffusion models** [47, 62, 133], and DP training of **language models** [141, 118, 137, 250], which collectively illustrate the extensive potential of DP generators across numerous applications such as density estimation, high-quality image generation, training downstream models, and model selection. In particular, Ghalebikesabi et al. [62] highlighted that certain training techniques advantageous for DP classification models [42], such as pre-training, utilization of large batch sizes, and augmentation multiplicity [56, 42], also show effectiveness when applied to training DP generators in diffusion models. Furthermore, the work by Jiang et al. [88] underscores the potential efficacy of training a DP Flow model within a compressed, lower-dimensional latent space. This strategy not only circumvents the substantial computational demands [178], but also synergizes well with DP protocols, given the direct correlation between the DP noise-to-signal-ratio and the model's

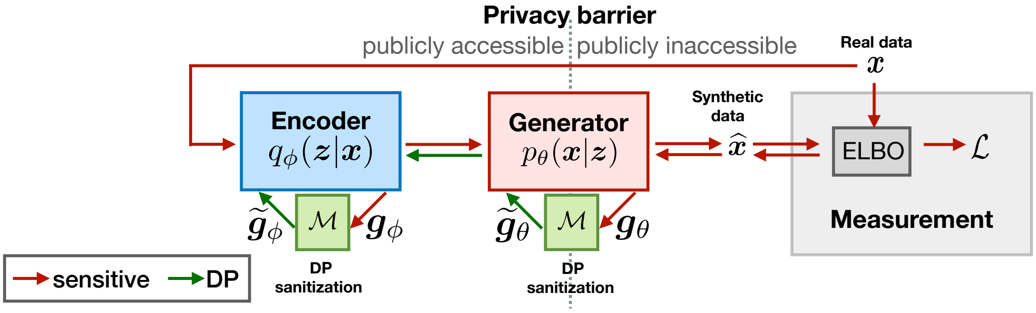


Figure 4.8: Diagram showcasing the DP training of a VAE (Section 4.3.2.2). This representation is also applicable to other DP model training scenarios conforming to privacy barrier **B4**, e.g., by replacing the trainable encoder with a non-trainable module.

dimensionality.

Privacy Analysis. The privacy analysis follows from the adopted DP mechanism for training the generators, similar to the standard case of training a DP classifier. A key consideration lies in the correct implementation and analysis of the privacy cost when the models comprise multiple trainable components, such as the encoder and decoder in the VAE. In such cases, simply incorporating the DP-SGD into the generator module and conducting a standard privacy accountant is inappropriate. This is due to the fact that each training example's influence is assimilated into the encoder's parameters. Consequently, every training example, even those absent from the current mini-batch, can affect all latent variables (which serve as inputs to the generator/decoder) in each iteration, rendering the per-example gradient clipping itself insufficient for bounding the sensitivity. A proper implementation would require either enforcing DP also on the encoder (i.e., applying DP-SGD on both the encoder and decoder) or factoring this into the privacy cost computation (i.e., the DP-SGD step on the decoder should be counted as full batch Gaussian mechanism instead of a subsampled one). Moreover, in situations where each sample in a mini-batch is used more than once, such as their use over multiple time steps when training diffusion models, the cost must be accounted for every such occurrence. To deal with this, one can refer to the *multiplicity* technique [56, 42, 62, 47], which averages all gradients resulting from each unique training sample before clipping them.

Analysis, Insights, Implications. Methods in this category are generally easy to implement, particularly for models with only a generator as the learnable component. This reduces training to the standard classification cases, demonstrating significant potential and achieving state-of-the-art generation quality when adapted to the latest generative modeling techniques [47, 62]. However, this privacy barrier setting may not be fully compatible with models containing multiple trainable components. The reason for this lies in the potential integration of training samples' effects into the parameters of components other than the generator (e.g., the encoder in VAEs, the discriminator in GANs), which substantially complicates the implementation of DP mechanisms and may lead to unexpectedly high privacy consumption. Moreover, DP methods are bounded by the expressive capability of the underlying generative model. Particularly in this category, which predominantly relies on explicit density models, the usage of simple critics (like static ℓ_1 or ℓ_2 loss functions) tends to restrict the capture of fine details, often delivering less desirable outcomes compared to trainable critics. For instance, VAEs have commonly produced blurrier images, whereas GANs pioneered the production of high-resolution photorealistic generations. While recent advancements in explicit density models have significantly improved their capabilities, particularly through innovative designs that enable training on extensive datasets, there is a potential limitation concerning their practical utility. This limitation primarily arises from the substantial need for sensitive training data, which is essential to achieve a satisfactory performance level with the resulting DP model in real-world applications. Looking forward, we envision future advancement on balancing the data efficiency and generation performance could largely improve the practicability of the DP methods under this category.

4.5 Discussion

4.5.1 Connection to Related Fields

While the data generation methods investigated in this work are mostly designed to capture the entire data distribution for general purposes, intriguing results are observed when the generator is intentionally guided towards enhancing its downstream utility for specific target tasks such as training neural network classifiers [29] and answering linear queries [126]. This can be achieved by employing objectives tailored for downstream tasks, rather than relying solely on general distribution divergence measures. If downstream tasks can be executed on a specific set of samples and do not require a complete understanding of the distribution, problem complexity can be further reduced by directly optimizing the synthetic samples instead of the generative models. This strategy, which trade-off the generality of general-purpose generative modeling for downstream utility, might be particularly beneficial considering the high complexity inherent to DP generation. Moreover, such framework naturally aligns with broader fields such as coreset generation, private query release, private Bayesian inference. In these scenarios, a set of synthetic data can be optimized to resemble real data for specific tasks [229, 11], substitute real data for answering queries to conserve the privacy budget under DP [73, 72], or support privacy-preserving computation of the posterior distribution [134, 185].

4.5.2 Relation to Other Summary Papers

Several related summary papers complement our work by focusing on different aspects. For instance, Tao et al. [207] benchmark multiple DP models for tabular data; Fan [55] and Cai et al. [22] discuss early DP GANs; Jordan et al. [90] and De Cristofaro [43] provide high-level overviews of DP synthetic data generation for non-expert audiences; Hu et al. [83] covers broad classes of DP data generation methods without focusing on the technical part of deep generative modeling; Lastly, Ponomareva et al. [165] offer a comprehensive summary of developing and deploying general DP ML models, supplementing our focus on the technical aspects of DP generative modeling.

4.5.3 Challenges and Future Directions

Public Knowledge. A promising future direction which holds significant practical relevance is the exploitation of public data/knowledge in training DP generative models. Recent studies have demonstrated promising improvements in DP generation introduced by leveraging public data [35, 126, 71, 133] and reported high-quality generation [62, 122] with the aid of such resources. A prevalent method for leveraging public knowledge involves utilizing large foundation models, initially pre-trained on public datasets, and subsequently fine-tuned to align with private data distributions for various applications. This approach is particularly relevant in the field of natural language processing (NLP), where the widespread availability of foundation models and the typically significant semantic overlap between public and private data renders DP fine-tuning relatively effective [118]. Additionally, the rapid growth of efficient fine-tuning techniques also show great potential for facilitating DP learning [247, 49]. While these advancements are particularly notable in the NLP domain, exploring the specific benefits and most effective strategies for applying these techniques to other data modalities is a topic that warrants further research. Furthermore, challenges that are generally associated with private learning on public data [213] call for further investigation. In particular, the unique difficulties specific to generative modeling, such as a small tolerance for distribution shift (between the public and private data distributions), warrant additional exploration.

Task-specific Generation. There exists a principled trade-off between the flexibility offered by general-purpose generative modeling and the utility of task-specific data generation. In particular, capturing a complete high-dimensional data distribution is a difficult task. This task becomes even harder when considering the privacy constraints, thus making the models highly data-demanding and almost impossible for DP model to achieve reasonable performance in practice. It has also been recently questioned to what extend a well-performing general-purpose DP generative model can be realized at all [200, 201]. While it is difficult to predict how these trade-off develop in the future, task-specific (or task-guided) data generation can greatly relax the objectives, leading to real-world useful DP synthetic data (see examples discussed in Section 4.5.1). On the other hand, such task-specific generation is particularly advantageous for scenarios where the synthetic data is intentionally designed to be useful only for specific (benign) tasks, thereby preventing potential unauthorized data misuse.

Conditional Generation. While the formulas presented throughout Section 4.4 are illustrated through unconditional generation for simplicity and clarity, in practice, DP generation is typically executed in a conditional manner, whereby samples are generated given specific input conditions. Although implementing conditional generation is technically straightforward for all generative network backbones [144, 192, 152, 236], it might necessitate additional consideration with respect to the privacy analysis. For instance, when modeling the class-conditional data feature distribution, an additional privacy budget may be allocated to learn the class label occurrence ratio for addressing class imbalance [70], contrasting with other methodologies that typically employ a data-independent uniform class-label distribution. Moreover, certain situations necessitate meticulous investigation into privacy implications and performance. Firstly, when the training process employs conditional (e.g., per-label class) sampling, additional consideration for privacy cost is imperative, as this contradicts the requirements of random sub-sampling incorporated in standard privacy cost computations. Secondly, some generative modules may integrate such conditional information in non-trivial ways (e.g., being embedded into the module parameters beyond mere gradients [95]). This integration can mean that the conditional input might no longer be protected under DP guarantees via a vanilla DP sanitization scheme. These scenarios necessitate further exploration to ensure the reliability of privacy protections and to facilitate the development of more effective utility-preserving DP generative models.

Federated Learning. DP data generation models have also shown promising potential in applications related to federated training [10, 240, 256, 216], facilitating tasks such as privacy-preserving data inspection and debugging that were previously infeasible due to privacy constraints. Specifically, Augenstein et al. [10] incorporated DP-SGD into the training of a GAN in a federated setting, where each client maintains a local GAN model and communicates the gradients to the server during each communication round, with the gradients being sanitized under DP noise. Moreover, Chen et al. [32] illustrated that the privacy barrier **B3** (Section 4.4.3) is seamlessly compatible with the federated training setting. In this context, only the upstream gradient (Equation 4.12) needs to be communicated, offering additional benefits such as improved communication efficiency. More recently, task-specific DP generation has proven particularly advantageous in alleviating non-iid challenges and enhancing convergence speed for federated learning [241, 228]. Although these approaches might still require a substantial amount of client local data and computational resources, the future development of efficient algorithms is anticipated to yield fruitful outcomes.

Evaluation and Auditing. Evaluating generative models has historically posed a significant challenge [132], and the same holds true for DP generation methods. While evaluating them based on specific downstream tasks has been a common approach in existing literature, it has become evident that relying solely on a single metric may be inadequate. This limitation arises from the general lack of alignment among various aspects, including downstream utility, statistical properties, and visual appearance [5, 201, 29, 59]. Consequently, there arises a need for future investigations into comprehensive metrics that consider mixed objectives to more effectively address a wide range of potential practical applications.

Furthermore, assessing the privacy guarantees of DP generators against real-world attacks (i.e., "auditing" [85, 147]), and quantifying the privacy risk associated with synthetic data [201, 81], presents a particularly intricate challenge for generative models. This complexity primarily arises from two key factors. Firstly, the measurement of privacy risks often conflicts with the primary objective of maximum likelihood, which aims to precisely fit the training data. While achieving an exact alignment with the training data aligns with training objectives, it raises a debatable question about compromising privacy protection. Deciding whether an exact match should be regarded as a privacy breach in such cases remains a matter of debate. Secondly, generative models typically exhibit low sensitivity to privacy attacks [74, 34], which diminishes the informativeness of computed auditing scores. These challenges highlight the need for dedicated design tailored to the auditing of DP generative models.

4.6 Conclusion

In summary, we introduce a unified view coupled with a novel taxonomy that effectively characterizes existing approaches in DP deep generative modeling. Our taxonomy, which encompasses critical aspects such as threat models, general formulation, detailed descriptions, privacy analysis, as well as insights and broader implications, provides a consolidated platform for systematically exploring potential innovative methodologies while leveraging the strengths of existing techniques. Furthermore, we present a comprehensive introduction to the core principles of DP and generative modeling, accompanied by substantial insights and discussions regarding essential considerations for future research in this area.

II

Part 2: Privacy attacks and defenses

While the previous part emphasized rigorous privacy guarantees in training generative models, we now shift our focus to practical aspects of privacy attacks and defenses. This transition complements our earlier discussions, effectively bridging the rigorous theoretical upper bounds of privacy risks with estimable lower bounds in real-world scenarios. In particular, our efforts are directed towards developing practical and effective privacy attacks against advanced generative models, as discussed in Chapter 5. These attacks not only highlight the necessity of devising dedicated privacy-preserving training techniques but also serve as a validation tool. Concurrently, we are exploring privacy defense strategies for general discriminative (classification) models. These strategies, detailed in Chapter 6, are designed to enhance the privacy-utility trade-off compared to the standard rigorous privacy-preserving training elaborated in the previous part.

In Chapter 5, we study privacy attacks targeting real-world application scenarios of advanced generative models. Specifically, we focus on membership inference attacks on diffusion models, designed to ascertain if specific query samples have been used in the training dataset of a target diffusion model that might be deployed or integrated with media editing tools. We execute a systematic analysis of the attack surface and present highly effective attack strategies, each meticulously tailored for different attack settings.

In Chapter 6, we present an effective defense mechanisms against membership inference attacks on classification models, addressing diverse attack settings. Our principal strategy involves moderating the training objective to a more attainable level while preserving the model's discriminative power. This method effectively diminishes the generalization gap and minimizes model overfitting and overlearning, consequently mitigating privacy risks without compromising the model's performance.

5 Data Forensics in Diffusion Models: A Systematic Analysis of Membership Privacy

Contents

Diffusion models have achieved tremendous success in image generation over recent years, establishing themselves as the state-of-the-art technology for AI-based image processing systems. Despite the numerous benefits brought by recent advances in diffusion models, there are also concerns about their potential misuse, particularly in privacy breaches and intellectual property infringement. Specifically, the unique characteristics of diffusion models expose new attack surfaces and vulnerabilities when deployed in real-world systems. With a thorough exploration of the attack surface, we present a comprehensive analysis framework of membership inference attacks on diffusion models, complemented with novel attack methods tailored to each attack scenario specifically relevant to diffusion models. Leveraging easily obtainable quantities, our approach proves to be highly effective, realizing near-perfect attack performance (>0.9 AUCROC) in realistic scenarios. Our extensive evaluation demonstrates the effectiveness of our method, highlighting the importance of considering privacy risks and intellectual property protection when using diffusion models in image generation systems.

This chapter is based on [262]: As a co-first author of [262], Dingfan Chen played a pivotal role in formulating the project idea, steering the experiments, and serving as the main writer of the paper. This paper is under submission at the time of composing this thesis.

5.1 Introduction

Deep generative modeling has made significant advancements over the past few years, giving rise to photo-realistic media generation tools with emerging commercial uses for art and design. In particular, the rapid improvement of denoising diffusion models [191, 150, 80, 196, 197, 198, 46] has not only greatly advanced the state-of-the-art in the image and video generation tasks but also solidified diffusion models as arguably the most promising generative framework to date, providing the foundation for powerful commercial models such as Stable Diffusion [178], Imagen [182], and DALL·E-2 [172].

Despite the remarkable success of recent diffusion models, the widespread use of online APIs and shared pre-trained models raises concerns about their potential risks in various areas. One major concern is the risk of data misuse and violations of privacy, as sensitive information pertaining to individual identities could be revealed. Additionally, malicious users may attempt to infer the original training data, further exacerbating privacy concerns. An example of such an attack is the membership inference attack (MIA) [188], which seeks to determine if a particular data record was used to train a machine learning model. This is particularly concerning in the context of diffusion models that serve as the backbone for online media editing tools, which are accessible to the public.

MIA is closely related to other concerns, such as intellectual property (IP) infringement and leakage of the content from specific training data samples during the development and deployment of diffusion models. Advanced diffusion models rely heavily on the usage of massive and diverse training data. However, with the commercialization of these models, there is a risk of data being harvested from the internet for model training purposes without proper regard for the IP rights of media creators. Moreover, such data might be regenerated by the model in a manner that results in the leakage of the complete data sample. Even worse, it is often impractical for developers to manually review all training samples for IP compliance or to scrutinize the regeneration process to prevent data leakage. In this context, MIA emerges as an essential foundation, not only for potential IP protection [194] but also for investigating and understanding data leakage [26, 27]. Our work propels the frontier of trustworthy development of diffusion models by presenting a systematic exploration of the associated attack surface, complemented by practical and effective MIA strategies.

Diffusion models possess several distinct features that set them apart from other generative models. First of all, the encoding process in diffusion models is unlearnable and fixed, following a standard procedure that is known to the public. While this eases the training of diffusion models on complex data distributions, it also presents vulnerabilities as attackers can easily and precisely imitate the encoding process, even if the model developer tries to hide it during model deployment (Section 5.5.3). In contrast, attacks on other generative models usually require approximating the encoding process in a lossy manner, e.g., through gradient-based optimization on the model internals [34] (Section 5.6.2). Additionally, the generation process in diffusion models is iterative, resulting in multiple intermediate outputs that may all reveal information about the training samples. Such information can be easily exploited by an attacker to construct dedicated attacks tailored to diffusion models under different deployment scenarios.

In this work, we pioneer the investigation of such risks associated with diffusion models. Specifically, we conduct the first systematic analysis of MIAs against diffusion models. While previous studies have explored MIAs in the context of both classification models [188, 183, 25, 180, 244, 195, 149] and other generative models [74, 78, 34], we highlight that diffusion models have unique properties and usage patterns that create new attack surface not covered by existing works. Furthermore, established methods are mostly inappropriate to our scenario, while potential adaptations would only yield suboptimal special cases of our proposed attack (Section 5.4.4). Instead, we thoroughly examine the attack vectors and identify three major attack scenarios that are most representative and prevalent in practice, given real-world APIs as reference. Moreover, we design novel attacks tailored to diffusion models based on their unique characteristics, consistently surpassing existing solutions by a substantial margin across various settings.

Contributions: In summary, we make the following contributions in this work:

- **Task-level:** We present a comprehensive analysis framework and conduct the first systematic investigation of MIAs on state-of-the-art diffusion models. With a thorough analysis of the potential attack surface, our study reveals the most realistic threat models reflecting real-world usage patterns, with which we aim to strategically guide and faithfully benchmark future research in related fields.
- **Approach-level:** We devise novel attack strategies, tailored to suit various attack scenarios. Supported by a theoretical foundation, our attacks use easily obtainable quantities, integrating simplicity, practicability and high effectiveness. Our enhancement techniques, such as truncation and calibration (Section 5.4.1-5.4.2), markedly improve attack performance across a spectrum of realistic settings. We anticipate broader applications of our approach, extending beyond MIAs and facilitating future advancements towards trustworthy deployment of diffusion models.
- **Insight-level:** We thoroughly evaluate and provide key insights into components impacting the effectiveness of attack strategies, demonstrating our approach's consistent efficacy across varied scenarios. Specifically, having only access to the API, our approach reaches >0.95 AUCROC on the CelebA dataset with 20k training samples, where previous work generally fails to report effective attacks. Moreover, our attack demonstrates substantial effectiveness when applied to real-world pre-trained Stable-Diffusion models trained on large-scale datasets with 2.3 billion samples, evidenced by an AUCROC of >0.7 and a >24% TPR@1% FPR. Our findings reveal a dual implication: while the common practice of sharing diffusion models poses a markedly high privacy risk, strong attack strategies exist that present the potential for monitoring sample usage during model training, serving needs such as IP protection and early detection of data leakage.

5.2 Related Work

Generative Models. Generative models aim to simulate the probability distribution of real data by defining a parametric family of densities and finding the optimal parameters. The optimal parameter is typically found by either maximizing the (lower bound of) likelihood of the real data or minimizing the (estimated) divergence between the generated and real data distributions. With the advancement in the expressive power of deep neural networks, recent generative models have achieved significant success in modeling high-dimensional data distributions. Different types of deep generative models have been developed in the literature, with generative adversarial networks (GANs) [65], variational autoencoders (VAEs) [101], and diffusion-based models [191, 196, 80] being the representative ones. In this work, we focus on diffusion models, which represent the current state-of-the-art deep generative framework [46] and serve as the backbone for various online media generation tools [182, 178, 172]. Additionally, we make connections and draw comparisons with GANs and VAEs (Section 5.5.6 and 5.6.2), which were the previous leading generative frameworks.

	white-box	**gray-box**			**black-box**	
		knowledgeable	agnostic	extension	specific	agnostic
[27]	✓	-	-	-	-	-
[50]	-	✓	-	✓	-	-
[82]	✓	✓	-	-	-	-
[103]	-	✓	-	-	-	-
[136]	✓	-	-	-	-	-
[157]	✓	-	-	-	-	-
[237]	-	-	-	✓	-	-
Ours	✓	✓	✓	✓	✓	✓

Table 5.1: Attack settings (defined in Table 5.2) of related works. ✓ indicates settings where new attacks have been proposed.

Membership Inference Attacks (MIAs). MIAs were introduced by Shokri et al. [188] and initially targeted black-box classification models where the full confidence score predictions are accessible by the attacker. Subsequent works have developed various approaches in attacking both white-box [149, 174] as well as black-box [188, 180, 244, 183, 195] classification models. In particular, it has been shown that the sample loss generally can serve as a discriminative signal that tells apart members from non-members [244]. Sablayrolles et al. [180] further showed that black-box attacks can approximate the performance of white-box MIA under mild assumptions on the model parameter distribution.

Recent works have explored such attacks on popular generative models such as GANs [34, 74] and VAEs [78]. Specifically, Hayes et al. [74] noted that disclosing the discriminator of a GAN can leak membership information in a white-box setting and proposed using a shadow model for black-box attacks. Hilprecht et al. [78] suggested the reconstruction error as a membership score for white-box VAE attacks and counting generated samples within an ϵ-ball of the query for a black-box membership score. Chen et al. [34] introduced a taxonomy of MIAs against GANs and proposed an optimization-based approach for attacks with only generator access and a distance-based approach for black-box setting with only synthetic samples available.

Our work presents the first systematic analysis of MIAs on diffusion models. Despite similarities in the training objectives with VAEs and comparable generation quality to GANs, diffusion models have distinct properties and unique attack vectors that can be considered and exploited by attackers. We thoroughly examine different attack scenarios specifically relevant to diffusion models (see Table 5.2) and leverage its intrinsic characteristics, such as the pre-defined encoding process and multi-step generation process, to conduct effective attacks. Algorithmically, our approach shares the same high-level concept with existing sample loss-based techniques [244, 180, 34], but differs fundamentally by exploiting the intrinsic properties of diffusion models. This makes the membership score both representative and discriminative, leading to improved performance. Notably, while there have been some very recent attempts, mostly unpublished, to investigate privacy attacks on diffusion models [103, 237, 136, 82, 27, 157, 50], these efforts only constitute a subset of the scenarios investigated in our work (see Table 5.1), with their proposed attacks generally fall into special and sub-optimal cases of our attack model.

5.3 Background

5.3.1 Diffusion Models

Formulation. Given observed samples x_0 from a distribution of interest, the goal of a generative model is to learn to model its true underlying distribution $p(x_0)$ and generate novel samples from it. Specifically, diffusion models use a forward noising process q, i.e., the "encoding" process, to gradually transform the data distribution into a standard Gaussian $\mathcal{N}(0, \boldsymbol{I})$. The models then learn to reverse this transformation through a learnable denoising function $p_{\boldsymbol{\theta}}$, i.e., the "decoding" process. Once the denoising function $p_{\boldsymbol{\theta}}$ is learned, generating new samples from the data distribution can be achieved by sampling from the standard Gaussian and then iteratively applying the reverse denoising steps $p_{\boldsymbol{\theta}}(\boldsymbol{x}_{t-1}|\boldsymbol{x}_t)$. Formally, the forward noising process can be written as follows,

$$q(\boldsymbol{x}_t|\boldsymbol{x}_{t-1}) = \mathcal{N}(\boldsymbol{x}_t; \sqrt{\alpha_t}\boldsymbol{x}_{t-1}, (1-\alpha_t)\boldsymbol{I}) \tag{5.1}$$

where the subscript t is the step index, and α_t is a scaling factor ($0 \leq \alpha_t \leq 1$) controlling the amount of information preserved in each noising step (where a larger α_t means more information is kept). Given a sufficiently large T and an appropriate schedule of α_T, the latent $\boldsymbol{x}_T$ at the final step forms a standard Gaussian distribution. Meanwhile, the forward process defined in Equation 5.1 allows direct sampling of the noisy latent $\boldsymbol{x}_t$ at an arbitrary step given the input data $\boldsymbol{x}_0$ [80]:

$$q(\boldsymbol{x}_t|\boldsymbol{x}_0) = \mathcal{N}(\boldsymbol{x}_t; \sqrt{\bar{\alpha}_t}\boldsymbol{x}_0, (1-\bar{\alpha}_t)\boldsymbol{I}) \tag{5.2}$$

$$\boldsymbol{x}_t = \sqrt{\bar{\alpha}_t}\boldsymbol{x}_0 + \sqrt{1-\bar{\alpha}_t}\epsilon \tag{5.3}$$

where $\bar{\alpha}_t = \prod_{s=0}^{t} \alpha_s$ and $\epsilon \sim \mathcal{N}(0, \boldsymbol{I})$ denotes a random noise sample. Moreover, the posterior $q(\boldsymbol{x}_{t-1}|\boldsymbol{x}_t, \boldsymbol{x}_0)$ can be computed using Bayes theorem:

$$\begin{aligned} q(\boldsymbol{x}_{t-1}|\boldsymbol{x}_t, \boldsymbol{x}_0) &= \mathcal{N}(\boldsymbol{x}_{t-1}; \mu_q(\boldsymbol{x}_t, \boldsymbol{x}_0), \Sigma_q(t)) \\ \mu_q(\boldsymbol{x}_t, \boldsymbol{x}_0) &= \frac{\sqrt{\alpha_t}(1-\bar{\alpha}_{t-1})\boldsymbol{x}_t + \sqrt{\bar{\alpha}_{t-1}}(1-\alpha_t)\boldsymbol{x}_0}{1-\bar{\alpha}_t} \\ \Sigma_q(t) &= \frac{(1-\alpha_t)(1-\bar{\alpha}_{t-1})}{1-\bar{\alpha}_t}\boldsymbol{I} \end{aligned} \tag{5.4}$$

The joint distribution for the reverse process can be formularized as:

$$p(\boldsymbol{x}_{0:T}) = p(\boldsymbol{x}_T)\prod_{t=1}^{T} p_{\boldsymbol{\theta}}(\boldsymbol{x}_{t-1}|\boldsymbol{x}_t) \tag{5.5}$$

with $p(\boldsymbol{x}_T) = \mathcal{N}(\boldsymbol{x}_T; 0, \boldsymbol{I})$, indicating that the latent distribution at the final step T is a standard Gaussian. The denoising function is modeled as a Gaussian using a neural network as follows:

$$p_{\boldsymbol{\theta}}(\boldsymbol{x}_{t-1}|\boldsymbol{x}_t) = \mathcal{N}(\boldsymbol{x}_{t-1}; \mu_{\boldsymbol{\theta}}(\boldsymbol{x}_t, t), \Sigma_{\boldsymbol{\theta}}(\boldsymbol{x}_t, t)) \tag{5.6}$$

Objective. The diffusion models are trained to maximize the variational lower bound (VLB), i.e., a lower bound of the log-likelihood of the observed data. Formally,

$$\begin{aligned} \log p(\boldsymbol{x}_0) \geq\ & \mathbb{E}_{q(\boldsymbol{x}_1|\boldsymbol{x}_0)}[\log p_{\boldsymbol{\theta}}(\boldsymbol{x}_0|\boldsymbol{x}_1)] - D_{\mathrm{KL}}(q(\boldsymbol{x}_T|\boldsymbol{x}_0)\|p(\boldsymbol{x}_T)) \\ & - \sum_{t=2}^{T} \mathbb{E}_{q(\boldsymbol{x}_t|\boldsymbol{x}_0)}[D_{\mathrm{KL}}(q(\boldsymbol{x}_{t-1}|\boldsymbol{x}_t, \boldsymbol{x}_0)\|p_{\boldsymbol{\theta}}(\boldsymbol{x}_{t-1}|\boldsymbol{x}_t))] \end{aligned}$$

where D_{KL} denotes the KL divergence. The training objective can be equivalently written as minimizing the negative VLB:

$$\boldsymbol{\theta}^* = \arg\min_{\boldsymbol{\theta}} \mathcal{L}_{vlb} = \arg\min_{\boldsymbol{\theta}} \{\mathcal{L}_0 + \mathcal{L}_1 + ... + \mathcal{L}_T\} \tag{5.7}$$

$$\mathcal{L}_0 = -\log p_{\boldsymbol{\theta}}(\boldsymbol{x}_0|\boldsymbol{x}_1) \tag{5.8}$$

$$\mathcal{L}_{t-1} = D_{\mathrm{KL}}(q(\boldsymbol{x}_{t-1}|\boldsymbol{x}_t, \boldsymbol{x}_0)\|p_{\boldsymbol{\theta}}(\boldsymbol{x}_{t-1}|\boldsymbol{x}_t)),\ 2 \leq t \leq T \tag{5.9}$$

$$\mathcal{L}_T = D_{\mathrm{KL}}(q(\boldsymbol{x}_T|\boldsymbol{x}_0)\|p(\boldsymbol{x}_T)) \tag{5.10}$$

In practice, $\mathcal{L}_0$ is computed by discretizing each color component into 256 bins, and evaluating the probability of $p_{\boldsymbol{\theta}}(\boldsymbol{x}_0|\boldsymbol{x}_1)$ landing in the correct bin [46, 150]. $\mathcal{L}_{t-1}$ in Equation 5.9 is computed by sampling from an arbitrary step of the forward noising process (Equation 5.3) and estimate $\mathcal{L}_{t-1}$ using Equation 5.4 and 5.6. Optimizing over the sum in Equation 5.7 on the training dataset is achieved by randomly sampling t for each image $\boldsymbol{x}_0$ in each mini-batch, i.e., approximating $\mathcal{L}_{vlb}$ using the expectation $\mathbb{E}_{t,\boldsymbol{x}_0,\epsilon}[\mathcal{L}_t]$.

Parameterization. Recent works develop different ways to parameterize $p_{\boldsymbol{\theta}}$ in Equation 5.6 for solving $\arg\min_{\boldsymbol{\theta}} \mathcal{L}_{t-1}$:

$$\arg\min_{\boldsymbol{\theta}} \mathcal{L}_{t-1}$$

$$= \arg\min_{\boldsymbol{\theta}} \frac{1-\bar{\alpha}_t}{2(1-\alpha_t)(1-\bar{\alpha}_{t-1})}\|\mu_{\boldsymbol{\theta}}(\boldsymbol{x}_t, t) - \mu_q\|_2^2 \tag{5.11}$$

$$= \arg\min_{\boldsymbol{\theta}} \frac{\bar{\alpha}_{t-1}(1-\alpha_t)}{2(1-\bar{\alpha}_t)(1-\bar{\alpha}_{t-1})}\|\hat{\boldsymbol{x}}_{\boldsymbol{\theta}}(\boldsymbol{x}_t, t) - \boldsymbol{x}_0\|_2^2 \tag{5.12}$$

$$= \arg\min_{\boldsymbol{\theta}} \frac{(1-\bar{\alpha}_t)(1-\alpha_t)}{2\alpha_t(1-\bar{\alpha}_{t-1})}\|\epsilon_0 - \hat{\epsilon}_{\boldsymbol{\theta}}(\boldsymbol{x}_t, t)\|_2^2 \tag{5.13}$$

$$= \arg\min_{\boldsymbol{\theta}} \frac{(1-\bar{\alpha}_t)(1-\alpha_t)}{2\alpha_t(1-\bar{\alpha}_{t-1})}\|s_{\boldsymbol{\theta}}(\boldsymbol{x}_t, t) - \nabla \log p(\boldsymbol{x}_t)\|_2^2 \tag{5.14}$$

The most obvious option is to let the neural network predict $\mu_{\boldsymbol{\theta}}(\boldsymbol{x}_t, t)$ directly (Equation 5.11). Alternatively, the network could predict $\boldsymbol{x}_0$ from noisy image $\boldsymbol{x}_t$ and time index t (Equation 5.12) [80]. The network could also predict the noise ϵ_0 that determines $\boldsymbol{x}_t$ from $\boldsymbol{x}_0$ (Equation 5.13) [182, 80] and the score of the image at an arbitrary noise level, i.e, the gradient of $\boldsymbol{x}_t$ in data space (Equation 5.14) [196, 197, 198].

5.3.2 Membership Inference

In MIA, attackers are given a query set $\mathbb{S} = \{(\boldsymbol{x}^i, m^i)\}_{i=1}^N$ with both member (training) and non-member (testing) samples $\boldsymbol{x}^i$ from the same distribution. The membership attribute m^i indicates if $\boldsymbol{x}^i$ is a member ($m^i = 1$ for members). The goal is to determine the membership attribute m^i of each sample $\boldsymbol{x}^i$. Each image is typically a distinct sample (i.e., $\boldsymbol{x}^i = \boldsymbol{x}^i_{\mathrm{img}}$), but for conditional generation, a sample could also contain text (i.e., $\boldsymbol{x}^i = (\boldsymbol{x}^i_{\mathrm{img}}, \boldsymbol{x}^i_{\mathrm{text}})$). The attack $\mathcal{A}(\boldsymbol{x}^i, \mathcal{M}(\boldsymbol{\theta}))$ predicts m^i for a given query $\boldsymbol{x}^i$ and a target diffusion model $\mathcal{M}$ parameterized by $\boldsymbol{\theta}$. The Bayes optimal attack $\mathcal{A}_{opt}(\boldsymbol{x}^i, \mathcal{M}(\boldsymbol{\theta}))$ is given by:

$$\mathcal{A}_{opt}(\boldsymbol{x}^i, \mathcal{M}(\boldsymbol{\theta})) = \mathbb{1}\left[\log \frac{P(m^i = 1|\boldsymbol{x}^i, \boldsymbol{\theta})}{P(m^i = 0|\boldsymbol{x}^i, \boldsymbol{\theta})} \geq 0\right] \tag{5.15}$$

	Model type	Image embedding	Model internals	Hyperparameters T, α_T
white-box (5.4.1)	✓	✓	□	✓
gray-box knowledgeable (5.4.2)	✓	✓	■	✓
gray-box agnostic (5.4.2, 5.5.3)	✓	✓	■	×
gray-box extension (5.6.3)	✓	×	■	✓
black-box specific (5.4.3)	✓	×	■	×
black-box agnostic (5.4.3)	×	×	■	×

Table 5.2: Taxonomy of attack settings. (×: without access; ✓: with access; ■: black-box; □: white-box). "Model type": Whether the underlying model is known to be a diffusion model. "Image embedding": Whether access to the image embedding during generation is available.

with $\mathbb{1}$ denoting the indicator function, and $P(m^i = 1|\boldsymbol{x}^i, \boldsymbol{\theta})$ representing the real underlying membership probability.

Our attack is motivated by the recent results showing the dependence of attack success rate on the sample loss [244, 180]: a large difference in expected loss between members and non-members leads to successful attacks [244]. Sablayrolles et al. [180] further prove that the Bayes optimal attack depends only on the sample loss under a mild posterior assumption of the model parameters, resulting in:

$$\mathcal{A}_{opt}(\boldsymbol{x}^i, \mathcal{M}(\boldsymbol{\theta})) = \mathbb{1}\left[\ell(\boldsymbol{\theta}, \boldsymbol{x}^i) < \tau(\boldsymbol{x}^i)\right] \tag{5.16}$$

where τ denotes a threshold function.

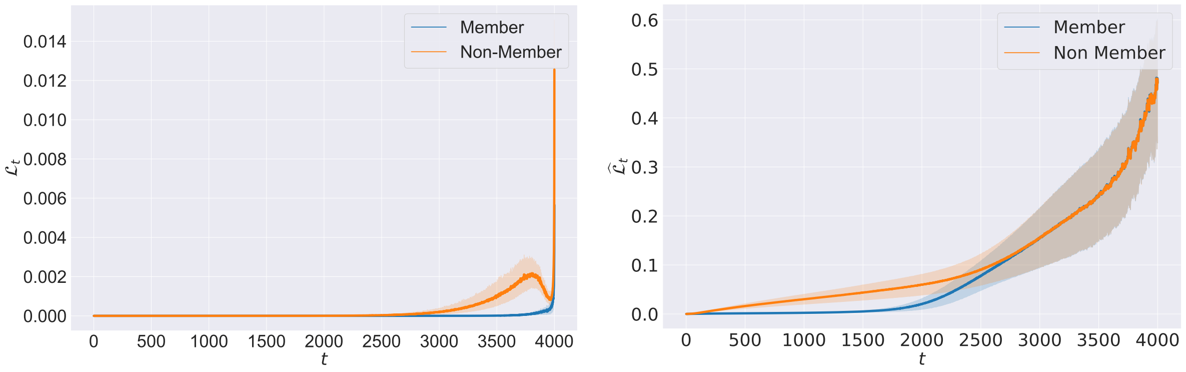
Equation 5.16 implies a sample is likely a training member if the target model shows a low loss on it. While many existing attacks [34, 180, 78] fit this paradigm, direct application to diffusion models would suggest using the VLB loss $\mathcal{L}_{vlb}$ (Equation 5.7) as the membership score (see discussions in Section 5.4.4). However, this approach (though adopted by previous works such as [82, 136]) results in sub-optimal performance (see Section 5.5.2 and 5.6.1) or even be infeasible under realistic threat models. Our proposed approach, tailored for each possible threat model, is described in detail in Section 5.4.

5.4 General Attack Pipeline

5.4.1 White-box Setting

Threat Model. We start by investigating the white-box setting, representing the most informed attacker scenario. In this scenario, attackers have complete access to the trained model parameters $\boldsymbol{\theta}$, as well as the necessary information for model implementation, such as the number of total steps T as well as the (schedule of) scaling factor α_t used in the forward and backward pass. This scenario reflects the common open-source practice of releasing the source code and the pre-trained model checkpoints for public use. These resources, often used as building blocks for image editing tools, are readily available online.

Approach. Existing results suggest that utilizing the sample loss is a viable approach for calculating membership score (see Section 5.3.2). However, relying solely on $\mathcal{L}_{vlb}$ results in subpar outcomes (see Section 5.5.2). We conjecture that this is mainly due to the following reasons: First, the randomness in the sampling process during training may cause the unequal weight of each term $\mathcal{L}_t$ in the total sum, leading to a deviation from the intended objective $\mathcal{L}_{vlb}$.



(a) Distribution of $\mathcal{L}_t$

(b) Distribution of $\widehat{\mathcal{L}_t}$

Figure 5.1: The distribution of $\mathcal{L}_t$ (5.1(a)) and $\widehat{\mathcal{L}_t}$ (5.1(b)) terms for each reverse denoising step t of a target diffusion model trained on CelebA with 5k samples.

More importantly, the large variance in the scale of each term $\mathcal{L}_t$ results in less informative terms dominating the $\mathcal{L}_{vlb}$ sum, leading to a less discriminative outcome for membership inference. As depicted in Figure 5.1(a), the terms with a larger value of t have the greatest impact on the VLB loss, but they may be less informative in determining membership. This is because they are close to the Gaussian noise endpoint $\boldsymbol{x}_T$ and contain limited information about the sample itself $\boldsymbol{x}_0$.

Hence, we propose using the independent terms $\mathcal{L}_t$ that are most discriminative for membership inference instead of $\mathcal{L}_{vlb}$ (the sum of all terms). For this, we focus on terms $0 \leq t \leq T_{\text{trun}}$, where T_{trun} is the point where the loss terms become significantly larger than the previous ones. A simple rule of thumb is to set T_{trun} to approximately $0.75T$, which achieves high effectiveness and is not sensitive to different datasets. In practice, a reasonable choice of T_{trun} can be selected on a small reference set. We also investigate various statistical functions to summarize the loss terms (expressed as a general function f in Equation 5.17) instead of just the sum as in $\mathcal{L}_{vlb}$ (see Section 5.5.2). We anticipate that a learnable function f may also be effective, but for simplicity and high attack effectiveness, we use simple statistics such as mean and median. Our attack can be formulated as follows:

$$\mathcal{A}(\boldsymbol{x}^i, \mathcal{M}(\boldsymbol{\theta})) = \mathbb{1}\left[f\left(\left\{\mathcal{L}_t(\boldsymbol{\theta}, \boldsymbol{x}^i)\right\}_{t=0}^{T_{\text{trun}}}\right) < \tau(\boldsymbol{x}^i)\right] \tag{5.17}$$

In line with existing results, the attack predicts that a sample belongs to the training set if its overall loss (summarized by f) is lower than a threshold $\tau(\boldsymbol{x}^i)$. The threshold function can be calibrated to each sample to account for the impact of sample difficulty on membership inference [180, 25].

5.4.2 Gray-box Setting

Threat Model. In this setting, we consider a more realistic scenario where the attacker does not have direct access to the model parameter $\boldsymbol{\theta}$, but can still execute the model. This closely resembles a situation where the model is available to the public through an online API, where the model owner allows others to use the essential functions of the model without disclosing the underlying model. The information that the attacker can exploit may vary depending on the information released through the API. For example, some APIs allow greater control over the generation process, while others do not. Here, we present our attack designed for scenarios that most closely resemble existing real-world APIs and defer the discussion of more relaxed settings to the next subsection.

Approach. Similar to attacking a white-box diffusion model, we still use the sample loss as the membership indicator. However, the attacker does not have direct access to the terms $\mathcal{L}_t$. Instead, what the attacker can access are the intermediate outputs $\hat{\boldsymbol{x}}_{\boldsymbol{\theta}}(\boldsymbol{x}_t, t)$ of the diffusion models applying t denoising steps given the image embedding $\boldsymbol{x}_T$. (We consider an image generation model here and discuss the extension to text-to-image models in Section 5.6.3.) Given a query image $\boldsymbol{x}_0$, the attacker first runs the forward pass to obtain the image embedding $\boldsymbol{x}_T$. Note that this requires knowledge or an educated guess about the total number of steps T and the scheduling of the scaling factor α_t. This information is typically displayed on online APIs (see Appendix D.1.3) that allow flexible control of the generation process (e.g., as the "num_inference_steps" and "scheduler" parameters.) The intermediate outputs can be obtained by controlling the number of inference steps t and extracting the corresponding output images displayed on the API. The attack can be formulated as:

$$\begin{aligned}\mathcal{A}(\boldsymbol{x}^i, \mathcal{M}(\boldsymbol{\theta})) &= \mathbb{1}\left[f\left(\left\{\widehat{\mathcal{L}_t}(\boldsymbol{\theta}, \boldsymbol{x}^i)\right\}_{t=0}^{T_{\text{trun}}}\right) < \tau(\boldsymbol{x}^i)\right] \\ \text{with} \quad \widehat{\mathcal{L}_t}(\boldsymbol{\theta}, \boldsymbol{x}^i) &= \|\hat{\boldsymbol{x}}_{\boldsymbol{\theta}}(\boldsymbol{x}_t^i, t) - \boldsymbol{x}_0^i\|_2^2\end{aligned} \tag{5.18}$$

Compared to the white-box setting, the attack assumption is slightly relaxed in that the attacker needs to estimate the loss terms based on the information typically available on online APIs. Each term of $\widehat{\mathcal{L}_t}$ differs from the ground-truth $\mathcal{L}_t$ used in the white-box case by a scaling factor (Equation 5.12). We deliberately do not use this scaling factor to reduce the impact of the attacker not knowing the exact α_t (this is accessible in some existing APIs but not all of them) and potentially making incorrect guesses in some cases. Additionally, we use the same truncation trick and explore several statistic functions f as in the white-box case to encourage distinguishability in the membership score.

Additionally, we explore the situation where the model owner may reduce the intermediate outputs by subsampling the inference steps, for example, to speed up the generation or limit potential privacy exposure. Formally, the attack in this case can be formulated as follows:

$$\begin{aligned}\mathcal{A}(\boldsymbol{x}^i, \mathcal{M}(\boldsymbol{\theta})) &= \mathbb{1}\left[f(\mathcal{S}) < \tau(\boldsymbol{x}^i)\right] \\ \text{with} \quad \mathcal{S} &\subseteq \left\{\widehat{\mathcal{L}_t}(\boldsymbol{\theta}, \boldsymbol{x}^i)\right\}_{t=0}^{T}\end{aligned} \tag{5.19}$$

That is, the attacker may only have access to a subset of the intermediate outputs from the reverse denoising steps of the diffusion model. We delve into the truncation techniques specific to this scenario in the experiment section.

5.4.3 Black-box Setting

Threat Model. In this scenario, attackers are limited to passively obtaining generated samples from well-trained generative models, without the ability to affect the generation process. This creates a realistic scenario, as there are no set assumptions about the attacker's abilities. We categorize the situation into two cases based on the attacker's knowledge of the synthetic data being generated by a diffusion model, referred to as "*known model type*" and "*unknown model type*". In the "*known model type*" case, the attacker recognizes that the accessible synthetic data was produced by a diffusion model and may exploit this information to design targeted attacks. For instance, this may correspond to common situations where the attacker can only collect final outputs from online diffusion model APIs but is not allowed to perform steerable generation, thereby preventing our gray-box attacks discussed in Section 5.4.2.

Approach. ***Model-specific Attack***: Once the attacker knows that the synthetic data set was generated by a diffusion model, a natural approach would be to train a shadow diffusion model to imitate the target diffusion model by using the synthetic data set as the training set. This enables the attacker to carry out an attack in the same way as in a white-box scenario. This type of attack is referred to as a *"model-specific attack"* to differentiate it from attacks that do not use any information about the generative models. Formally,

$$\mathcal{A}(\boldsymbol{x}^i, \mathcal{M}(\boldsymbol{\theta})) = \mathbb{1}\left[f\left(\{\mathcal{L}_t(\boldsymbol{\theta}^s, \boldsymbol{x}^i)\}_{t=0}^{T_{\text{trun}}}\right) < \tau(\boldsymbol{x}^i)\right] \tag{5.20}$$

where $\boldsymbol{\theta}^s$ represents the parameters of the shadow model, which were obtained by training the diffusion model on the synthetic data generated by the target model $\mathcal{M}(\boldsymbol{\theta})$.

Model-agnostic Attack: In the absence of any additional information except for the synthetic sample set, the attacker's last resort is to use *model-agnostic attacks*. Several options exist in the literature, such as GAN-Leak [34], which uses the Euclidean distance to the nearest neighbor in the synthetic set as a proxy for the sample loss and membership score, and Monte-Carlo [78], which counts the number of generated samples within an ϵ-ball of the query using a carefully designed distance metric. In line with previous work, we use the distance of the query image to its nearest neighbor in the synthetic set as the membership score. Furthermore, we enhance the distance metric by using a pre-trained feature extractor (trained on the large-scale public ImageNet [45] dataset) and further refine the distance calculation by leveraging label information if available. Formally,

$$\mathcal{A}(\boldsymbol{x}^i, \mathcal{M}(\boldsymbol{\theta})) = \mathbb{1}\left[\min_k \left\{\ell_{dis}(\boldsymbol{x}^i, \boldsymbol{s}^k)\right\}_{k=1}^{K} < \tau(\boldsymbol{x}^i)\right] \tag{5.21}$$

with $\boldsymbol{s}^k \sim p_{\boldsymbol{\theta}}$ representing the samples generated by the target model parameterized by $\boldsymbol{\theta}$. K denotes the total number of synthetic samples, and ℓ_{dis} is the cosine distance in the feature space of a pre-trained ImageNet classifier, where the feature space is determined by the output of the second last layer.

5.4.4 Analysis & Insights

While our attack and several previous ones fall within a general likelihood ratio formulation (Equation 5.15 in Section 5.3.2), and therefore demonstrate analogous algorithmic components, our attack possesses distinctive features that are critical to its superior effectiveness. Specifically, the white-box attack in [78] uses reconstruction error as the membership score. Adapting it to diffusion models (has been realized in [82]) results in a special case of our gray-box attack (using the loss term at the final time step), but with suboptimal configurations, i.e., truncating all steps except the final one or using the "Min" statistical function (see results in Section 5.5.3). The adaptation of the GAN-Leaks [34] white-box attack to diffusion models would suggest optimizing the latent code using a gradient-based method to find the nearest neighbour in the output space of the target diffusion model and using the nearest-neighbour distance as the membership score. This approach will be upper-bounded by the aforementioned white-box attack of [78] (that does not require optimization over the latent space), and thus further upper-bounded by our methods. Seen from a more abstract perspective, the GAN-Leaks framework (suggests using the approximated data-likelihood) would translate into our white-box attack with a "Sum" statistical function (has been adopted by [82, 136]), which is also a suboptimal special case of ours. Furthermore, while previous black-box model-agnostic attacks are applicable to diffusion models, we show in Sections 5.5.4 that our attack generally outperforms previous ones (Figures 5.6(a)-5.6(b)).

The primary factors motivating the specialized features of our attack design lie in the intrinsic properties of diffusion models. Firstly, diffusion models generate data iteratively, a characteristic that sets them apart from most other types of generative models. This formulation offers key opportunities for the attack to exploit knowledge from multiple iterations, instead of relying solely on the singular one-shot signal as done in previous attacks. Moreover, the loss terms produced from different iterations may convey varying amounts of information. This variability necessitates specialized treatment, such as the truncation technique used in our method. Specifically, by the construction of diffusion models, the Signal-to-Noise Ratio (SNR) for the latent variable distribution at each time step conditioned on the clean sample $q(\boldsymbol{x}_t|\boldsymbol{x}_0)$, decreases monotonically as the time step increases [99]. In other words, as $q(\boldsymbol{x}_t|\boldsymbol{x}_0)=\mathcal{N}(\boldsymbol{x}_t; \sqrt{\bar{\alpha}_t}\boldsymbol{x}_0, (1-\bar{\alpha}_t)\boldsymbol{I})$ (Equation 5.2), $\mathrm{SNR}(t)=\frac{\mu^2}{\sigma^2}=\frac{\bar{\alpha}_t}{1-\bar{\alpha}_t}$, with $\mathrm{SNR}(t) < \mathrm{SNR}(s)$ for all $t > s$. This inherent property results in loss terms that are closer to the Gaussian noise end becoming larger, since the latent variable tends to carry less informative signals about the samples $\boldsymbol{x}_0$ as t increases (as illustrated in Figure 5.1). Our attack design accommodates these unequal weights for each term by permitting arbitrary statistical functions that can capture the distinctive characteristics of various scenarios. Additionally, our method explicitly eliminates terms that, while dominant in scale, provide less discriminative information.

5.5 Experiment Analysis

In this section, we present the first systematic evaluation of membership inference attacks against state-of-the-art diffusion models. Our comprehensive study encompasses all viable threat models, ranging from the most knowledgeable white-box setting to the most practical black-box one. Importantly, we make key discoveries linking the attack effectiveness to the threat model, data set, model architecture, and training configuration, leading to practical implications for securing the deployment of diffusion models in real-world settings.

5.5.1 Setup

Target Models and Datasets. In line with previous research, we conduct experiments on benchmark datasets with diverse characteristics, including **CelebA**, **CIFAR-10**, **Laion2B-improved-aesthetics**, and **COCO-2017**. We evaluate state-of-the-art diffusion models using their official PyTorch implementation: **Improved Diffusion** [150], **Guided Diffusion** [46] and pre-trained **Stable Diffusion** [178]. By default, we set the number of denoising steps T to be 4000 and adopt a standard linear scheduler for α_t. We mainly focus on the unconditional image generation task and investigate text-to-image generation model in Section 5.6.3. All experiments were conducted on a single NVIDIA A100 GPU. Notably, our target models generate high-quality output (Table 5.8 and Figure D.1), surpassing the results in previous works [74, 78, 34], demonstrating the high practical value of our study.

Attack Evaluation. We adopt the standard assessment procedures in MIA literature [34, 74, 180, 183, 188]. We assess the attack effectiveness on a balanced query set $\mathcal{S}$, i.e., with an equal

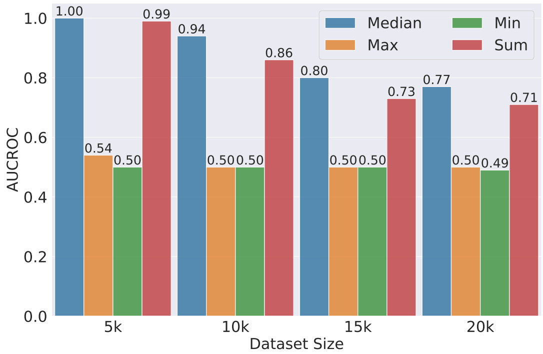


Figure 5.2: The **white-box** attack AUCROC when applying *different statistic function f* (*Sum*, *Median*, *Min*, and *Max*) to the entire loss trajectory $\{\mathcal{L}_t\}_{t=0}^{T}$ on CelebA. The *"Sum"* function corresponds to the direct use of $\mathcal{L}_{vlb}$ for MIA.

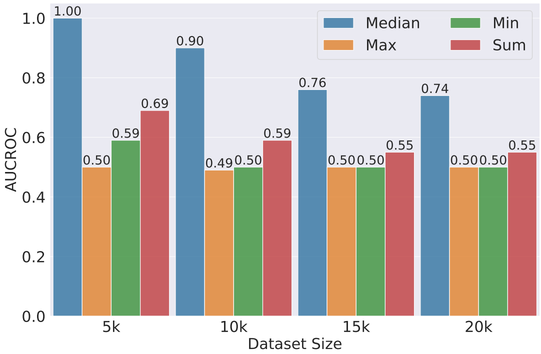
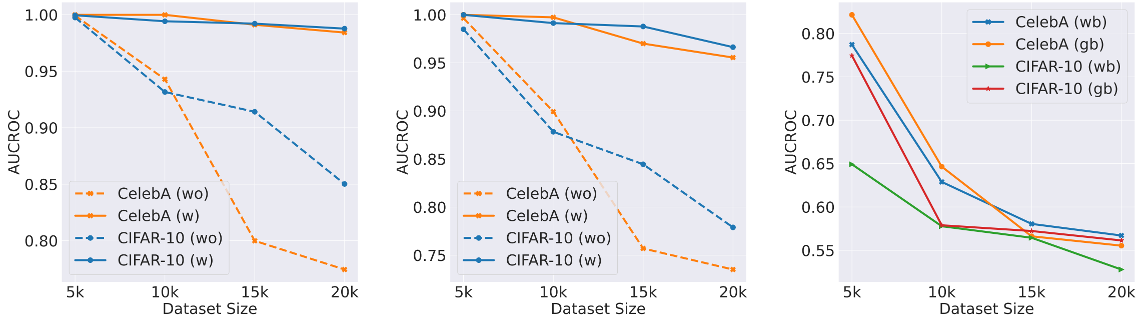


Figure 5.3: The **gray-box** attack AUCROC when applying *different statistic function f* (*Sum*, *Median*, *Min*, and *Max*) to the entire loss trajectory estimated based on the intermediate outputs (i.e., $\{\widehat{\mathcal{L}}_t\}_{t=0}^{T}$) on CelebA.



(a) White-box

(b) Gray-box

(c) Black-box (model-specific)

Figure 5.4: The attack AUCROC across different *dataset sizes* and *attack scenarios*. The results obtained with (indicated as "w" and shown as solid lines) and without (indicated as "wo" and shown as dashed lines) applying our truncation techniques are compared. We present the best results for each case and the truncation step is always set to be $0.75T$.

number of members and non-members. The attack performance is evaluated by measuring the area under the receiver operating characteristic curve (**AUCROC**), which is obtained by varying τ. The complete **ROC curve** is also provided for clear visualization of the attack's properties [25]. Moreover, we evaluate the truth false positive rate under a certain low false positive rate (i.e., **TPR@1%FPR** and **TPR@0.1%FPR**) to demonstrate the attack performance with realistic scenarios [25]. Additionally, following [74], the attack **Accuracy** and **F1 Score** are calculated by setting the threshold to the median value of the membership scores over the query set. All these metrics have a value ranging from 0 to 1, with a higher value indicating a more effective attack.

5.5.2 Evaluation on White-box Attack

Effectiveness of Sample Losses as Membership Score. We first assess the feasibility of inferring membership with white-box access to a target diffusion model based on the sample loss terms $\{\mathcal{L}_t\}_{t=0}^{T}$ and/or the VLB loss $\mathcal{L}_{vlb}$ (which represents the sum of all loss terms). As

Size	ref	T	$0.975T$	$0.875T$	$0.75T$	$0.625T$	$0.5T$
5k	1.00	0.54	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**
10k	0.94	0.50	0.97	**1.00**	**1.00**	0.99	0.93
15k	0.80	0.50	0.81	**0.99**	**0.99**	0.96	0.80
20k	0.77	0.50	0.65	0.97	**0.98**	0.95	0.77

Table 5.3: The **white-box** Attack AUCROC for different truncating steps T_{trun} in CelebA trained with different training data size. The **"ref"** (meaning "reference") column represents the best results that can be achieved without any truncation (but may correspond to different statistic functions). The rest of the results correspond to using a *"Max"* statistic function. The columns, from left to right, represent increasing levels of truncation, excluding the top $[0, 100, 500, 1000, 1500, 2000]$ steps, respectively. The column labeled "T" (in gray) represents "no truncation". The best results (selected to four decimal places) for each configuration are highlighted in bold.

shown in Figure 5.2, even simply using $\mathcal{L}_{vlb}$ as the membership score achieves promising attack performance. For instance, we obtain 0.62 AUCROC when using $\mathcal{L}_{vlb}$ to attack the target model trained with 20k data samples. The performance improves to 0.68 when we explore various options of the statistical function $f \in \{Sum, Median, Max, Min\}$ and select the best, which is the *"Median"* in this setting. As a reference, previous work reported a maximum AUCROC of 0.61 under comparable conditions (i.e., white-box attack with the same size and type of split in the training set, without using additional reference data) when attacking GANs [34]. The *"Average"* function is not considered as it is equivalent to *"Sum"* in terms of discrimination. While we expect that a more complicated design of function f might lead to improved results, we observe that a simple data-independent function works sufficiently well in most cases and stick to such choices throughout our evaluation.

Performance Gain from Truncation. To address the instability and indistinguishability caused by large variations in the magnitude of $\mathcal{L}_t$ (as discussed in Section 5.4.4 and depicted in Figure 5.1(a)), we improve our attacks by truncating the loss trajectory. This involves excluding the initial denoising steps that have limited relevance to membership but have high values that can easily dominate the statistical function. We present the detailed results in Table D.4 (in Appendix) and present the results for the best configuration (f is selected to be *"Max"* and T_{trun} is set to be $0.75T$) in Figure 5.4(a). As shown in Figure 5.4(a), our truncation techniques consistently improve the attack performance across various training configurations. In the most challenging setting with a dataset size larger than 15k, the improvement is particularly significant: by around 0.2 AUCROC on CIFAR-10 and 0.25 on CelebA, respectively.

We further show that the performance gain is *not* sensitive to the particular choice of the truncation step T_{trun} and training configurations. As can be seen from Table 5.3 (See Figure D.2(a) for the visualization), a boost in the attack performance can be achieved with a relatively large range of possible values of T_{trun} (e.g., when T_{trun} is roughly in the range from $0.875T$ to $0.75T$). Moreover, the best value turns out to be consistent across different training settings (i.e., the training set size and dataset in our experiments). This suggests a high practical value of our technique such that the attacker may be able to determine the appropriate parameter on any available reference dataset and use such selected parameters for completing the attack. We set by default the statistic function to be a *"Max"* function and T_{trun} to be $0.75T$ for our white-box attack when adopting truncation techniques, which empirically leads to promising performance across various situations.

Effect of Dataset Size. The size of the training dataset is a key determinant of the membership risk associated with machine learning models, as previously noted in the literature [188, 74, 34]. As the number of training samples grows, the model becomes unable to capture the point-wise delta distribution and moves from memorization to generalization. Specifically, previous studies have shown that MIA performance tends to be less effective (< 0.6 AUCROC) when the training dataset size exceeds 10k [78, 34].

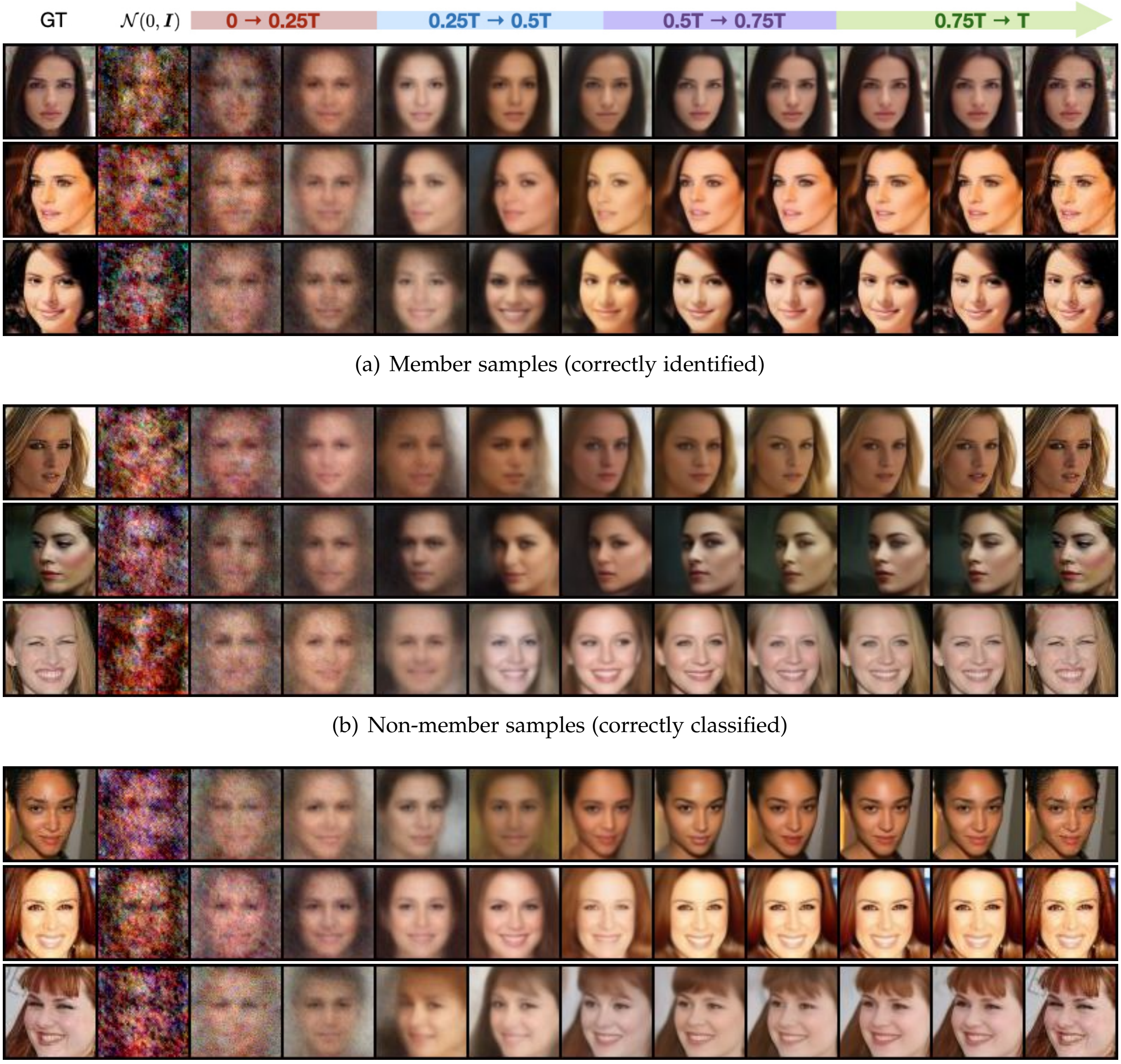
Consistent with these findings, our results show a tendency of decline in the attack AUCROC as the size of the training dataset grows (refer to Figure 5.4(a)). However, we observe a consistently high level of attack performance throughout our evaluations, even in cases where previous attacks have typically failed. For example, when the training set size is up to 10k, our attack achieves near-perfect AUCROC on both datasets, and even with a training set size of 20k, our attack remains highly effective (0.98 attack AUCROC for CelebA and 0.99 for CIFAR-10) after applying our truncation techniques. These results highlight the potential of using sample losses for effective attacks and the significant privacy risk incurred by the common practice of sharing diffusion models in open source.

5.5.3 Evaluation on Gray-box Attack

Estimated Losses as Membership Score. We further consider the real-world scenario where third-party providers, such as Amazon AWS, offer API services to create images using diffusion models. In this scenario, attackers typically have access to the generated images at any inference step of the deployed diffusion model (e.g., by specifying the inference step parameter and obtaining the displayed output), but may not have knowledge of the ground-truth loss terms $\mathcal{L}_t$ (which requires knowing the exact values of α_t). As discussed in Section 5.4.2, the attack must estimate the loss based on the intermediate output shown on the API. We evaluate the attack performance based on our proposed estimation in Equation 5.18.

We present our results obtained by using the whole estimated loss trajectories $\{\widehat{\mathcal{L}}_t\}_{t=0}^{T}$ in Figure 5.3. As can be seen, we demonstrate promising performance with an AUCROC of 0.9 for datasets with up to 10k training samples, and an AUCROC value of 0.74 when the dataset size grows to 20k. Despite a slight decrease in comparison to the white-box setting, our gray-box already achieves a reasonable level of performance in its vanilla form, suggesting the potential effectiveness of our formulation. Our results also reveal that "*Median*" and "*Sum*" statistical functions perform better than the others, with "*Median*" showing the best performance in most cases across different training configurations. This naturally follows our intuition that using a robust statistic that captures the discriminative factors may be preferable over simply aggregating the available information. Moreover, when compared to the white-box case, the loss terms used by our gray-box method generally exhibit higher variance and magnitude (caused by the difference in the scaling factors between the white-box and gray-box loss terms). This observation partially explains the superior performance of different functions in each scenario. Specifically, "Median" function is a more robust choice for the gray-box attack, while the "Max" function is a more discriminatory choice for the white-box attack.

Performance Gain from Truncation. Similar to the case in the white-box setting, not all of the estimated loss terms are informative in distinguishing between members and non-members. Moreover, recall that the generation process in a diffusion model is designed to mimic the reverse denoising process, resulting in noisier outputs in the early denoising stages and thus a larger difference when compared with the clean query sample. By construction, this makes these loss terms corresponding to the earlier denoising steps to have larger magnitudes and to possibly dominate the attack prediction.



(a) Member samples (correctly identified)

(b) Non-member samples (correctly classified)

(c) Member samples (but classified as non-members)

Figure 5.5: The query images together with their reconstruction triggered by our gray-box attack. The first column presents the ground-truth query images, while the last column refers to the final generated images from Guided-Diffusion models trained on CelebA (20k). The intermediate results are plotted per 400 time steps.

Our truncation technique is an effective solution for addressing this issue in the gray-box setting. As demonstrated in Figure 5.4(b) and detailed in Table D.5 (in the Appendix D), our truncation technique consistently improves the attack AUCROC by up to 0.22 on CelebA and 0.2 on CIFAR-10, particularly in challenging cases where the training set size is larger than 10k. These cases typically result in less successful attacks with AUCROC less than 0.6 for existing works [34], whereas we achieve highly effective attacks with AUCROC around 0.95 throughout our evaluation.

We further validate our intuition (discussed in Section 5.4.4) via qualitative results in Figure 5.5, where we generally observed that: (1) Starting from the noise end, the intermediate results from the first $0.25T$ steps do not manifest significant visual distinction, which supports our truncation technique to eliminate or reduce the influence of less informative terms. (2) For member data, the intermediate results begin to visually resemble the query at around the $0.5T$ time step counted from the noise end. In contrast, non-member images require more steps (approximately up to $0.75T$) to reach a similar level of visual similarity. This discrepancy suggests that the target diffusion model indeed displays distinct behaviors for member vs. non-member images, which can be exploited by an adversary. (3) Both member and non-member images can be reconstructed to a high degree of visual similarity by the final step. This observation clarifies why relying solely on the final reconstruction difference (as suggested by prior works [78, 34]) may not yield optimal effectiveness. (4) While some member samples might display complex visual patterns (and/or be underrepresented in the distribution) making them more challenging to reconstruct and thus harder for an MIA to detect (see examples in Figure 5.5(c)), they still tend to be successfully reconstructed at earlier time steps compared to non-members. Consequently, it remains possible for a stronger attack (e.g., with carefully tuned hyperparameters) to detect these samples.

We also studied the impact of various truncation step options on the attack performance. As shown in Figure D.2(b) (in the Appendix), the optimal choice remains largely stable across different training setups (see detailed quantitative results in Table D.4). The results indicate that improvements can be achieved with a wide range of reasonable choices. Based on these findings, we set the default truncation step to be $0.75T$ and the "*Median*" statistical function as the default for evaluating gray-box attacks with truncation techniques.

Adaptive Defenses. While some of the diffusion APIs expose all the relevant hyperparameters for generation (and potential attack) and allow controllable synthesis, model owners may decide to withhold certain information to protect commercial interests and preserve privacy, which creates extra challenges for attackers. We study the impact of withholding hyperparameters in diffusion model APIs from an adaptive defense perspective.

We take a more in-depth investigation into the case where the adopted scheduler of α_t is not accessible. While the official implementation supports two scheduler options, we evaluate our attack performance by using a different scheduler than the one used to train the target model. This simulates a worst-case scenario where the attacker guesses the hyperparameter incorrectly. Our results show that our gray-box attacks remain effective even when using a different scheduler (see Table 5.4), with AUCROC values of 0.91 and 0.65 for datasets of 5k and 20k, respectively. These results suggest that even with a different scheduler, samples can still be mapped to descriptive embeddings in the latent space, revealing information for attack during the reverse generation process. Additionally, as there are only a few options for the scheduler and the forward process is largely the same or highly similar for most diffusion models, it is likely that the attacker can guess the correct scheduler. This implies that withholding the scheduler may not eliminate the privacy risk.

	5k				20k			
	Median	Sum	Min	Max	Median	Sum	Min	Max
w/o	0.62	0.56	0.56	0.49	0.53	0.50	0.49	0.50
w	**0.91**	0.65	0.56	0.52	**0.65**	0.54	0.49	0.50

Table 5.4: The **gray-box** attack AUCROC on CelebA under wrong guessing of (the scheduler for) α_t with ("w") or without ("wo") applying the truncation technique. The truncation step is set to be the default value $T_{trun} = 0.75T$. We highlight the best performance in each configuration in bold.

We also consider the case where the model owner may choose to suppress the intermediate outputs. As demonstrated in Table 5.5, even when 75% or 50% of the intermediate outputs during the reverse generation process are suppressed, our attack remains highly effective. While such suppression reduces the amount of information leaked to the public, thus diminishing potential risks, we posit that a well-designed attack using appropriate statistical techniques can still be successful. This premise is supported by an examination of the loss term distribution: as shown in Figure 5.1(b), an attack can exploit a certain range of the discriminative region. Even with substantial suppression, if the attacker can extract a subset of the intermediate outputs within such region, an inference attack can still be successfully executed.

	75%				50%			
	Median	Sum	Min	Max	Median	Sum	Min	Max
w/o	**1.00**	0.69	0.61	0.51	**1.00**	0.69	0.57	0.50
w	**1.00**	0.94	0.61	0.50	**1.00**	0.94	0.57	0.50

Table 5.5: The **gray-box** attack AUCROC on CelebA with 5k training samples when the model owner suppresses the intermediate output (with 75% or 50% suppression ratio). We evaluate both with ("w") and without ("wo") the truncation technique, where the truncation step is always set to $T_{trun} = 0.75T$. We highlight the best performance in each configuration in bold.

5.5.4 Evaluation on Black-box Attack

Model-Specific Attack and Cross-model Generalization. The model owner may decide to hide intermediate results when deploying an API, limiting the attacker's access to only the final synthetic output and limiting control over the generation process. In extreme cases, the attacker may only have access to the final synthetic output without any intermediate results. However, such APIs may still provide clues about the underlying model used [172, 182]. In such cases, training a shadow model as a proxy of the target and conducting the attack on the shadow model would be a good strategy.

With a proxy model in hand, the attacker can apply either white-box or gray-box attack techniques discussed previously. We present results using the default settings for both white-box and gray-box attacks with truncation in Figure 5.4(c). We observe that the attack performance decreases as the dataset size increases, but reasonable levels of AUCROC values above 0.6 are still generally obtained. The performance of the gray-box and white-box attacks is generally comparable. By default, we use the gray-box attack with truncation for further evaluation due to its overall stability.

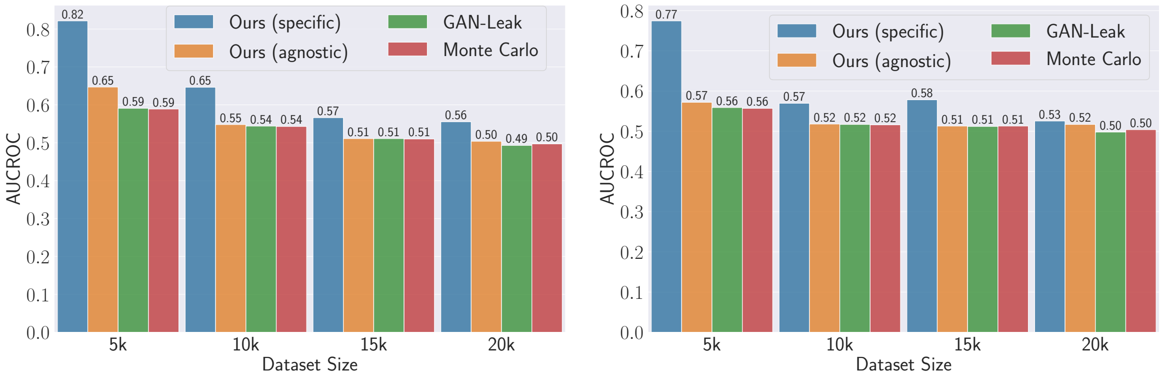
We take a step further into the investigation of the cross-architecture generalization of our model-specific black-box attack. Specifically, we study the scenario where the shadow model has a different architecture and may adopt a different setup of the key hyperparameters than the target model. As seen in Table 5.6, there is a slight decrease in attack AUCROC (0.77) compared to when the shadow model and target model have the same architecture (AUCROC 0.82). Furthermore, changing the key hyperparameter (the denoising step in our case) also results in a slight decrease in AUCROC from 0.77 to 0.73, but the change is not substantial. This aligns with previous research findings, as changing the architecture can cause the shadow model to be less similar to the target model.

However, the difference in architecture may have less impact on attacks against generative models compared to classification models. In the black-box scenario, membership information in generative models is completely contained in the generated distribution, which can still be captured by a shadow model with a different architecture. In contrast, for classification models, membership information is mainly represented by their specific responses to each query, which can vary greatly between models with different architectures. Therefore, attacks based on shadow models remain relatively effective in cross-architecture scenarios for generative models, unlike MIAs against classification models that tend to become less effective [74, 188].

Diffusion Steps	AUCROC	Accuracy	F1-Score	TPR@1%FPR
2000	0.73	0.68	0.68	5.99%
4000	0.77	0.69	0.69	7.34%
6000	0.76	0.68	0.68	6.77%

Table 5.6: The **black-box** attack performance when attacking a *guided diffusion model* with an *improved diffusion model* as the shadow model. Different settings of the diffusion steps are considered when training the shadow model. The experiments were carried out on the CelebA dataset with 5k training samples.

Model-based vs. Model-agnostic. For the least informed attack scenario, attackers would have to rely on a model-agnostic method, i.e., they cannot use any extra knowledge of the target model except blindly collecting generated samples from it. In this case, we build upon existing methods that calculate the distance between the query sample and generated samples (closer distance indicates higher membership probability). We improve upon these methods by enhancing the distance metrics, i.e., we use a pre-trained ImageNet classifier as a feature extractor and compute the cosine distance between features as the metrics. Our modification leverages the rich semantic information from the pre-trained feature extractor to improve the discriminative power of the resulting membership score. The comparisons to existing methods are shown in Figures 5.6(a) and 5.6(b). As shown, our model-agnostic attack performs slightly better than existing methods, while our model-specific attack greatly improves by leveraging slightly more information that is always freely available even in a black-box setting.



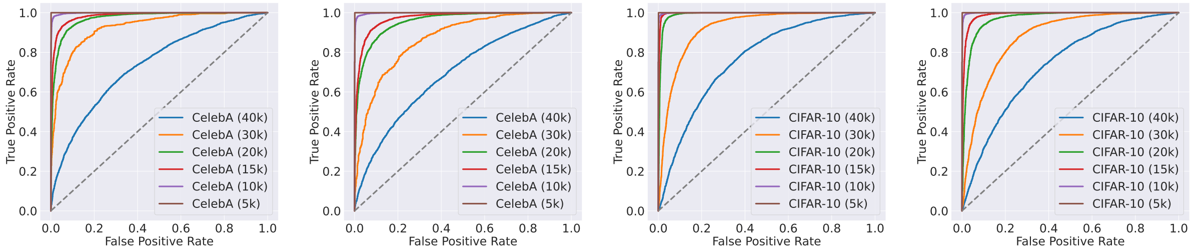
(a) CelebA

(b) CIFAR-10

Figure 5.6: The **black-box** attack AUCROC on 5.6(a) CelebA and 5.6(b) CIFAR-10, respectively. We adopt the gray-box attack (with truncation) on the shadow model for our model-specific attack.

5.5.5 Exploring Larger Dataset Sizes

While previous studies mainly reported effective attacks on datasets with no more than 20k training data samples, our attack performance has not yet reached its saturation point in such scenarios, maintaining an AUCROC of over 0.95 (for both gray-box and white-box cases). To



(a) white-box (CelebA) (b) gray-box (CelebA) (c) white-box (CIFAR-10) (d) gray-box (CIFAR-10)

Figure 5.7: ROC curves of our **white-box** and **gray-box** (with truncation techniques) attacks on CelebA and CIFAR-10.

Dataset	Dataset Size	AUCROC	TPR@0.1%FPR	TPR@1%FPR
CIFAR-10	20k	0.97	8.34%	41.57%
	30k	0.87	1.52%	10.06%
	40k	0.73	0.36%	4.48%
CelebA	20k	0.98	15.22%	51.06%
	30k	0.91	8.19%	34.14%
	40k	0.69	1.18%	6.06%

Table 5.7: The **gray-box** attack performance against guided diffusion models with varying dataset size.

push the boundaries of our approach, we further experimented with guided diffusion models trained with a larger size of training data (up to the maximum effective size for the adopted benchmarking datasets that still leaves a sufficient amount of data for MIA evaluation). As presented in Table 5.7, our approach is still capable of extracting membership information, achieving an AUCROC that generally exceeds 0.7. Furthermore, the attack maintains a high TPR under specific low FPR thresholds (0.1% and 1%). Specifically, the TPR is always substantially higher than the FPR at least three times greater, indicating that the attack could successfully identify significantly more member samples than it misclassified non-members as members (refer to Figure 5.7(a) for the complete ROC curves), which points to a strong attack. Notably, an MIA is deemed successful "if it can reliably violate the privacy of even just a few users in a sensitive dataset" [25].

5.5.6 MIA Comparison between Diffusion Models and GANs

Our results presented in previous sections suggest that diffusion models may be generally more vulnerable to MIAs in comparison to other popular generative models such as GANs under various attack scenarios. In this section, we closely examine their behavior under MIAs in a similar setting. We train and evaluate both diffusion and GAN models on the same sample set using the same attack method (i.e., the model-agnostic black-box attack from [34]). We consider the widely used models that represent the state-of-the-art or previous state-of-the-art at their release time. These include two GAN models, namely the StyleGAN [94] and progressive GAN (PGGAN) [92], and the "Guided" and "Improved" diffusion models. The results in Table 5.8 show that diffusion models have higher attack AUCROC compared to GAN models, with PGGAN having 0.56 and Improved Diffusion having 0.62, while the difference in the TPR@1%FPR becomes much more significant. These also support that diffusion models tend to be more vulnerable to MIA attacks than GAN models, even when considering only model-agnostic attacks, not to mention the exceptionally high privacy risk when our dedicated approaches are applied.

	Guided	Improved	StyleGAN	PGGAN
FID	22.46	24.78	25.89	55.28
AUCROC	0.59	0.62	0.51	0.57
TPR@1%FPR	5.14%	6.11%	1.02%	4.04%

Table 5.8: The generation quality (FID) and attack performance (AUCROC, TPR@1%FPR) of various generative models trained on the CelebA dataset with 5k samples. The model-agnostic attack from [34] was evaluated. The FID was calculated based on 40k generated images from each generative model.

5.6 Discussion

In this section, we highlight several key insights and their practical implications, as well as discuss possible concerns regarding our attack formulation.

5.6.1 Comparison with Concurrent Works

While our study covers a broad spectrum, which subsumes several existing and concurrent works as discussed in Section 5.2, we also demonstrate superior advantages in terms of attack effectiveness and algorithmic design. Table 5.9 and 5.10 present results on benchmark datasets with 20k samples. Our approach consistently outperforms other methods by a noticeable margin, even including those unpublished works that offer comparable evaluations, across all metrics.

Method-wise, our approach also stands out in its practicability and modeling superiority. For comparison, [27] necessitates the training of >10 shadow models, with each using 50% of the target dataset (>10k samples), rendering it notably sample-inefficient. The methods of [50, 103] rely on the reconstruction loss at a specific intermediate step as the membership score, but with hyperparameters that require extensive tuning and are non-transferable across datasets. Both [82] and [136] align with our sub-optimal attack setup, mirroring our vanilla gray-box ("min" and "sum") and white-box ("sum") attacks without truncation. [237] employs the final reconstruction loss as the membership score and requires a shadow dataset with over 3k samples for training its attack predictor. Remarkably, our proposed approach surpasses methods like [26, 50, 103, 82, 136], offering clearly better practicability than [26] and [82]. Moreover, our method possesses hyperparameters that are easily transferable between datasets, needing <200 shadow samples from a relevant distribution for adjustment.

5.6.2 Privacy Risks of Diffusion Models

Our results in Section 5.5 (particularly in Section 5.5.6) show that MIAs have a notably higher attack success rate when targeting diffusion models compared to other popular generation models like GANs across various attack scenarios. This possibly can mainly be attributed to the objective used by diffusion models. The objective, which is to maximize the log-likelihood lower bound on all training samples, can result in a loss landscape that locally minimizes the loss around each training sample, potentially leading to a spike in the distribution if not

Dataset	Method	AUCROC	TPR@0.1%FPR	TPR@1%FPR
CIFAR-10	[27]	0.98	27.93%	90.63%
	[82]	0.93	19.98%	88.47%
	[136]	0.93	20.10%	87.99%
	Ours	**0.98**	**30.68%**	**93.46%**
CelebA	[27]	0.97	30.97%	92.27%
	[82]	0.92	23.01%	89.21%
	[136]	0.91	21.08%	87.19%
	Ours	**0.98**	**33.49%**	**95.12%**

Table 5.9: Comparison to existing works under **white-box** setting.

Dataset	Method	AUCROC	TPR@0.1%FPR	TPR@1%FPR
CIFAR-10	[50]	0.91	4.19%	27.32%
	[82]	0.86	0.82%	10.09%
	[103]	0.93	5.21%	34.21%
	Ours	**0.98**	**8.34%**	**41.57%**
CelebA	[50]	0.96	13.98%	48.76%
	[82]	0.92	9.98%	36.91%
	[103]	0.94	12.06%	40.24%
	Ours	**0.98**	**15.22%**	**51.06%**

Table 5.10: Comparison to existing works under **gray-box** knowledgeable setting.

properly regularized. This inevitably leaves clues for attackers to successfully conduct their attacks. In contrast, the adversarial objective used in GANs indirectly guides the generator to produce samples that resemble the training data, while also preventing exact memorization through adversarial updates. These indicate that diffusion models may intrinsically pose a higher privacy risk and should be used with caution in real-world applications, especially considering their widespread use as a standard media generation tool.

Additionally, it is relatively easy to reduce privacy risk for other generative models by only releasing the functional part (e.g. the generator) and keeping the unnecessary part (e.g. the encoder in VAEs or the discriminator in GANs) private [34]. However, this is normally not the case for diffusion models, since the unnecessary part of diffusion models (the forward process) is fixed, unlearned, identical or highly similar across models and settings, making it easy for the attacker to guess and mimic the real process. As a result, an attack generally requires less effort to associate each query sample with its latent variable and estimate the likelihood needed for the attack. This can be seen in the qualitative results shown in Figure 5.5: the reconstruction is more accurate when compared to previous cases that required gradient-based optimization [34]. This characteristic of diffusion models thus poses higher potential risks in deployment scenarios.

5.6.3 Conditional Generation

Our approach can be seamlessly extended to conditional generation models, such as the text-to-image stable diffusion models. Specifically, we consider each query sample as a combination of an image and its accompanying text description. We extract multiple images generated at various diffusion steps from the target diffusion models for the query text and derive the membership score via Equation 5.18. However, the loss is computed only on the image component $\boldsymbol{x}^i_{\text{img}}$, and the text $\boldsymbol{x}^i_{\text{text}}$ serves as an additional input to the model. We investigate two publicly released pre-trained models, stable-diffusion-v1.4 and stable-diffusion-v1.5, both from the official Huggingface repository. We use the inference implementation without any modifications to collect the generated images at different steps, to simulate the gray-box setting. For evaluation, we use COCO-2017 [121] as the non-member set, which exhibits a distribution similar to the member set (i.e., Laion2B-improved-aesthetics [186]) used by the pre-trained stable diffusion models. All images are resized to a resolution of 512×512 for evaluation. As Table 5.11 demonstrates, our approach effectively extracts membership information from these real-world diffusion models, trained on large-scale datasets, specifically, with 2.3 billion samples. Remarkably, our method maintains a significant level of TPR (exceeding 24%) even at a low FPR of 1%. This demonstrates its potential in accurately tracing the usage of specific samples during the training of a diffusion model, while concurrently highlighting the privacy risks associated with potential training data leakage when deploying or sharing such models that operate on sensitive data.

Models	AUCROC	F1-Score	TPR@1%FPR
stable-diffusion-v1.4	0.73	0.71	24.21 %
stable-diffusion-v1.5	0.74	0.71	25.66 %

Table 5.11: Our attack performance on the pre-trained stable diffusion models.

5.6.4 Potential Defenses

As presented in Section 5.5.4, limiting the information available to attackers is generally effective in protecting against such attacks. A slight decrease in attack performance can occur when the model developer hides important parameters, causing the attacker to make incorrect guesses, while a larger degradation happens when the model owner further prevents controllable generation (comparing gray-box to black-box model-specific attacks) and even obscures the sources of synthetic samples (comparing model-specific to model-agnostic attacks). However, these measures can come at the cost of a degraded user experience and may not be a sustainable solution.

Providing rigorous privacy guarantees is another option for the defense. Differential privacy (DP) [53] is a widely used technique that ensures protection against privacy attacks. To prevent privacy leakage from machine learning models, DP incorporates adding random noise to the gradients during training to reduce the impact of each individual sample on the model parameter and thus hide the presence of the data in the training set [1]. However, DP training inevitably hampers the model utility and significantly increases the computational cost during training. While notable recent progress has been achieved in developing DP generative models, these advancements are largely limited to simple datasets like MNIST and Fashion-MNIST and do not offer a practical solution for the complex datasets considered in this work (for example, see generation results in [47]). We believe a more in-depth investigation into developing efficient and effective defense mechanisms for MIA on diffusion models is required but leave it as future work as it is orthogonal to our contributions in this work.

5.7 Conclusion

In this work, we present the first systematic analysis of membership inference attacks against diffusion models. Our study presents, for the first time, the key attack vectors that are particularly relevant for real-world deployment scenarios of diffusion models. Moreover, we propose our novel attack approaches tailored to each attack scenario. Our methods exploit readily available information while delivering promising performance across a broad range of settings, thereby demonstrating high potential for application scenarios that necessitate accurate auditing of data usage when developing and deploying diffusion models. Our findings, coupled with our insights, highlight the high potential privacy risks associated with diffusion models, an area we believe warrants further exploration. To facilitate future research in this field, the source code implementation will be made openly available upon publication.

6 RelaxLoss: Defending Membership Inference Attacks without Losing Utility

Contents

As a long-term threat to the privacy of training data, membership inference attacks (MIAs) emerge ubiquitously in machine learning models. Existing works evidence strong connection between the distinguishability of the training and testing loss distributions and the model's vulnerability to MIAs. Motivated by existing results, we propose a novel training framework based on a *relaxed loss* (**RelaxLoss**) with a more achievable learning target, which leads to narrowed generalization gap and reduced privacy leakage. RelaxLoss is applicable to any classification model with added benefits of easy implementation and negligible overhead. Through extensive evaluations on five datasets with diverse modalities (images, medical data, transaction records), our approach consistently outperforms state-of-the-art defense mechanisms in terms of resilience against MIAs as well as model utility. Our defense is the first that can withstand a wide range of attacks while preserving (or even improving) the target model's utility.

This chapter is based on [33]: As the first author of [33], Dingfan Chen proposed the project idea, implemented the algorithms, conducted all experiments, and served as the main writer of the paper. This paper was selected as a spotlight paper in ICLR 2022 (among the top 20% of the accepted papers). The source code is available on Github [1]

6.1 Introduction

While deep learning (DL) models have achieved tremendous success in the past few years, their deployments in many sensitive domains (e.g., medical, financial) bring privacy concerns since data misuse in these domains induces severe privacy risks to individuals. In particular, modern deep neural networks (NN) are prone to memorize training data due to their high

[1] https://github.com/DingfanChen/RelaxLoss

capacity, making them vulnerable to privacy attacks that extract detailed information about the individuals from models [188, 193, 244] .

In membership inference attack (MIA), an adversary attempts to identify whether a specific data sample was used to train a target victim model. This threat is pervasive in various data domains (e.g., images, medical data, transaction records) and inevitably poses serious privacy threats to individuals [188, 148, 183], even given only black-box access (query inputs in, posterior predictions out) [188, 183, 195] or partially observed output predictions (e.g., top-k predicted labels) [38].

Significant advances have been achieved to defend against MIAs. Conventionally, regularization methods designed for mitigating overfitting such as dropout [199] and weight-decay [60] are regarded as defense mechanisms [183, 87, 188]. However, as conveyed by [98, 97], vanilla regularization techniques (which are not designed for MIA), despite slight improvement towards reducing the generalization gap, are generally unable to eliminate MIA. In contrast, recent works design defenses tailored to MIA. A common strategy among such defenses is adversarial training [66, 65], where a surrogate attack model (represented as a NN) is used to approximate the real attack and subsequently the target model is modified to maximize prediction errors of the surrogate attacker via adversarial training. This strategy contributes to remarkable success in defending NN-based attacks [148, 87]. However, these methods are greatly restricted by strong assumptions on attack models, thereby failing to generalize to novel attacks unanticipated by the defender (e.g., a simple metric-based attack) [195]. In order to defend attacks beyond the surrogate one, differentially private (DP) training techniques [1, 158, 159] that provide strict guarantees against MIA are exploited. Nevertheless, as evidenced by [170, 87, 74, 86, 34, 97], incorporating DP constraints inevitably compromises model utility and increases computation cost.

In this paper, we present an effective defense against MIAs while avoiding negative impacts on the defender's model utility. Our approach is built on two main insights: *(i)* the optimal attack only depends on the sample loss under mild assumptions of the model parameters [180]; *(ii)* a large difference between the training loss and the testing loss provably causes high membership privacy risks [244]. By intentionally 'relaxing' the target training loss to a level which is more achievable for the test loss, our approach narrows the loss gap and reduces the distinguishability between the training and testing loss distributions, effectively preventing various types of attacks in practice. Moreover, our approach allows for a utility-preserving (or even improving) defense, greatly improving upon previous results. As a practical benefit, our approach is easy to implement and can be integrated into any classification models with minimal overhead.

Contributions. In summary, we make the following contributions:

- We propose **RelaxLoss**, a simple yet effective defense mechanism to strengthen a target model's resilience against MIAs without degrading its utility. To the best of our knowledge, our approach for the first time addresses a wide range of attacks while preserving (or even improving) the model utility.

- We derive our method from a Bayesian optimal attacker and provide both empirical and analytical evidence supporting the main principles of our approach.

- Extensive evaluations on five datasets with diverse modalities demonstrate that our method outperforms state-of-the-art approaches by a large margin in membership inference protection and privacy-utility trade-off.

6.2 Related Work

Membership Inference Attack. Inferring membership information from deep NNs has been investigated in various application scenarios, ranging from the white-box setting where the whole target model is released [149, 174] to the black-box setting where the complete/partial output predictions are accessible to the adversary [188, 183, 244, 180, 195, 38, 84, 217]. An adversary first determines the most informative features (depending on the application scenarios) that faithfully reflect the sample membership (e.g., logits/posterior predictions [188, 183, 87], loss values [244, 180], and gradient norms [149, 174]), and subsequently extracts common patterns in these features among the training samples for identifying membership. In this work, we work towards an effective defense by suppressing the common patterns that an optimal attack relies on.

Defense. Existing defense mechanisms against MIA are mainly divided into three main categories: *(i)* regularization techniques to alleviate model overfitting, *(ii)* adversarial training to confuse surrogate attackers, and *(iii)* a differentially private mechanism offering rigorous privacy guarantees. Our proposed approach can be regarded as a regularization technique owing to its effect in reducing generalization gap. Unlike previous regularization techniques, our method is explicitly tailored towards defending MIAs by reducing the information that an attacker can exploit, leading to significantly better defense effectiveness. Algorithmically, our approach shares similarity with techniques that suppress the target model's confidence score predictions (e.g., label-smoothing [68, 146] and confidence-penalty [163]), but ours is fundamentally different in the sense that we modulate the loss distribution with gradient ascent.

Previous state-of-the-art defense mechanisms against MIA, such as Memguard [87] and Adversarial Regularization [148], are built on top of the idea of adversarial training [66, 65]. Such approaches usually rely on strong assumptions about attack models, making their effectiveness highly dependent on the similarity between the surrogate and the real attacker [195]. In contrast, our method does not rely on any assumptions about the attack model, and has shown consistent effectiveness across different attacker types.

Differential privacy [51, 53, 1, 158] provides strict worst-case guarantees against arbitrarily powerful attackers that exceed practical limits, but inevitably sacrifices model utility [170, 87, 74, 34, 97, 86] and meanwhile increases computation burden [64, 41]. In contrast, we focus on practically realizable attacks for utility-preserving and computationally efficient defense.

6.3 Preliminaries

Notations. We denote by $\boldsymbol{z}_i = (\boldsymbol{x}_i, \boldsymbol{y}_i)$ one data sample, where $\boldsymbol{x}_i$ and $\boldsymbol{y}_i$ are the feature and the one-hot label vector, respectively. $f(\cdot\,; \boldsymbol{\theta})$ represents a classification model parametrized by $\boldsymbol{\theta}$, and $\boldsymbol{p} = f(\boldsymbol{x}; \boldsymbol{\theta}) \in [0,1]^C$ denotes the predicted posterior scores (after the final softmax layer) where C denotes the number of classes. $\mathbb{1}$ denotes the indicator function, i.e., $\mathbb{1}[p]$ equals 1 if the predicate p is true, else 0. We use subscripts for sample index and superscripts for class index.

Attacker's Assumptions. We consider the standard setting of MIA: the attacker has access to a query set $\mathcal{S} = \{(\boldsymbol{z}_i, m_i)\}_{i=1}^N$ containing both member (training) and non-member (testing) samples drawn from the same data distribution P_{data}, where m_i is the membership attribute ($m_i = 1$ if $\boldsymbol{z}_i$ is a member). The task is to infer the value of the membership attribute m_i

associated with each query sample $\boldsymbol{z}_i$. We design defense for a general attack with full access to the target model. The attack $\mathcal{A}(\boldsymbol{z}_i, f(\cdot;\boldsymbol{\theta}))$ is a binary classifier which predicts m_i for a given query sample $\boldsymbol{z}_i$ and a target model parametrized by $\boldsymbol{\theta}$. The Bayes optimal attack $\mathcal{A}_{opt}(\boldsymbol{z}_i, f(\cdot;\boldsymbol{\theta}))$ will output 1 if the query sample is more likely to be contained in the training set, based on the real underlying membership probability $P(m_i = 1|\boldsymbol{z}_i, \boldsymbol{\theta})$, which is usually formulated as a non-negative log ratio:

$$\mathcal{A}_{opt}(\boldsymbol{z}_i, f(\cdot;\boldsymbol{\theta})) = \mathbb{1}\left[\log \frac{P(m_i = 1|\boldsymbol{z}_i, \boldsymbol{\theta})}{P(m_i = 0|\boldsymbol{z}_i, \boldsymbol{\theta})} \geq 0\right] \tag{6.1}$$

Defender's Assumptions. We closely mimic an assumption-free scenario in designing our defense method. In particular, we consider a knowledge-limited defender which: *(i)* does not have access to additional public (unlabelled) training data (in contrast to [158, 159]); and *(ii)* lacks prior knowledge of the attack strategy (in contrast to [87, 148]). For added rigor, we also study attacker's countermeasures to our defense in Section 6.6.4.

6.4 RelaxLoss

The ultimate goal of the defender is two-fold: *(i) privacy:* reducing distinguishability of member and non-member samples; *(ii) utility:* avoiding the sacrifice of the target model's performance. We hereby introduce each component of our method targeting at privacy (Section 6.4.1) and utility (Section 6.4.2).

6.4.1 Privacy: Reduce Distinguishability via Relaxed Target Loss

We begin by exploiting the dependence of attack success rate on the sample loss [244, 180]. In particular, a large gap in the expected loss values on the member and non-member data, i.e., $\mathbb{E}[\ell]_{\text{non}} - \mathbb{E}[\ell]_{\text{mem}}$, is proved to be sufficient for conducting attacks [244]. Along this line, [180] further show that the Bayes optimal attack only depends on the sample loss under a mild posterior assumption of the model parameter: $\mathcal{P}(\boldsymbol{\theta}|\boldsymbol{z}_1, .., \boldsymbol{z}_n) \propto e^{-\frac{1}{T}\sum_{i=1}^{N} m_i \cdot \ell(\boldsymbol{\theta}, \boldsymbol{z}_i)}$[2]. Formally,

$$\mathcal{A}_{opt}(\boldsymbol{z}_i, f(\cdot;\boldsymbol{\theta})) = \mathbb{1}[-\ell(\boldsymbol{\theta}, \boldsymbol{z}_i) > \tau(\boldsymbol{z}_i)] \tag{6.2}$$

where τ denotes a threshold function [3]. Intuitively, Equation 6.2 shows that $\boldsymbol{z}_i$ is likely to be used for training if the target model exhibits small loss value on it. These results motivate our approach to mitigate MIAs by reducing distinguishablitiy between the member and non-member loss distributions.

Relaxing Loss Target with Gradient Ascent Directly operating on member and non-member loss distributions, however, is impractical, since the exact distributions are intractable and a large amount of additional hold-out samples are required for estimating the distribution of non-member data. In order to bypass these issues and reduce the distinguishability between the member and non-member loss distributions, we propose to simplify the problem by considering the mean of the loss distributions, and subsequently set a *more achievable* mean

[2] This corresponds to a Bayesian perspective, i.e., $\boldsymbol{\theta}$ is regarded as a random variable that minimizes the empirical risk $\sum_{i=1}^{N} m_i \cdot \ell(\boldsymbol{\theta}, \boldsymbol{z}_i)$. T is the temperature that captures the stochasticity.

[3] We summarize both the strategy MALT and MAST from [180] in Equation 6.2, where τ is a constant function for MALT.

value for the target loss, where the loss target is relaxed to a level that is easier to be achieved for the non-member data.

Algorithmically, instead of pursuing zero training loss of the target model, we relax the target mean loss value α to be larger than zero and apply a *gradient ascent* step as long as the average loss of the current batch is smaller than α.

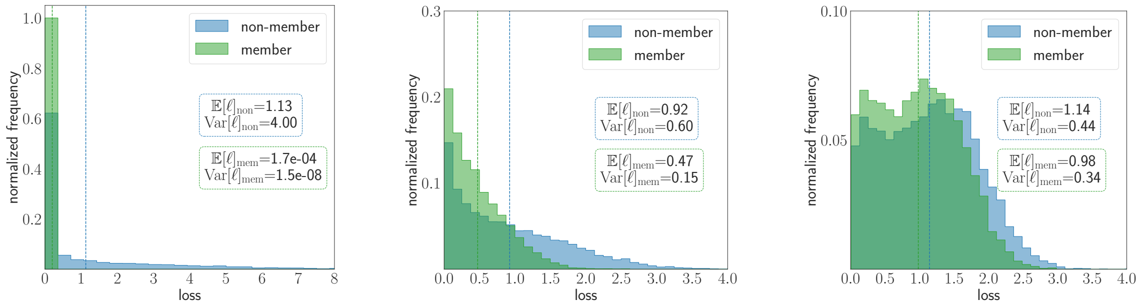
Algorithm 2: RelaxLoss

```
Input: Dataset $\{(\boldsymbol{x}_i, \boldsymbol{y}_i)\}_{i=1}^{N}$, training epochs $E$, learning rates $\tau$, batch size $B$, number of
       output classes $C$, target loss value $\alpha$
Output: Model $f(\cdot;\boldsymbol{\theta})$ with parameters $\boldsymbol{\theta}$
Initialize model parameter $\boldsymbol{\theta}$ ;
for epoch in $\{1, ..., E\}$ do
    for batch_index in $\{1, ..., K\}$ do
        Get sample batch $\{(\boldsymbol{x}_i, \boldsymbol{y}_i)\}_{i=1}^{B}$
        Perform forward pass: $\boldsymbol{p}_i = f(\boldsymbol{x}_i; \boldsymbol{\theta})$
        Compute cross entropy loss $\mathcal{L}(\boldsymbol{\theta})$ on the batch
        if $\mathcal{L}(\boldsymbol{\theta}) \geq \alpha$ then
            // gradient descent
            $\boldsymbol{\theta} \leftarrow \boldsymbol{\theta} - \tau \cdot \nabla\mathcal{L}(\boldsymbol{\theta})$
        else
            if epoch $\%2 = 0$ then
                // gradient ascent
                $\boldsymbol{\theta} \leftarrow \boldsymbol{\theta} + \tau \cdot \nabla\mathcal{L}(\boldsymbol{\theta})$
            else
                // posterior flattening
                Construct softlabel $\boldsymbol{t}_i$ with
                    $t_i^c = \begin{cases} p_i^c & \text{if } y_i^c = 1 \\ (1 - p_i^c)/(C-1) & \text{otherwise} \end{cases}$
                Compute cross entropy loss with the softlabel: [a])
                 $\ell(\boldsymbol{\theta}, \boldsymbol{z}_i) = -\sum_{c=1}^{C} \mathrm{sg}[t_i^c] \log p_i^c$ $\mathcal{L}(\boldsymbol{\theta}) = \frac{1}{B}\sum_{i=1}^{B} \ell(\boldsymbol{\theta}, \boldsymbol{z}_i)$
                Update model parameters: $\boldsymbol{\theta} \leftarrow \boldsymbol{\theta} - \tau\nabla\mathcal{L}(\boldsymbol{\theta})$
            end
        end
    end
end
return model $f(\cdot\,;\boldsymbol{\theta})$
```

[a] sg stands for the stopgradient operator that is defined as identity at forward pass and has zero partial derivatives, i.e., $\boldsymbol{t}_i$ is a non-updated constant.

6.4.2 Utility: Apply Posterior Flattening and Normal Gradient Operations

With the relaxed target loss, the predicted posterior score of the ground-truth class p^{gt} is no longer maximized towards 1. If the probability mass of all non-ground-truth classes $1 - p^{gt}$ concentrates on only few of them (e.g., hard samples that are close to the decision boundary between two classes), it is very likely that one non-ground-truth class has a score larger than



(a) Vanilla

(b) Ours ($\alpha = 0.5$)

(c) Ours ($\alpha = 1.0$)

Figure 6.1: Loss histograms on CIFAR-10 with ResNet20 architecture when applying (a) vanilla training, (b) our method with $\alpha = 0.5$, and (c) our method with $\alpha = 1.0$. The empirical mean and variance of the loss distributions are shown in the figure. The AUC of a loss thresholding attack equals to 0.84 in (a), 0.67 in (b), and 0.57 in (c). We observe that our method fits the target mean (Section 6.4.1), increases the variance of the training loss distribution and reduces the distinguishability between member and non-member distributions (Section 6.5).

p^{gt} (i.e., $\max_{c,c \neq gt} p^c > p^{gt}$), thus leading to incorrect predictions. To address this issue, we propose to encourage a large margin between the prediction score of the ground-truth-class and the others by *flattening the target posterior* scores for non-ground-truth classes. Specifically, we dynamically construct softlabels during each epoch by: *(i)* retaining the score of the ground-truth class, i.e., the current predicted value p^{gt}, and *(ii)* re-allocating the remaining probability mass evenly to all non-ground-truth classes.

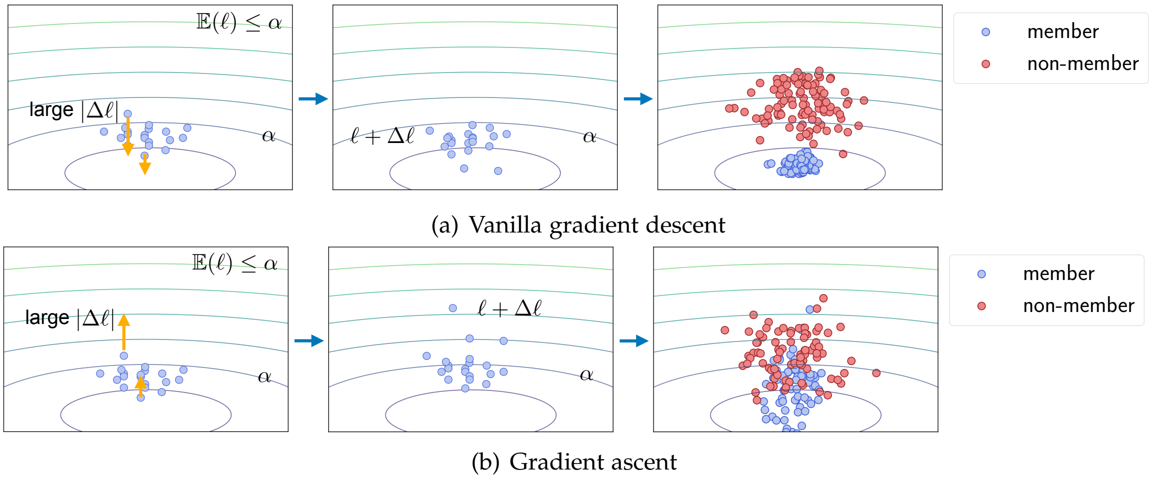
In summary, we run a repetitive training strategy to balance privacy and utility, which consists of two steps: *(i)* if the model is not well-trained, i.e., the current loss is larger than the target mean value α, we run a normal gradient descent step; *(ii)* otherwise, we apply gradient ascent or the posterior flattening step (See Algorithm 2).

6.5 Analytical Insights

In this section, we analyze the key properties that explain the effectiveness of RelaxLoss. We provide both analytical and empirical evidence showing that RelaxLoss can *(i)* reduce the generalization gap, and *(ii)* increase the variance of the training loss distribution, both contributing to mitigating MIAs.

RelaxLoss reduce the generalization gap. We apply RelaxLoss to CIFAR-10 dataset and plot the resulting loss histograms in Figure 6.1. With a more achievable learning target, RelaxLoss blurs the gap between the member and non-member loss distributions (Figure 6.1), which naturally leads to a narrowed generalization gap (Appendix Figure E.1) and reduced privacy leakage [244].

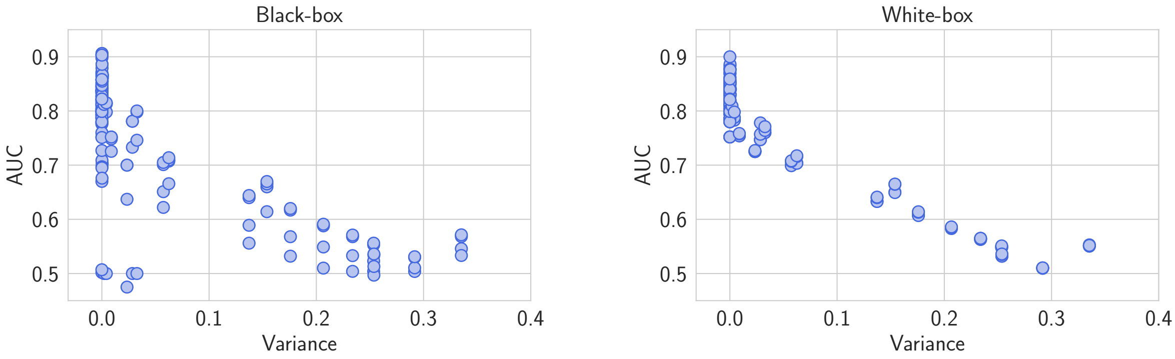
RelaxLoss increases the variance of the training loss distribution. We observe that RelaxLoss spreads out the training loss distribution (i.e., increase the variance) due to its gradient ascent step (Appendix E.1.1): large loss samples tend to have a more significant increase in its loss value during the gradient ascent step (See Figure 6.2 for demonstration). In contrast, except DP-SGD, existing defense methods do not have this property (Appendix E.3.9). Intuition suggests that the increase of training loss variance suppresses the common pattern among training losses and reduces the information that can be exploited by an attacker, thus contributing to the protection against attacks. To verify the association between the variance increasing



(a) Vanilla gradient descent

(b) Gradient ascent

Figure 6.2: Comparison between vanilla gradient descent and the gradient ascent step in RelaxLoss (demonstrated in 2D). The loss contour lines are plotted in the figure, the *bottom* part of which corresponds to a *low* loss region. The target loss level α is visualized in the figure. Training with vanilla gradient descent step results in near zero loss for member samples, and large loss values for non-member samples. In contrast, a large loss value ℓ tends to trigger large update $|\Delta\ell|$ during the gradient ascent step. As a result, RelaxLoss spreads out the training loss distribution and blurs the gap between the distributions (Section 6.5).



(a)

(b)

Figure 6.3: Correlation between the training set loss variance and the MIA performance on CIFAR-10 (ResNet20). Each point in the figure corresponds to one target model trained with different defense mechanisms. The Pearson's correlation coefficients equal to: (a) -0.77 for black-box attacks; (b) -0.94 for white-box attacks; and -0.85 if considering black-box and white-box attacks together.

effect and the defense effectiveness, we conduct experiments on CIFAR-10 dataset and measure the Pearson's correlation coefficients between the training loss variance and the attack AUC (Figure 6.3). With an overall score of -0.85 (-0.77 and -0.94 for black-box and white-box attacks, respectively), we conclude a fairly strong negative relationship. When considering a typical Gaussian assumption of the loss distributions [244, 114], we further show this variance increasing property helps to lower an upper bound of the attack AUC, and provide formal analysis in Appendix E.1.2.

6.6 Experiments

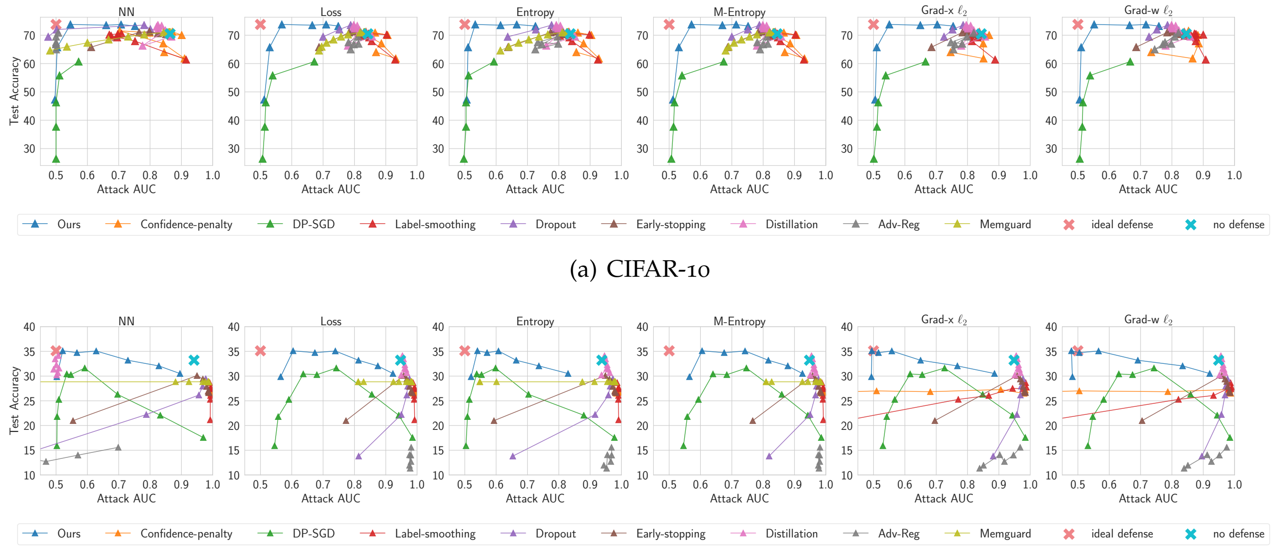
In this section, we rigorously evaluate the effectiveness of our defense across a wide range of datasets with diverse modalities, various strong attack models, and eight defense baselines representing the previous state of the art.

6.6.1 Experimental Setup

Settings. We set up seven target models, trained on five datasets (CIFAR-10, CIFAR-100, CH-MNIST, Texas100, Purchase100) with diverse modalities (natural and medical images, medical records, and shopping histories). For image datasets, we adopt a 20-layer ResNet [76] and an 11-layer VGG [190]; and for non-image datasets, we adopt MLPs with the same architecture and same training protocol as in previous works [148, 87]. We evenly split each dataset into five folds and use each fold as the training/testing set for the target/shadow model[4], and use the last fold for training the surrogate attack model (for [87, 188]). We fix the random seed and training setting for a fair comparison among different defense methods. See Appendix E.2 for implementation details.

Attack methods. To broadly handle attack methods in our defense evaluation, we consider attacks in a variety of application scenarios (*black-box* and *white-box*) and strategies. We consider the following state-of-the-art attacks from two categories: *(i) White-box attacks*: Both [149, 174] are based on gradient norm thresholding. We denote these attacks by the type of gradient followed by its norm. **Grad-x** and **Grad-w** stand for the gradient w.r.t. the *input* and the *model parameters*, respectively; *(ii) Black-box attacks*: [183] (denoted as **NN**, standing for the proposed neural network attack model). We adopt the implementation provided by [87] and use the complete logits prediction as input to the attacker. [180] (denoted as **Loss** for their loss thresholding method). We use a general threshold independent of the query sample, as the adaptive thresholding version is more expensive computational-wise with no or only marginal improvements. [195] (denoted as **Entropy** and **M-Entropy** for their proposed attack by thresholding the prediction entropy and a modified version, respectively.). We exclude attacks that only use partial output predictions (e.g., top-1 predicted label) from our evaluation as they are strictly weaker than the attacks we include above [38].

Evaluation metrics We evaluate along two fronts: utility (measured by **test accuracy** of the victim model) and privacy. For privacy, in line with previous works [195, 87, 180, 188, 148, 183], we consider the following two metrics: *(i)* attack **accuracy**: We evaluate the attack accuracy on a balanced query set, where a random guessing baseline corresponds to 50% accuracy. For threshold-based attack methods, following [195], we select the threshold value to be the one with the best attack accuracy on the shadow model and shadow dataset; *(ii)* attack **AUC**: The area under the receiver operating characteristic curve (AUC), corresponding to an integral over all possible threshold values, represents the degree of separability. A perfect defense mechanism corresponds to AUC=0.5.



(a) CIFAR-10

(b) CIFAR-100

Figure 6.4: Comparisons of all defense mechanisms on CIFAR-10 and CIFAT-100 dataset with ResNet20 architecture. Each subplot corresponds to one attack method. The x-axis corresponds to the attack AUC *(the lower the better)* while the y-axis is the target model's test accuracy *(the higher the better)*. For visualization purposes, we plot the ideal defense on the *top-left corner*, whose x-coordinate equals 0.5 and y-coordinate is set to be the highest test accuracy among all models.

6.6.2 Comparison to Baselines

Defense Baselines. We consider two state-of-the-art defense methods: **Memguard** [87] and Adversarial regularization (**Adv-Reg**) [148]. Additionally, we compare to five regularization methods: **Early-stopping**, **Dropout** [199], **Label-smoothing** [68, 146], **Confidence-penalty** [163] and (Self-)**Distillation** [79, 255]. Moreover, we compare to differential private mechanism, i.e., Differentially private stochastic gradient descent (**DP-SGD**) [1][5]. We exclude defenses that additionally require public (unlabelled) data [158, 159] for training the target model from our evaluation.

Privacy-utility trade-off. We vary the hyperparameters that best describe the privacy-utility trade-off of each method across their effective ranges (See Appendix E.2 for details) and plot the corresponding privacy-utility curves. We set the attack AUC value (*privacy*) as x-axis and the target model's performance (*utility*) as the y-axis. A better defense exhibits a privacy-utility curve approaching the top-left corner, i.e., high utility and low privacy risk. As shown in Figure 6.4(a) and 6.4(b) (and Appendix Figure E.3-E.7), we observe that our method improves the privacy-utility trade-off over baselines for almost all cases: *(i)* Previous state-of-the-art defenses (Memguard and Adv-Reg) are effective for the **NN** attack, but can hardly generalize to the other types of attack, which is also verified in [195]. Moreover, as a test-time defense, Memguard is not applicable to white-box attacks. In contrast, our method is consistently effective irrespective of the attack model and applicable to all types of attacks.

[4] Shadow models are used for training the attack models in the **NN**-based attack and selecting the optimal threshold for all metric-based attacks.

[5] In line with previous work [38], we adopt small noise scale (<0.5) for maintaining target model's utility at a decent level, which leads to meaninglessly large ϵ values.

(a)

Dataset	N_{train}
CIFAR-10	12000
CIFAR-100	12000
CH-MNIST	1000
Texas100	13466
Purchase100	39465

(b)

	CIFAR10 (ResNet20)		CIFAR10 (VGG11)		CIFAR100 (ResNet20)		CIFAR100 (VGG11)		CH-MNIST		Texas100		Purchase100	
	top-1	top-5	top-1	top-5	top-1	top-5	top-1	top-5	top-1	top-5	top-1	top-5	top-1	top-5
wo defense	70.5	96.6	73.8	97.0	33.2	63.0	41.4	67.5	77.1	99.6	52.3	82.6	89.1	99.8
with defense	73.8	98.2	74.4	97.8	35.1	67.7	41.4	69.9	78.4	99.7	55.3	86.8	89.1	99.6
Δ	4.68	1.66	0.81	0.82	5.72	7.46	0.00	3.56	1.69	0.10	5.74	5.08	0.00	-0.20

Table 6.1: (a) Size of the target model's training set. (b) Target model's test accuracy (in %) with and without (wo) applying our defense. The *relative* difference (Δ) is in % and the increase is highlighted in green and decrease in red. See Appendix E.3.2 for more details.

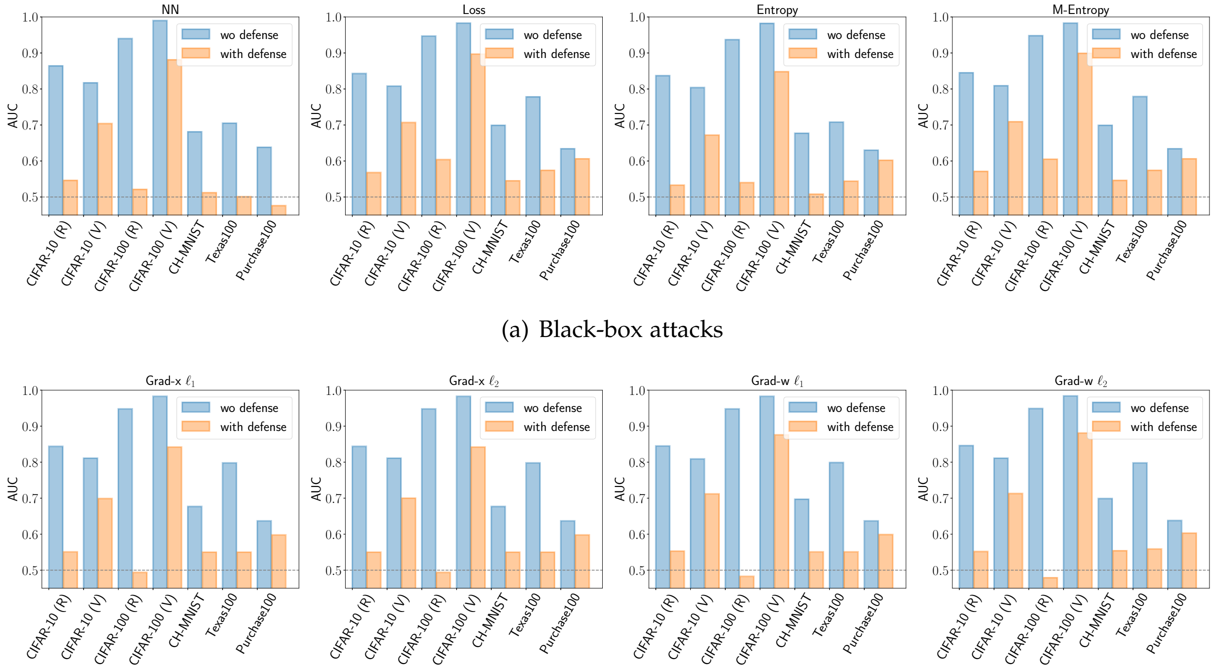
(ii) In comparison with other regularization methods, our method showcases significantly better defense effectiveness. Specifically, when compared with Early-stopping, the generally most effective regularization-based defense baseline, our approach decreases the attack AUC (i.e., relative percentage change) by up to 26% on CIFAR-10 and 46% on CIFAR-100 for a same level of utility. *(iii)* DP-SGD is generally the most effective defense baseline, despite the meaningless large ϵ values, which is consistent with [38]. In comparison with DP-SGD, our method improves the target model's test accuracy by around 16% on CIFAR-10 and 12% on CIFAR-100 (relative percentage increase) across different privacy levels. *(iv)* (See detailed results in Appendix E.3.8) Our approach is the only one that exhibit consistent defense effectiveness across various data modalities and model architectures, while the best baseline methods can only show effectiveness for at most one data modality (e.g., DP-SGD for images, and Label-smoothing for transaction records).

6.6.3 RelaxLoss Vs. Attacks

As can be seen from the privacy-utility curves in Figure 6.4 and Appendix E.3.8, our approach is the only one that can consistently defend various MIAs without sacrificing the model utility. We then evaluate *to what extend the attacks can be defended without loss in the model utility*. To this end, we select α corresponds to the model with the *lowest* privacy risk (lowest attack AUC averaged over all attacks), under the constraint that the defended model achieve a top-1 test accuracy *not worse than* the undefended model.

Utility. Table 6.1 summarizes the test accuracy of target models defended with our method. Compared with vanilla training, our method achieves a consistent improvement (up to 7.46%) in terms of utility across different datasets and model architectures, albeit a 0.2% accuracy drop for a saturated top-5 accuracy (99.6% compared to 99.8%).

Privacy. Figure 6.5 shows the membership privacy risk (**AUC**) of the target models in Table 6.1. We observe that our method is consistently effective for all types of attacks, datasets, and model architectures. In particular, our method consistently reduces the attack AUC: *(i)* to <0.6 for all non-image datasets; *(ii)* from 0.7 to 0.55 for CH-MNIST; *(iii)* from >0.8 to 0.55 for CIFAR-10 (R) and from >0.9 to <0.6 for CIFAR-100 (R). We also include the attack **accuracy** values in Appendix Table E.3, which shows our method reduces most attacks to a random-guessing level. We thus conclude that our method improves both target models' utility as well as their resilience against MIAs.



(a) Black-box attacks

(b) White-box attacks

Figure 6.5: Attack AUC on target models trained with and without (wo) applying our defense method. Each subplot is titled with the corresponding attack method's name. (R) and (V) denotes ResNet and VGG network, respectively. The corresponding target model's utility is shown in Table 6.1.

6.6.4 Adaptive Attack

We further analyze the robustness of our method against attack's countermeasures. Namely, we consider the situations where attackers have full knowledge about our defense mechanism and the selected hyperparameters, and have tailored the attacks to our defense method. We simulate the adaptive attacks by: *(i)* training shadow models with the same configuration used for training our defended target models in Table 6.1, *(ii)* simulating the adaptive attacks using the calibrated shadow models. We report the *highest* attack **accuracy** (i.e., worst-case privacy risk) among different *adaptive* attacks in Table 6.2 (See Appendix E.3.3 for details). We observe that despite being less effective in defending against adaptive attacks than non-adaptive attacks, our method still greatly decreases the *highest adaptive* attacker's accuracy by 13.6%-37.6% compared to vanilla training.

	CIFAR10 (ResNet20)	CIFAR10 (VGG11)	CIFAR100 (ResNet20)	CIFAR100 (VGG11)	CH-MNIST	Texas100	Purchase100
w/o defense	87.3	80.7	92.6	97.5	67.1	79.0	65.7
w/ defense (non-adaptive)	50.0	50.0	50.0	50.0	50.7	50.0	50.1
Δ (non-adaptive)	-42.7	-38.0	-46.0	-48.7	-24.4	-36.7	-23.9
w/ defense (adaptive)	56.0	68.2	57.8	84.2	56.6	53.8	56.0
Δ (adaptive)	-35.9	-15.5	-37.6	-13.6	-15.6	-31.9	-14.8

Table 6.2: The *highest* attack **accuracy** (in %) among different adaptive attacks (and the corresponding non-adaptive attack accuracy is shown for reference) evaluated on the target model with (w/) or without (w/o) defense. Δ corresponds to the *relative* difference (in %) in attack accuracy when applying our defense compared to vanilla training. The used target models are the same as in Table 6.1.

6.6.5 Ablation Study

We study the impact of each component of our approach and plot the results in Figure 6.6. We observe that while applying posterior flattening alone (without gradient ascent) has limited effects, using it together with gradient ascent indeed improves the model's test accuracy across a wide range of attack AUC, which validates the necessity of all components of our method.

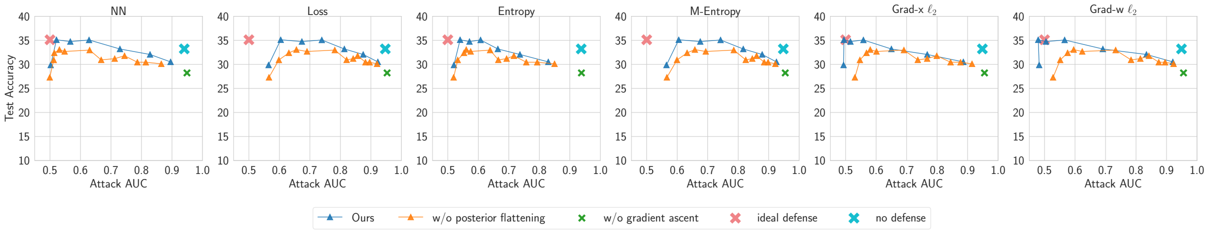


Figure 6.6: Ablation study on CIFAR-100 with ResNet architecture. We validate the necessity of our gradient ascent (Section 6.4.1) and posterior flattening step (Section 6.4.2)

6.7 Discussion

Properties of RelaxLoss. RelaxLoss enjoys several properties which explain its superiority over existing defense methods. In particular, we provide empirical and analytical evidence showing that in contrast to most existing methods (Appendix E.3.9), RelaxLoss reduces the generalization gap and spreads out the training loss distributions (Section 6.5), thereby effectively defeating MIAs. Moreover, we observe that RelaxLoss soften the decision boundaries (Appendix E.3.10), which contributes to improving model generalization [253, 163].

Practicality. We consider the practicality of our method from the following aspects: *(i) Hyperparameter tuning*: Our method involves a single hyperparameter α that controls the trade-off between privacy and utility. A fine-grained grid search on a validation set (i.e., first estimating the privacy-utility trade-off with varying value of α, and subsequently selecting the α corresponding to the desired privacy/utility level) allows precise control over the expected privacy/utility level of the target model. *(ii) Computation cost*: Our method incurs negligible additional computation cost when compared with backpropagation in vanilla training (Appendix E.3.6). In contrast, baseline methods generally suffer from a larger computation burden. For instance, Memguard slows down the inference due to its test-time optimization step, while the training speed of DP-SGD and Adv-Reg is greatly hindered by per-sample gradient computation [64, 41] and adversarial update step, respectively.

6.8 Conclusion

In this paper, we present *RelaxLoss*, a novel training scheme that is highly effective in protecting against privacy attacks while improving the utility of target models. Our primary insight is that membership privacy risks can be reduced by narrowing the gap between the loss distributions. We validate the effectiveness of our method on a wide range of datasets and models, and evidence its superiority when compared with eight defense baselines which represent previous state of the art. As RelaxLoss exhibits superior protection and performance and is easy to be implemented in various machine learning models, we expect it to be highly practical and widely used.

III

Part 3: Application

In this part, we apply the previously discussed privacy-preserving data generation techniques in Part I to more challenging real-world applications, while also considering the practical privacy attacks explored in Part II.

In Chapter 7, we conduct a comprehensive investigation of DP generation techniques applied to real-world gene expression data with complex high-dimensional distribution. Specifically, we introduce an evaluation framework that assesses various aspects of the synthetic data, including downstream utility, statistical fidelity, and biological plausibility. We examine a range of representative methods and present key insights into their advantages, drawbacks, and implications for potential future improvements.

7 Towards Biologically Plausible and Private Gene Expression Data Generation

Contents

Generative models trained with Differential Privacy (DP) are becoming increasingly prominent in the creation of synthetic data for downstream applications. Existing literature, however, primarily focuses on basic benchmarking datasets and tends to report promising results only for elementary metrics and relatively simple data distributions. In this paper, we initiate a systematic analysis of how DP generative models perform in their natural application scenarios, specifically focusing on real-world gene expression data. We conduct a comprehensive analysis of five representative DP generation methods, examining them from various angles, such as downstream utility, statistical properties, and biological plausibility.

Our extensive evaluation illuminates the unique characteristics of each DP generation method, offering critical insights into the strengths and weaknesses of each approach, and uncovering intriguing possibilities for future developments. Perhaps surprisingly, our analysis reveals that most methods are capable of achieving seemingly reasonable downstream utility, according to the standard evaluation metrics considered in existing literature. Nevertheless, we find that none of the DP methods are able to accurately capture the biological characteristics of the real dataset. This observation suggests a potential over-optimistic assessment of current methodologies in this field and underscores a pressing need for future enhancements in model design.

This chapter is based on [31]: As a co-first author of [31], Dingfan Chen implemented all the DP generation methods, conducted the data generation experiments, and was the main writer of the paper. This paper has been accepted by PETs 2024.

7.1 Introduction

Genomic data is considered a goldmine for medical researchers, enabling them to tackle a wide array of challenges. These challenges range from identifying patients at risk of specific diseases, to developing tailored drugs to enhance treatment reliability and reduce care duration. Gene expression data stands as one of the most extensively utilized forms of genomic data. More specifically, in a cell, the instructions on how to build the cell's proteins are encoded in the DNA as genes. In order to produce proteins, copies of the genes are made from the DNA in the form of messenger RNAs (mRNAs) which are then translated into proteins. The more mRNA copies are made of a gene, the more of the corresponding protein can be produced. Conditions such as environmental stimuli or diseases can alter the kind and quantity of proteins that are being produced. Thus, the cell's response to such conditions is reflected in the transcription of genes, i.e., the strength of their expression. Measuring gene expression has therefore become an essential biomedical tool in order to understand how a cell, tissue or organism responds to the conditions it is exposed to [231, 39].

Nevertheless, the use of gene expression data is not without danger, as it can threaten patient privacy [153]. The precise nature of the information it contains could attract the interest of malicious entities, capable of exploiting it for multiple purposes. For example, an insurance company could choose to raise the coverage cost for a patient predisposed to a serious illness. Additionally, publishing information about a person's genetic predispositions for stigmatized diseases can severely impact their social life and societal acceptance.

In light of these concerns, there arose a need to protect individual privacy and avoid such problems, leading to exploration of methods that are able to generate synthetic data backed by rigorous privacy guarantees. Such approaches involve creating synthetic datasets that reflect the characteristics of real gene expression data while providing strong theoretical differential privacy (DP) guarantees. Nonetheless, employing DP entails introducing randomness during the training process, which inevitably compromises the quality of the produced synthetic data. Furthermore, as we strive for stronger privacy guarantees, the randomness required for privacy increases proportionally, further affecting the quality of the synthetic data to a larger extent. This underscores the well-known trade-off between privacy and utility.

Despite significant advances in DP data generation methods that report both good generation quality and privacy guarantees, the majority of quality assessment have unfortunately focused solely on downstream utility. A notable gap persists in evaluations that overlook the preservation of essential statistical and biological characteristics. These characteristics are, however, crucial for ensuring the fidelity and applicability of the generated data. In real-world scenarios, the challenge becomes even more pronounced due to the vast feature space inherent in gene expression data, which stands in stark contrast to the often limited number of available samples. Consequently, the effectiveness of existing methods, previously tested primarily on basic benchmark datasets with relatively simple distributions, remains unclear when applied to real-world gene expression data.

In this paper, we fill this gap by presenting the first systematic quality assessment of synthetic gene expression data produced by five benchmark DP generation models with diverse characteristics. Our assessment encompasses five metrics, spanning various aspects from downstream utility, statistical fidelity, to biological plausibility. Our extensive experimental results reveal intriguing findings:(1) significant privacy risks do exist if the generative models are trained non-privately, while DP training (even with a high privacy budget of $\varepsilon = 100$) greatly mitigates such risks; (2) almost all methods manage to achieve seemingly near-perfect performance in terms of standard utility metrics while providing a reasonably strong privacy guarantee (e.g., $\varepsilon \leq 10$), yet none of the DP models succeed in producing biologically plausible data.

In summary, the key contributions of our study are outlined below:

- Our work presents the first comprehensive and systematic analysis of DP generation methods applied to real-world gene expression data. Our extensive investigation encompasses five diverse generation models, five metrics targeting three principal aspects, providing the first comprehensive view for the current state of real-world applicability of DP generation methods.

- Our analysis reveals crucial insights, highlighting the limitations of existing evaluations that predominantly focus on a single aspect, namely, downstream utility. In contrast, our thorough assessment establishes a reliable evaluation framework that effectively addresses the misconceptions arising from these one-dimensional evaluations.

- Our compelling findings, complemented by an in-depth discussion, offer fresh perspectives for the future development in the related field. With our systematic assessment, we aim to steer DP generation methods towards improved practicality in real-world applications involving sensitive data.

7.2 Related Work

7.2.1 Models for Synthetic Gene Expression Data

Various types of generative models have been employed for generating synthetic gene expression data. Variational autoencoders and deep Boltzmann machines have been used to generate data that aids in designing studies and planning analysis for large experiments [214]. Generative adversarial networks have been exploited for generating gene expression data to combat the challenges of low sample sizes via data augmentation, which is specifically motivated by the unfavorable ratio of samples to features in these datasets [105, 135]. Additionally, synthetic gene expression data has also been used to train imputation methods for handling missing data [156]. However, none of these methods ensure privacy during the whole data generation process. Given that genome-related data, including the gene expression data, is highly privacy-sensitive [153], applying existing works in real-world scenarios becomes challenging due to privacy regulations.

To the best of our knowledge, there is a lack of research delving into the differentially private generation of synthetic gene expression data. While some studies, like [210], have investigated the private generation of synthetic data within the realm of medical data at large, a dedicated focus on gene expression data remains notably absent.

7.2.2 Measuring Quality of Synthetic Gene Expression Data

A variety of methods have been applied in the past to assess the quality of synthetic gene expression data from a biological standpoint. These methods have been used both in the context of bulk as well as single-cell RNA-seq data. Bulk RNA-sequencing refers to the process of sequencing the mRNA transcripts from a sample containing a collection of many cells [117]. The resulting data thus reports the average expression strength of each gene across these cells. Single-cell RNA-sequencing on the other hand, first separates the cells present in the sample before sequencing each individually, generating an expression profile at cell resolution rather than sample resolution [117, 205]. The methods used for evaluating this data comprise the comparison of expression data distributions [9, 214, 251] by looking at mean and median expressions, proportion of zero counts (in single-cell cases) and coefficients of variation. Also, metrics related to functional biology have been applied [225, 214, 105, 156, 135], including preservation of gene-gene correlations, gene ontology terms, differentially expressed genes and clusters in reduced dimensional space, using for example t-SNE, PCA, UMAP or after feature selection.

7.3 Preliminaries

7.3.1 Threat Model

The objective of an adversary is to infer private information about individuals in the training datasets by launching various privacy attacks, such as membership inference attack (MIA), which aims to ascertain if a particular data point was used in training the dataset. We consider two common scenarios for synthetic data generation from an attack standpoint:

- A trained generator generates the synthetic data (e.g., Section 7.4.1-7.4.4). In this case, the adversary can have either black-box access or white-box access to the generator. Black-box access means the adversary can only access the synthetic data generated by querying the model through an API. White-box access allows the adversary to access the generator's internal state, including its parameters.
- The synthetic data is directly generated without using any generator (e.g., Section 7.4.5). In this scenario, the adversary only has access to the synthetic data.

While our privacy model protects against the most powerful adversaries, as discussed below in Section 7.3.2, our experiments consider the scenario with the most knowledgeable adversary who has white-box access to the trained generator, as well as the practical scenario where only the synthetic data is accessible.

7.3.2 Privacy Model

We aim to develop a solution that protects against potential attacks as delineated in our threat model in Section 7.3.1. Specifically, we adopt differential privacy (DP), which ensures the difficulty to infer the presence of any record in the training dataset, even when the adversary has white-/black-box access to the trained generator and/or to the synthetic data. As a result, any potential negative impact on an individual's privacy cannot be attributed to their involvement in the training phase (up to ε and δ). For instance, if an insurance company accesses the generator or the synthetic data (from DP generation methods) and decides to increase an individual's insurance premium, such a decision cannot be attributed to the individual's data presented in the training dataset.

Definition 7.3.1 ((ε, δ)-DP [53])**.** A randomized mechanism $\mathcal{M}$ with range $\mathcal{R}$ is (ε, δ)-DP, if

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

holds for any subset of outputs $\mathcal{O} \subseteq \mathcal{R}$ and for any adjacent datasets $\mathcal{D}$ and $\mathcal{D}'$, where $\mathcal{D}$ and $\mathcal{D}'$ differ from each other by adding or removing one training example, i.e., $\mathcal{D}' = \mathcal{D} \cup \{x\}$ or $\mathcal{D} = \mathcal{D}' \cup \{x\}$ for a data sample x. The privacy parameter ε is the upper bound of privacy loss, and δ is the probability of breaching DP constraints. Smaller values of both ε and δ translate to stronger DP guarantees and better privacy protection.

Definition 7.3.2 (Gaussian Mechanism [53])**.** Let $f : X \to \mathbb{R}^d$ be an arbitrary d-dimensional function with L_2-(global) sensitivity Δ_f^2:

$$\Delta_f^2 = \max_{\mathcal{D},\mathcal{D}'} \|f(\mathcal{D}) - f(\mathcal{D}')\|_2 \tag{7.1}$$

The Gaussian Mechanism $\mathcal{M}_\sigma$, parameterized by σ, adds noise into the output, i.e.,

$$\mathcal{M}_\sigma(\boldsymbol{x}) = f(\boldsymbol{x}) + \mathcal{N}(0, \sigma^2 \boldsymbol{I}). \tag{7.2}$$

For $\varepsilon, \delta \in (0,1)$, $\mathcal{M}_\sigma$ is (ε, δ)-DP if $\sigma \geq \sqrt{2 \ln(1.25/\delta)} \Delta_2 f / \varepsilon$.

Theorem 7.3.1 (Post-processing Theorem)**.** If $\mathcal{M}$ satisfies (ε, δ)-DP, $F \circ \mathcal{M}$ will satisfy (ε, δ)-DP for any data-independent function F with $\circ$ denoting the composition operator.

The post-processing theorem guarantees that if a DP generation model is (ε, δ)-DP, releasing the trained generator and the synthetic dataset will also be privacy-preserving, with the privacy cost bounded by ε and δ.

7.3.3 Biological Criteria

Differential Expression. When diseases and other pathological conditions affect the body, they can alter gene activation within cells, contributing to the manifestation of symptoms. The specific set of genes whose expression levels vary from one disease to another are commonly referred to as *differentially expressed (DE) genes*. Identifying DE genes that distinguish between two conditions is a fundamental step in gene expression analysis [7, 40, 176, 177]. Differential expression can occur as either *up-regulation* or *down-regulation*, meaning that the expression of genes is significantly *increased* or *decreased* in one condition compared to another, respectively (see Section 7.5.3 for the formal definition).

Gene Co-Expression. Genes that are involved in the same biological pathways often form a functional group or *module*, meaning they collectively respond to a condition by similar changes in the expression strength. For example, all genes involved in fighting off a bacterial infection will be activated together when such a pathogen enters the body. Such genes are referred to as *co-expressed*. In order to identify activated or inactivated biological pathways, detecting such modules of co-expressed genes is a common step in the analysis of gene-expression data [154, 106]. Specifically, co-expression between a pair of genes with indices j and k is quantified using their *Pearson correlation coefficient* r_{jk} with

$$r_{jk} = \frac{\sum_{i=1}^{n}(x_j^{(i)} - \bar{x}_j)(x_k^{(i)} - \bar{x}_k)}{\sqrt{\sum_{i=1}^{n}(x_j^{(i)} - \bar{x}_j)^2}\sqrt{\sum_{i=1}^{n}(x_k^{(i)} - \bar{x}_k)^2}}, \tag{7.3}$$

where $x_j^{(i)}$ and $x_k^{(i)}$ are the expression values of genes j and k in sample i, respectively, while $\bar{x}_j$ and $\bar{x}_k$ are the mean expression values of the two genes across n biological samples. Groups of genes with high Pearson correlation coefficients are considered *modules* of co-expressed genes, with $r_{jk} > 0.7$ are typically considered as biologically significant co-expressions.

7.4 Models

Given the real dataset $\mathcal{D} = \{(\boldsymbol{x}^{(i)}, y^{(i)})\}_{i=1}^n$ consisting of n samples $(\boldsymbol{x}^{(i)}, y^{(i)})$ with $\boldsymbol{x}^{(i)} \in \mathbb{R}^d$ and $y^{(i)} \in \{1, ..., C\}$ denoting the features and class labels respectively, the objective of the generation methods is to capture the real underlying distribution $p(\boldsymbol{x}, y)$ and generate synthetic data samples $(\tilde{\boldsymbol{x}}, \tilde{y})$ that mimic the statistical characteristics of the real samples from $\mathcal{D}$. In our case, the feature vector $\boldsymbol{x}^{(i)}$ represents the gene expression level and the class label $y^{(i)}$ corresponds to the disease type, with d and C denoting the feature dimension and number of label classes, respectively.

In this work, we explore the most prominent categories of (DP) generation methods found in the literature: (1) *density estimation (probability distribution fitting)*, (2) *graphical models-based* methods, (3) *marginal-based* methods, and (4) *deep generative models*. A summary of these methods and their diverse characteristics can be found in Table 7.1.

Method	Category	Attribute type	DP sanitization
RON-Gauss	Density estimation	continuous only	one-shot
VAE	Deep generative model	continuous	iterative
GAN	Deep generative model	continuous	iterative
Private-PGM	Graphical model	discrete only	one-shot
PrivSyn	Marginal	discrete only	one-shot

Table 7.1: Summary of Models.

7.4.1 RON-Gauss

RON-Gauss [28] generates synthetic data by drawing samples from a multivariate Gaussian distribution fitted in a projected space of the real data. Specifically, it operates by executing the following steps: Firstly, the data is pre-processed to ensure it possesses bounded sensitivity and adheres to the regularity conditions for the Diaconis-Freedman-Meckes effect (which guarantees the data will exhibit Gaussian-like distribution after projection with high probability). Next, a random orthonormal (RON) projection is applied on the pre-processed data, i.e., $\overline{\boldsymbol{X}} = \boldsymbol{W}^T\boldsymbol{X}$ with $\boldsymbol{W} \in \mathbb{R}^{d \times p}$ signifying the RON projection matrix and $\boldsymbol{X}$ representing the pre-processed data matrix. Subsequently, a multivariate Gaussian model is fitted onto the projected data. During the inference stage, new samples are drawn from the fitted Gaussian distribution and are inversely projected into the original data space to form synthetic data samples. To maintain privacy, DP noise is added into both the mean and covariance of the fitted Gaussian distribution. Moreover, the Gaussian model is independently applied to each label class to facilitate label-conditional generation, which aligns with the concept of a Gaussian mixture model (GMM), where each label class forms a mode of the GMM. The detailed algorithm is presented in Algorithm 3.

Algorithm 3: RON-Gauss

```
Input: Dataset $\mathcal{D} = \{(\boldsymbol{x}^{(i)}, y^{(i)})\}_{i=1}^{n}$, projection dimension $p$, noise scale $\sigma$
Output: Synthetic dataset $\mathcal{S}$
for $c$ in $\{1, ..., C\}$ do
    (1) Extract samples with label class $c$ to form data matrix $\boldsymbol{X}_c \in \mathbb{R}^{d \times n_c}$;
    (2) Pre-process data and compute the mean:
    • Pre-normalize:$\boldsymbol{x}^{(i)} := \boldsymbol{x}^{(i)} / \|\boldsymbol{x}^{(i)}\|_2 \quad \forall \boldsymbol{x}^{(i)} \in \boldsymbol{X}_c$
    • Compute the DP mean: $\boldsymbol{\mu}_c = \frac{1}{n_c} \sum_{i=1}^{n_c} \boldsymbol{x}^{(i)} + \mathcal{N}(0, \sigma^2 \boldsymbol{I})$
    • Center the data: $\boldsymbol{x}^{(i)} := \boldsymbol{x}^{(i)} - \boldsymbol{\mu}_c \quad \forall \boldsymbol{x}^{(i)} \in \boldsymbol{X}_c$
    • Re-normalize: $\boldsymbol{x}^{(i)} := \boldsymbol{x}^{(i)} / \|\boldsymbol{x}^{(i)}\|_2 \quad \forall \boldsymbol{x}^{(i)} \in \boldsymbol{X}_c$
    (3) Apply RON projection: $\overline{\boldsymbol{X}}_c := \boldsymbol{W}^T \boldsymbol{X}_c \in \mathbb{R}^{p \times n_c}$;
    (4) Derive the DP covariance: $\boldsymbol{\Sigma}_c = \frac{1}{n_c} \overline{\boldsymbol{X}}_c \overline{\boldsymbol{X}}_c^T + \mathcal{N}(0, \sigma^2 \boldsymbol{I})$;
    (5) Synthesize data for class $c$ by drawing samples from the Gaussian distribution
        $\widetilde{\boldsymbol{x}}^{(i)} \sim \mathcal{N}(\boldsymbol{W}^T \boldsymbol{\mu}_c, \boldsymbol{\Sigma}_c)$;
    (6) Inversely project and recenter: $\widetilde{\boldsymbol{x}}^{(i)} := \boldsymbol{W} \widetilde{\boldsymbol{x}}^{(i)} + \boldsymbol{\mu}_c$ and construct the synthetic set
        $\mathcal{S}_c = \{(\widetilde{\boldsymbol{x}}^{(i)}, c)\}_{i=1}^{n_c}$;
end
return Synthetic dataset $\mathcal{S} = \mathcal{S}_1 \cup \cdots \cup \mathcal{S}_C$
```

7.4.2 VAE

The Variational Autoencoder (VAE) [100] is a type of deep generative model that consists of both an encoder and a decoder. During training, these two components are cascaded and optimized to reconstruct data under pre-defined similarity metrics such as L_1/L_2 loss. The encoder (denoted as q_ϕ) maps input data $\boldsymbol{x}$ into a latent space, while the decoder (denoted as p_θ) maps the encoded latent representation back into the data space. Meanwhile, VAE regularizes the encoder by imposing a prior P_z over the latent code distribution. This regularization encourages the latent code to form a simple distribution that is amenable to sampling. During inference, new latent codes $\boldsymbol{z}$ are sampled from the prior distribution P_z and then fed into the decoder to generated synthetic samples. The formal VAE objective is composed of a reconstruction term and a prior regularization term:

$$\min_{\theta,\phi} \mathcal{L}_{VAE} = -\mathbb{E}_{q_\phi(\boldsymbol{z}|\boldsymbol{x})}[p_\theta(\boldsymbol{x}|\boldsymbol{z})] + KL(q_\phi(\boldsymbol{z}|\boldsymbol{x}) \| P_z) \tag{7.4}$$

where $KL(\cdot\|\cdot)$ denotes the KL divergence, $\boldsymbol{z}$ and $\boldsymbol{x}$ stand for the latent code and the real data, respectively. $q_\phi(\boldsymbol{z}|\boldsymbol{x})$ represents the probabilistic encoder parameterized by ϕ, and $p_\theta(\boldsymbol{x}|\boldsymbol{z})$ represents the probabilistic decoder parameterized by θ. In practice, the prior P_z is always chosen to be a unimodal Gaussian distribution and $\boldsymbol{z}$ is sampled using the reparameterization trick, facilitating a closed-form derivation of the second term.

We employ the class conditional (CVAE) [192] for label-conditional generation. In this framework, both the encoder and the decoder receive additional (one-hot) label information y. Formally, the training objective can be expressed as:

$$\min_{\theta,\phi} \mathcal{L}_{CVAE} = -\mathbb{E}_{q_\phi(\boldsymbol{z}|\boldsymbol{x},y)}[p_\theta(\boldsymbol{x}|\boldsymbol{z},y)] + KL(q_\phi(\boldsymbol{z}|\boldsymbol{x},y) \| P_z) \tag{7.5}$$

During the generation process, labels are generated based on their occurrence rates in the real dataset. Privacy constraints is incorporated in the training stage by replacing the regular stochastic gradient descent (SGD) update with DP-SGD [1], which involves clipping the per-example gradients and adding calibrated random noise to the mini-batch gradients.

7.4.3 GAN

The Generative Adversarial Network (GAN) [65] is another widely used type of deep generative model. It comprises two neural network components, a generator G_θ and a discriminator D_ϕ, which are trained simultaneously in an adversarial manner. The generator takes random noise $\boldsymbol{z}$ (latent code) as input and generates samples that approximate the distribution of the training data. Conversely, the discriminator evaluates both generator-generated samples and real training data samples, aiming to distinguish between the two sources. Throughout training, these two modules engage in a competitive process, each adapting to the other: the generator seeks to generate progressively more realistic samples to deceive the discriminator, while the discriminator learns to distinguish the two sources more accurately. The standard GAN training objective can be formulated as

$$\min_\theta \max_\phi \mathbb{E}_{\boldsymbol{x}\sim P_{\text{data}}}[\log(D_\phi(\boldsymbol{x}))] + \mathbb{E}_{\boldsymbol{z}\sim P_z}[\log(1 - D_\phi(G_\theta(\boldsymbol{z})))] \tag{7.6}$$

where θ, ϕ denote the parameters of the generator and the discriminator respectively. P_{data} stands for the real data distribution, and the P_z is the prior distribution of the latent code. The first term in the objective prompts the discriminator to output high scores for real data samples. In contrast, the second term encourages the discriminator to assign lower scores to generated samples, while the generator is optimized to maximize the discriminator's output score. During inference, the generator will receive new latent code samples z drawn from the known prior distribution P_z, often standard Gaussian, and produce synthetic data samples.

For private training, we adopt the DP Wasserstein GAN (DP-WGAN) [6] implementation and its conditional variant to integrate label information during generation. Specifically, the Wasserstein distance [8] is used as the training objective with the label information acting as auxiliary input for both the generator and discriminator:

$$\min_\theta \max_\phi \mathbb{E}_{\boldsymbol{x}\sim P_{\text{data}}}[D_\phi(\boldsymbol{x}, y)] - \mathbb{E}_{\boldsymbol{z}\sim P_z}[D_\phi(G_\theta(\boldsymbol{z}, y), y)] \tag{7.7}$$

The DP guarantee is ensured by employing DP-SGD for discriminator updates, which in turn guarantee the privacy of the whole GAN model and the synthetic data due to the post-processing theorem (Theorem 7.3.1).

7.4.4 Private-PGM

The Private Probabilistic Graphical Models (Private-PGM) framework [138] is designed to construct undirected graphical models from DP noisy measurements over low-dimensional marginals, which facilitates the generation of new synthetic samples via sampling from the learned graphical model. Specifically, Private-PGM operates on records consisting of discrete attributes. Formally, a record is denoted as $\boldsymbol{x} = (x_1, ..., x_d, x_{d+1})$ where each feature attribute x_i for all $i \in \{1, ..., d\}$ and the label $y = x_{d+1}$ fall within a discrete finite domain. Let $\mathcal{C}$ represent a collection of *measurement sets*, where each $C \in \mathcal{C}$ is a subset of $\{1, ..., d+1\}$ (i.e., the combinations of attributes), and let $\boldsymbol{v}_C$ define the marginal probability vector on C. Private-PGM first obtains DP noisy measurements $\boldsymbol{m}_C = \boldsymbol{Q}_C \boldsymbol{v}_C + \mathcal{N}(0, \sigma_C^2 \boldsymbol{I})$ with $\boldsymbol{Q}_C$ denoting the linear marginal query set over measurement set C and $\mathcal{N}(0, \sigma_C^2 \boldsymbol{I})$ representing the noise introduced by the Gaussian mechanism (with σ_C the noise scale determined by the desired privacy level ε_C and δ. Refer to Definition 7.3.2). Subsequently, it estimates the marginal $\hat{\boldsymbol{v}}$ that best explain all the noisy measurement $\hat{\boldsymbol{v}} = \arg\min_{\boldsymbol{v}} \|\mathcal{Q}\boldsymbol{v} - \boldsymbol{m}\|$ where $\mathcal{Q}$ is a block-diagonal matrix with diagonal blocks $\{\mathcal{Q}_C\}_{C\in\mathcal{C}}$ (i.e., combining all the query set $\mathcal{Q}_C$) and $\boldsymbol{m} = (\boldsymbol{m}_C)_{C\in\mathcal{C}}$ the combined vector of measurements. Meanwhile, it estimates the parameter of the graphical model using existing graph inference and learning algorithms such as belief propagation on a junction tree.

In general, $\mathcal{Q}_C$ can represent a complex set of linear queries expressed over C, and its selection can be adaptively tailored to downstream objectives. In our work, we adhere to the default implementationwhere $\mathcal{Q}_C$ is set to be an identity matrix. This configuration renders the measurement $\boldsymbol{m}_C$ equivalent to the corresponding noisy marginal $\boldsymbol{v}_C + \mathcal{N}(0, \sigma_C^2\boldsymbol{I})$. Moreover, for computational feasibility, we adopt the basic configuration offered by the official implementation that sets $\mathcal{C} = \big\{\{1\}, ..., \{d+1\}\big\} \cup \big\{\{1, d+1\}, ..., \{d, d+1\}\big\}$, which encompasses all one-way marginals as well as the 2-way marginals associated with the label attribute. The privacy budget is allocated uniformly across each measurement, i.e., $\varepsilon_C = \varepsilon/|\mathcal{C}|$ with ε the total privacy cost due to sequential composition.

7.4.5 PrivSyn

Similar to Private-PGM, PrivSyn [258] operates on data with discrete attributes to obtain measurable (noisy) marginals. However, while Private-PGM explicitly constructs factorized sparse graphical models, PrivSyn directly generates data from the noisy marginal measurements. This approach inherently allows the use of an implicitly dense graphical model, enhancing its expressiveness capacity.

PrivSyn is structured to execute the following steps sequentially:

- *Marginal selection*: This step selects the most informative marginals from the candidate set to optimize the privacy-utility trade-off.
- *Noise addition*: DP noise is added to the selected marginal measurements, ensuring privacy guarantee.
- *Post-processing*: This phase ensures consistency from the noisy measurements. It addresses issues such as negative marginal measurements, cases where probabilities do not sum up to 1, and aligning different marginals that share common attributes.
- *Data Synthesis*: Starting with a randomly initialized synthetic dataset, this step iteratively updates it to ensure alignment with the marginal measurements.

In our experimental evaluation, we omit the more involving 2-way marginal selection step for our dataset, as this step is prohibited by the significant computation and privacy costs, which scale quadratically with the feature dimensions. Instead, we utilize all 2-way marginals linked with the label attribute, aligning with the approach taken in Private-PGM to ensure a fair comparison. Apart from this, we adhere to the default configuration of the official implementation, which allocates the privacy budget at a ratio of $1:8$ between publishing the 1-way and 2-way marginals.

7.5 Multi-Dimensional Evaluation of Synthetic Gene Expression Data

Our study delved into a comprehensive assessment of various models. This evaluation was executed through a meticulous analysis of model performance across three main aspects: **utility**

(Section 7.5.1), **statistical** (Section 7.5.2), and **biological** (Section 7.5.3) evaluation. Each aspect encompasses distinct metrics: *machine learning efficacy* for **utility** evaluation, *marginal* (*histogram intersection*) and *joint* (*distance to closest record*) closeness for **statistical** evaluation, as well as *differential expression* and *gene co-expression* for **biological** evaluation.

7.5.1 Utility Evaluation

7.5.1.1 *Machine Learning Efficacy*

Evaluating the quality of synthetic data typically involves a standard procedure of assessing its performance within a downstream task. This evaluation determines whether the synthetic data, when used as a replacement for the real data, can accomplish the desired task with comparable effectiveness. This is executed by training machine learning models on real (train) data and evaluating their performance on held-out (test) data. Subsequently, a parallel model is trained on synthetic data and evaluated using the same held-out data. The choice of evaluation metrics is determined by the specific nature of the task at hand. In our work, we adopt the standard *accuracy* score for evaluating the disease classification task.

7.5.2 Statistical Evaluation

Utility-based metrics, however, often offer an incomplete perspective due to their narrow evaluation lens, presenting a single facet of the model's performance, which can occasionally lead to misleading impressions. In order to address this potential bias, it becomes crucial to incorporate additional statistical metrics that emphasize the fidelity of the generation process. This entails assessing how effectively the model captures both the marginal distribution and the underlying joint distribution of the data, providing a more comprehensive understanding of its performance.

7.5.2.1 *Histogram Intersection*

The *histogram intersection* serves as a prevalent qualitative tool for visualizing one-dimensional data (i.e., single columns/attributes), enabling a comprehensive exploration of the data's distribution characteristics. Understanding such single-dimensional distributions can be pivotal for subsequent pre-processing and analysis steps. Prior studies have harnessed this metric to compare the distributions of synthetic and real data by selecting specific attributes from the real dataset and overlaying the histograms of the corresponding real data onto the synthetic ones. This technique, referred to as *distribution matching plots*, provides a qualitative assessment of how closely the two distributions align.

However, relying solely on qualitative measures has its limitations, particularly when confronted with large feature sets like gene expression data. Manually visualizing each column becomes impractical. This necessitates a quantitative approach that maintains a similar essence but can be aggregated to yield a single score. The *histogram intersection metric* proposed in [4] is applicable in such scenario. It is computed as the sum of the minimum probability values between the real data column and the synthetic data column. This sum is subsequently averaged across the various columns in the dataset (see Equation 7.9). In contrast to other analogous techniques like the *Wasserstein distance*[1] or *Jensen-Shannon divergence*

[1] https://www.wikiwand.com/en/Wasserstein_metric

score[2], the *histogram intersection* score demonstrates superior performance and exhibits a strong correlation with other metrics [4] [3]. This quantitative approach strikes a balance between comprehensiveness and practicality, making it an effective tool for evaluating the quality of the generated data.

$$p_c = \frac{s_c}{|\mathcal{D}|\Delta_i} \quad q_c = \frac{t_c}{|\mathcal{S}|\Delta_i} \tag{7.8}$$

$$\text{HI}(\boldsymbol{p}_i, \boldsymbol{q}_i) = \sum_c \min(p_c, q_c) \tag{7.9}$$

$$\text{Overlap Score} = \frac{1}{d} \sum_i \text{HI}(\boldsymbol{p}_i, \boldsymbol{q}_i) \tag{7.10}$$

where $\mathbf{p}_i$ and $\mathbf{q}_i$ denote the histogram representations of the probability distributions for the real ($\mathcal{D}$) and synthetic ($\mathcal{S}$) datasets within feature i, respectively. The terms p_c and q_c represent the proportions of category c for feature i, with s_c/t_c denoting the counts of real/synthetic samples in category c. The factor Δ_i is introduced as a normalization term, specifying the bin size for numerical features. The term $\text{HI}(\mathbf{p}_i, \mathbf{q}_i)$ represents the histogram intersection score for feature i. The dimensionality of the feature space is denoted by d. The *Overlap Score* is computed by averaging the histogram intersection scores across all features.

7.5.2.2 *Distance to Closest Record*

The *distance to closest record* metric aims to measure the similarity between the *joint* distribution of real and synthetic data. Obtaining an exact measurement of the joint distribution is inherently challenging and always infeasible, as the underlying probability distribution of the real data is unknown and generally intractable. To circumvent this, we approximate the alignment of joint distributions using k-nearest neighbors (KNN). This involves computing the Euclidean distance between each synthetic data sample and its k nearest neighbors in either the held-out or training set. The objective is to evaluate the plausibility of each synthetic sample being real. The final KNN Distance score is the average across all synthetic dataset samples and various k values, as defined in Equation 7.12.

$$d_k(\tilde{\boldsymbol{x}}) = \text{first}_k\Big(\text{sort}\Big(\{\|\tilde{\boldsymbol{x}} - \boldsymbol{x}\|_2 \,|\, \forall \boldsymbol{x} \in \mathcal{D}_{\text{train/test}}\}\Big)\Big) \tag{7.11}$$

$$\text{KNN Distance Score} = \frac{1}{|\mathcal{S}| \cdot k} \sum_{\tilde{\boldsymbol{x}} \in \mathcal{S}} \sum_{i=1}^{k} d_{k,i}(\tilde{\boldsymbol{x}}) \tag{7.12}$$

where $\mathcal{S}$ denotes the synthetic set, $\mathcal{D}$ the real dataset, $d_k(\tilde{\boldsymbol{x}})$ is a sequence contain the k smallest values of distances (where $\text{sort}(\cdot)$ represents the sorting operation in ascending order and $\text{first}_k(\cdot)$ denotes the operation for retrieving the first k elements from the sorted sequence), with $d_{k,i}(\tilde{\boldsymbol{x}})$ denoting the i-th element of $d_k(\tilde{\boldsymbol{x}})$.

7.5.3 Biological Evaluation

7.5.3.1 *Differential Expression*

There are several methods to measure differential expression, but many of them make strong assumptions on the distribution underlying gene expression data [7, 177, 131]. However, the

[2] https://www.wikiwand.com/en/Jensen-Shannon_divergence

[3] The histogram intersection metric defined here also corresponds to 1 - total variation distance, a popular metric that quantifies the similarity between two probability distributions.

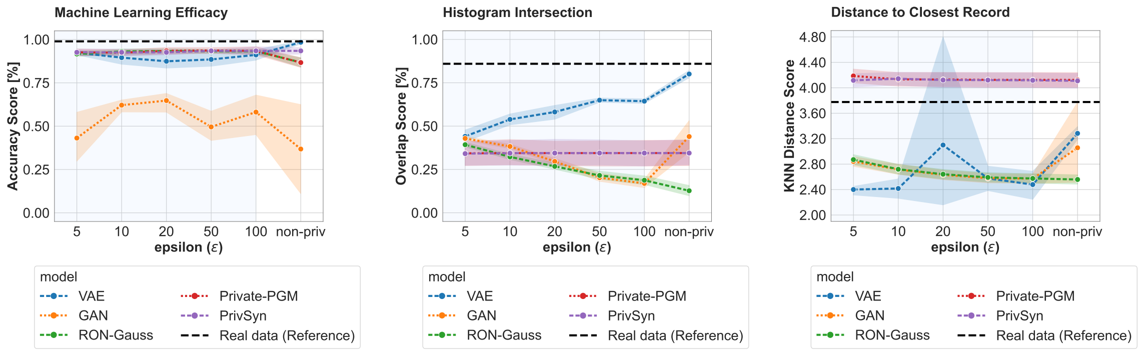
question of which distribution gene expression data follows has been subject to debate for many years [44]. To avoid making any (potentially false) assumptions regarding the distribution, we chose a non-parametric test for the identification of differentially expressed genes, namely the *Wilcoxon signed rank test* [235]. For each pair of conditions in the data, the test was conducted on the expression values of each gene measured across the samples of the respective condition. We ran the test using the pairwise Wilcoxon function from the R-package *scran* (version 1.26.2) and using the alternative hypothesis for each side to differentiate between up- and down-regulation. We considered a gene as differentially expressed between two conditions if the p-value was at most 0.05. The reconstruction of DE-genes by different generative models M at varying privacy levels ε is commonly quantified via the mean true positive rate (TPR) defined as follows.

$$\mathrm{TPR}_{\mathcal{M},\varepsilon} = \frac{\sum\limits_{\{a_i,a_j\}\subset\mathcal{A},a_i\neq a_j} \left(\mathrm{TPR}^{\mathrm{up}}_{a_i,a_j,\mathcal{M},\varepsilon} + \mathrm{TPR}^{\mathrm{down}}_{a_i,a_j,\mathcal{M},\varepsilon}\right)}{2\cdot\binom{|\mathcal{A}|}{2}} \tag{7.13}$$

where $\mathcal{A}$ denotes the set of condition pairs (each pair representing different disease types distinguished by unique label classes in our case). $\mathrm{TPR}^{\mathrm{up\ (down)}}_{a_i,a_j,\mathcal{M},\varepsilon}$ signifies the true positive rate for identifying up-regulated (or down-regulated) DE-genes within synthetic data generated by the model $\mathcal{M}$ under a given privacy budget ε, in comparison to the actual DE-genes observed in the real dataset for conditions a_i and a_j. $\binom{|\mathcal{A}|}{2}$ represents the count of all possible unordered condition pairs.

7.5.3.2 *Gene Co-Expression*

To assess if groups of co-expressed genes that are present in the real data were preserved in the synthetic data, we applied *hCoCena* [154], an R-package that enables the integration of different gene expression datasets, i.e., the real and the synthetic data in our case, and their subsequent joint co-expression analysis. The tool creates a gene co-expression network for each set, which is a weighted graph G = (V, E), where the nodes V represent genes, edges E represent co-expressions and the edges are weighted with the co-expression strength. The weight w is computed as the *Pearson Correlation Coefficient* r between their expression values across samples, such that $w(e_{i,j})$=$r(x_i, x_j)$, where x_i and x_j are the expression values of gene i and j, respectively. Afterwards, genes that are not significantly strongly co-expressed according to a user-defined correlation cut-off with any other gene are discarded to only include strong co-expressions that are potentially biologically meaningful. A gene co-expression network is created for each dataset. We then used these co-expression networks to identify the number of co-expressions (i.e., graph edges) that were correctly reconstructed in the synthetic data and the number of spurious co-expressions introduced in the synthetic data that did not exist in the real data. Additionally, modules of strongly co-expressed genes were identified in the network of the real dataset using the *Leiden community detection algorithm*. We investigated the **mean group fold-changes (GFCs)** for the detected modules across conditions in the real and the synthetic data. GFCs are a metric for the average expression of a module in a group of samples, i.e. all samples of a particular experimental condition, essentially representing the activation or deactivation of the module under the given condition.



(a) (b) (c)

Figure 7.1: Utility Evaluation by Machine Learning Efficacy, and Statistical Evaluation by Histogram Intersection and Distance to Closest Record. Shown in (a) are the Accuracy Scores for the *Machine Learning Efficacy* metric across 5 various models for the DP-case (blue shading) with varying ε values, alongside the non-private case. Similarly, (b) and (c) display the Overlap Score and K-Nearest Neighbors Distance Score for the *Histogram Intersection* metric and *Distance to Closest Record* metric, respectively. Evaluations encompassed two seeds for training split creation and two synthetic dataset randomizations. The presented values represent means across these randomization seeds. The black dashed line represents the reference score on actual train-test data, signifying the best attainable score.

7.6 Evaluation

7.6.1 Dataset

The generative models were trained on a bulk RNA-seq dataset compiled by Warnat-Herresthal *et al.* [232]. The dataset is structured as a matrix, with rows corresponding to *samples* and columns to *features*. Each row represents a biological specimen obtained from a patient, while each column indicates the expression level of a particular gene. The expression levels are quantified by RNA-seq counts, with higher integer values indicating greater gene activity. It comprises samples from 5 disease classes, 4 classes of which are types of leukemia and the fifth class is the category "*Other*", which is made up of samples from various other diseases as well as healthy controls. The 4 leukemia types are acute myeloid leukemia ("*AML*"), acute lymphocytic leukemia ("*ALL*"), chronic myeloid leukemia ("*CML*") and chronic lymphocytic leukemia ("*CLL*"). Sample counts per class are listed in Table 7.2. As per the original publication, the data were normalized with DeSeq2 [131] to account for varying sequencing depths and RNA composition, which is necessary to compare expression levels of different samples and conduct a DE-gene analysis. Given the high dimensionality of the features (more than 12k genes) and the comparatively low sample size (1181), we reduced the feature space to 958 genes. Notably, even this reduced feature dimension remains significantly high, especially when compared with standard benchmarking datasets which typically comprise merely dozens of features. These 958 genes were not selected randomly but based on their characterization as *landmark genes* in the LINCS L1000 project [203]. The landmark genes were identified as representative genes that, when measured, allow the inference of around 20k other genes.

Pre-processing and Post-processing. In accordance with standard practices, we pre-processed

Class	AML	ALL	CML	CLL	Other	**Total**
# Samples	508	12	14	13	634	**1181**

Table 7.2: Dataset summary. Listed are the different sample classes present in the dataset and the number of samples in each class.

our data prior to model training. For RON-Gauss, GAN, and VAE, which operate on continuous data expected to be well-centered, we standardized each feature by subtracting its mean and dividing by its standard deviation. Conversely, for Private-PGM and PrivSyn which rely on discrete representations for computing marginals, we discretized each feature into four bins based on its quantiles: <25%, 25%-50%, 50%-75%, and >75%. This approach was chosen to accurately represent up- and down-regulation, while also maintaining a condensed format (resulting in a limited number of bins after discretization) for an optimized privacy-utility trade-off.

After training and generation, we implemented the following post-processing measures:

- For RON-Gauss, GAN, and VAE: We reverted the standardization by multiplying the generated data features by the standard deviation and adding back the mean (both the standard deviation and the mean were pre-computed on the real dataset).
- For Private-PGM and PrivSyn: We mapped the generated discrete data back to the original continuous mean value associated with each bin.

We verify the efficacy of our approach via our preliminary experiments: the continuous pre- and post-precessing was proved to be lossless, while the discrete one did *not* affect the biological and utility evaluation.

In line with the common evaluation protocol adopted in DP literature, we do *not* incorporate DP into the pre- and post-processing process, and the label class occurrence ratio is treated as public information and used during generation. This approach aids in producing meaningful evaluation results and offers a more accurate indication of performance, particularly given our challenging setup. However, it is crucial to note that in real-world applications, all such processes including the hyperparameter selection [160] would require DP sanitization to ensure stringent privacy protection. Although implementing such sanitization is generally technically straightforward (e.g., either computed on public data or using DP techniques such as Algorithm 2 in [212] and [63] for DP sanitization techniques applicable to continuous and discrete processing, respectively), it can lead to considerable utility loss in bio-data, mainly due to limited sample sizes, which warrants further discussion and investigation.

7.6.2 Setup

We follow the official implementation for methods that offer open-source code: RON-Gauss[4], GAN[5], Private-PGM[6], PrivSyn[7] and adopt the default hyperparameter setting. We use the authorized RDP accountant implementation adopted by both TensorFlow privacy[8] and Opacus[9] for VAE and GAN. For the VAE model, which lacks an official DP implementation, we tuned

[4] https://github.com/inspire-group/RON-Gauss/tree/master

[5] https://github.com/nesl/nist_differential_privacy_synthetic_data_challenge/

[6] https://github.com/ryan112358/private-pgm

[7] https://github.com/usnistgov/PrivacyEngCollabSpace/tree/master/tools/de-identification/Differential-Privacy-Synthetic-Data-Challenge-Algorithms/DPSyn

[8] https://www.tensorflow.org/responsible_ai/privacy/tutorials/classification_privacy

[9] https://opacus.ai/api/accounting/rdp.html

the key hyperparameters (including the weight of the reconstruction loss term, the number of training iterations, the gradient clipping bound, the batch size, and the latent dimension) via grid-search. We repeat the experiments over different random seeds and report the mean and standard deviation over these seeds by default. For biological evaluation where results from different seed cannot be aggregated, we detail the outcomes for each individual seed separately. The δ is set to be 10^{-5} by default across our experiments.

7.7 Experiments

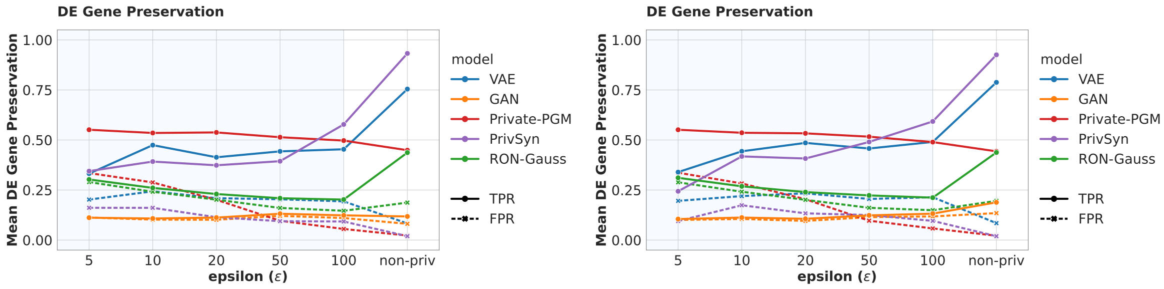


Figure 7.2: Biological Evaluation by DE-Gene Preservation. Shown is the preservation of DE-genes (true positive rate (TPR): solid lines; false positive rate (FPR): dashed lines) across the tested models for the DP-case (indicated by blue shading) with different values of ε and the non-private case. The evaluation was performed for two different seeds used for creating the training split (left and right plot). The presented values are means across two different seeds set for generating the data (except for Private-PGM and PrivSyn, where seeding is not possible).

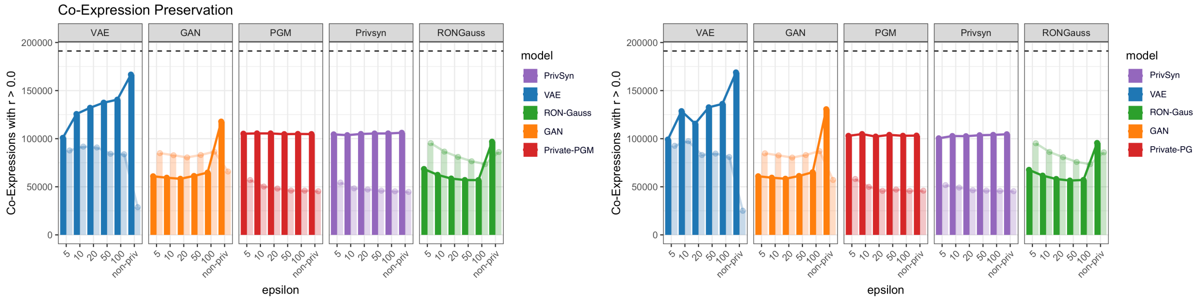


Figure 7.3: Biological Evaluation by Co-Expression Preservation for $r > 0$. Shown is the co-expression preservation across the tested models for different values of ε as well as the non-private case for two different seeds used for creating the training split (left and right plot). Specifically, non-transparent bars give the number of correctly reconstructed co-expressions with Pearson Correlation Coefficient $r > 0$ and an associated p-value < 0.05, while semi-transparent bars give the number of co-expressions introduced by the model that did not exist in the real data. The dashed black line indicates the number of co-expressions in the real data. All values shown are means across two different seeds set for generating the data (except for Private-PGM and PrivSyn, where seeding is not possible).

We study five different generative models: VAE, GAN, RON-Gauss, Private-PGM, and PrivSyn, which encompass diverse categories, attribute types, and DP sanitation approaches, as summarized in Table 7.1. Our assessment was conducted under two scenarios: initially, without the imposition of DP constraints, and subsequently, with DP integration using values of epsilon

(ε) ranging from 5 to 100, signifying a spectrum from high to low privacy levels. We did not reduce the privacy budget to values smaller than 5, as the models fail to achieve reasonable results at this threshold. For models such as VAE and GAN that were not originally designed with DP protections, we incorporate DP to the gradients following the DP-SGD framework [1] to create their respective private variants. Conversely, for inherently privacy-centric models like RON-Gauss, Private-PGM, and PrivSyn, we set the noise scale to be zero to simulate their non-private counterparts. These diverse models were then evaluated using the metrics detailed in Section 7.5. We set the real data as *Reference* (See the dashed black lines in Figure 7.1), which represents the score of each metric when applied to the real training data and then evaluated on the real held-out (i.e., test) data.

7.7.1 Utility Evaluation

For the Machine Learning Efficacy metric, given the classification nature of the task (predicting diverse disease types using gene expression data), we employ a widely-used and straightforward machine learning approach known as logistic regression. This model undergoes training as outlined in Section 7.5.1.1. The chosen evaluation metric is the *accuracy* score.

Results and Findings. The outcomes, depicted in Figure 7.1(a), portray the machine learning utility scores for various generative models across differing privacy levels, ranging from high ($\varepsilon = 5$) to low ($\varepsilon = 100$). In the non-private context (termed as non-priv in Figure 7.1(a)), we observe that all five models—with the exception of GAN —exhibit a substantial utility score ranging from 86% to 98%. This shows a moderate decrease of 0.5% to 12% relative to the reference point set by real data (black dashed line). Within the private realm ($\varepsilon = 5, 10, 20, 50, 100$), models such as Private-PGM, PrivSyn, and RON-Gauss display consistent high utility, encountering a reduction of less than 7.4% in very high privacy conditions ($\varepsilon = 5$) to a 5.8% drop in situations with lower privacy ($\varepsilon = 100$). Notably, these models demonstrate a higher utility as ε increases. Remarkably, the utility metric easily saturates, even with a simple probabilistic model (i.e., the unimodal Gaussian as in RON-Gauss), while the VAE exhibits slight advantages in the non-private case. The GAN model generally performs worse in terms of the utility metric and exhibits relatively high variance, potentially due to the unstable nature of its adversarial training process, which is exacerbated in our dataset with limited samples.

7.7.2 Statistical Evaluation

7.7.2.1 *Histogram Intersection*

We initiate by subjecting the numerical column to min-max pre-processing, a technique that rescales values to fit within the range of 0 to 1. Following this normalization, a discretization binning process is employed, utilizing 25, 50, and 100 bin size, which provides an approximated representation of the numerical column's distribution, and thus ensures tractability. No additional pre-processing steps are required for the discrete and categorical columns. Our computation of the *Overlap Score* adheres to the definition in Equation 7.10.

Results and Findings. Figure 7.1(b) illustrates the overlap score, which serves as the mean of the histogram intersection scores between the columns of real and synthetic data, as detailed in Section 7.5.2.1. In general, across both private and non-private settings, most models exhibit subpar performance on this metric. An exception stands out: the VAE model (depicted by the blue line). Remarkably, it showcases a consistent overlap score of ~80%, experiencing only a 6.8% relative drop compared to the reference set by the real data (black dashed line). This performance trend consistently improves from the high privacy setting ($\varepsilon = 5$) to the low privacy setting ($\varepsilon = 100$), indicating that the synthetic data's marginal distribution increasingly resembles that of the real data, with the relaxation of privacy constraints. However, this does not uniformly apply to all models. For instance, the RON-Gauss model (represented by the green line) shows an unexpected behavior—its overlap score is higher in the very high privacy setting ($\varepsilon = 5$) compared to the non-private setting, exhibiting an 85% drop in performance relative to the real data reference. This outcome is surprising given that this model involves continuous attribute types, which should typically lead to a moderately increasing overlap score as ε increases. Similarly, the GAN model follows a similar trend to RON-Gauss, but it demonstrates a higher overlap score in the non-private setting, with a reduced relative drop of 48.8%. We conjecture that such seemingly abnormal behavior may be partially explained by the fact that both GAN and RON-Gauss did not capture the marginal distributions faithfully even in the non-private case, which making the results more influenced by randomness than by the learnability. The inferior performance of the Private-PGM and PrivSyn models in this metric can potentially be attributed to the loss in precision resulting from the reverse transformation inherent in the discretization process, which may dominate the additional information loss incurred by privacy constraints. Interestingly, despite the modest performance in this metric, the same models excel in the private setting for the machine learning efficacy metric. This underlines the necessity of evaluating synthetic data from various generative models across an array of metrics to gain a comprehensive understanding of their behavior relative to real data.

7.7.2.2 *Distance to Closest Record*

This metric aims to approximate the likelihood that a synthetic data sample originates from the distribution of real data samples. This measurement relies on the K-Nearest Neighbors (KNN) approximation technique. In our experimental setup, we specifically set the value of k to 10, which dictates the computation of the KNN Distance Score according to Equation 7.12. Our procedure involves fitting the KNN classifier from SKlearn[10] on the real test data. Subsequently, we compute the 10-NN distances from the test set to each sample within the synthetic dataset. Figure 7.1(c) shows the averaged 10-NN distance score for different epsilon values (x-axis) and diverse generative models. A higher proximity of this score to the reference established by the real data implies a greater likelihood that the *joint* distribution of real and synthetic data aligns closely. Scores falling below the reference point set by real data imply that the synthetic data samples are closely aligned with the distribution of the real test data. However, it is essential to exercise caution while interpreting these results due to the relatively small size of the test set. Making assertive conclusions based solely on these findings might be premature.

Results and Findings. Intuitively, we anticipate that the score for this metric should be lower, indicating closer alignment to the real data reference (depicted by the black dashed line) in the non-private setting. As privacy levels increase, we expect a moderate increase in the distance—moving from $\varepsilon = 100$ to $\varepsilon = 5$. This examination aims to substantiate the assertions made by prior study [161] that this metric has the potential to quantify privacy. However, the results illustrated in Figure 7.1(c) present a counter-intuitive observation. All models, excluding the graphical-based models Private-PGM and PrivSyn, demonstrate distances below the real data reference. This holds true for both private and non-private scenarios. Notably, the VAE model stands out, exhibiting a low distance to the closest test record (i.e closest to the real data reference but still falls below the black dashed line). This shows a relative drop of

[10] https://scikit-learn.org/stable/modules/generated/sklearn.neighbors.KNeighborsClassifier.html

48% when contrasted with the reference established by real data. Notably, the Private-PGM and PrivSyn models, which yield unsatisfactory outcomes in the histogram intersection metric, also exhibit the most substantial distances to the real data reference. This persistent distance above the black dashed line further indicates that the reverse discretization process could lead to a loss of precision in these models. Additionally, for the VAE model, across the $\varepsilon = 5$ to $\varepsilon = 50$ range, there's a pronounced variance in scores across different experimental random seeds. This variance might offer insights into the model's sensitivity behavior in the private setting.

7.7.2.3 *Summary of Utility and Statistical Evaluation*

The observed results of Figure 7.1 underscore the necessity of assessing diverse metrics when evaluating synthetic data. Moreover, it brings to light an intriguing revelation: even if the synthetic data strays from both marginal and joint distributions, it still exhibits the capacity to maintain substantial downstream utility tasks. This observation reinforces the significance of a comprehensive evaluation approach that considers various aspects of data behavior and performance.

7.7.3 Biological Evaluation

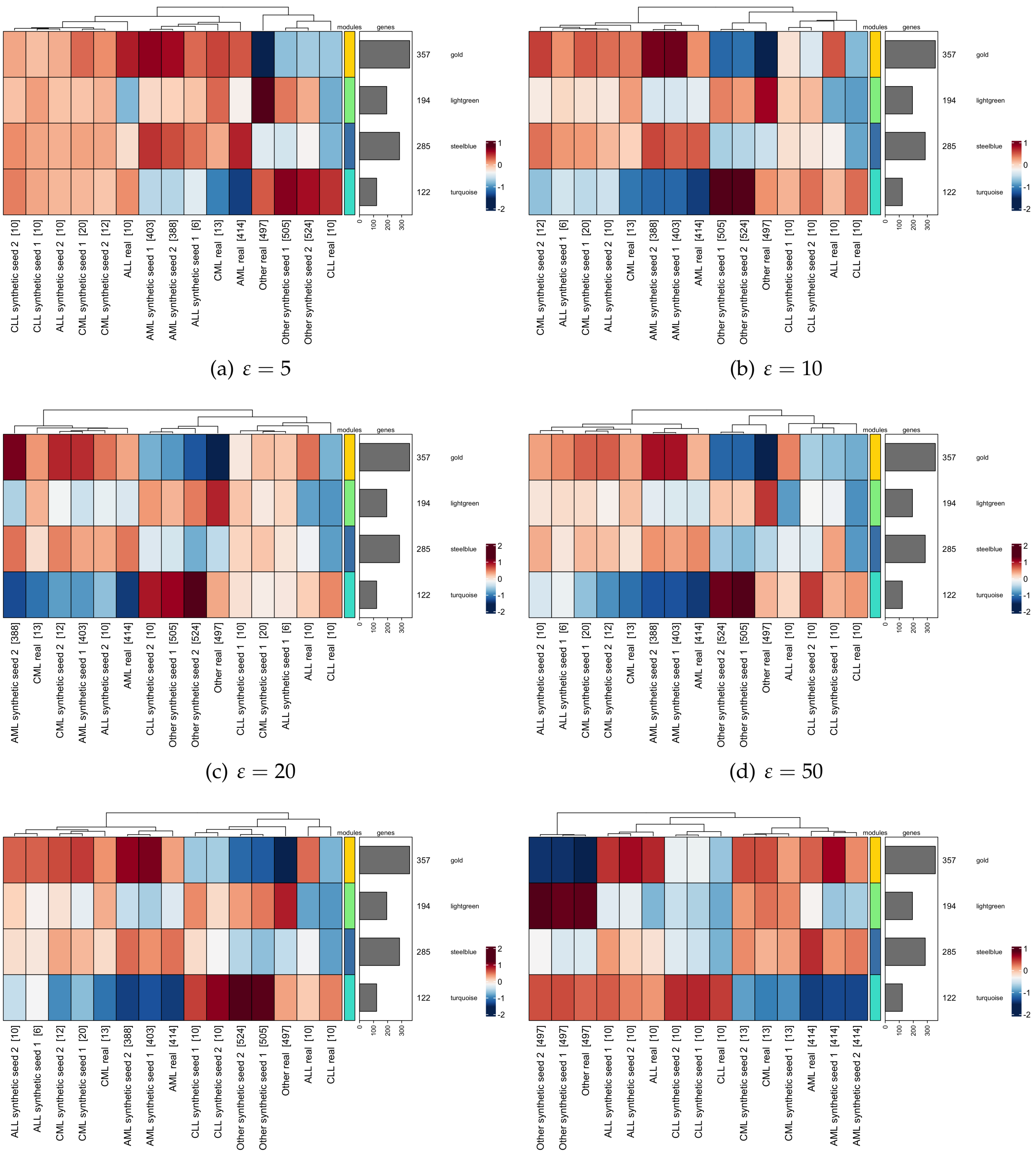
To evaluate the different models for biological soundness, we assessed their capabilities of maintaining two biological aspects in the generated synthetic data: (1) the preservation of differential expression by assessing the TPR and FPR of reconstructed DE-genes per model and across privacy parameters and (2) the preservation of co-expressions between genes, i.e., their Pearson Correlation Coefficients r as well as the activation of co-expressed modules. The models were evaluated once without the constraint of DP and then with DP using ε = 100, 50, 20, 10, 5.

7.7.3.1 *Differential Expression*

We first compared the models' ability to maintain DE-genes in a **non-private** setting. As shown in Figure 7.2, it can be observed that the TPR was high for PrivSyn and VAE models, reaching more than 75% on both data split seeds. RON-Gauss , Private-PGM and GAN showed subpar results, with the GAN model performing particularly poorly. Regarding the FPR, all models maintained rates below 25%, with PrivSyn and Private-PGM reaching FPRs close to zero.
For the **DP** training setting, we observe from Figure 7.2 the following:

- VAE: At a privacy parameter ε = 100, the TPR decreases noticeably in comparison to the non-DP setting from around 75% on average to approximately 50%. As ε is reduced further, the TPR continues to show a decreasing tendency, albeit at a less steep rate. Even at the lowest privacy budget of $\varepsilon = 5$, the TPR of VAE remains higher than that of the GAN in the non-DP setting. Moreover, VAE shows better or equal TPR than PrivSyn at low ε values, and outperforms RON-Gauss across all ε but underperforms Private-PGM once DP is introduced. The FPR increases slightly when introducing DP but remains largely stable for different values of ε.

- GAN:The TPR of DE-genes in the GAN model observed under non-DP conditions remains poor at the introduction of DP and decreasing ε, staying below 20%, while the FPR remains stable (around 10%) across all ε.



(a) $\varepsilon = 5$

(b) $\varepsilon = 10$

(c) $\varepsilon = 20$

(d) $\varepsilon = 50$

(e) $\varepsilon = 100$

(f) Non-private

Figure 7.4: Activation patterns of co-expressed gene modules in VAE for $r > 0$. Shown are the Group Fold Changes (GFCs) of gene modules (*rows*) in the real and the synthetic data sampled with two different seeds. The dendrograms representing the hierarchical clustering of the sample groups differentiated by label class and seed, with each *column* corresponding to a distinct group. Optimally, samples with the same label classes should be adjacent, indicating that they are clustered together. Numbers on the right indicate the number of genes per module, numbers in square brackets on the bottom indicate the number of samples per condition and dataset. Darker shades of red imply activation of the gene module, while darker shades of blue indicate deactivation.

- RON-Gauss: The TPR of the RON-Gauss model drops when introducing DP. Intriguingly, and somewhat against expectations, it exhibits a slight improvement as the privacy loss ε decreases, yet it still only attains low values (around 30%). Concurrently, the FPR steadily increases with decreasing ε, eventually approaching the TPR.
- Private-PGM: The TPR of Private-PGM exhibits a slight increase with decreasing ε, with Private-PGM outperforming all other models for $\varepsilon \leq 50$. Conversely, the FPR rate also increases drastically, reaching around 35%.
- PrivSyn: While PrivSyn showed near perfect TPR in the non-DP case, it is strongly impacted by the introduction of DP, falling below VAE and Private-PGM for $\varepsilon \leq 50$. This performance loss is similarly reflected in the increasing FPR with decreasing ε values.

In summary, while PrivSyn and VAE demonstrate good preservation of DE genes in the non-DP setting, their performances drop when introducing DP and they are surpassed by the Private-PGM model. However, this boost in DE-gene preservation of the Private-PGM model is accompanied by an increasing false positive rate, possibly indicating that the Private-PGM model generally tends to generate more DE genes, however without biological correctness. Both the GAN and the RON-Gauss models perform poorly on this metric, especially in the DP case.

7.7.3.2 *Co-expression*

Here, we investigated both the general preservation of co-expressed genes as well as the activation and deactivation of strongly co-expressed gene sets, so-called *modules*, detected in the real data. The preserved co-expressions as well as the activation patterns of co-expressed gene modules were assessed once for all positive correlations identified in the data (r > 0) (Figure 7.3, Appendix F.5-F.8) and once after filtering for only highly co-expressed genes (r > 0.7) (Appendix Figure F.9-F.11). The latter is motivated by the typical interest in strongly correlated genes during co-expression analyses. The detection of gene modules was performed on the real data for these respective filtering thresholds. In both cases, co-expressions were filtered for associated p-values < 0.05.

We first investigate the **non-private** setting. When considering all co-expressions with r > 0, the VAE reconstructed most of the them while only introducing few false ones that did not exist in the real data (Figure 7.3). The GAN model had less correctly and more incorrectly reconstructed co-expressions than the VAE (Figure 7.3) and the patterns of activation in the gene modules do not match the real data (Appendix Figure F.5). The Private-PGM and PrivSyn models had very similar performances, with more correctly than incorrectly reconstructed co-expressions, however only reconstructing half of the co-expressions found in the real data (Figure 7.3). The activation of the gene modules was well reconstructed (Appendix Figure F.6, F.7). The number of correctly and incorrectly reconstructed co-expressions was almost equal for the RON-Gauss model (Figure 7.3) and activation patterns in the gene modules were almost entirely lost (Appendix Figure F.8). Reducing the co-expressions to only those with $r > 0.7$, only the VAE model and one of the sampling seeds for the GAN yielded any results. The VAE correctly reconstructed most co-expressions from the real set but additionally introduced an almost equal number of incorrect co-expressions (Appendix Figure F.9). Activation patterns in gene modules were well preserved (Appendix Figure F.10). In the data generated by the GAN, the number of incorrectly introduced co-expressions was very high (Appendix Figure F.9) and activation patterns in the gene modules remained poor (Appendix Figure F.11).

For the **DP** setting, we list below our findings:

- VAE: For all co-expressions with $r > 0$, the number of correct co-expressions reconstructed by the VAE reduced gradually when introducing DP with decreasing ε (Figure 7.3). Meanwhile, the number of incorrect co-expressions more than doubled. Activation patterns of gene modules were well maintained for the classes *CML*, *AML* and *Other* at ε = 100 and 50. For lower ε, the characteristic patterns of the modules were increasingly lost as illustrated in Figure 7.4, indicated by the increasing lack of distinctive colors. While the order of gene modules (rows) is fixed to improve comparability, the order of sample groups (columns) is dictated by their hierarchical clustering. This is intended, since it illustrates similarity between module expression of different conditions in the different datasets. In the case of biologically high-quality synthetic data, synthetic samples are expected to co-locate with real samples of the same condition. Note that the results are only shown for one seed used for splitting the dataset for training. If the synthetic data successfully captured the co-expression modules, disease classes are expected to cluster together across synthetic and real data. Focusing only on highly co-expressed genes with $r > 0.7$, a high number of co-expressions is introduced that do not occur in the real data (Appendix Figure F.9). Preservation of module activation is comparable to that observed when selecting co-expressions with $r > 0$ (Appendix Figure F.10).

- GAN: When considering all co-expressions with $r > 0$, the number of correctly reconstructed co-expressions decreased and the number of incorrect ones increased when introducing DP with $\varepsilon = 100$ and reducing this value did not impact the metric further (Figure 7.3). The module activation patterns from the real data are almost entirely lost with the modules demonstrating homogeneous activation (Appendix Figure F.5). When filtering for $r > 0.7$ there were no co-expressions left for any of the ε-values.

- Private-PGM & PrivSyn: The Private-PGM and PrivSyn models demonstrated similar behavior, with the number of reconstructed co-expressions barely being affected by introducing varying levels of privacy in comparison to the non-DP setting (Figure 7.3). Private-PGM maintained the module activation patters for very high ε-values (100 and 50) and for lower ε (20, 10, 5) patterns of large classes such as *AML* and *Other* where maintained, but degraded for the smaller classes (Appendix Figure F.6). Similar results are observed for PrivSyn, with the exception that the degradation of activation patterns already starts at $\varepsilon = 50$ (Appendix Figure F.7). Like the GAN, both models did not generate any significant co-expressions exceeding $r > 0.7$.

- RON-Gauss: While in the non-DP setting, the number of incorrect co-expressions was still slightly lower than that of correct ones, this changes in the DP-setting (Figure 7.3). Decreasing values of ε, however, not only gradually increased the number of incorrectly reconstructed ones, but also that of the correctly reconstructed co-expressions. The gene modules lose their distinctive patterns, showing uniform activation and thus the synthetic data is clustering distinctly away from the real data for all ε (Appendix Figure F.8). As was the case for all models but the VAE, no high co-expressions with $r > 0.7$ were generated by the RON-Gauss model.

In summary, all models except the VAE struggled at correctly recreating strong co-expressions and even the VAE was prone to introducing a high number of incorrect co-expressions for $r > 0.7$. Also for weaker co-expressions, introducing DP strongly impaired the utility of the data both in terms of general co-expressions as well as the activation and inactivation of highly co-expressed modules, with only high ε-values of 100 and 50 maintaining the co-expression structure in the data of some models but not offering any considerable privacy.

7.8 Discussion and Future Directions

Private vs. non-private synthetic data. The biological evaluation of the different models yielded that some model types are capable of generating synthetic data with high biological utility in the non-DP setting. However, the incorporation of DP, though essential for maintaining privacy, significantly hampers their performance. In examining the generally top-performing VAE models through membership inference attacks (Section 7.3.1), we found that non-DP training poses a considerable privacy risk, with AUC-ROC scores of 0.949 and 0.614 for white-box (implemented following [78]) and black-box attacks (implemented following [34]), respectively. Notably, setting the privacy budget at a relative high level of ε=100 resulted in a rapid decline of AUC-ROC scores to around 0.52 in both scenarios. While such high privacy budgets ε =100/50 in some cases still allowed good reconstruction of biology properties as measured by our metrics, these budgets are generally too high to be considered strictly privacy-preserving.

Challenges of low sample regime. As has become apparent in the analysis of activation patterns of co-expressed modules, classes with low sample counts were the first to lose their activation patterns with decreasing privacy budgets. However, such low sample sizes are highly common in gene expression datasets given the often low availability of sampling material. This is particularly the case for rare diseases or samples that can only be acquired with invasive and/or risky medical procedures. Another point that requires addressing is the feature space. The results presented here were achieved on a strongly reduced feature space of approximately 1000 genes, with gene expression datasets often comprising 20-times as many features. The observed limitations of differentially private data generation can thus be expected to increase further when attempting to generate full sets.

Comparing models. The biological evaluation indicated that some model architectures (VAE, PrivSyn and sometimes Private-PGM) are better than others (GAN, RON-Gauss) at learning and generating such highly complex, non-normally distributed data like gene expressions. In general, VAE stands out with the best overall performance, likely because of their substantial expressive capacity, which outperforms simpler probabilistic models like RON-Gauss and methods dependent on low-dimensional approximations, such as PrivSyn and Private-PGM. Moreover, VAEs benefit from stable training processes, advantageous in scenarios with limited samples, unlike the less stable GANs. However, incorporating privacy into this process presents challenges, while maintaining biological utility in a privacy-preserving manner requires further research and possibly more data.

Dependent data. In certain scenarios where the dataset used contains dependent records—such as those associated with the same individual (e.g., single-cell data), a transition to a more advanced level of protection becomes imperative, wherein the goal shifts to preserving each group of dependent records (referred to as Group-level DP). However, this elevation in privacy protection comes with the trade-off of injecting more noise, potentially leading to a greater compromise in the quality of the synthetic data. Furthermore, the task of defining a set of dependent records is not always straightforward. For instance, while it is evident that individuals within the same family often share a common genomic heritage, the extent of relatedness to consider when forming such groups remains ambiguous. Determining whether to include only immediate family members like parents and siblings or to encompass more distant relatives poses an additional challenge. Due to these intricate aspects of privacy considerations, we opt to exclude single-cell datasets from our analysis, despite their potential size advantage for assessing non-DP generative models.

General-purpose synthetic data vs. task-specific data. Providing general-purpose private synthetic data that is useful for all kinds of downstream tasks while preserving statistical and biological properties is still a highly challenging task. Having accurate generators would also imply a strong model and insights for the respective domain, which is often not the case for many bio-medical applications. In addition, small sub-population might not be represented and suffer from mode collapse issues of the generator. It has also been recently questions to what extend such an ultimate solution can be achieved at all [200, 201]. While it is difficult to predict how these trade-off develop in the future, the increased available of such medical data will have a positive effect. In addition, task-specific data generation (e.g. [29]) in a data distillation approach can relax the objectives, but is also departing from the goal of preserving statistic and biological properties by mostly focusing on downstream utility.

7.9 Conclusions

We provide the first systematic analysis of non-private and differentially private generation of gene expression data that covers five diverse modeling approaches ranging from simple density estimation over graphical models to deep generative models. Our analysis encompasses a diverse set of metrics that shed light on the quality of the generated data in terms of statistical and biological properties as well as down-stream utility. A key message of our work is that such a broad evaluation is necessary in order to understand the limitations of current generators. Overall, simple estimators fall behind in performance but equally very complex models like GAN are suffering from the low sample regime as typically encountered in bio-medical applications. While downstream utility can be strong, the synthetic data itself might not retain statistical nor biological properties. Adding privacy preserving estimation and learning of the generators amplifies these problems. A general model recommendation is difficult to provide, as these trade-offs will shift as more data is going to become available in the future. However, we see a tendency that the evaluated graphical models have retained better the differential expression and the variational autoencoder retained better the co-expression - in particular when privacy is added. We will release our setup and evaluation framework in order to further drive progress in this domain.

Conclusion and Future Work 8

Contents

Recent advances in machine learning, while significant, depend heavily on the availability of large-scale, high-quality data. However, in many real-world scenarios involving sensitive information, these advancements may be significantly restricted due to privacy concerns and regulatory barriers that limit data sharing. Meanwhile, the increased usage of personal data in commercial machine learning applications amplifies the risk of privacy violations. Such hurdles present considerable challenges, which can dampen the momentum of progress and innovation in the general field of machine learning.

This thesis aims to address practical privacy issues and devise solutions to these data challenges. We approach the problem from the following main perspectives:

- **Privacy-preserving Generative Modeling:** We investigate how to generate synthetic data with strict privacy guarantees. The primary aim is to create synthetic datasets that preserve the analytical utility of the original data, such as for training machine learning algorithms, while strictly limiting the risk of disclosing any individual's private information.

- **Privacy Attacks and Defenses**: We assess the vulnerabilities inherent in machine learning models and the potential privacy breaches that could occur when these models are deployed, particularly the likelihood of them leaking private information. Moreover, we investigate countermeasures that can be implemented to fortify these models against such privacy attacks.

- **Application**: We pioneer in applying privacy-preserving data generation methods to real-world sensitive datasets, involving a systematic study of various representative methods with unique characteristics. Moreover, we establish a comprehensive evaluation framework that accurately captures the different facets of the generated data, particularly its applicability in real-world downstream analyses.

We provide a comprehensive summary of our contributions aimed at addressing the aforementioned aspect of privacy in machine learning. Additionally, we outline potential future directions for research, highlighting areas that could benefit from further exploration and improvement.

8.1 Discussions of Contributions

8.1.1 Privacy-preserving Generative Modeling

Significant progress has been made in incorporating DP guarantees into the training of deep generative models, yielding promising results in sanitizing high-dimensional samples for arbitrary downstream tasks [24, 32, 239, 245, 14]. Unfortunately, these methods still face challenges in producing high-fidelity sanitized data that is broadly useful in real-world scenarios, often due to inherent difficulties and practical engineering hurdles.

In conclusion, our proposed novel modifications to the DP training paradigms for deep generative models enhance previous methods by tackling the core challenges identified in this field: hyperparameter tuning and training stability, modeling complexity, as well as fragmented research, as introduced in Chapter 1. Accordingly, our solution fall into three aspects: sanitization scheme (Chapter 2), generation framework (Chapter 3), and a unified perspective (Chapter 4) of this problem.

- **Gradient Sanitization Scheme.** To alleviate the difficulty of hyperparameter search and to improve training stability, in Chapter 2, we introduce a novel gradient sanitization scheme that can be naturally integrated into the training of a generative adversarial network. Our primary insight posits that privacy-preserving training, often at the expense of utility, need only be applied to the generator—the component that will be released to the public. In contrast, the discriminator, which is often discarded post-training, can be optimally trained without privacy constraints. Furthermore, our approach effectively utilizes the Lipschitz property inherent in the discriminator from the Wasserstein training objective, enabling the achievement of precise sensitivity estimates without the need for exhaustive hyperparameter tuning.

- **Generation Framework.** We tackle the inherent modeling complexity of generating private high-dimensional data by proposing a new framework, as detailed in Chapter 3. Instead of training deep generative models with DP constraints, we directly optimize a small set of samples guided by discriminative information targeting downstream utility, which is a more attainable objective. We introduce a simple yet effective method to synthesizing a set of representative samples that reflects the original data for training downstream neural networks. Our findings challenge prevailing thinking and offer new insights to advance the field of private data generation.

- **Unified View.** We introduce a unified view, coupled with a novel taxonomy, that effectively characterizes existing approaches and integrates the otherwise fragmented research in DP deep generative modeling, as detailed in Chapter 4. Our taxonomy covers critical aspects such as threat models, general formulations, detailed descriptions, privacy analysis, as well as insights and broader implications, providing a holistic design surface for systematically exploring innovative methodologies and building upon the strengths of existing techniques. Furthermore, we present an in-depth introduction to the core principles of DP and generative modeling, enriched with substantial insights and discussions on key considerations for future research in this field.

8.1.2 Privacy Attacks and Defenses

While a significant amount of research on privacy attacks and defenses in machine learning systems has been presented in recent years, the pursuit of understanding and mitigating these issues continues due to their adversarial nature. Stronger attacks could emerge from efforts to devise strategies that are customized for each application scenario, while stronger defenses are required to counteract these attacks. The fundamental goal in analyzing privacy attacks is to augment their effectiveness and practicability, while the focus of designing defenses is to establish an ideal balance between privacy preservation and functional utility.

Our investigation encompasses both a systematic analysis of attacks against advanced generative models (Chapter 5) and the development of effective defenses on general discriminative models (Chapter 6).

- **Privacy Attack for Advanced Generative Models.** We provides a pioneering systematic analysis of membership inference attacks on diffusion models, the state-of-the-art generative backbone widely adopted in modern media editing tools. We uncover the key attack vectors relevant to the deployment of diffusion models in real-world contexts and introduce effective attack strategies, capitalizing on easily obtainable information to achieve robust performance across various settings. Our results highlight the significant potential privacy risk associated with diffusion models, with which we aim to motivate further research into related topics.
- **Defense against Privacy Attacks in Discriminative Models.** We introduce a novel training approach that effectively protects against membership inference attacks while preserving the utility of the target models. Our key insight is that membership privacy risks can be mitigated by reducing the discrepancy between training and testing loss distributions. This insight has led to the development of RelaxLoss, which modifies the training objective to a more attainable goal. Our approach provides effective protection and is straightforward to implement across various machine learning models, demonstrating its practicality and potential for widespread application.

8.1.3 Application

Despite the considerable progress achieved by recent research in privacy-preserving data generation, the deployment of these methods to address real-world, complex, high-dimensional data across different modalities continues to present a significant challenge. The adoption of these privacy techniques in sensitive domains, as well as their proper integration into various application scenarios, remains largely unexplored. Moreover, the unique and diverse characteristics of data in sensitive domains call for specialized advancements in representation, modeling, and evaluation methodologies—areas that the current body of research has not yet sufficiently addressed.

- **Privacy-preserving Generation of Gene Expression Data.** We present the first comprehensive analysis of gene expression data generation, covering five methodologies ranging from simple density estimators to advanced deep generative models. Our analysis utilizes a broad spectrum of metrics to evaluate the properties of the synthetic data, including statistical and biological characteristics, as well as its downstream utility. Our critical investigation highlights the necessity of such extensive evaluations to understand the limitations of current data generators. While most existing methods, including the most basic parametric Gaussian density estimation, can yield near-perfect downstream utility with strong DP guarantees, they often fail to preserve the statistical and biological characteristics of the original data. The challenge of generating DP synthetic data that preserves statistical fidelity and is biologically plausible remains an open issue, necessitating future research.

8.2 Future Directions

The research detailed in this thesis has shown significant promise in understanding and mitigating the privacy risks associated with the development and deployment of machine learning models. A central aspect of this advancement has been the exploration of viable solutions, notably privacy-preserving data generation, to tackle the data privacy challenge. However, there are still open challenges and opportunities for future development that broaden the scope of our current research and warrant further investigation. In the following subsections, we present a series of potential future research directions that we intend to explore.

8.2.1 Privacy-preserving Generative Modeling

Exploiting Public Knowledge. As privacy-preserving algorithms inherently suffer from high sample complexity, a promising future direction which holds significant practical relevance is the exploitation of public knowledge in training DP generative models. While recent studies have demonstrated great promise in improving private classification models [158, 159], privacy query release [125], DP generation [35, 126, 71, 133] and reported high-quality generation [62, 122] with the aid of such resources, current research relies heavily on strong assumptions about the public data distribution (e.g., assuming that public and private data come from the same distribution) while the specifics of its usefulness and the most effective way to utilize these resources are still unclear. Moreover, the challenges commonly linked with private learning on public data necessitate additional scrutiny [213]. In particular, the unique difficulties specific to generative modeling, such as a small tolerance for distribution shift, calls for more in-depth investigation. Future research should focus on practical cases where public data originates from diverse sources, presenting distribution shifts, and develop innovative methods to tackle these distributional disparities.

Task-specific Generation. Fitting a complete high-dimensional data distribution for general purpose is a complex task, and privacy constraints further exacerbate this challenge by increasing the model's data requirements. A principled trade-off emerges between the flexibility provided by general-purpose generative modeling and the utility of task-specific data generation. A natural solution to alleviate the complexity issue is to incorporate prior knowledge to simplify the task. For instance, one could utilize the knowledge of potential downstream tasks to direct the training of DP generation methods. This might involve using a task-dependent supervision loss, such as an additional classification loss for a downstream classification task. Furthermore, generating data specifically for defined tasks has the added advantage of being useful for predetermined benign applications, thus reducing the risk of potential unauthorized data misuse. While promising results have been achieved with DP generation of synthetic data for standard downstream tasks such as answering linear queries [73, 72], Bayesian estimation [134, 185], and training classification models [29], developing specialized solutions for different scenarios may necessitate significant modifications to the existing frameworks, highlighting a need for further research in this area.

Application in Multi-party Interaction. DP data generation finds a compelling use case in multi-party interactions, where the need for privacy-protected data exchange is paramount. For instance, DP data generation has shown promising potential in the realm of federated learning [10, 32, 256, 216], enabling tasks such as privacy-preserving data inspection and debugging that were previously infeasible due to privacy constraints. Task-specific DP generation can be particularly useful in these contexts, as it has shown potential in addressing non-iid issues and in contributing to the faster convergence of federated learning models [241, 228]. While these methods may still require substantial amounts of local data and computational resources from the participants, future development of efficient algorithms is anticipated to lead to promising results.

8.2.2 Privacy Attacks and Defense

Expanding Attack Surfaces. The recent proliferation of machine learning applications in everyday life inevitably leads to new vulnerabilities and expands potential attack surfaces, increasing the demand for enhanced privacy protection. For instance, the increasing need for training in a distributed or federated setting, which involves information exchange between parties, opens up possibilities for privacy attacks by malicious participants or untrusted central servers. Sensitive data could potentially be reconstructed from transmitted intermediate signals [263], which however, is hard to defend using existing solutions designed for the standard centralized training setting. While operations such as secure aggregation have been proposed, their protection level under potential strong privacy attacks is not fully understood. Furthermore, the evolving complexity of interactions between the service providers and data providers highlights the necessity for a more in-depth examination of privacy-preserving mechanisms. This may necessitate different levels of privacy protection beyond the customer data privacy, such as the model privacy of the service provider for intellectual property, which warrants future development of appropriate protection mechanisms that exhibit resilience even under advanced threat models.

Privacy Auditing. Assessing the privacy guarantees of privacy-preserving techniques via real-world attack simulations, commonly referred to as "privacy auditing" [85, 147], presents substantial potential and calls for the development of potent attacks. Specifically, while such techniques have been investigated in the context of auditing classification models, the exploration of computationally efficient attacks suitable for more complex models, such as the generative ones, remains an open challenge. This complexity primarily stems from several factors. Firstly, to fully leverage the attack capabilities permitted by the associated threat model, auditing attacks often require the repetitive training of multiple models on neighboring data subsets, a process that significantly increases computational demands. This necessitates the invention of efficient approximation methods. Moreover, generative models generally exhibit low sensitivity to privacy attacks [74, 34], resulting in less informative auditing results. These challenges highlight a pressing need for strategic design of more efficient attacks, particularly those tailored for the auditing of DP generative models.

8.2.3 Broader View

Our long-term objective is to develop trustworthy machine learning systems, wherein privacy-preserving learning is a pivotal element, anchoring the foundational trust in artificial intelligence systems. As we look to the future, our ambition is to delve into the broader field of these subjects, tackling other related challenges such as ethics, interpretability, robustness, and fairness.

Holistic Assessment and Design of Trustworthy Systems. The pursuit of trustworthiness in AI systems encompasses a range of facets, some of which may synergize and reinforce each other, while others might present conflicting objectives. This complex landscape necessitates a holistic approach in both the assessment and design phases, requiring a deep understanding and careful integration of these diverse elements. A key challenge lies in identifying and addressing both the aligned and conflicting dimensions. For instance, the trade-off between transparency and privacy in AI algorithms exemplifies such a conflict. To effectively tackle these challenges, future work should be committed to multidisciplinary approaches that integrate various elements to achieve an optimized balance.

Risks Related to Foundation Models. The future of developing effective machine learning systems is closely linked to the advancement of foundation models, i.e., large ML models trained on a vast quantity of data at scale and can be adapted to a wide range of downstream tasks. These models are crucial in establishing the backbone for numerous downstream applications, paving the way for a diverse range of AI technologies. Aiming for these foundation models to function in a trustworthy and privacy-preserving way, to conform to ethical standards, and to provide interpretability and fairness is crucial for future research and practice in the general field of machine learning. Initiatives that focus on aligning foundation models with ethical standards, enhancing methods for transparency and interpretability, and bolstering them against adversarial threats are of great importance and necessitate concerted efforts. Ultimately, the aim of this research is to develop foundation models that are not only technologically advanced but also resonate with the broader goal of creating AI that is as ethically responsible as it is revolutionary.

List of Algorithms

List of Figures

List of Tables

Bibliography

[1] Martin Abadi, Andy Chu, Ian Goodfellow, H Brendan McMahan, Ilya Mironov, Kunal Talwar, and Li Zhang. Deep learning with differential privacy. In *ACM SIGSAC Conference on Computer and Communications Security (CCS)*, 2016. Cited on pages 3, 14, 16, 18, 28, 29, 30, 41, 86, 90, 91, 97, 110, 118, 168, 169, 178, and 179.

[2] Nazmiye Ceren Abay, Yan Zhou, Murat Kantarcioglu, Bhavani Thuraisingham, and Latanya Sweeney. Privacy preserving synthetic data release using deep learning. In *Machine Learning and Knowledge Discovery in Databases: European Conference, ECML PKDD 2018*. Springer, 2019. Cited on pages 49 and 58.

[3] Gergely Acs, Luca Melis, Claude Castelluccia, and Emiliano De Cristofaro. Differentially private mixture of generative neural networks. *IEEE Transactions on Knowledge and Data Engineering*, 31(6), 2018. Cited on pages 49 and 58.

[4] Tejumade Afonja, Dingfan Chen, and Mario Fritz. Margctgan: A "marginally" better ctgan for the low sample regime. In *DAGM German Conference on Pattern Recognition (GCPR)*, 2023. Cited on pages 10, 112, and 113.

[5] Ahmed Alaa, Boris Van Breugel, Evgeny S Saveliev, and Mihaela van der Schaar. How faithful is your synthetic data? sample-level metrics for evaluating and auditing generative models. In *International Conference on Machine Learning (ICML)*. PMLR, 2022. Cited on page 62.

[6] Moustafa Alzantot and Mani Srivastava. Differential Privacy Synthetic Data Generation using WGANs, 2019. Cited on pages 49, 53, and 110.

[7] Simon Anders and Wolfgang Huber. Differential expression analysis for sequence count data. *Genome Biology*, 11(10), 2010. Cited on pages 107 and 113.

[8] Martin Arjovsky, Soumith Chintala, and Léon Bottou. Wasserstein generative adversarial networks. In *International Conference on Machine Learning (ICML)*, 2017. Cited on pages 4, 14, 18, 46, 56, and 110.

[9] Alemu Takele Assefa, Jo Vandesompele, and Olivier Thas. Spsimseq: semi-parametric simulation of bulk and single-cell rna-sequencing data. *Bioinformatics*, 36(10), 2020. Cited on page 106.

[10] Sean Augenstein, H. Brendan McMahan, Daniel Ramage, Swaroop Ramaswamy, Peter Kairouz, Mingqing Chen, Rajiv Mathews, and Blaise Agüera y Arcas. Generative models for effective ML on private, decentralized datasets. In *International Conference on Learning Representations (ICLR)*, 2020. Cited on pages 15, 19, 22, 23, 61, 131, and 161.

[11] Olivier Bachem, Mario Lucic, and Andreas Krause. Practical coreset constructions for machine learning. *arXiv preprint arXiv:1703.06476*, 2017. Cited on page 60.

[12] Borja Balle, Gilles Barthe, and Marco Gaboardi. Privacy amplification by subsampling: Tight analyses via couplings and divergences. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2018. Cited on page 19.

[13] Borja Balle, Gilles Barthe, Marco Gaboardi, Justin Hsu, and Tetsuya Sato. Hypothesis testing interpretations and renyi differential privacy. In *International Conference on Artificial Intelligence and Statistics (AISTATS)*. PMLR, 2020. Cited on page 40.

[14] Brett K Beaulieu-Jones, Zhiwei Steven Wu, Chris Williams, and Casey S Greene. Privacy-preserving generative deep neural networks support clinical data sharing. biorxiv. *DOI*, 10, 2017. Cited on pages 2, 13, 14, 16, 26, 38, 49, 53, and 128.

[15] Eden Belouadah and Adrian Popescu. Il2m: Class incremental learning with dual memory. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, 2019. Cited on page 33.

[16] James Bennett, Stan Lanning, et al. The netflix prize. In *Proceedings of KDD cup and workshop*, volume 2007. Citeseer, 2007. Cited on page 13.

[17] Alex Bie, Gautam Kamath, and Guojun Zhang. Private gans, revisited. *arXiv preprint arXiv:2302.02936*, February 2023. Cited on pages 49 and 55.

[18] Mikołaj Bińkowski, Dougal J. Sutherland, Michael Arbel, and Arthur Gretton. Demystifying MMD GANs. In *International Conference on Learning Representations (ICLR)*, 2018. Cited on pages 46 and 50.

[19] Avrim Blum, Katrina Ligett, and Aaron Roth. A learning theory approach to noninteractive database privacy. *Journal of the ACM (JACM)*, 60(2), 2013. Cited on pages 2, 14, 26, 29, and 38.

[20] Nicolas Bonneel, Julien Rabin, Gabriel Peyré, and Hanspeter Pfister. Sliced and radon wasserstein barycenters of measures. *Journal of Mathematical Imaging and Vision (JMIV)*, 51, 2015. Cited on page 51.

[21] Andrew Brock, Jeff Donahue, and Karen Simonyan. Large scale gan training for high fidelity natural image synthesis. In *International Conference on Learning Representations (ICLR)*, 2018. Cited on page 20.

[22] Zhipeng Cai, Zuobin Xiong, Honghui Xu, Peng Wang, Wei Li, and Yi Pan. Generative adversarial networks: A survey toward private and secure applications. *ACM Computing Surveys (CSUR)*, 54(6), 2021. Cited on page 60.

[23] Sebastian Caldas, Peter Wu, Tian Li, Jakub Konečnỳ, H Brendan McMahan, Virginia Smith, and Ameet Talwalkar. Leaf: A benchmark for federated settings. *arXiv preprint arXiv:1812.01097*, 2018. Cited on pages 23 and 161.

[24] Tianshi Cao, Alex Bie, Arash Vahdat, Sanja Fidler, and Karsten Kreis. Don't generate me: Training differentially private generative models with sinkhorn divergence. *Advances in Neural Information Processing Systems (NeurIPS)*, 2021. Cited on pages 2, 3, 13, 26, 29, 30, 38, 49, 56, 128, 169, and 177.

[25] Nicholas Carlini, Steve Chien, Milad Nasr, Shuang Song, Andreas Terzis, and Florian Tramer. Membership inference attacks from first principles. In *IEEE Symposium on Security and Privacy (SP)*. IEEE, 2022. Cited on pages 66, 72, 76, and 83.

[26] Nicholas Carlini, Florian Tramer, Eric Wallace, Matthew Jagielski, Ariel Herbert-Voss, Katherine Lee, Adam Roberts, Tom Brown, Dawn Song, Ulfar Erlingsson, et al. Extracting training data from large language models. In *USENIX Security Symposium (USENIX Security)*, 2021. Cited on pages 5, 66, and 84.

[27] Nicolas Carlini, Jamie Hayes, Milad Nasr, Matthew Jagielski, Vikash Sehwag, Florian Tramer, Borja Balle, Daphne Ippolito, and Eric Wallace. Extracting training data from diffusion models. In *USENIX Security Symposium (USENIX Security)*, 2023. Cited on pages 5, 66, 68, 84, and 85.

[28] Thee Chanyaswad, Changchang Liu, and Prateek Mittal. Ron-gauss: Enhancing utility in non-interactive private data release. *Proceedings on Privacy Enhancing Technologies (PETs)*, 1, 2019. Cited on page 108.

[29] Dingfan Chen, Raouf Kerkouche, and Mario Fritz. Private set generation with discriminative information. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2022. Cited on pages 8, 9, 25, 49, 54, 55, 60, 62, 125, and 130.

[30] Dingfan Chen, Raouf Kerkouche, and Mario Fritz. A unified view of differentially private deep generative modeling. *arXiv preprint arXiv:2309.15696*, 2023. Cited on pages 8, 9, and 37.

[31] Dingfan Chen*, Marie Oestreich*, Tejumade Afonja, Raouf Kerkouche, Matthias Becker, and Mario Fritz. Towards biologically plausible and private gene expression data generation. In *Proceedings on Privacy Enhancing Technologies (PoPETs)*, 2024. Cited on pages 9, 10, and 104.

[32] Dingfan Chen, Tribhuvanesh Orekondy, and Mario Fritz. Gs-wgan: A gradient-sanitized approach for learning differentially private generators. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2020. Cited on pages 2, 3, 8, 9, 13, 26, 29, 30, 34, 38, 49, 56, 57, 62, 128, 131, 168, 169, 170, and 177.

[33] Dingfan Chen, Ning Yu, and Mario Fritz. Relaxloss: Defending membership inference attacks without losing utility. In *International Conference on Learning Representations (ICLR)*, 2022. Cited on pages 9, 10, and 89.

[34] Dingfan Chen, Ning Yu, Yang Zhang, and Mario Fritz. Gan-leaks: A taxonomy of membership inference attacks against generative models. In *ACM SIGSAC Conference on Computer and Communications Security (CCS)*, 2020. Cited on pages 5, 10, 16, 62, 66, 68, 71, 74, 75, 77, 78, 79, 80, 83, 84, 85, 90, 91, 124, 131, and 181.

[35] Dongjie Chen, Sen-ching Samson Cheung, Chen-Nee Chuah, and Sally Ozonoff. Differentially private generative adversarial networks with model inversion. In *IEEE International Workshop on Information Forensics and Security (WIFS)*. IEEE, 2021. Cited on pages 49, 55, 60, and 130.

[36] Jia-Wei Chen, Chia-Mu Yu, Ching-Chia Kao, Tzai-Wei Pang, and Chun-Shien Lu. Dpgen: Differentially private generative energy-guided network for natural image synthesis. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022. Cited on pages 49 and 173.

[37] Qingrong Chen, Chong Xiang, Minhui Xue, Bo Li, Nikita Borisov, Dali Kaarfar, and Haojin Zhu. Differentially private data generative models. *arXiv preprint arXiv:1812.02274*, 2018. Cited on pages 49 and 58.

[38] Christopher A Choquette Choo, Florian Tramer, Nicholas Carlini, and Nicolas Papernot. Label-only membership inference attacks. In *International conference on machine learning (ICML)*. PMLR, 2021. Cited on pages 90, 91, 96, 97, and 98.

[39] Ana Conesa, Pedro Madrigal, Sonia Tarazona, David Gomez-Cabrero, Alejandra Cervera, Andrew McPherson, Michał Wojciech Szcześniak, Daniel J Gaffney, Laura L Elo, Xuegong Zhang, and et al. A survey of best practices for rna-seq data analysis. *Genome Biology*, 17(1), 2016. Cited on page 104.

[40] Juliana Costa-Silva, Douglas Domingues, and Fabricio Martins Lopes. Rna-seq differential expression analysis: An extended review and a software tool. *Plos One*, 12(12), 2017. Cited on page 107.

[41] Felix Dangel, Frederik Kunstner, and Philipp Hennig. Backpack: Packing more into backprop. In *International Conference on Learning Representations (ICLR)*, 2020. Cited on pages 5, 91, and 100.

[42] Soham De, Leonard Berrada, Jamie Hayes, Samuel L Smith, and Borja Balle. Unlocking high-accuracy differentially private image classification through scale. *arXiv preprint arXiv:2204.13650*, 2022. Cited on pages 58 and 59.

[43] Emiliano De Cristofaro. What is synthetic data? the good, the bad, and the ugly. *arXiv preprint arXiv:2303.01230*, 2023. Cited on page 60.

[44] Laurence de Torrenté, Samuel Zimmerman, Masako Suzuki, Maximilian Christopeit, John M Greally, and Jessica C Mar. The shape of gene expression distributions matter: how incorporating distribution shape improves the interpretation of cancer transcriptomic data. *BMC Bioinformatics*, 21, 2020. Cited on page 114.

[45] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2009. Cited on pages 13 and 74.

[46] Prafulla Dhariwal and Alexander Nichol. Diffusion models beat GANs on image synthesis. *Advances in Neural Information Processing Systems (NeurIPS)*, 2021. Cited on pages 66, 67, 70, and 75.

[47] Tim Dockhorn, Tianshi Cao, Arash Vahdat, and Karsten Kreis. Differentially private diffusion models. *arXiv preprint arXiv:2210.09929*, 2022. Cited on pages 49, 58, 59, 86, and 173.

[48] Tian Dong, Bo Zhao, and Lingjuan Lyu. Privacy for free: How does dataset condensation help privacy? In *International Conference on Machine Learning (ICML)*, 2022. Cited on page 29.

[49] Haonan Duan, Adam Dziedzic, Nicolas Papernot, and Franziska Boenisch. Flocks of stochastic parrots: Differentially private prompt learning for large language models. *arXiv preprint arXiv:2305.15594*, 2023. Cited on page 61.

[50] Jinhao Duan, Fei Kong, Shiqi Wang, Xiaoshuang Shi, and Kaidi Xu. Are diffusion models vulnerable to membership inference attacks? *arXiv preprint arXiv:2302.01316*, 2023. Cited on pages 68, 84, and 85.

[51] Cynthia Dwork. Differential privacy: A survey of results. In *International Conference on Theory and Applications of Models of Computation (TAMC)*. Springer, 2008. Cited on pages 2, 13, 15, 25, 26, 38, 39, and 91.

[52] Cynthia Dwork, Moni Naor, Omer Reingold, Guy N Rothblum, and Salil Vadhan. On the complexity of differentially private data release: efficient algorithms and hardness results. In *Proceedings of the forty-first annual ACM symposium on Theory of computing (STOC)*, 2009. Cited on pages 2, 13, 25, and 38.

[53] Cynthia Dwork, Aaron Roth, et al. The algorithmic foundations of differential privacy. *Foundations and Trends® in Theoretical Computer Science*, 9(3–4), 2014. Cited on pages 16, 27, 40, 86, 91, 107, and 173.

[54] Cynthia Dwork, Guy N Rothblum, and Salil Vadhan. Boosting and differential privacy. In *IEEE 51st Annual Symposium on Foundations of Computer Science (FOCS)*, 2010. Cited on page 14.

[55] Liyue Fan. A survey of differentially private generative adversarial networks. In *The AAAI Workshop on Privacy-Preserving Artificial Intelligence*, page 8, 2020. Cited on page 60.

[56] Stanislav Fort, Andrew Brock, Razvan Pascanu, Soham De, and Samuel L Smith. Drawing multiple augmentation samples per image during training efficiently decreases test error. *arXiv preprint arXiv:2105.13343*, 2021. Cited on pages 58 and 59.

[57] Lorenzo Frigerio, Anderson Santana de Oliveira, Laurent Gomez, and Patrick Duverger. Differentially private generative adversarial networks for time series, continuous, and discrete open data. In *IFIP International Conference on ICT Systems Security and Privacy Protection*. Springer, 2019. Cited on pages 14, 49, and 53.

[58] Benjamin CM Fung, Ke Wang, Rui Chen, and Philip S Yu. Privacy-preserving data publishing: A survey of recent developments. *ACM Computing Surveys (Csur)*, 42(4), 2010. Cited on pages 2, 13, 25, and 38.

[59] Georgi Ganev, Bristena Oprisanu, and Emiliano De Cristofaro. Robin hood and matthew effects: Differential privacy has disparate impact on synthetic data. In *International Conference on Machine Learning (ICLR)*. PMLR, 2022. Cited on page 62.

[60] Stuart Geman, Elie Bienenstock, and René Doursat. Neural networks and the bias/variance dilemma. *Neural Computation*, 4(1), 1992. Cited on page 90.

[61] Aude Genevay, Gabriel Peyré, and Marco Cuturi. Learning generative models with sinkhorn divergences. In *International Conference on Artificial Intelligence and Statistics (AISTATS)*. PMLR, 2018. Cited on page 46.

[62] Sahra Ghalebikesabi, Leonard Berrada, Sven Gowal, Ira Ktena, Robert Stanforth, Jamie Hayes, Soham De, Samuel L Smith, Olivia Wiles, and Borja Balle. Differentially private diffusion models generate useful synthetic images. *arXiv preprint arXiv:2302.13861*, 2023. Cited on pages 38, 49, 58, 59, 60, and 130.

[63] Jennifer Gillenwater, Matthew Joseph, and Alex Kulesza. Differentially private quantiles. In *International Conference on Machine Learning (ICML)*. PMLR, 2021. Cited on page 116.

[64] Ian Goodfellow. Efficient per-example gradient computations. *arXiv preprint arXiv:1510.01799*, 2015. Cited on pages 5, 91, and 100.

[65] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. In *Advances in neural information processing systems (NeurIPS)*, 2014. Cited on pages 14, 16, 46, 67, 90, 91, and 110.

[66] Ian J Goodfellow, Jonathon Shlens, and Christian Szegedy. Explaining and harnessing adversarial examples. In *International Conference on Learning Representations (ICLR)*, 2015. Cited on pages 90 and 91.

[67] Ishaan Gulrajani, Faruk Ahmed, Martin Arjovsky, Vincent Dumoulin, and Aaron C Courville. Improved training of wasserstein gans. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2017. Cited on pages 4, 14, 18, 46, 53, 56, and 161.

[68] Chuan Guo, Geoff Pleiss, Yu Sun, and Kilian Q Weinberger. On calibration of modern neural networks. In *International Conference on Machine Learning (ICML)*. PMLR, 2017. Cited on pages 91 and 97.

[69] Frederik Harder, Kamil Adamczewski, and Mijung Park. Differentially private mean embeddings with random features (dp-merf) for simple & practical synthetic data generation. *arXiv preprint arXiv:2002.11603*, 2020. Cited on page 20.

[70] Frederik Harder, Kamil Adamczewski, and Mijung Park. Dp-merf: Differentially private mean embeddings with randomfeatures for practical privacy-preserving data generation. In *International conference on artificial intelligence and statistics (AISTAT)*. PMLR, 2021. Cited on pages 3, 26, 30, 38, 49, 50, 51, 61, 169, 170, and 174.

[71] Fredrik Harder, Milad Jalali Asadabadi, Danica J Sutherland, and Mijung Park. Differentially private data generation needs better features. *arXiv preprint arXiv:2205.12900*, 2022. Cited on pages 49, 51, 60, 130, and 174.

[72] Moritz Hardt, Katrina Ligett, and Frank McSherry. A simple and practical algorithm for differentially private data release. *Advances in neural information processing systems (NeurIPS)*, 2012. Cited on pages 29, 60, and 130.

[73] Moritz Hardt and Guy N Rothblum. A multiplicative weights mechanism for privacy-preserving data analysis. In *IEEE 51st Annual Symposium on Foundations of Computer Science (FOCS)*, 2010. Cited on pages 2, 14, 26, 29, 38, 60, and 130.

[74] Jamie Hayes, Luca Melis, George Danezis, and Emiliano De Cristofaro. Logan: Membership inference attacks against generative models. *Proceedings on Privacy Enhancing Technologies (PETs)*, 2019. Cited on pages 5, 62, 66, 68, 75, 76, 78, 82, 90, 91, and 131.

[75] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. In *Proceedings of the IEEE international conference on computer vision (ICCV)*, 2015. Cited on page 30.

[76] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition (CVPR)*, 2016. Cited on page 96.

[77] Martin Heusel, Hubert Ramsauer, Thomas Unterthiner, Bernhard Nessler, and Sepp Hochreiter. Gans trained by a two time-scale update rule converge to a local nash equilibrium. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2017. Cited on pages 19 and 161.

[78] Benjamin Hilprecht, Martin Härterich, and Daniel Bernau. Monte Carlo and reconstruction membership inference attacks against generative models. *Proceedings on Privacy Enhancing Technologies (PETs)*, 2019(4), 2019. Cited on pages 66, 68, 71, 74, 75, 78, 80, and 124.

[79] Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531*, 2015. Cited on page 97.

[80] Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. *Advances in Neural Information Processing Systems (NeurIPS)*, 2020. Cited on pages 47, 66, 67, 69, and 70.

[81] Florimond Houssiau, James Jordon, Samuel N Cohen, Owen Daniel, Andrew Elliott, James Geddes, Callum Mole, Camila Rangel-Smith, and Lukasz Szpruch. Tapas: a toolbox for adversarial privacy auditing of synthetic data. *arXiv preprint arXiv:2211.06550*, 2022. Cited on page 62.

[82] Hailong Hu and Jun Pang. Membership inference of diffusion models. *arXiv preprint arXiv:2301.09956*, 2023. Cited on pages 68, 71, 74, 75, 84, and 85.

[83] Yuzheng Hu, Fan Wu, Qinbin Li, Yunhui Long, Gonzalo Munilla Garrido, Chang Ge, Bolin Ding, David Forsyth, Bo Li, and Dawn Song. Sok: Privacy-preserving data synthesis. *arXiv preprint arXiv:2307.02106*, 2023. Cited on page 60.

[84] Bo Hui, Yuchen Yang, Haolin Yuan, Philippe Burlina, Neil Zhenqiang Gong, and Yinzhi Cao. Practical blind membership inference attack via differential comparisons. In *The Network and Distributed System Security Symposium (NDSS)*, 2021. Cited on page 91.

[85] Matthew Jagielski, Jonathan Ullman, and Alina Oprea. Auditing differentially private machine learning: How private is private sgd? *Advances in Neural Information Processing Systems (NeurIPS)*, 2020. Cited on pages 62 and 131.

[86] Bargav Jayaraman and David Evans. Evaluating differentially private machine learning in practice. In *28th USENIX Security Symposium (USENIX Security)*, 2019. Cited on pages 5, 90, and 91.

[87] Jinyuan Jia, Ahmed Salem, Michael Backes, Yang Zhang, and Neil Zhenqiang Gong. Memguard: Defending against black-box membership inference attacks via adversarial examples. In *ACM SIGSAC Conference on Computer and Communications Security (CCS)*, 2019. Cited on pages 5, 90, 91, 92, 96, and 97.

[88] Dihong Jiang and Sun Sun. Dp-lflow: Differentially private latent flow for scalable sensitive image generation. *ICML Deployable Generative AI Workshop*, 2023. Cited on pages 49 and 58.

[89] Dihong Jiang, Guojun Zhang, Mahdi Karami, Xi Chen, Yunfeng Shao, and Yaoliang Yu. Dp^2-vae: Differentially private pre-trained variational autoencoders. *arXiv preprint arXiv:2208.03409*, 2022. Cited on page 49.

[90] James Jordon, Lukasz Szpruch, Florimond Houssiau, Mirko Bottarelli, Giovanni Cherubin, Carsten Maple, Samuel N Cohen, and Adrian Weller. Synthetic data–what, why and how? *arXiv preprint arXiv:2205.03257*, 2022. Cited on page 60.

[91] Peter Kairouz, Sewoong Oh, and Pramod Viswanath. The composition theorem for differential privacy. In *International conference on machine learning (ICML)*. PMLR, 2015. Cited on page 189.

[92] Tero Karras, Timo Aila, Samuli Laine, and Jaakko Lehtinen. Progressive growing of GANs for improved quality, stability, and variation. In *International Conference on Learning Representations (ICLR)*, 2018. Cited on page 83.

[93] Tero Karras, Miika Aittala, Janne Hellsten, Samuli Laine, Jaakko Lehtinen, and Timo Aila. Training generative adversarial networks with limited data. *Advances in Neural Information Processing Systems (NeurIPS)*, 2020. Cited on page 26.

[94] Tero Karras, Samuli Laine, and Timo Aila. A style-based generator architecture for generative adversarial networks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019. Cited on page 83.

[95] Tero Karras, Samuli Laine, Miika Aittala, Janne Hellsten, Jaakko Lehtinen, and Timo Aila. Analyzing and improving the image quality of stylegan. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition (CVPR)*, 2020. Cited on page 61.

[96] Jakob Nikolas Kather, Cleo-Aron Weis, Francesco Bianconi, Susanne M Melchers, Lothar R Schad, Timo Gaiser, Alexander Marx, and Frank Gerrit Zöllner. Multi-class texture analysis in colorectal cancer histology. *Scientific reports*, 6(1), 2016. Cited on page 191.

[97] Yigitcan Kaya and Tudor Dumitras. When does data augmentation help with membership inference attacks? In *International Conference on Machine Learning (ICML)*. PMLR, 2021. Cited on pages 5, 90, and 91.

[98] Yigitcan Kaya, Sanghyun Hong, and Tudor Dumitras. On the effectiveness of regularization against membership inference attacks. *arXiv preprint arXiv:2006.05336*, 2020. Cited on page 90.

[99] Diederik Kingma, Tim Salimans, Ben Poole, and Jonathan Ho. Variational diffusion models. *Advances in neural information processing systems (NeurIPS)*, 2021. Cited on page 75.

[100] Diederik P. Kingma and Max Welling. Auto-Encoding Variational Bayes. In *International Conference on Learning Representations (ICLR)*, 2014. Cited on pages 47 and 109.

[101] Diederik P. Kingma and Max Welling. Auto-encoding variational Bayes. In *International Conference on Learning Representations (ICLR)*, 2014. Cited on page 67.

[102] Durk P Kingma and Prafulla Dhariwal. Glow: Generative flow with invertible 1x1 convolutions. *Advances in neural information processing systems (NeurIPS)*, 2018. Cited on page 47.

[103] Fei Kong, Jinhao Duan, RuiPeng Ma, Hengtao Shen, Xiaofeng Zhu, Xiaoshuang Shi, and Kaidi Xu. An efficient membership inference attack for the diffusion model by proximal initialization. *arXiv preprint arXiv:2305.18355*, 2023. Cited on pages 68, 84, and 85.

[104] Alex Krizhevsky, Geoffrey Hinton, et al. Learning multiple layers of features from tiny images. *Master's thesis, Department of Computer Science, University of Toronto*, 2009. Cited on pages 171, 182, and 190.

[105] Snehalika Lall, Sumanta Ray, and Sanghamitra Bandyopadhyay. Lsh-gan enables in-silico generation of cells for small sample high dimensional scrna-seq data. *Communications Biology*, 5(1), 2022. Cited on pages 105 and 106.

[106] Peter Langfelder and Steve Horvath. Wgcna: an r package for weighted correlation network analysis. *BMC Bioinformatics*, 9, 2008. Cited on page 107.

[107] Hugo Larochelle and Iain Murray. The neural autoregressive distribution estimator. In *International conference on artificial intelligence and statistics (AISTATS)*. JMLR Workshop and Conference Proceedings, 2011. Cited on page 47.

[108] Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11), 1998. Cited on pages 19, 30, 161, and 168.

[109] Shiye Lei and Dacheng Tao. A comprehensive survey to dataset distillation. *arXiv preprint arXiv:2301.05603*, 2023. Cited on page 25.

[110] David D Lewis, Yiming Yang, Tony G Rose, and Fan Li. Rcv1: A new benchmark collection for text categorization research. *Journal of Machine Learning Research (JMLR)*, 5(Apr), 2004. Cited on page 13.

[111] Chun-Liang Li, Wei-Cheng Chang, Yu Cheng, Yiming Yang, and Barnabás Póczos. Mmd gan: Towards deeper understanding of moment matching network. *Advances in neural information processing systems (NeurIPS)*, 2017. Cited on pages 46 and 50.

[112] Chunyuan Li, Hao Liu, Changyou Chen, Yunchen Pu, Liqun Chen, Ricardo Henao, and Lawrence Carin. Alice: Towards understanding adversarial learning for joint distribution matching. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2017. Cited on pages 19 and 161.

[113] Jeffrey Li, Mikhail Khodak, Sebastian Caldas, and Ameet Talwalkar. Differentially private meta-learning. In *International Conference on Learning Representations (ICLR)*, 2020. Cited on pages 4 and 14.

[114] Junnan Li, Richard Socher, and Steven CH Hoi. Dividemix: Learning with noisy labels as semi-supervised learning. In *International Conference on Learning Representations (ICLR)*, 2019. Cited on page 95.

[115] Ninghui Li, Min Lyu, Dong Su, and Weining Yang. Differential privacy: From theory to practice. *Synthesis Lectures on Information Security, Privacy, & Trust*, 8(4), 2016. Cited on page 14.

[116] Ninghui Li, Wahbeh Qardaji, and Dong Su. On sampling, anonymization, and differential privacy or, k-anonymization meets differential privacy. In *ACM Symposium on Information, Computer and Communications Security (ASIACCS)*, 2012. Cited on page 19.

[117] Xinmin Li and Cun-Yu Wang. From bulk, single-cell to spatial rna sequencing. *International journal of oral science*, 13(1), 2021. Cited on page 106.

[118] Xuechen Li, Florian Tramer, Percy Liang, and Tatsunori Hashimoto. Large language models can be strong differentially private learners. In *International Conference on Learning Representations (ICLR)*, 2022. Cited on pages 58 and 61.

[119] Zhizhong Li and Derek Hoiem. Learning without forgetting. *IEEE transactions on pattern analysis and machine intelligence (TPAMI)*, 40(12), 2017. Cited on page 32.

[120] Seng Pei Liew, Tsubasa Takahashi, and Michihiko Ueno. Pearl: Data synthesis via private embeddings and adversarial reconstruction learning. In *International Conference on Learning Representations*, 2021. Cited on pages 49, 51, and 174.

[121] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. Microsoft COCO: Common objects in context. In *Computer Vision–ECCV 2014: 13th European Conference*. Springer, 2014. Cited on page 86.

[122] Zinan Lin, Sivakanth Gopi, Janardhan Kulkarni, Harsha Nori, and Sergey Yekhanin. Differentially private synthetic data via foundation model apis 1: Images. *arXiv preprint arXiv:2305.15560*, 2023. Cited on pages 49, 60, 130, 173, and 175.

[123] Zinan Lin, Ashish Khetan, Giulia Fanti, and Sewoong Oh. Pacgan: The power of two samples in generative adversarial networks. *Advances in neural information processing systems (NeurIPS)*, 2018. Cited on page 189.

[124] Zinan Lin, Vyas Sekar, and Giulia Fanti. On the privacy properties of gan-generated samples. In *International Conference on Artificial Intelligence and Statistics (AISTATS)*. PMLR, 2021. Cited on page 189.

[125] Terrance Liu, Giuseppe Vietri, Thomas Steinke, Jonathan Ullman, and Steven Wu. Leveraging public data for practical private query release. In *International Conference on Machine Learning (ICML)*. PMLR, 2021. Cited on page 130.

[126] Terrance Liu, Giuseppe Vietri, and Steven Z Wu. Iterative methods for private synthetic data: Unifying framework and new methods. *Advances in Neural Information Processing Systems (NeurIPS)*, 2021. Cited on pages 60 and 130.

[127] Yaoyao Liu, Yuting Su, An-An Liu, Bernt Schiele, and Qianru Sun. Mnemonics training: Multi-class incremental learning without forgetting. In *Proceedings of the IEEE/CVF conference on Computer Vision and Pattern Recognition (CVPR)*, 2020. Cited on page 29.

[128] Ziwei Liu, Ping Luo, Xiaogang Wang, and Xiaoou Tang. Deep learning face attributes in the wild. In *Proceedings of International Conference on Computer Vision (ICCV)*, 2015. Cited on page 181.

[129] Yunhui Long, Suxin Lin, Zhuolin Yang, Carl A Gunter, and Bo Li. Scalable differentially private generative student model via pate. *arXiv preprint arXiv:1906.09338*, 2019. Cited on pages 3, 15, 20, 38, 49, 56, and 57.

[130] Yunhui Long, Boxin Wang, Zhuolin Yang, Bhavya Kailkhura, Aston Zhang, Carl A. Gunter, and Bo Li. G-PATE: Scalable differentially private data generator via private aggregation of teacher discriminators. In A. Beygelzimer, Y. Dauphin, P. Liang, and J. Wortman Vaughan, editors, *Advances in Neural Information Processing Systems (NeurIPS)*, 2021. Cited on pages 29, 30, 168, 169, and 170.

[131] Michael I Love, Wolfgang Huber, and Simon Anders. Moderated estimation of fold change and dispersion for rna-seq data with deseq2. *Genome Biology*, 15(12), 2014. Cited on pages 113 and 115.

[132] Mario Lucic, Karol Kurach, Marcin Michalski, Sylvain Gelly, and Olivier Bousquet. Are gans created equal? a large-scale study. *Advances in neural information processing systems (NeurIPS)*, 2018. Cited on page 62.

[133] Saiyue Lyu, Margarita Vinaroz, Michael F Liu, and Mijung Park. Differentially private latent diffusion models. *arXiv preprint arXiv:2305.15759*, 2023. Cited on pages 49, 58, 60, and 130.

[134] Dionysis Manousakas, Zuheng Xu, Cecilia Mascolo, and Trevor Campbell. Bayesian pseudo-coresets. *Advances in Neural Information Processing Systems (NeurIPS)*, 2020. Cited on pages 60 and 130.

[135] Mohamed Marouf, Pierre Machart, Vikas Bansal, Christoph Kilian, Daniel S Magruder, Christian F Krebs, and Stefan Bonn. Realistic in silico generation and augmentation of single-cell rna-seq data using generative adversarial networks. *Nature communications*, 11, 2020. Cited on pages 105 and 106.

[136] Tomoya Matsumoto, Takayuki Miura, and Naoto Yanai. Membership inference attacks against diffusion models. *arXiv preprint arXiv:2302.03262*, 2023. Cited on pages 68, 71, 75, 84, and 85.

[137] Justus Mattern, Zhijing Jin, Benjamin Weggenmann, Bernhard Schoelkopf, and Mrinmaya Sachan. Differentially private language models for secure data sharing. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, 2022. Cited on page 58.

[138] Ryan Mckenna, Daniel Sheldon, and Gerome Miklau. Graphical-model based estimation and inference for differential privacy. In *International Conference on Machine Learning (ICML)*. PMLR, 2019. Cited on pages 14 and 110.

[139] Brendan McMahan, Daniel Ramage, Kunal Talwar, and Li Zhang. Learning differentially private recurrent language models. In *International Conference on Learning Representations (ICLR)*, 2018. Cited on page 15.

[140] H Brendan McMahan, Eider Moore, Daniel Ramage, Seth Hampson, et al. Communication-efficient learning of deep networks from decentralized data. In *International Conference on Artificial Intelligence and Statistics (AISTATS)*, 2017. Cited on page 19.

[141] H. Brendan McMahan, Daniel Ramage, Kunal Talwar, and Li Zhang. Learning differentially private recurrent language models. In *International Conference on Learning Representations (ICLR)*, 2018. Cited on page 58.

[142] Ilya Mironov. Rényi differential privacy. In *IEEE 30th Computer Security Foundations Symposium (CSF)*, 2017. Cited on pages 15, 16, 40, 157, and 165.

[143] Ilya Mironov, Kunal Talwar, and Li Zhang. Rényi differential privacy of the sampled gaussian mechanism. *arXiv preprint arXiv:1908.10530*, 2019. Cited on pages 28, 29, 165, 168, 178, and 179.

[144] Mehdi Mirza and Simon Osindero. Conditional generative adversarial nets. *arXiv preprint arXiv:1411.1784*, 2014. Cited on page 61.

[145] Baharan Mirzasoleiman, Jeff Bilmes, and Jure Leskovec. Coresets for data-efficient training of machine learning models. In *International Conference on Machine Learning (ICML)*. PMLR, 2020. Cited on page 29.

[146] Rafael Müller, Simon Kornblith, and Geoffrey Hinton. When does label smoothing help? *Advances in neural information processing systems (NeurIPS)*, 2019. Cited on pages 91 and 97.

[147] Milad Nasr, Jamie Hayes, Thomas Steinke, Borja Balle, Florian Tramèr, Matthew Jagielski, Nicholas Carlini, and Andreas Terzis. Tight auditing of differentially private machine learning. In *USENIX Security Symposium (USENIX Security)*, 2023. Cited on pages 62 and 131.

[148] Milad Nasr, Reza Shokri, and Amir Houmansadr. Machine learning with membership privacy using adversarial regularization. In *ACM SIGSAC Conference on Computer and Communications Security (CCS)*, 2018. Cited on pages 90, 91, 92, 96, 97, and 191.

[149] Milad Nasr, Reza Shokri, and Amir Houmansadr. Comprehensive privacy analysis of deep learning: Passive and active white-box inference attacks against centralized and federated learning. In *IEEE symposium on security and privacy (SP)*. IEEE, 2019. Cited on pages 66, 68, 91, and 96.

[150] Alexander Quinn Nichol and Prafulla Dhariwal. Improved denoising diffusion probabilistic models. In *International Conference on Machine Learning (ICML)*. PMLR, 2021. Cited on pages 66, 70, and 75.

[151] Sebastian Nowozin, Botond Cseke, and Ryota Tomioka. f-gan: Training generative neural samplers using variational divergence minimization. *Advances in neural information processing systems (NeurIPS)*, 2016. Cited on page 46.

[152] Augustus Odena, Christopher Olah, and Jonathon Shlens. Conditional image synthesis with auxiliary classifier gans. In *International conference on machine learning (ICML)*. PMLR, 2017. Cited on pages 53 and 61.

[153] Marie Oestreich, Dingfan Chen, Joachim L Schultze, Mario Fritz, and Matthias Becker. Privacy considerations for sharing genomics data. *EXCLI journal*, 20, 2021. Cited on pages 10, 104, and 105.

[154] Marie Oestreich, Lisa Holsten, Shobhit Agrawal, Kilian Dahm, Philipp Koch, Han Jin, Matthias Becker, and Thomas Ulas. hcocena: horizontal integration and analysis of transcriptomics datasets. *Bioinformatics*, 38, 2022. Cited on pages 107 and 114.

[155] Aaron van den Oord, Sander Dieleman, Heiga Zen, Karen Simonyan, Oriol Vinyals, Alex Graves, Nal Kalchbrenner, Andrew Senior, and Koray Kavukcuoglu. Wavenet: A generative model for raw audio. *arXiv preprint arXiv:1609.03499*, 2016. Cited on page 47.

[156] Diksha Pandey and Perumal P Onkara. Improved downstream functional analysis of single-cell rna-sequence data using dgan. *Scientific Reports*, 13(1), 2023. Cited on pages 105 and 106.

[157] Yan Pang, Tianhao Wang, Xuhui Kang, Mengdi Huai, and Yang Zhang. White-box membership inference attacks against diffusion models. *arXiv preprint arXiv:2308.06405*, 2023. Cited on page 68.

[158] Nicolas Papernot, Martín Abadi, Ulfar Erlingsson, Ian Goodfellow, and Kunal Talwar. Semi-supervised knowledge transfer for deep learning from private training data. In *International Conference on Learning Representations (ICLR)*, 2017. Cited on pages 29, 41, 90, 91, 92, 97, and 130.

[159] Nicolas Papernot, Shuang Song, Ilya Mironov, Ananth Raghunathan, Kunal Talwar, and Úlfar Erlingsson. Scalable private learning with pate. In *International Conference on Learning Representations (ICLR)*, 2018. Cited on pages 29, 41, 42, 90, 92, 97, and 130.

[160] Nicolas Papernot and Thomas Steinke. Hyperparameter tuning with renyi differential privacy. In *International Conference on Learning Representations (ICLR)*, 2021. Cited on page 116.

[161] Noseong Park, Mahmoud Mohammadi, Kshitij Gorde, Sushil Jajodia, Hongkyu Park, and Youngmin Kim. Data synthesis based on generative adversarial networks. *arXiv preprint arXiv:1806.03384*, 2018. Cited on page 119.

[162] 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 (JMLR)*, 12, 2011. Cited on pages 19 and 162.

[163] Gabriel Pereyra, George Tucker, Jan Chorowski, Łukasz Kaiser, and Geoffrey Hinton. Regularizing neural networks by penalizing confident output distributions. *arXiv preprint arXiv:1701.06548*, 2017. Cited on pages 91, 97, 100, 192, and 204.

[164] Bjarne Pfitzner and Bert Arnrich. Dpd-fvae: Synthetic data generation using federated variational autoencoders with differentially-private decoder. *arXiv preprint arXiv:2211.11591*, 2022. Cited on page 58.

[165] Natalia Ponomareva, Hussein Hazimeh, Alex Kurakin, Zheng Xu, Carson Denison, H Brendan McMahan, Sergei Vassilvitskii, Steve Chien, and Abhradeep Thakurta. How to dp-fy ml: A practical guide to machine learning with differential privacy. *Journal of Artificial Intelligence Research*, 77:1113–1201, 2023. Cited on pages 3, 39, and 60.

[166] Ameya Prabhu, Philip HS Torr, and Puneet K Dokania. Gdumb: A simple approach that questions our progress in continual learning. In *European conference on computer vision (ECCV)*. Springer, 2020. Cited on page 33.

[167] Julien Rabin, Gabriel Peyré, Julie Delon, and Marc Bernot. Wasserstein barycenter and its application to texture mixing. In *Proceedings of the Third international conference on Scale Space and Variational Methods in Computer Vision (SSVM)*, pages 435–446, 2011. Cited on page 51.

[168] Alec Radford, Luke Metz, and Soumith Chintala. Unsupervised representation learning with deep convolutional generative adversarial networks. In Yoshua Bengio and Yann LeCun, editors, *International Conference on Learning Representations (ICLR)*, 2016. Cited on pages 20 and 34.

[169] Ali Rahimi and Benjamin Recht. Random features for large-scale kernel machines. *Advances in neural information processing systems (NeurIPS)*, 2007. Cited on page 51.

[170] Md Atiqur Rahman, Tanzila Rahman, Robert Laganière, Noman Mohammed, and Yang Wang. Membership inference attack against differentially private deep learning model. *Transactions on Data Privacy*, 11(1), 2018. Cited on pages 5, 90, and 91.

[171] Alain Rakotomamonjy and Ralaivola Liva. Differentially private sliced wasserstein distance. In *International Conference on Machine Learning (ICML)*. PMLR, 2021. Cited on pages 49, 50, 51, 174, and 175.

[172] Aditya Ramesh, Prafulla Dhariwal, Alex Nichol, Casey Chu, and Mark Chen. Hierarchical text-conditional image generation with CLIP latents. *arXiv preprint arXiv:2204.06125*, 2022. Cited on pages 66, 67, and 81.

[173] Sylvestre-Alvise Rebuffi, Alexander Kolesnikov, Georg Sperl, and Christoph H Lampert. icarl: Incremental classifier and representation learning. In *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition (CVPR)*, 2017. Cited on pages 29, 33, and 169.

[174] Shahbaz Rezaei and Xin Liu. Towards the infeasibility of membership inference on deep models. *arXiv preprint arXiv:2005.13702*, 2020. Cited on pages 68, 91, and 96.

[175] Danilo Rezende and Shakir Mohamed. Variational inference with normalizing flows. In *International conference on machine learning (ICML)*. PMLR, 2015. Cited on page 47.

[176] Matthew E Ritchie, Belinda Phipson, Di Wu, Yifang Hu, Charity W Law, Wei Shi, and Gordon K Smyth. limma powers differential expression analyses for rna-sequencing and microarray studies. *Nucleic acids research*, 43, 2015. Cited on page 107.

[177] Mark D Robinson, Davis J McCarthy, and Gordon K Smyth. edger: a bioconductor package for differential expression analysis of digital gene expression data. *Bioinformatics*, 26(1), 2010. Cited on pages 107 and 113.

[178] Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-resolution image synthesis with latent diffusion models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022. Cited on pages 58, 66, 67, and 75.

[179] Aaron Roth and Tim Roughgarden. Interactive privacy via the median mechanism. In *Proceedings of the forty-second ACM symposium on Theory of computing (STOC)*, 2010. Cited on pages 2, 14, 26, 29, and 38.

[180] Alexandre Sablayrolles, Matthijs Douze, Yann Ollivier, Cordelia Schmid, and Hervé Jégou. White-box vs black-box: Bayes optimal strategies for membership inference. In *International Conference on Machine Learning (ICML)*. PMLR, 2019. Cited on pages 6, 66, 68, 71, 72, 75, 90, 91, 92, 96, 189, and 194.

[181] Noveen Sachdeva and Julian McAuley. Data distillation: A survey. *arXiv preprint arXiv:2301.04272*, 2023. Cited on page 25.

[182] Chitwan Saharia, William Chan, Saurabh Saxena, Lala Li, Jay Whang, Emily Denton, Seyed Kamyar Seyed Ghasemipour, Burcu Karagol Ayan, S. Sara Mahdavi, Rapha Gontijo Lopes, et al. Photorealistic text-to-image diffusion models with deep language understanding. *Advances in Neural Information Processing Systems (NeurIPS)*, 2022. Cited on pages 66, 67, 70, and 81.

[183] Ahmed Salem, Yang Zhang, Mathias Humbert, Mario Fritz, and Michael Backes. Ml-leaks: Model and data independent membership inference attacks and defenses on machine learning models. In *Annual Network and Distributed System Security Symposium (NDSS)*, 2019. Cited on pages 66, 68, 75, 90, 91, and 96.

[184] Tim Salimans, Ian Goodfellow, Wojciech Zaremba, Vicki Cheung, Alec Radford, and Xi Chen. Improved techniques for training gans. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2016. Cited on pages 19 and 161.

[185] Terrance D Savitsky, Matthew R Williams, and Jingchen Hu. Bayesian pseudo posterior mechanism under asymptotic differential privacy. *Journal of Machine Learning Research (JMLR)*, 23, 2022. Cited on pages 60 and 130.

[186] Christoph Schuhmann, Romain Beaumont, Richard Vencu, Cade Gordon, Ross Wightman, Mehdi Cherti, Theo Coombes, Aarush Katta, Clayton Mullis, Mitchell Wortsman, et al. Laion-5b: An open large-scale dataset for training next generation image-text models. *Advances in Neural Information Processing Systems (NeurIPS)*, 2022. Cited on page 86.

[187] Jonathan Schwarz, Wojciech Czarnecki, Jelena Luketina, Agnieszka Grabska-Barwinska, Yee Whye Teh, Razvan Pascanu, and Raia Hadsell. Progress & compress: A scalable framework for continual learning. In *International Conference on Machine Learning (ICML)*. PMLR, 2018. Cited on page 32.

[188] Reza Shokri, Marco Stronati, Congzheng Song, and Vitaly Shmatikov. Membership inference attacks against machine learning models. In *IEEE Symposium on Security and Privacy (SP)*. IEEE, 2017. Cited on pages 66, 68, 75, 78, 82, 90, 91, 96, and 191.

[189] Ravid Shwartz-Ziv and Naftali Tishby. Opening the black box of deep neural networks via information. *arXiv preprint arXiv:1703.00810*, 2017. Cited on page 47.

[190] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. In Yoshua Bengio and Yann LeCun, editors, *International Conference on Learning Representations (ICLR)*, 2015. Cited on page 96.

[191] Jascha Sohl-Dickstein, Eric Weiss, Niru Maheswaranathan, and Surya Ganguli. Deep unsupervised learning using nonequilibrium thermodynamics. In *International Conference on Machine Learning (ICML)*. PMLR, 2015. Cited on pages 47, 66, and 67.

[192] Kihyuk Sohn, Honglak Lee, and Xinchen Yan. Learning structured output representation using deep conditional generative models. *Advances in neural information processing systems (NeurIPS)*, 2015. Cited on pages 61 and 109.

[193] Congzheng Song, Thomas Ristenpart, and Vitaly Shmatikov. Machine learning models that remember too much. In *ACM SIGSAC Conference on Computer and Communications Security (CCS)*, 2017. Cited on page 90.

[194] Congzheng Song and Reza Shokri. Robust membership encoding: Inference attacks and copyright protection for deep learning. *arXiv preprint arXiv:1909.12982*, 2019. Cited on pages 5 and 66.

[195] Liwei Song and Prateek Mittal. Systematic evaluation of privacy risks of machine learning models. In *USENIX Security Symposium (USENIX Security)*, 2021. Cited on pages 66, 68, 90, 91, 96, 97, 191, and 195.

[196] Yang Song and Stefano Ermon. Generative modeling by estimating gradients of the data distribution. *Advances in Neural Information Processing Systems (NeurIPS)*, 2019. Cited on pages 47, 66, 67, and 70.

[197] Yang Song and Stefano Ermon. Improved techniques for training score-based generative models. *Advances in Neural Information Processing Systems (NeurIPS)*, 2020. Cited on pages 66 and 70.

[198] Yang Song, Jascha Sohl-Dickstein, Diederik P. Kingma, Abhishek Kumar, Stefano Ermon, and Ben Poole. Score-based generative modeling through stochastic differential equations. In *International Conference on Learning Representations (ICLR)*, 2021. Cited on pages 66 and 70.

[199] Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. Dropout: a simple way to prevent neural networks from overfitting. *The Journal of Machine Learning Research (JMLR)*, 2014. Cited on pages 90 and 97.

[200] Theresa Stadler, Bristena Oprisanu, and Carmela Troncoso. Synthetic data – anonymisation groundhog day. In *USENIX Security Symposium (USENIX Security)*, 2022. Cited on pages 61 and 125.

[201] Theresa Stadler and Carmela Troncoso. Why the search for a privacy-preserving data sharing mechanism is failing. *Nature Computational Science*, 2(4):208–210, 2022. Cited on pages 61, 62, and 125.

[202] Ton Steerneman. On the total variation and hellinger distance between signed measures; an application to product measures. *Proceedings of the American Mathematical Society (AMS)*, 88(4):684–688, 1983. Cited on page 189.

[203] Aravind Subramanian, Rajiv Narayan, Steven M Corsello, David D Peck, Ted E Natoli, Xiaodong Lu, Joshua Gould, John F Davis, Andrew A Tubelli, Jacob K Asiedu, and et al. A next generation connectivity map: L1000 platform and the first 1,000,000 profiles. *Cell*, 171(6), 2017. Cited on page 115.

[204] Shun Takagi, Tsubasa Takahashi, Yang Cao, and Masatoshi Yoshikawa. P3gm: Private high-dimensional data release via privacy preserving phased generative model. In *International Conference on Data Engineering (ICDE)*. IEEE, 2021. Cited on pages 49 and 58.

[205] Fuchou Tang, Catalin Barbacioru, Yangzhou Wang, Ellen Nordman, Clarence Lee, Nanlan Xu, Xiaohui Wang, John Bodeau, Brian B Tuch, Asim Siddiqui, and et al. mrna-seq whole-transcriptome analysis of a single cell. *Nature Methods*, 6(5), 2009. Cited on page 106.

[206] Uthaipon Tao Tantipongpipat, Chris Waites, Digvijay Boob, Amaresh Ankit Siva, and Rachel Cummings. Differentially private synthetic mixed-type data generation for unsupervised learning. *Intelligent Decision Technologies*, 15(4), 2021. Cited on page 49.

[207] Yuchao Tao, Ryan McKenna, Michael Hay, Ashwin Machanavajjhala, and Gerome Miklau. Benchmarking differentially private synthetic data generation algorithms. *arXiv preprint arXiv:2112.09238*, 2021. Cited on page 60.

[208] Naftali Tishby, Fernando C. Pereira, and William Bialek. The information bottleneck method. In *Proc. of the 37-th Annual Allerton Conference on Communication, Control and Computing*, 1999. Cited on page 47.

[209] Mariya Toneva, Alessandro Sordoni, Remi Tachet des Combes, Adam Trischler, Yoshua Bengio, and Geoffrey J Gordon. An empirical study of example forgetting during deep neural network learning. *arXiv preprint arXiv:1812.05159*, 2018. Cited on page 29.

[210] Amirsina Torfi, Edward A Fox, and Chandan K Reddy. Differentially private synthetic medical data generation using convolutional gans. *Information Sciences*, 586, 2022. Cited on page 105.

[211] Reihaneh Torkzadehmahani, Peter Kairouz, and Benedict Paten. Dp-cgan: Differentially private synthetic data and label generation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, 2019. Cited on pages 14, 16, 19, 20, 29, 30, 49, 53, and 169.

[212] Florian Tramer and Dan Boneh. Differentially private learning needs better features (or much more data). In *International Conference on Learning Representations (ICLR)*, 2021. Cited on pages 26, 42, and 116.

[213] Florian Tramèr, Gautam Kamath, and Nicholas Carlini. Considerations for differentially private learning with large-scale public pretraining. *arXiv preprint arXiv:2212.06470*, 2022. Cited on pages 61 and 130.

[214] Martin Treppner, Adrián Salas-Bastos, Moritz Hess, Stefan Lenz, Tanja Vogel, and Harald Binder. Synthetic single cell rna sequencing data from small pilot studies using deep generative models. *Scientific Reports*, 11(1), 2021. Cited on pages 105 and 106.

[215] Aleksei Triastcyn and Boi Faltings. Generating artificial data for private deep learning. *arXiv preprint arXiv:1803.03148*, 2018. Cited on pages 49 and 53.

[216] Aleksei Triastcyn and Boi Faltings. Federated generative privacy. *IEEE Intelligent Systems*, 35(4):50–57, 2020. Cited on pages 61 and 131.

[217] Stacey Truex, Ling Liu, Mehmet Emre Gursoy, Lei Yu, and Wenqi Wei. Demystifying membership inference attacks in machine learning as a service. *IEEE Transactions on Services Computing*, 2019. Cited on page 91.

[218] Dmitry Ulyanov, Andrea Vedaldi, and Victor Lempitsky. Instance normalization: The missing ingredient for fast stylization. *arXiv preprint arXiv:1607.08022*, 2016. Cited on page 30.

[219] Dmitry Ulyanov, Andrea Vedaldi, and Victor Lempitsky. Deep image prior. In *Proceedings of the IEEE conference on computer vision and pattern recognition (CVPR)*, 2018. Cited on page 34.

[220] Aaron Van den Oord, Nal Kalchbrenner, Lasse Espeholt, Oriol Vinyals, Alex Graves, et al. Conditional image generation with pixelcnn decoders. *Advances in neural information processing systems (NeurIPS)*, 2016. Cited on page 47.

[221] Aäron Van Den Oord, Nal Kalchbrenner, and Koray Kavukcuoglu. Pixel recurrent neural networks. In *International conference on machine learning (ICML)*. PMLR, 2016. Cited on page 47.

[222] Tim Van Erven and Peter Harremos. Rényi divergence and kullback-leibler divergence. *IEEE Transactions on Information Theory*, 60(7), 2014. Cited on page 157.

[223] Giuseppe Vietri, Grace Tian, Mark Bun, Thomas Steinke, and Steven Wu. New oracle-efficient algorithms for private synthetic data release. In *International Conference on Machine Learning (ICML)*. PMLR, 2020. Cited on pages 2, 26, and 38.

[224] Margarita Vinaroz, Mohammad-Amin Charusaie, Frederik Harder, Kamil Adamczewski, and Mi Jung Park. Hermite polynomial features for private data generation. In *International Conference on Machine Learning (ICML)*. PMLR, 2022. Cited on pages 49, 50, 51, and 174.

[225] Ramon Viñas, Helena Andrés-Terré, Pietro Liò, and Kevin Bryson. Adversarial generation of gene expression data. *Bioinformatics*, 38(3), 2022. Cited on page 106.

[226] Chris Waites and Rachel Cummings. Differentially private normalizing flows for privacy-preserving density estimation. In *Proceedings of the AAAI/ACM Conference on AI, Ethics, and Society*, 2021. Cited on pages 49 and 58.

[227] Boxin Wang, Fan Wu, Yunhui Long, Luka Rimanic, Ce Zhang, and Bo Li. Datalens: Scalable privacy preserving training via gradient compression and aggregation. In *ACM SIGSAC Conference on Computer and Communications Security (CCS)*, 2021. Cited on pages 13, 29, 30, 49, 56, 57, 168, 169, and 177.

[228] Hui-Po Wang, Dingfan Chen, Raouf Kerkouche, and Mario Fritz. Fedlap-dp: Federated learning by sharing differentially private loss approximations. *Proceedings on Privacy Enhancing Technologies (PoPETs)*, 2024. Cited on pages 10, 62, and 131.

[229] Tongzhou Wang, Jun-Yan Zhu, Antonio Torralba, and Alexei A Efros. Dataset distillation. *arXiv preprint arXiv:1811.10959*, 2018. Cited on pages 28, 29, 60, and 170.

[230] Yu-Xiang Wang, Borja Balle, and Shiva Prasad Kasiviswanathan. Subsampled renyi differential privacy and analytical moments accountant. In *International Conference on Artificial Intelligence and Statistics (AISTATS)*, 2019. Cited on pages 19, 157, 158, and 165.

[231] Zhong Wang, Mark Gerstein, and Michael Snyder. Rna-seq: a revolutionary tool for transcriptomics. *Nature Reviews. Genetics*, 10(1), 2009. Cited on page 104.

[232] Stefanie Warnat-Herresthal, Konstantinos Perrakis, Bernd Taschler, Matthias Becker, Kevin Baßler, Marc Beyer, Patrick Günther, Jonas Schulte-Schrepping, Lea Seep, Kathrin Klee, and et al. Scalable prediction of acute myeloid leukemia using high-dimensional machine learning and blood transcriptomics. *iScience*, 23(1), 2020. Cited on page 115.

[233] Chengkun Wei, Minghu Zhao, Zhikun Zhang, Min Chen, Wenlong Meng, Bo Liu, Yuan Fan, and Wenzhi Chen. Dpmlbench: Holistic evaluation of differentially private machine learning. *arXiv preprint arXiv:2305.05900*, 2023. Cited on page 25.

[234] Max Welling. Herding dynamical weights to learn. In *Proceedings of the 26th Annual International Conference on Machine Learning (ICML)*, 2009. Cited on page 29.

[235] Frank Wilcoxon. Individual comparisons by ranking methods. *Biometrics Bulletin*, 1(6), 1945. Cited on page 114.

[236] Christina Winkler, Daniel Worrall, Emiel Hoogeboom, and Max Welling. Learning likelihoods with conditional normalizing flows. *arXiv preprint arXiv:1912.00042*, 2019. Cited on page 61.

[237] Yixin Wu, Ning Yu, Zheng Li, Michael Backes, and Yang Zhang. Membership inference attacks against text-to-image generation models. *arXiv preprint arXiv:2210.00968*, 2022. Cited on pages 68 and 84.

[238] Han Xiao, Kashif Rasul, and Roland Vollgraf. Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms, 2017. Cited on pages 19, 30, 161, and 168.

[239] Liyang Xie, Kaixiang Lin, Shu Wang, Fei Wang, and Jiayu Zhou. Differentially private generative adversarial network. *arXiv preprint arXiv:1802.06739*, 2018. Cited on pages 2, 14, 16, 20, 26, 38, 49, 53, and 128.

[240] Bangzhou Xin, Wei Yang, Yangyang Geng, Sheng Chen, Shaowei Wang, and Liusheng Huang. Private fl-gan: Differential privacy synthetic data generation based on federated learning. In *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2020. Cited on page 61.

[241] Yuanhao Xiong, Ruochen Wang, Minhao Cheng, Felix Yu, and Cho-Jui Hsieh. Feddm: Iterative distribution matching for communication-efficient federated learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2023. Cited on pages 62 and 131.

[242] Chugui Xu, Ju Ren, Deyu Zhang, Yaoxue Zhang, Zhan Qin, and Kui Ren. Ganobfuscator: Mitigating information leakage under gan via differential privacy. *IEEE Transactions on Information Forensics and Security (TIFS)*, 14(9), 2019. Cited on pages 49 and 53.

[243] Yilin Yang, Kamil Adamczewski, Danica J Sutherland, Xiaoxiao Li, and Mijung Park. Differentially private neural tangent kernels for privacy-preserving data generation. *arXiv preprint arXiv:2303.01687*, 2023. Cited on pages 49, 51, and 174.

[244] Samuel Yeom, Irene Giacomelli, Matt Fredrikson, and Somesh Jha. Privacy risk in machine learning: Analyzing the connection to overfitting. In *IEEE 31st Computer Security Foundations Symposium (CSF)*. IEEE, 2018. Cited on pages 6, 66, 68, 71, 90, 91, 92, 94, and 95.

[245] Jinsung Yoon, James Jordon, and Mihaela van der Schaar. PATE-GAN: Generating synthetic data with differential privacy guarantees. In *International Conference on Learning Representations (ICLR)*, 2019. Cited on pages 2, 14, 15, 20, 26, 38, 49, 53, and 128.

[246] Ashkan Yousefpour, Igor Shilov, Alexandre Sablayrolles, Davide Testuggine, Karthik Prasad, Mani Malek, John Nguyen, Sayan Ghosh, Akash Bharadwaj, Jessica Zhao, Graham Cormode, and Ilya Mironov. Opacus: User-friendly differential privacy library in PyTorch. *arXiv preprint arXiv:2109.12298*, 2021. Cited on page 168.

[247] Da Yu, Saurabh Naik, Arturs Backurs, Sivakanth Gopi, Huseyin A Inan, Gautam Kamath, Janardhan Kulkarni, Yin Tat Lee, Andre Manoel, Lukas Wutschitz, et al. Differentially private fine-tuning of language models. *arXiv preprint arXiv:2110.06500*, 2021. Cited on page 61.

[248] Ning Yu, Vladislav Skripniuk, Dingfan Chen, Larry Davis, and Mario Fritz. Responsible disclosure of generative models using scalable fingerprinting. In *International Conference on Learning Representations (ICLR)*, 2022. Cited on page 10.

[249] Ruonan Yu, Songhua Liu, and Xinchao Wang. Dataset distillation: A comprehensive review. *arXiv preprint arXiv:2301.07014*, 2023. Cited on page 25.

[250] Xiang Yue, Huseyin A Inan, Xuechen Li, Girish Kumar, Julia McAnallen, Hoda Shajari, Huan Sun, David Levitan, and Robert Sim. Synthetic text generation with differential privacy: A simple and practical recipe. In *The 61st Annual Meeting Of The Association For Computational Linguistics*, 2023. Cited on page 58.

[251] Luke Zappia, Belinda Phipson, and Alicia Oshlack. Splatter: simulation of single-cell rna sequencing data. *Genome Biology*, 18(1), 2017. Cited on page 106.

[252] Friedemann Zenke, Ben Poole, and Surya Ganguli. Continual learning through synaptic intelligence. In *International Conference on Machine Learning (ICML)*. PMLR, 2017. Cited on page 169.

[253] Hongyi Zhang, Moustapha Cisse, Yann N Dauphin, and David Lopez-Paz. mixup: Beyond empirical risk minimization. In *International Conference on Learning Representations (ICLR)*, 2018. Cited on pages 100 and 204.

[254] Jun Zhang, Graham Cormode, Cecilia M Procopiuc, Divesh Srivastava, and Xiaokui Xiao. Privbayes: Private data release via bayesian networks. *ACM Transactions on Database Systems (TODS)*, 42(4), 2017. Cited on page 14.

[255] Linfeng Zhang, Jiebo Song, Anni Gao, Jingwei Chen, Chenglong Bao, and Kaisheng Ma. Be your own teacher: Improve the performance of convolutional neural networks via self distillation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, 2019. Cited on pages 97 and 193.

[256] Longling Zhang, Bochen Shen, Ahmed Barnawi, Shan Xi, Neeraj Kumar, and Yi Wu. Feddpgan: federated differentially private generative adversarial networks framework for the detection of covid-19 pneumonia. *Information Systems Frontiers*, 23(6):1403–1415, 2021. Cited on pages 61 and 131.

[257] Xinyang Zhang, Shouling Ji, and Ting Wang. Differentially private releasing via deep generative model (technical report). *arXiv preprint arXiv:1801.01594*, 2018. Cited on pages 14, 16, 20, 49, 53, and 55.

[258] Zhikun Zhang, Tianhao Wang, Ninghui Li, Jean Honorio, Michael Backes, Shibo He, Jiming Chen, and Yang Zhang. Privsyn: Differentially private data synthesis. In *USENIX Security Symposium (USENIX Security)*, 2021. Cited on page 111.

[259] Bo Zhao and Hakan Bilen. Dataset condensation with differentiable siamese augmentation. In *International Conference on Machine Learning (ICML)*. PMLR, 2021. Cited on pages 27, 28, 29, 166, 170, and 171.

[260] Bo Zhao and Hakan Bilen. Dataset condensation with distribution matching. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision (WACV)*, 2023. Cited on page 170.

[261] Bo Zhao, Konda Reddy Mopuri, and Hakan Bilen. Dataset condensation with gradient matching. In *International Conference on Learning Representations (ICLR)*, 2021. Cited on pages 27, 28, 29, 166, 170, and 171.

[262] Derui Zhu*, Dingfan Chen*, Jens Grossklags, and Mario Fritz. Data forensics in diffusion models: A systematic analysis of membership privacy. *arXiv preprint arXiv:2302.07801*, 2023. Cited on pages 8, 9, and 65.

[263] Ligeng Zhu, Zhijian Liu, and Song Han. Deep leakage from gradients. *Advances in neural information processing systems (NeurIPS)*, 2019. Cited on page 131.

A

GS-WGAN: A Gradient-Sanitized Approach for Learning Differentially Private Generators

These supplementary materials include the privacy analysis (§A.1), the algorithm pseudocode (§A.2), the details of experiment setup (§A.3), and additional results (§A.4). Our source code is available at Github: https://github.com/DingfanChen/GS-WGAN.

A.1 Privacy Analysis

The privacy cost (ε) computation including: *(i)* bounding the privacy loss for our gradient sanitization mechanism using RDP; *(ii)* applying analytical moments accountant of subsampled RDP [230] for a tighter upper bound on the RDP parameters; *(iii)* tracking the overall privacy cost: multiplying the RDP orders by the number of training iterations and converting the resulting RDP orders to an (ε, δ) pair (Definition 3.2 [142]). We below present the theoretical results.

Theorem 2.4.1. Each generator update step satisfies $(\lambda, 2B\lambda/\sigma^2)$-RDP where B is the batch size.

Proof. Let $f = \text{clip}(\boldsymbol{g}_G^{\text{upstream}}, C)$, i.e., the clipped gradient before being sanitized. The sensitivity can be derived via the triangle inequality:

$$\Delta_2 f = \max_{S,S'} \|f(S) - f(S')\|_2 \leq 2C \tag{A.1}$$

with $C = 1$ in our case. Hence, we have $\mathcal{M}_{\sigma,C}$ is $(\lambda, 2\lambda/\sigma^2)$-RDP.
Each generator update step (which operates on a batch of data) can be expressed as

$$\hat{\boldsymbol{g}}_G = \frac{1}{B} \sum_{i=1}^{B} \mathcal{M}_{\sigma,C}(\nabla_{G(\boldsymbol{z}_i)} \mathcal{L}_G(\boldsymbol{\theta}_G)) \cdot J_{\boldsymbol{\theta}_G} G(\boldsymbol{z}_i; \boldsymbol{\theta}_G) \tag{A.2}$$

This can be seen as a composition of B Gaussian mechanisms. Concretely, we want to bound the Rényi divergence $D_\lambda(\hat{\boldsymbol{g}}_G(S) \| \hat{\boldsymbol{g}}_G(S'))$ with S, S' denoting the neighbouring datasets. We use the following properties of Rényi divergence [222]:
(i) Data-processing inequality : $D_\lambda(P_Y \| Q_Y) \leq D_\lambda(P_X \| Q_X)$ if the transition probabilities $A(Y|X)$ in the Markov chain $X \rightarrow Y$ is fixed.
(ii) Additivity : For arbitrary distributions $P_1, .., P_N$ and $Q_1, ..., Q_N$ let $P^N = P_1 \times \cdots \times P_N$ and $Q^N = Q_1 \times \cdots \times Q_N$. Then $D_\lambda(P^N \| Q^N) = \sum_{n=1}^{N} D_\lambda(P_n \| Q_n)$
Let u and v denote the output distribution of the sanitization mechanism $\mathcal{M}_{\sigma,C}$ when applied on S and S' respectively, and h the post-processing function (i.e., multiplication with the local

Jacobian). We have,

$$\begin{aligned} D_\lambda(\hat{\mathbf{g}}_G(S), \hat{\mathbf{g}}_G(S')) &\leq D_\lambda\Big(h_1(u_1) * \cdots * h_B(u_B) \| h_1(v_1) * \cdots * h_B(v_B)\Big) && \text{(A.3)} \\ &\leq D_\lambda\Big(\big(h_1(u_1), \cdots, h_B(u_B)\big) \| \big(h_1(v_1), \cdots, h_B(v_B)\big)\Big) && \text{(A.4)} \\ &= \sum_b D_\lambda((h_b(u_b) \| h_b(v_b)) && \text{(A.5)} \\ &\leq \sum_b D_\lambda(u_b \| v_b) && \text{(A.6)} \\ &\leq B \cdot \max_b D_\lambda(u_b \| v_b) && \text{(A.7)} \\ &\leq B \cdot 2\lambda/\sigma^2 && \text{(A.8)} \end{aligned}$$

where (3)(4)(6) are based on the data-processing theorem; (5) follows from the additivity; and the last equation follows from the $(\lambda, 2\lambda/\sigma^2)$-RDP of $\mathcal{M}_{\sigma,C}$. □

Theorem A.1.1. (RDP for Subsampled Mechanisms [230]) Given a dataset containing n datapoints with domain $\mathcal{X}$ and a randomized mechanism $\mathcal{M}$ that takes an input from $\mathcal{X}^m$ for $m \leq n$, let the randomized algorithm $\mathcal{M} \circ$ **subsample** be defined as: *(i)* **subsample**: subsample without replacement m datapoints of the dataset (with subsampling rate $\gamma = m/n$); *(ii)* apply $\mathcal{M}$: a randomized algorithm taking the subsampled dataset as the input. For all integers $\lambda \geq 2$, if $\mathcal{M}$ is $(\lambda, \epsilon(\lambda))$-RDP, then $\mathcal{M} \circ$ **subsample** is $(\lambda, \epsilon'(\lambda))$-RDP where

$$\begin{aligned} \epsilon'(\lambda) \leq & \frac{1}{\lambda - 1} \log \Bigg(1 + \gamma^2 \binom{\lambda}{2} \min \left\{4(e^{\epsilon(2)} - 1), e^{\epsilon(2)} \min \{2, (e^{\epsilon(\infty)} - 1)^2\}\right\} \\ & + \sum_{j=3}^{\lambda} \gamma^j \binom{\lambda}{j} e^{(j-1)\epsilon(j)} \min\{2, (e^{\epsilon(\infty)} - 1)^j\}\Bigg) \end{aligned}$$

In practice, we adopt the official implementation of [230] [1] for computing the accumulated privacy cost (i.e., tracking the RDP orders and converting RDP to (ε, δ)-DP).

A.2 Algorithm

We present the pseudocode of our proposed method in Algorithm 4 (Centralized setup) and Algorithm 5 (Federated setup).

[1] https://github.com/yuxiangw/autodp

Algorithm 4: Centralized GS-WGAN Training

```
Input: Dataset $S$, subsampling rate $\gamma$, noise scale $\sigma$, warm-start iterations $T_w$,
       training iterations $T$, learning rates $\eta_D$ and $\eta_G$, the number of discriminator
       iterations per generator iteration $n_{dis}$, batch size $B$
Output: Differentially Private generator $G$ with parameters $\boldsymbol{\theta}_G$, total privacy cost $\varepsilon$
Subsample (without replacement) the dataset $S$ into subsets $\{S_k\}_{k=1}^{K}$ with rate $\gamma$
  $(K=1/\gamma)$;
for $k$ in $\{1,...,K\}$ in parallel do
    Initialize non-private generator $\boldsymbol{\theta}_G^k$, discriminator $\boldsymbol{\theta}_D^k$ for $step$ in $\{1,...,T_w\}$ do
        for $t$ in $\{1,...,n_{dis}\}$ do
            Sample batch $\{\boldsymbol{x}_i\}_{i=1}^{B} \subseteq S_k$ ;
            Sample batch $\{\boldsymbol{z}_i\}_{i=1}^{B}$ with $\boldsymbol{z}_i \sim P_z$ ;
            $\boldsymbol{\theta}_D^k \leftarrow \boldsymbol{\theta}_D^k - \eta_D \cdot \frac{1}{B}\sum_i \nabla_{\boldsymbol{\theta}_D^k} \mathcal{L}_D(\boldsymbol{\theta}_D^k; \boldsymbol{x}_i, G(\boldsymbol{z}_i; \boldsymbol{\theta}_G^k))$ ;
        end
        $\boldsymbol{\theta}_G^k \leftarrow \boldsymbol{\theta}_G^k - \eta_G \cdot \frac{1}{B}\sum_i \nabla_{\boldsymbol{\theta}_G^k} \mathcal{L}_G(\boldsymbol{\theta}_G^k; G(\boldsymbol{z}_i; \boldsymbol{\theta}_G^k), \boldsymbol{\theta}_D^k)$ ;
    end
    Initialize private generator $\boldsymbol{\theta}_G$ ;
    for $step$ in $\{1,...,T\}$ do
        Sample subset index $k \sim \mathcal{U}[1,K]$ ;
        for $t$ in $\{1,...,n_{dis}\}$ do
            Sample batch $\{\boldsymbol{x}_i\}_{i=1}^{B} \subseteq S_k$ ;
            Sample batch $\{\boldsymbol{z}_i\}_{i=1}^{B}$ with $\boldsymbol{z}_i \sim P_z$ ;
            $\boldsymbol{\theta}_D^k \leftarrow \boldsymbol{\theta}_D^k - \eta_D \cdot \frac{1}{B}\sum_i \nabla_{\boldsymbol{\theta}_D^k} \mathcal{L}_D(\boldsymbol{\theta}_D^k; \boldsymbol{x}_i, G(\boldsymbol{z}_i; \boldsymbol{\theta}_G))$ ;
        end
        $\boldsymbol{\theta}_G \leftarrow \boldsymbol{\theta}_G - \eta_G \cdot \frac{1}{B}\sum_i \mathcal{M}_{\sigma,C}(\boldsymbol{\theta}_G; G(\boldsymbol{z}_i; \boldsymbol{\theta}_G), \boldsymbol{\theta}_D^k) \cdot \boldsymbol{J}_{\boldsymbol{\theta}_G} G(\boldsymbol{z}_i; \boldsymbol{\theta}_G)$ ;
        Accumulate privacy cost $\varepsilon$ ;
    end
end
return Generator $G(\cdot\,; \boldsymbol{\theta}_G)$, privacy cost $\varepsilon$
```

Algorithm 5: Federated (Decentralized) GS-WGAN Training

```
Input: Client index set $\{1, ..., K\}$, noise scale $\sigma$, warm-start iterations $T_w$, training
    iterations $T$, learning rates $\eta_D$ and $\eta_G$, the number of discriminator iterations
    per generator iteration $n_{dis}$, batch size $B$
Output: Differentially Private generator $G$ with parameters $\boldsymbol{\theta}_G$, total privacy cost $\varepsilon$
for each client $k$ in $\{1, ..., K\}$ in parallel do
    ClientWarmStart($k$)
end
Initialize private generator $\boldsymbol{\theta}_G$ ;
for $step$ in $\{1, ..., T\}$ do
    Sample client index $k \sim \mathcal{U}[1, K]$ ;
    for $t$ in $\{1, ..., n_{dis}\}$ do
        Sample batch $\{\boldsymbol{z}_i\}_{i=1}^{B}$ with $\boldsymbol{z}_i \sim P_z$ ;
        $\{\hat{\boldsymbol{g}}_i^{\text{up}}\}_{i=1}^{B} \leftarrow$ ClientUpdate($k, G(\boldsymbol{z}_i; \boldsymbol{\theta}_G)$)
    end
    $\boldsymbol{\theta}_G \leftarrow \boldsymbol{\theta}_G - \eta_G \cdot \frac{1}{B} \sum_i \hat{\boldsymbol{g}}_i^{\text{up}} \cdot \boldsymbol{J}_{\boldsymbol{\theta}_G} G(\boldsymbol{z}_i; \boldsymbol{\theta}_G)$ ;
    Accumulate privacy cost $\varepsilon$ ;
end
return Generator $G(\cdot\,; \boldsymbol{\theta}_G)$, privacy cost $\varepsilon$

Procedure ClientWarmStart($k$)
    Get local dataset $S_k$ ;
    Initialize local generator $\boldsymbol{\theta}_G^k$, discriminator $\boldsymbol{\theta}_D^k$ ;
    for $step$ in $\{1, ..., T_w\}$ do
        for $t$ in $\{1, ..., n_{dis}\}$ do
            Sample batch $\{\boldsymbol{x}_i\}_{i=1}^{B} \subseteq S_k$ ;
            Sample batch $\{\boldsymbol{z}_i\}_{i=1}^{B}$ with $\boldsymbol{z}_i \sim P_z$ ;
            $\boldsymbol{\theta}_D^k \leftarrow \boldsymbol{\theta}_D^k - \eta_D \cdot \frac{1}{B} \sum_i \nabla_{\boldsymbol{\theta}_D^k} \mathcal{L}_D(\boldsymbol{\theta}_D^k; \boldsymbol{x}_i, G(\boldsymbol{z}_i; \boldsymbol{\theta}_G^k))$ ;
        end
        $\boldsymbol{\theta}_G^k \leftarrow \boldsymbol{\theta}_G^k - \eta_G \cdot \frac{1}{B} \sum_i \nabla_{\boldsymbol{\theta}_G^k} \mathcal{L}_G(\boldsymbol{\theta}_G^k; G(\boldsymbol{z}_i; \boldsymbol{\theta}_G^k), \boldsymbol{\theta}_D^k)$ ;
    end

Procedure ClientUpdate($k, G(\boldsymbol{z}_i; \boldsymbol{\theta}_G)$)
    Get local dataset $S_k$, local discriminator $D(\cdot\,; \boldsymbol{\theta}_D^k)$ ;
    Sample batch $\{\boldsymbol{x}_i\}_{i=1}^{B} \subseteq S_k$ ;
    $\boldsymbol{\theta}_D^k \leftarrow \boldsymbol{\theta}_D^k - \eta_D \cdot \frac{1}{B} \sum_i \nabla_{\boldsymbol{\theta}_D^k} \mathcal{L}_D(\boldsymbol{\theta}_D^k; \boldsymbol{x}_i, G(\boldsymbol{z}_i; \boldsymbol{\theta}_G))$ ;
    return $\mathcal{M}_{\sigma,C}(\boldsymbol{\theta}_G; G(\boldsymbol{z}_i; \boldsymbol{\theta}_G), \boldsymbol{\theta}_D^k)$
```

A.3 Experiment Setup

A.3.1 Hyperparameters

We adopt the hyperparameters setting in [67] for the GAN training, and list below the hyperparameters relevant for privacy computation.

Centralized Setting. We use by default a subsampling rate of γ=1/1000, noise scale σ=1.07, pretraining (warm-start) for 2K iterations and subsequently training for 20K iterations.

Federated Setting. We use by default a noise scale σ=1.07, pretraining (warm-start) for 2K iterations and subsequently training for 30K iterations.

A.3.2 Datasets

Centralized Setting. **MNIST** [108] and **Fashion-MNIST** [238] datasets contain 60K training images and 10K testing images. Each image has dimension 28×28 and belongs to one of the 10 classes.

Federated Setting. **Federated EMNIST** [23] dataset contains 28×28 gray-scale images of handwritten letters and numbers, grouped by user. The entire dataset contains 3400 users with 671,585 training examples and 77,483 testing examples. Following [10], the users are filtered by the prediction accuracy of a 36-class (10 numeric digits + 26 letters) CNN classifier. For evaluating the sample quality, we train GAN models on the users' data which yields classification accuracy $\geq$ 93.9% (866 users); For simulating the debugging task, we randomly choose 50% of the users and pre-process their data by flipping the pixel intensities. To mimic the real-world situation where the server is blind to the erroneous pre-processing, users with low classification accuracy $\leq$88.2% are selected (2136 users) as they are suspected to be affected by erroneous flipping (with bug). Note that only a fraction of them is indeed affected by the bug (1720 with bug, 416 without bug). This has the realistic property that the client data is non-IID and poses additional difficulties in the GAN training.

A.3.3 Evaluation Metrics

In line with previous literature, we use Inception Score (IS) [184, 112] and Frechet Inception Distance (FID) [77] for measuring sample quality, and classification accuracy for evaluating the usefulness of generated samples. We present below a detailed explanation of the evaluation metrics we adopted in the experiments.

Inception Score (IS). Formally, the IS is defined as follows,

$$\text{IS} = \exp\left(\mathbb{E}_{\boldsymbol{x}\sim G(\boldsymbol{z})} D_{KL}(P(y|\boldsymbol{x}) \| P(y))\right)$$

which corresponds to exponential of the KL divergence between the conditional class $P(y|\boldsymbol{x})$ and the marginal class distribution $P(y)$, where both $P(y|\boldsymbol{x})$ and $P(y)$ are measured by the output distribution of a pre-trained classifier when passing the generated samples as input. Intuitively, the IS should exhibit a high value if $P(y|\boldsymbol{x})$ has low entropy (i.e., the generated images are sharp and contain clear objects) and $P(y)$ is of high entropy (i.e., the generated samples have a high diversity covering all the different classes). In our experiments, we use pre-trained classifiers on the real datasets (with test accuracy equals to 99.25%, 93.75%, 92.16% on the MNIST, Fashion-MNIST and Federated EMNIST dataset respectively) [2] for computing the IS.

Frechet Inception Distance (FID). The FID is formularized as follows,

$$\mathrm{FID} = \|\mu_r - \mu_g\|^2 + \mathrm{tr}(\Sigma_r + \Sigma_g - 2(\Sigma_r\Sigma_g)^{1/2})$$

where $\boldsymbol{x}_r \sim \mathcal{N}(\mu_r, \Sigma_r)$ and $\boldsymbol{x}_g \sim \mathcal{N}(\mu_g, \Sigma_g)$ are the 2048-dimensional activations of the Inception-v3 pool3 layer for real and generated samples respectively. A lower FID value indicates a smaller discrepancy between the real and generated samples, which corresponds to a better sample quality and diversity. Following previous works [3] , we rescale the images and convert them to RGB by repeating the grayscale channel three times before inputting them to the Inception network.

Classification Accuracy. We consider the following classification models in our experiments: Multi-layer Perceptron (MLP), Convolutional Neural Network (CNN), AdaBoost (adaboost), Bagging (bagging), Bernoulli Naive Bayes (bernoulli nb), Decision tree (decision tree), Gaussian Naive Bayes (gaussian nb), Gradient Boosting (gbm), Linear Discriminant Analysis (lda), Linear Support Vector Machine (linear svc), Logistic Regression (logistic reg), Random Forest (random forest), and XGBoost (xgboost). For implementing the CNN model, we use two hidden layers (with dropout) each containing 32 and 64 kernels and apply ReLU as the activation function. For implementing the MLP, we use one hidden layer with 100 neurons and set ReLU as the activation function. All the other classification models are implemented using the default hyperparameters supplied by the scikit-learn [162] package.

A.3.4 Baseline Methods

We present more details about the implementation of the baseline methods. In particular, we provide the default value of the privacy hyperparameters below.

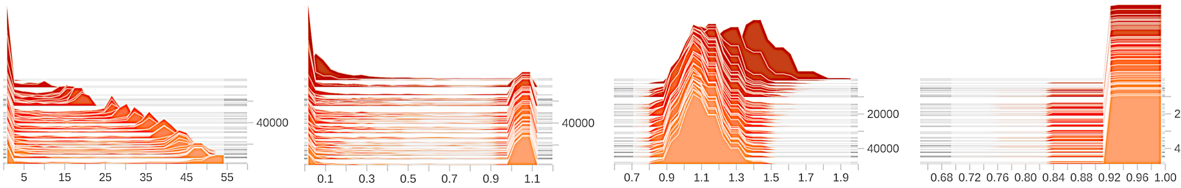
DP-Merf (AE) [4] We use as default a batch size=500 (γ=1/120), noise scale σ=0.588, training iteration=600 (epoch=5) for implementing DP-Merf, and batch size=500, noise scale σ=0.686, training iteration=2040 (epoch=17) for implementing DP-Merf AE.
DP-SGD GAN [5] We set the default hyper-parameters as follows: gradient clipping bound C=1.1, noise scale σ=2.1, batch size=600, training iterations=30K.
G-PATE We use 2000 teacher discriminators with batch size of 30 and set noise scales σ_1=600 and σ_2=100, consensus threshold T=0.5. A random projection with projection dimension=10 is applied.
PATE-GAN [6] When extending PATE-GAN to high-dimensional image datasets, we observe that after a few iterations, the generated samples are classified as fake by all teacher discriminators and the learning signals (gradients) for student discriminator and the generator vanish. Consequently, the training stuck at the early stage where the losses remain unchanged and no progress can be observed. While this issue is well resolved by careful design of the prior distribution, as reported in the original paper, we find that this technique has a limited effect when

[2] https://github.com/ChunyuanLI/MNIST_Inception_Score
[3] https://github.com/google/compare_gan
[4] https://github.com/frhrdr/Differentially-Private-Mean-Embeddings-with-Random-Features-for-Synthetic-Data-Generation
[5] https://github.com/reihaneh-torkzadehmahani/DP-CGAN
[6] https://bitbucket.org/mvdschaar/mlforhealthlabpub/src/2534877d99c8fdf19cbade16057990171e249ef3/alg/pategan/

applied to the high-dimensional image dataset. In addition, we make the following attempts to address this issue: *(i)* changing the network initialization *(ii)* increasing (or decreasing) the network capacity of the student discriminator, the teacher discriminators, and the generator *(iii)* increasing the number of iterations for updating the student discriminator and/or the generator. Despite some progress in preserving the gradients for larger iterations, none of the above attempts successfully eliminate the issue, as the training inevitably gets stuck within 1K iterations.

A.4 Additional Results

Effects of gradient clipping. We show in Figure A.1 the gradient norm distribution before and after gradient clipping. The clipping bound is set to be 1.1 for DP-SGD and 1 for our method. In contrast to DP-SGD, the clipping operation distorts less information in our framework, witnessed by a much smaller difference in the average gradient norm before and after the clipping. Moreover, the gradients used in our method exhibit much less variance both before and after the clipping compared with DP-SGD.



(a) DP-SGD (before) (b) DP-SGD (after) (c) Ours (before) (d) Ours (after)

Figure A.1: Effects of gradient clipping.

Comparison to Baselines. We provide the detailed quantitative results in Table A.1 and A.2, which are supplementary to Table 2.1 in the main paper. We show in parentheses the calibrated accuracy, i.e., the absolute accuracy of each classifier trained on generated data divided by the accuracy when trained on real data. The results are averaged over five runs.

	Real	GAN (non-private)	G-PATE	DP-SGD GAN	DP-Merf	DP-Merf AE	Ours
MLP	0.98	0.84 (85%)	0.25 (26%)	0.60 (61%)	0.63 (64%)	0.54 (55%)	0.79 (81%)
CNN	0.99	0.84 (85%)	0.51 (52%)	0.64 (65%)	0.63 (64%)	0.68 (69%)	0.80 (81%)
adaboost	0.73	0.28 (39%)	0.11 (16%)	0.32 (44%)	0.38 (52%)	0.21 (29%)	0.21 (29%)
bagging	0.93	0.46 (49%)	0.36 (38%)	0.44 (47%)	0.43 (46%)	0.33 (35%)	0.45 (48%)
bernoulli nb	0.84	0.80 (95%)	0.71 (84%)	0.62 (74%)	0.76 (90%)	0.50 (60%)	0.77 (92%)
decision tree	0.88	0.40 (45%)	0.13 (14%)	0.36 (41%)	0.29 (33%)	0.27 (31%)	0.35 (40%)
gaussian nb	0.56	0.71 (126%)	0.61 (110%)	0.37 (66%)	0.57 (102%)	0.17 (30%)	0.64 (114%)
gbm	0.91	0.50 (55%)	0.11 (12%)	0.45 (49%)	0.36 (40%)	0.20 (22%)	0.39 (43%)
lda	0.88	0.84 (95%)	0.60 (68%)	0.59 (67%)	0.72 (82%)	0.55 (63%)	0.78 (89%)
linear svc	0.92	0.81 (88%)	0.24 (26%)	0.56 (61%)	0.58 (63%)	0.43 (47%)	0.76 (83%)
logistic reg	0.93	0.83 (90%)	0.26 (28%)	0.60 (65%)	0.66 (71%)	0.55 (59%)	0.79 (85%)
random forest	0.97	0.39 (41%)	0.33 (34%)	0.63 (65%)	0.66 (68%)	0.45 (46%)	0.52 (54%)
xgboost	0.91	0.44 (49%)	0.15 (16%)	0.60 (66%)	0.70 (77%)	0.54 (59%)	0.50 (55%)
Average	0.88	0.63 (71%)	0.34 (40%)	0.52 (59%)	0.57 (66%)	0.42 (47%)	0.60 (69%)

Table A.1: Classification accuracy on MNIST ($\varepsilon = 10, \delta = 10^{-5}$).

Privacy-utility Curves. We show in Figure A.2 the privacy-utility curves of different methods when applied to the Fashion-MNIST dataset. We evaluate over three runs and show the

	Real	GAN (non-private)	G-PATE	DP-SGD GAN	DP-Merf	DP-Merf AE	Ours
MLP	0.88	0.77 (88%)	0.30 (34%)	0.50 (57%)	0.56 (64%)	0.56 (64%)	0.65 (74%)
CNN	0.91	0.73 (80%)	0.50 (54%)	0.46 (51%)	0.54 (59%)	0.62 (68%)	0.64 (70%)
adaboost	0.56	0.41 (74%)	0.42 (75%)	0.21 (38%)	0.33 (59%)	0.26 (46%)	0.25 (45%)
bagging	0.84	0.57 (68%)	0.38 (45%)	0.32 (38%)	0.40 (47%)	0.45 (54%)	0.47 (56%)
bernoulli nb	0.65	0.59 (91%)	0.57 (88%)	0.50 (77%)	0.62 (95%)	0.54 (83%)	0.55 (85%)
decision tree	0.79	0.53 (67%)	0.24 (30%)	0.33 (42%)	0.25 (32%)	0.36 (46%)	0.40 (51%)
gaussian nb	0.59	0.55 (93%)	0.57 (97%)	0.28 (47%)	0.59 (100%)	0.12 (20%)	0.48 (81%)
gbm	0.83	0.44 (53%)	0.25 (30%)	0.38 (46%)	0.27 (33%)	0.30 (36%)	0.38 (46%)
lda	0.80	0.77 (96%)	0.55 (69%)	0.55 (69%)	0.67 (84%)	0.65 (81%)	0.67 (84%)
linear svc	0.84	0.77 (91%)	0.30 (36%)	0.39 (46%)	0.46 (55%)	0.40 (48%)	0.65 (77%)
logistic reg	0.84	0.76 (90%)	0.35 (42%)	0.51 (61%)	0.59 (70%)	0.50 (60%)	0.68 (81%)
random forest	0.88	0.69 (78%)	0.33 (37%)	0.51 (58%)	0.61 (69%)	0.55 (63%)	0.54 (61%)
xgboost	0.83	0.65 (78%)	0.49 (59%)	0.52 (63%)	0.62 (75%)	0.55 (66%)	0.47 (57%)
Average	0.79	0.61 (77%)	0.40 (54%)	0.42 (53%)	0.50 (65%)	0.45 (56%)	0.53 (67%)

Table A.2: Classification accuracy on Fashion-MNIST ($\varepsilon = 10, \delta = 10^{-5}$).

corresponding mean and standard deviation. Similar to the results shown in Figure 2.4 in the main paper, our method achieves a consistent improvement over prior methods across a broad range of privacy budget ε.

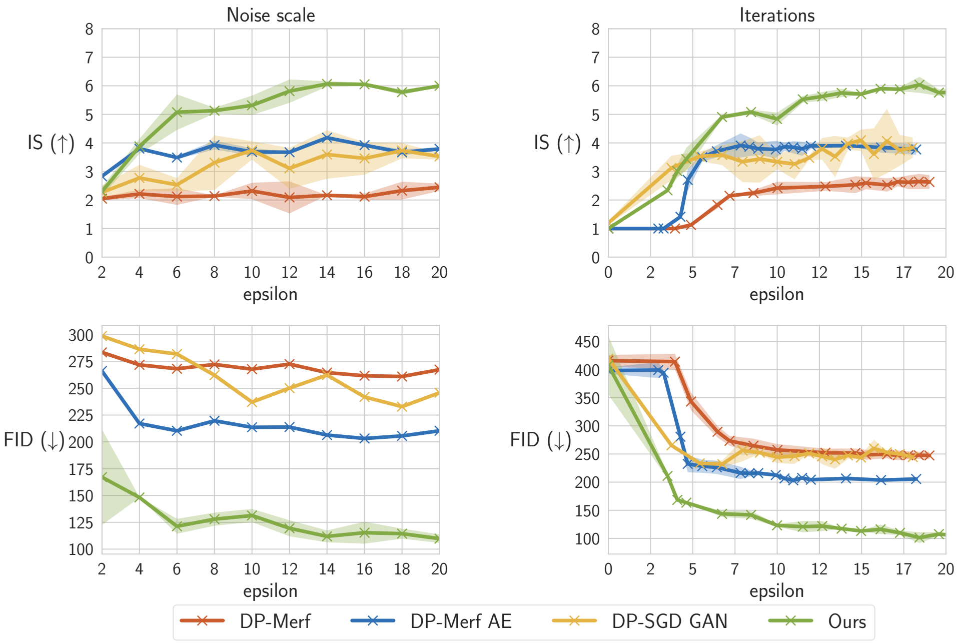


Figure A.2: Privacy-utility trade-off on Fashion-MNIST with $\delta = 10^{-5}$. (Left: Effects of noise scale. Right: Effects of Iterations.)

Private Set Generation with Discriminative Information B

These supplementary materials include the privacy analysis (§B.1), the details of the adopted algorithms (§B.2), and the details of experiment setup (§B.3), and additional results and discussions (§B.4). The source code is available at https://github.com/DingfanChen/Private-Set.

B.1 Privacy Analysis

Our privacy computation is based on the notion of Rényi-DP, which we recall as follows.

Definition B.1.1. (Rényi Differential Privacy (RDP) [142]). A randomized mechanism $\mathcal{M}$ is (α, ε)-RDP with order α, if

$$D_\alpha(\mathcal{M}(\mathcal{D}) \| \mathcal{M}(\mathcal{D}')) = \frac{1}{\alpha - 1} \log \mathbb{E}_{x \sim \mathcal{M}(\mathcal{D})} \left[\left(\frac{Pr[\mathcal{M}(\mathcal{D}) = x]}{Pr[\mathcal{M}(\mathcal{D}') = x]} \right)^{\alpha - 1} \right] \leq \varepsilon \tag{B.1}$$

holds for any adjacent datasets $\mathcal{D}$ and $\mathcal{D}'$, where $D_\alpha(P\|Q) = \frac{1}{\alpha-1}\log \mathbb{E}_{x\sim Q}[(P(x)/Q(x))^\alpha]$ is the Rényi divergence of order $\alpha > 1$ between the distributions P and Q.

To compute the privacy cost of our approach, we numerically compute $D_\alpha(\mathcal{M}(\mathcal{D})\|\mathcal{M}(\mathcal{D}'))$ in Definition B.1.1 for a range of orders α [143, 230] in each training step that requires access to the real gradient $g_{\boldsymbol{\theta}}^{\mathcal{D}}$. To obtain the overall accumulated privacy cost over multiple training iterations, we use the composition properties of RDP summarized by the following theorem.

Theorem B.1.1. (Adaptive Composition of RDP [143]). Let $f : \mathcal{D} \to \mathcal{R}_1$ be (α, ε_1)-RDP and $g : \mathcal{R}_1 \times \mathcal{D} \to \mathcal{R}_2$ be (α, ε_2)-RDP, then the mechanism defined as (X, Y), where $X \sim f(\mathcal{D})$ and $Y \sim g(X, \mathcal{D})$, satisfies $(\alpha, \varepsilon_1 + \varepsilon_2)$-RDP

In total, our private set generation (PSG) approach (shown in Algorithm 1 of the main paper) and the generator prior variant (shown in Algorithm 2) can be regarded as a composition over RTK (i.e., the number of iterations where the real gradient is used) homogenous subsampled Gaussian mechanisms (with the subsampling ratio $= B/N$) in terms of the privacy cost.

Lastly, we use the following theorem to convert (α, ε)-RDP to (ε, δ)-DP.

Theorem B.1.2. (From RDP to (ε, δ)-DP [142]). If $\mathcal{M}$ is a (α, ε)-RDP mechanism, then $\mathcal{M}$ is also $(\varepsilon + \frac{\log 1/\delta}{\alpha - 1}, \delta)$-DP for any $0 < \delta < 1$.

B.2 Algorithms

B.2.1 Objective

The distance $\mathcal{L}_{\text{dis}}$ (in Equation 5 of the main paper) between the real and synthetic gradients is defined to be the sum of cosine distance at each layer [261, 259]. Let $\boldsymbol{\theta}^l$ denote the weight at the l-th layer, the distance can be formularized as follows,

$$\mathcal{L}_{\text{dis}}(\nabla_{\boldsymbol{\theta}}\mathcal{L}(\mathcal{S},\boldsymbol{\theta}_t),\nabla_{\boldsymbol{\theta}}\mathcal{L}(\mathcal{D},\boldsymbol{\theta}_t)) = \sum_{l=1}^{L} d(\nabla_{\boldsymbol{\theta}^l}\mathcal{L}(\mathcal{S},\boldsymbol{\theta}_t),\nabla_{\boldsymbol{\theta}^l}\mathcal{L}(\mathcal{D},\boldsymbol{\theta}_t))$$

where d denotes the cosine distance between the gradients at each layer:

$$d(\boldsymbol{A},\boldsymbol{B}) = \sum_{i=1}^{out}\left(1 - \frac{\boldsymbol{A}_{i\cdot}\cdot\boldsymbol{B}_{i\cdot}}{\|\boldsymbol{A}_{i\cdot}\|\|\boldsymbol{B}_{i\cdot}\|}\right)$$

$\boldsymbol{A}_{i\cdot}$ and $\boldsymbol{B}_{i\cdot}$ are the flattened gradient vectors to each output node i. For FC layers, $\boldsymbol{\theta}^l$ is a 2D tensor with dimension $out \times in$ and the flattened gradient vector has dimension in, while for Conv layer, $\boldsymbol{\theta}^l$ is a 4D tensor with dimensionality $out \times in \times h \times w$ and the flattened vector has dimension $in \times h \times w$. out,in, h, w corresponds to the number of output and input channels, kernel height, and width, respectively.

B.2.2 Generator Prior

We present the pseudocode of the generator prior experiments (Section 6 of the main paper) in Algorithm 2, which is supplementary to Figure 3.4-3.6 and Equation 3.8 of the main paper.

The only difference to the original PSG formulation is that the samples are restricted to be the output of a generator network and the updates are conducted on the generator network parameters (See Figure 3.4 for the illustration and see Figure 1 in the main paper for a comparison). Note that the generator network is freshly initialized (i.e., untrained) when we commence the training process, thereby restricting the prior image to originate only from the convolutional structure instead of utilizing additional public knowledge for data. Additionally, we fix the random latent code z_i during the whole training process to guarantee that there is no other randomness/degree of freedom except that introduced by the generator network itself. While it is possible to allow random sampling of the latent code and generate changeable $\mathcal{S}$ to mimic the training of generative models (i.e., train a generative network using the gradient matching loss), we observe that the training easily fails in the early stage. We argue that this also indicates that training a generative network is a harder task than training a set of samples directly, which explains the better convergence behavior and superior final performance of our formulation in comparison to existing works (which build on top of deep generative networks).

Algorithm 2: Private Set Generation with Generator Prior

```
Input: Dataset $\mathcal{D} = \{(\boldsymbol{x}_i, y_i)\}_{i=1}^{N}$, learning rate for update network parameters $\tau_{\boldsymbol{\theta}}$ and
    $\tau_{\boldsymbol{\varphi}}$, sampling probability $\rho$, DP noise scale $\sigma$, gradient clipping bound $C$, number
    of runs $R$, outer iterations $T$, inner iterations $J$, batches $K$, number of classes $L$,
    number of samples per class (spc), desired privacy cost $\varepsilon$ given a pre-defined $\delta$
Output: Synthetic set $\mathcal{S}$
Compute the DP noise scale $\sigma$ numerically so that the privacy cost equals to $\varepsilon$ after the
  training; Initialize model parameter $\boldsymbol{\varphi}$ of the conditional generator $G$;
for $c$ in $\{1, ..., L\}$ do
    for sample_index in spc do
        $y_i^{\mathcal{S}} = c$ ;
        Sample $\boldsymbol{z}_i \sim \mathcal{N}(0, I)$ ($\boldsymbol{z}_i$ is fixed for each corresponding synthetic sample during
          the training) ;
        $\boldsymbol{x}^{\mathcal{S}} = G(\boldsymbol{z}_i, y_i^{\mathcal{S}}; \boldsymbol{\varphi})$;
        Insert $(\boldsymbol{x}_i^{\mathcal{S}}, y_i^{\mathcal{S}})$ into $\mathcal{S}$;
    end
end
for run in $\{1, ..., R\}$ do
    Initialize model parameter $\boldsymbol{\theta}_0 \sim P_{\boldsymbol{\theta}_0}$;
    for outer_iter in $\{1, ..., T\}$ do
        $\boldsymbol{\theta}_{t+1} = \boldsymbol{\theta}_t$
        for batch_index in $\{1, ..., K\}$ do
            Sample a batch $\{(x_i, y_i)\}_{i=1}^{B_k}$, where each $(x_i, y_i)$ from $\mathcal{D}$ is uniformly
              sampled with probability $\rho$; for each $(\boldsymbol{x}_i, y_i)$ in the batch do
                // Compute per-example gradients on real data
                        $g_{\boldsymbol{\theta}_t}^{\mathcal{D}}(\boldsymbol{x}_i) = \ell(F(\boldsymbol{x}_i; \boldsymbol{\theta}_t), y_i)$
                // Clip gradients
                        $\widetilde{g_{\boldsymbol{\theta}_t}^{\mathcal{D}}}(\boldsymbol{x}_i) = g_{\boldsymbol{\theta}_t}^{\mathcal{D}}(\boldsymbol{x}_i) \cdot \min(1, C / \|g_{\boldsymbol{\theta}_t}^{\mathcal{D}}(\boldsymbol{x}_i)\|_2)$
            end
            // Add noise to average gradient with Gaussian mechanism
                    $\widetilde{g_{\boldsymbol{\theta}_t}^{\mathcal{D}}} = \frac{1}{B_k} \sum_{i=1}^{B_k} (\widetilde{g_{\boldsymbol{\theta}_t}^{\mathcal{D}}}(\boldsymbol{x}_i) + \mathcal{N}(0, \sigma^2 C^2 I))$
            // Compute parameter gradients on synthetic data and update $G$
                $g_{\boldsymbol{\theta}_t}^{\mathcal{S}} = \nabla_{\boldsymbol{\theta}} \mathcal{L}(\mathcal{S}, \boldsymbol{\theta}_t)) = \frac{1}{M} \sum_{i=1}^{M} \ell(F(\boldsymbol{x}_i^{\mathcal{S}}; \boldsymbol{\theta}_t), y_i^{\mathcal{S}})$ where $\boldsymbol{x}_i^{\mathcal{S}} = G(\boldsymbol{z}_i, y_i^{\mathcal{S}}; \boldsymbol{\varphi})$
                $\boldsymbol{\varphi} = \boldsymbol{\varphi} - \tau_{\boldsymbol{\varphi}} \cdot \nabla_{\boldsymbol{\varphi}} \mathcal{L}_{\text{dis}}(g_{\boldsymbol{\theta}_t}^{\mathcal{S}}, \widetilde{g_{\boldsymbol{\theta}_t}^{\mathcal{D}}})$
        end
        for inner_iter in $\{1, ..., J\}$ do
            // Update network parameter using $\mathcal{S}$
                    $\mathcal{S} = \{G(\boldsymbol{z}_i, y_i^{\mathcal{S}}; \boldsymbol{\varphi}), y_i^{\mathcal{S}}\}_{i=1}^{M}$
                    $\boldsymbol{\theta}_t = \boldsymbol{\theta}_t - \tau_{\boldsymbol{\theta}} \cdot \nabla_{\boldsymbol{\theta}} \mathcal{L}(\mathcal{S}, \boldsymbol{\theta}_t)$
        end
    end
end
return Synthetic set $\mathcal{S}$
```

B.3 Experiment Setup

B.3.1 Datasets

MNIST [108] dataset contains 28×28 grayscale images of digit numbers. The dataset comprises 60K training images and 10K testing images in total. The task is to classify the image into one of the 10 classes based on the digit number it contains.

Fashion-MNIST [238] dataset consists of 28×28 grayscale images fashion products of 10 categories. The total dataset size is 60K for the training set and 10K for the testing set, respectively. The task is to classify the fashion product given in the images.

B.3.2 Required Resources and Computational Complexity

All our models and methods are implemented in PyTorch. Our experiments are conducted with Nvidia Tesla V100 and Quadro RTX8000 GPUs and a common configuration with 16GB GPU memory is sufficient for conducting all our experiments.

In comparison to normal non-private training, the major part of the additional memory and computation cost is introduced by the DP-SGD [1] step (for the per-sample gradient computation) that sanitizes the parameter gradient on real data, while the other steps (including the update on $\mathcal{S}$, and the updates of $F(\cdot;\boldsymbol{\theta})$ on $\mathcal{S}$ are equivalent to multiple calls of the normal non-private forward and backward passes (whose costs have lower magnitude than the DP-SGD step). Moreover, our formulation requires much less computational and memory consumption than previous works that require training multiple instances of the generative modules [32, 130, 227].

B.3.3 Hyperparameters

Training. We set the default value of hyperparameters as follows: batch size $= 256$ for both computing the parameter gradients in the outer iterations and for update the classifier F in the inner iterations, gradient clipping bound $C = 0.1$, $R = 1000$ for $\varepsilon = 10$ (and $R = 200$ for $\varepsilon = 1$), $K = 10$. The number of inner J and outer T iterations are dependent on the number of samples per class (spc), as more samples generally requires more iterations till convergence: (T, J) is set to be $(1, 1)$, $(10, 50)$, $(20, 25)$ and $(50, 10)$ for $spc = 1, 10, 20, 50$, respectively. The DP noise scale σ is calculated numerically[1] [1, 143] so that the privacy cost equals to ε after the training (with RTK steps in total that consume privacy budget), given that $\delta = 10^{-5}$. The learning rate is set to be $\tau_{\boldsymbol{\theta}} = 0.01$ (and $\tau_{\boldsymbol{\varphi}} = 0.01$ for training with generator prior) and $\tau_{\mathcal{S}} = 0.1$ for updating the network parameters and samples, respectively. We use SGD optimizer for the classifier F, and samples $\mathcal{S}$ (with momentum$= 0.5$), while we use Adam optimizer for the generator G if trained with prior. For the training process, no data augmentation is adopted. Our implementation of the DP-SGD step and the uniform data sampling operation is based on the Opacus [246] [2] package.

Evaluation. We set the epoch to be 40 and 300 when training the downstream classification models on the synthetic data with "full" size ($spc = 6000$) and small size ($spc \in \{1, 10, 20, 50\}$),

[1] Based on Google's TensorFlow privacy under version ≤ 0.8.0: https://github.com/tensorflow/privacy/blob/master/tensorflow_privacy/privacy/analysis/rdp_accountant.py

[2] https://opacus.ai/

respectively, to guarantee the convergence of downstream model training and maintain the evaluation efficiency. We set the learning rate to be 0.01 at the beginning and decrease it (by multiplying with 0.1) when half of the total epoch is achieved. We use SGD optimizer with momentum= 0.9, weight decay= $5 \cdot 10^{-4}$ and set batch size $= 256$ for training the classifier. Random cropping and re-scaling are adopted as data augmentation when training the classification model.

B.3.4 Baseline Methods

We present more details about the implementation of the baseline methods. In particular, we provide the default value of the privacy hyperparameters below.

DP-Merf [70] [3] For $\varepsilon = 1$ we use as the default hyperparameters setting provided in the official implementation: DP noise scale $\sigma = 5.0$, training epoch $= 5$, while for $\varepsilon = 10$, the DP noise scale is $\sigma = 0.568$.

DP-CGAN [211] [4] We set the default hyper-parameters as follows: gradient clipping bound $C = 1.1$, noise scale $\sigma = 2.1$, batch size= 600 and total number of training iterations= $30K$. We exclude this model from evaluation at $\varepsilon = 1$ as the required noise scale is too large for the training to make progress, which is consistent with the results in literature [70, 32].

GS-WGAN [32] [5] We adopt the default configuration provided by the official implementation ($\varepsilon = 10$): the subsampling rate $= 1/1000$, DP noise scale $\sigma = 1.07$, batch size $= 32$. Following [32], we pretrain (warm-start) the model for $2K$ iterations, and subsequently train for 20K iterations. Similar to the case for DP-CGAN, we exclude this model from evaluation at $\varepsilon = 1$ as the required noise scale is too large for the training to be stable.

For **G-PATE** [130], **DataLens** [227] and **DP-Sinkhorn** [24], we present the same results as reported in the original papers (Table 3.1 of the main paper) as reference, as they are either not directly comparable to ours or not open-sourced.

B.3.5 Private Continual Learning

Setting. The experiments presented in Section 5.2 of the main paper correspond to the class-incremental learning setting [173] where the data partition at each stage contains data from disjoint subsets of label classes. And the task protocol is sequentially learning to classify a given sample into all the classes seen so far. For our experiments on SplitMNIST and SplitFashionMNIST benchmarks [252], the datasets are split into 5 partitions each containing samples of 2 label classes. The evaluation task is thus binary classification for the first stage, while two more classes are included after each following stage.

While a clear definition of the private continual learning setting is, to the best of our knowledge, missing in the literature, we introduce a basic case where privacy can be strictly protected during the whole training process. In brief, we need to guarantee that all the information that is delivered to another party/stage should be privacy-preserving.

Hence, for the **DP-SGD** [1] baseline, the classification model is initialized to be a 10-class classifier, and is updated (fine-tuned) via DP-SGD at each training stage on each data partition.

[3] https://github.com/frhrdr/dp-merf

[4] https://github.com/reihaneh-torkzadehmahani/DP-CGAN

[5] https://github.com/DingfanChen/GS-WGAN

During the whole process, the model is transferred between different parties while privacy is guaranteed by DP-SGD training.

And for the private generation methods, i.e., **DP-Merf** and **Ours**, we use a fixed privacy budget to train a private generative model or a private synthetic set for each partition/stage. Subsequently, such a generative model or synthetic set is transferred between parties for conducting different training stages. For evaluation, a n-class classifier is initialized and then trained on the transferred private synthetic samples for each stage, where n is the total number of label classes seen so far. In our experiments, both methods only exploit information from the local partition for the generation, i.e., our private set is optimized on a freshly initialized classification network at each stage and for DP-Merf the mean embedding is taken over the local partition. While our formulation can be adjusted to (and may be further improved by) more advanced training strategies designed for continual learning to eliminate forgetting, many of such strategies are not directly compatible with private training as they require access to old data. We believe that our introduced private continual learning setting is of independent interest and leave an in-depth investigation of this topic as future work.

Hyperparameters. We use the default values for the hyperparameters as shown in Section B.3.3 and B.3.4, except that the training epoch is set to be 10 for **DP-SGD** and the runs $R = 200$ for **Ours**, to balance the convergence, forgetting effect, and evaluation efficiency. Moreover, the DP noise scale is calibrated to each *partition* of the data.

B.4 Additional Results and Discussions

B.4.1 Dataset Distillation Basis

In this paper, we propose to use the gradient matching technique [261, 259] (among existing dataset distillation approaches) as a basis for private set generation. In the following, we briefly discuss other popular dataset condensation approaches that achieve competitive performance for non-private tasks but appear less suitable for private learning. For example, [229] requires solving a nested optimization problem, which makes it hard to quantify the individual's effect (i.e., the sensitivity) and thus difficult to impose DP into the training. In addition, [260] relies on "per-class" feature aggregation as the only source of supervision to guide the synthetic data towards representing its target label class. However, this "per-class" operation contradicts label privacy and the requirement of uniform sampling for the privacy cost computation. In contrast, our formulation adopts uniform sampling (which is compatible with DP) and exploits the (inherently class-dependent) gradient signals to generate representative samples.

B.4.2 Computation Time

Under the default setting (See Section B.3.2 and B.3.3), it takes around 4.5 hours and 11 hours to train the synthetic data for the case of $spc = 10$ and $spc = 20$, respectively. To the best of our knowledge, our method is more efficient than existing works that require pre-training of (multiple) models [32, 130], but requires more running time than methods that use static pre-computed features [70]. Moreover, we see a tendency that the distilled dataset requires less time on downstream tasks compared to samples from generative models due to the smaller (distilled) sample size.

B.4.3 Evaluation on Colored Images

In this section, we provide additional evaluation results on colored image benchmark dataset. On CIFAR-10 [104] dataset, We use the same default setting as described in Section B.3.3 and adjust the network architectures to the input dimension ($32 \times 32 \times 3$). We summarize in Figure B.1 the quantitative results of downstream utility when varying the number of samples per class ($spc \in \{1, 10, 20\}$) and show as reference the results when training non-privately (We show here the results when applying uniform sampling of the data instead of the original per-class sampling approach [261, 259] also for the non-private baseline for controlled comparison). Additionally, we show in Figure B.1 the synthetic images when training under DP ($\varepsilon = 10$), and in B.2 the results when training non-privately. We observe that while the synthetic samples look noisy and non-informative, they do provide useful features for downstream classifiers, leading to a decent level of performance. Note that colored images are generally challenging for private learning. In fact, this makes our work the first one that is able to report non-trivial performance on this dataset.

	1	10	20
non-private	30.0	48.6	52.6
$\varepsilon = 10$	28.9	40.3	42.6

Table B.1: Test accuracy (%) on real data of downstream ConvNet classifier on CIFAR-10.

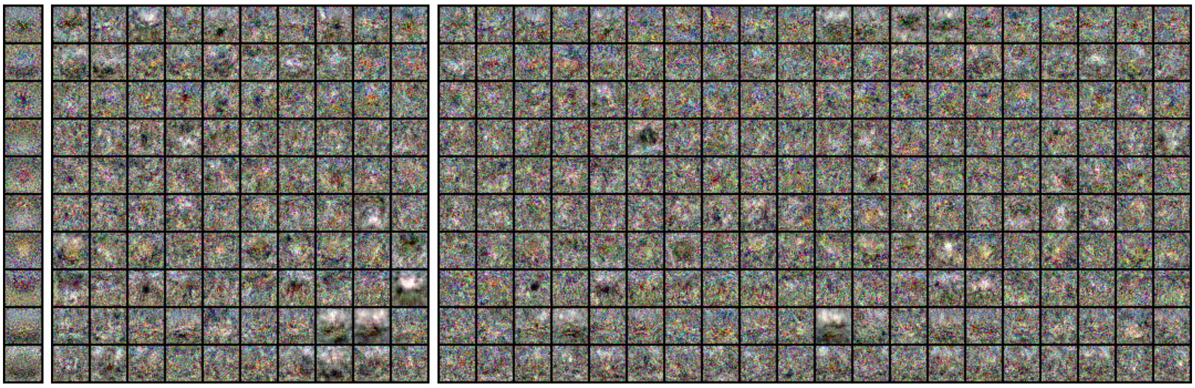


Figure B.1: CIFAR-10 ($\varepsilon = 10$)

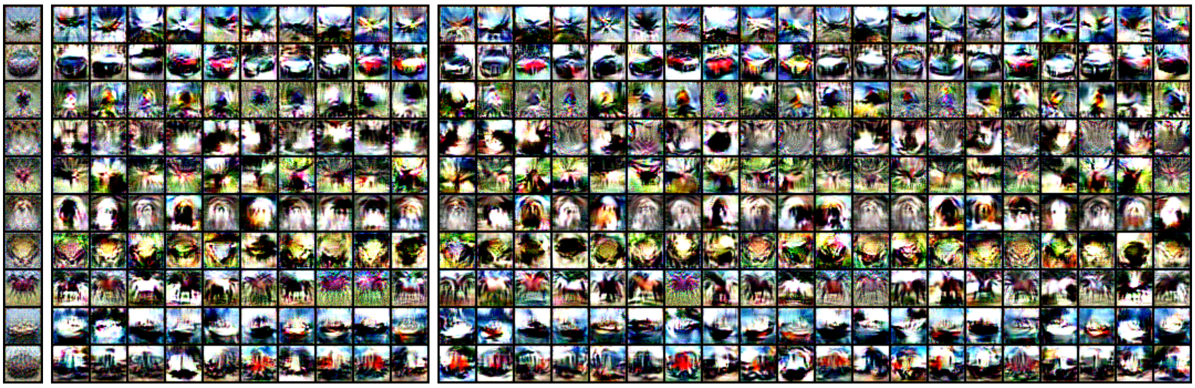


Figure B.2: CIFAR-10 (non-private)

C A Unified View of Differentially Private Deep Generative Modeling

This appendix provides additional details to the main paper presented in Chapter 4: we provide additional notes in §C.1, additional sensitivity analysis in §C.2, and additional background on DP in §C.3.

C.1 Additional Notes on Potential Methods with Privacy Barrier B1

In the DP deep generative modeling literature, existing approaches with privacy barrier between **Real data** and **Measurement** (Section 4.4.1) typically release sanitized features in a *condensed and aggregated* form. In this sense, recent approaches, which may deviate from the general "mean embedding" formulation (as shown in Equation 4.3-4.4), but still publish a sanitized statistical summary of the private dataset, such as **DPSDA** [122], fall into this category. Specifically, **DPSDA** sanitizes a count histogram that summarizes the distribution of real data and employs it as a measurement to refine the synthetic data distribution, thereby rendering it more similar to the real private data distribution.

However, one might wonder if it is feasible to release a DP database in the *original* form of the real data, *prior to* the training of a generative model. A positive example of this idea can be found in the Small Database Mechanism (**SmallDB**) in the context of private query release, introduced in Section 4.1 of [53]. This mechanism outputs a sanitized database in the same form as the original data, by selecting the database (from all possible sets of the data universe) via the exponential mechanism with a utility function of the negative error to the query release problem (difference in the query answer on the synthetic versus the real database). However, as the name suggests, the use of such an algorithm is largely limited to small (low-dimensional) datasets. This is mainly due to the exponential growth of the data universe with dimensionality, which drastically increases the computational burden and undermines the accuracy guarantees.

While **DP-GEN** [36] attempted to apply a similar idea to deep generative models, the output space of their generation method only supports (has non-zero probability) combinations of its input *private* dataset (See detailed proofs in Appendix B of [47]), instead of the entire data universe. This invalidates their claimed privacy guarantee, and the performance of a proper implementation of such a "direct database release" approach on high-dimensional data remains unclear.

C.2 Additional Sensitivity Analysis

C.2.1 Privacy barrier B1

Sensitivity of DP-Merf [70] and the General Formulation in Section 4.4.1. It can be clearly seen that the L_2-sensitivity for the replace-one notion is $\frac{2}{m}$, where $m = |\mathcal{D}|$ represents the size of the private dataset, as demonstrated in the original paper. Subsequently, we proceed to derive a conservative bound for the sensitivity value in the DP-Merf method under the add-or-remove-one DP notion, which can be generalized to other approaches within the same category (Section 4.4.1), including [120, 224, 71, 243]. For the add-one case, we let $m = |\mathcal{D}|$ and assume, without loss of generality, that $\mathcal{D}' = \mathcal{D} \cup \{\boldsymbol{x}'_{m+1}\}$ and $\boldsymbol{x}'_i = \boldsymbol{x}_i$ for all $i = 1, ..., m$.

$$
\begin{aligned}
\Delta^2 &= \max_{\mathcal{D},\mathcal{D}'} \left\| \frac{1}{m+1} \sum_{i=1}^{m+1} \phi(\boldsymbol{x}'_i) - \frac{1}{m} \sum_{i=1}^{m} \phi(\boldsymbol{x}_i) \right\|_2 \\
&= \max_{\boldsymbol{x}'_{m+1}, \boldsymbol{A}} \left\| \frac{1}{m+1} (\phi(\boldsymbol{x}'_{m+1}) + \boldsymbol{A}) - \frac{1}{m} \boldsymbol{A} \right\|_2 \\
&= \max_{\boldsymbol{x}'_{m+1}, \boldsymbol{A}} \left\| \frac{1}{(m+1)m} \boldsymbol{A} - \frac{1}{m+1} \phi(\boldsymbol{x}'_{m+1}) \right\|_2 \\
&\leq \max_{\boldsymbol{A}} \left\| \frac{1}{(m+1)m} \boldsymbol{A} \right\|_2 + \max_{\boldsymbol{x}'_{m+1}} \left\| \frac{1}{m+1} \phi(\boldsymbol{x}'_{m+1}) \right\|_2 \\
&\leq \frac{1}{(m+1)m} m + \frac{1}{m+1} = \frac{2}{m+1}
\end{aligned}
$$

where $\boldsymbol{A} = \sum_{i=1}^{m} \phi(\boldsymbol{x}_i)$ for brevity. The inequalities follow from the triangle inequality and the fact that $\|\phi(\cdot)\|_2 = 1$

Similarly, for the remove-one case, we let $m = |\mathcal{D}|$, $\mathcal{D}' \cup \{\boldsymbol{x}_m\} = \mathcal{D}$ and $\boldsymbol{x}'_i = \boldsymbol{x}_i$ for all $i = 1, ..., m-1$.

$$
\begin{aligned}
\Delta^2 &= \max_{\mathcal{D},\mathcal{D}'} \left\| \frac{1}{m-1} \sum_{i=1}^{m-1} \phi(\boldsymbol{x}'_i) - \frac{1}{m} \sum_{i=1}^{m} \phi(\boldsymbol{x}_i) \right\|_2 \\
&= \max_{\boldsymbol{x}_m, \boldsymbol{A}} \left\| \frac{1}{m-1} \boldsymbol{A} - \frac{1}{m} (\boldsymbol{A} + \phi(\boldsymbol{x}_m)) \right\|_2 \\
&= \max_{\boldsymbol{x}_m, \boldsymbol{A}} \left\| \frac{1}{(m-1)m} \boldsymbol{A} - \frac{1}{m} \phi(\boldsymbol{x}_m) \right\|_2 \\
&\leq \max_{\boldsymbol{A}} \left\| \frac{1}{(m-1)m} \boldsymbol{A} \right\|_2 + \max_{\boldsymbol{x}_m} \left\| \frac{1}{m} \phi(\boldsymbol{x}_m) \right\|_2 \\
&\leq \frac{1}{(m-1)m} (m-1) + \frac{1}{m} = \frac{2}{m}
\end{aligned}
$$

with $\boldsymbol{A} = \sum_{i=1}^{m-1} \phi(\boldsymbol{x}_i)$. The inequalities follow from the triangle inequality and the fact that $\|\phi(\cdot)\|_2 = 1$

Sensitivity of DP-SWD [171]. The sensitivity is calculated as the maximum difference over two embeddings, determined after performing random projections on two neighboring datasets. The "replace-one" notion is adopted to simplify the analysis. With a probability of at least $1 - \delta$, it can be shown that:

$$
\|\boldsymbol{XU} - \boldsymbol{X}'\boldsymbol{U}\|_F^2 \leq w(k, \delta)
$$

with $w(k,\delta) = \frac{k}{d} + \frac{2}{3}\ln\frac{1}{\delta} + \frac{2}{d}\sqrt{k\frac{d-1}{d+2}\ln\frac{1}{\delta}}$. Here $\boldsymbol{X}, \boldsymbol{X}'$ denote data matrices in $\mathbb{R}^{|\mathcal{D}|\times d}$ for neighboring datasets $\mathcal{D}, \mathcal{D}'$ under the bounded-DP notion, while $\boldsymbol{U} \in \mathbb{R}^{d\times k}$ represents the random projection matrix with each column independently drawn from $\mathbb{S}^{d-1}$. Additionally, it is ensured that $\|\boldsymbol{X}_{i,:} - \boldsymbol{X}'_{i,:}\|_2 \leq 1$ for all i by pre-processing the dataset, making each sample record have unit norm. To prove the desired result, the sensitivity is first transformed into a summation of k i.i.d random variables following the beta distribution $B(1/2,(d-1)/2)$, which then allows the application of Bernstein's inequality to establish concentration bounds for the summation. For a more detailed proof, please refer to Appendix 8.1-8.2 in [171].

Sensitivity of DPSDA [122]. The core component of DPSDA is the method of constructing a nearest neighbors histogram that describes the real data distribution while providing DP guarantees (refer to Algorithm 2 in [122]). Specifically, for every real sample $\boldsymbol{x}_i$ in the private dataset $\mathcal{D}$, the algorithm identifies its nearest synthetic counterparts and constructs a histogram. This histogram represents the frequency of each existing synthetic sample $\boldsymbol{s}_k$ being the closest to the real samples. Given a synthetic dataset consisting of n samples $\{\boldsymbol{s}_k\}_{k=1}^{n}$ and let $m = |\mathcal{D}|$:

$$h_j = \left|i : i \in [m], j = \underset{k\in[n]}{\arg\min}\, d(\boldsymbol{x}_i, \boldsymbol{s}_k)\right| \quad \text{for } j = 1, ..., n$$

where $\boldsymbol{h} = (h_1, ..., h_n)$ builds up the histogram with each h_j reflecting the number of real samples for which the corresponding synthetic sample $\boldsymbol{s}_j$ is the nearest neighbor, based on the distance metric d. Subsequently, DP Gaussian noise is added to the histogram for providing privacy guarantees: $\boldsymbol{h} = \boldsymbol{h} + \mathcal{N}(0, \sigma\boldsymbol{I})$.

For the add-or-remove-one notion, we can assume that w.l.o.g. the neighboring datasets $\mathcal{D}, \mathcal{D}'$ satisfy $\mathcal{D}' \cup \{\boldsymbol{x}_m\} = \mathcal{D}$ (or $\mathcal{D}' = \mathcal{D} \cup \{\boldsymbol{x}_m\}$). Let $\boldsymbol{s}_j$ be the closest synthetic sample to $\boldsymbol{x}_m$ and $\boldsymbol{h}, \boldsymbol{h}'$ represent the histograms on $\mathcal{D}$ and $\mathcal{D}'$ respectively. The L_2-sensitivity is then given by:

$$\begin{aligned}
\Delta^2 &= \max_{\mathcal{D},\mathcal{D}'} \|(h_1, \cdots, h_n) - (h'_1, \cdots, h'_n)\|_2 \\
&= \max_{h_j, h'_j} \|(0, ..., 0, h_j - h'_j, 0, ..., 0)\|_2 \\
&= 1
\end{aligned}$$

For the replace-one notion, we define neighboring datasets $\mathcal{D}, \mathcal{D}'$ to satisfy $\mathcal{D}' \cup \{\boldsymbol{x}_m\} = \mathcal{D} \cup \{\boldsymbol{x}'_m\}$ with $\boldsymbol{x}_m \neq \boldsymbol{x}'_m$. The L_2-sensitivity is defined by:

$$\begin{aligned}
\Delta^2 &= \max_{\mathcal{D},\mathcal{D}'} \|(h_1, \cdots, h_n) - (h'_1, \cdots, h'_n)\|_2 \\
&= \max_{h_j, h'_j, h_k, h'_k} \|(0, ..., 0, h_j - h'_j, 0, ..., 0, h_k - h'_k, 0, ..., 0)\|_2 \\
&= \sqrt{1^2 + 1^2} = \sqrt{2}
\end{aligned}$$

where $\boldsymbol{s}_j$ and $\boldsymbol{s}_k$ are the closet synthetic samples to $\boldsymbol{x}_m$ and $\boldsymbol{x}'_m$ respectively, while w.l.o.g. $j < k$.

C.2.2 Privacy barrier **B2**

The sensitivity analysis for methods in this category inherits the approach used in the DP-SGD and the PATE framework, which is presented below.

Sensitivity of DP-SGD (Section 4.2.1.1). The main component of the DP-SGD algorithm can be formalized as follows:

$$\text{Clip: } \bar{g}_t(\boldsymbol{x}_i) \leftarrow g_t(\boldsymbol{x}_i) / \max\left(1, \frac{\|g_t(\boldsymbol{x}_i)\|_2}{C}\right)$$
$$\text{Add noise: } \widetilde{g}_t \leftarrow \frac{1}{B}\Big(\sum_i \bar{g}_t(\boldsymbol{x}_i) + \mathcal{N}(0, \sigma^2 C^2 \boldsymbol{I})\Big)$$

where $g_t(\boldsymbol{x}_i) = \nabla_{\boldsymbol{\theta}_t}\mathcal{L}(\boldsymbol{\theta}_t, \boldsymbol{x}_i)$ denotes the gradient on sample $\boldsymbol{x}_i$ at iteration t, C represents the clipping bound, B is the batch size, σ is the noise scale, and the summation is taken over all samples in the batch. The sensitivity in DP-SGD is computed as:

$$\Delta^2 = \max_{\mathcal{D},\mathcal{D}'} \|\sum_i \bar{g}_t(\boldsymbol{x}_i) - \sum_i \bar{g}_t(\boldsymbol{x}_i')\|_2$$

For the add-or-remove-one DP notion, let $\mathcal{D}', \mathcal{D}$ only differ in the existence of $\boldsymbol{x}_i'$, i.e., $\mathcal{D}' = \mathcal{D} \cup \{\boldsymbol{x}_i'\}$, it is easy to see that

$$\Delta^2 = \max_{\boldsymbol{x}_i'} \|\bar{g}_t(\boldsymbol{x}_i')\|_2 \leq C$$

For the replace-one DP notion, w.l.o.g. let $\mathcal{D}' \cup \{\boldsymbol{x}_i'\} = \mathcal{D} \cup \{\boldsymbol{x}_i\}$, thus

$$\Delta^2 = \max_{\boldsymbol{x}_i', \boldsymbol{x}_i} \|\bar{g}_t(\boldsymbol{x}_i) - \bar{g}_t(\boldsymbol{x}_i')\|_2 \leq 2C$$

due to the triangle inequality.

Sensitivity of PATE (Section 4.2.1.2). Given m teachers, c possible label classes and an input vector $\boldsymbol{x}$, the "votes" of teachers that assign class j to a query input $\bar{\boldsymbol{x}}$ is denoted as:

$$n_j(\bar{\boldsymbol{x}}) = |i : i \in [m], f_i(\bar{\boldsymbol{x}}) = j| \quad \text{for } j = 1, ..., c$$

with f_i denotes the i-th teacher model. And the histogram of the teachers' vote histogram is:

$$\bar{n}(\bar{\boldsymbol{x}}) = (n_1, \cdots, n_c) \in \mathbb{N}^c$$

As each training data sample only influences a single teacher due to the disjoint partitioning, changing one data sample in the training dataset—whether it's removal, addition, or replacement—will at most alter the votes (by 1) for two classes, denoted here as classes i and j, on any possible query sample $\bar{\boldsymbol{x}}$. Let the vote histograms resulting from neighboring datasets $\mathcal{D}, \mathcal{D}'$ be $(n_1, \cdots, n_c)$ and $(n_1', \cdots, n_c')$ respectively, the global sensitivity can be represented as:

$$\begin{aligned}
\Delta^1 &= \max_{\mathcal{D},\mathcal{D}'} \|(n_1, \cdots, n_c) - (n_1', \cdots, n_c')\|_1 \\
&= \max_{n_i, n_i', n_j, n_j'} \|(0, ..., 0, n_i - n_i', 0, ..., 0, n_j - n_j', 0, ..., 0)\|_1 \\
&= \max_{n_i, n_i'} |n_i - n_i'| + \max_{n_j, n_j'} |n_j - n_j'| \leq 2 \\
\Delta^2 &= \max_{n_i, n_i', n_j, n_j'} \|(0, ..., 0, n_i - n_i', 0, ..., 0, n_j - n_j', 0, ..., 0)\|_2 \\
&= \max_{n_i, n_i', n_j, n_j'} \sqrt{(n_i - n_i')^2 + (n_j - n_j')^2} \leq \sqrt{2}
\end{aligned}$$

This holds for all possible query samples $\bar{\boldsymbol{x}}$.

The L_1- and L_2-sensitivities calibrate the two variants of noise mechanisms used in PATE: the Gaussian NoisyMax (GNMax) and the max-of-Laplacian (LNMax). The GNMax is defined as:

$$\text{PATE}_\sigma(\bar{\boldsymbol{x}}) = \arg\max_{j\in[c]}\{n_j(\bar{\boldsymbol{x}}) + \mathcal{N}(0,\sigma^2)\}$$

and the LNMax as:

$$\text{PATE}_\gamma(\bar{\boldsymbol{x}}) = \arg\max_{j\in[c]}\{n_j(\bar{\boldsymbol{x}}) + Lap(1/\gamma)\}$$

C.2.3 Privacy barrier **B3**

Sensitivity of GS-WGAN [32] and DP-Sinkhorn [24]. The sensitivity for both GS-WGAN and DP-Sinkhorn can be derived via triangle inequality:

$$\begin{aligned}\Delta^2 &= \max_{\mathcal{D},\mathcal{D}'} \|f(g_G^{\text{upstream}}) - f({g'_G}^{\text{upstream}})\|_2 \\ &\leq \max_{\mathcal{D}} \|f(g_G^{\text{upstream}})\|_2 + \max_{\mathcal{D}'} \|f({g'_G}^{\text{upstream}})\|_2 \\ &\leq 2C\end{aligned}$$

with f denoting the gradient clipping operation and C the clipping bound. Notably, no matter which privacy notion is used, both terms ($\max_{\mathcal{D}} \|f(g_G^{\text{upstream}})\|_2$ and $\max_{\mathcal{D}'} \|f({g'_G}^{\text{upstream}})\|_2$) are upper-bounded by the gradient clipping bound C.

Sensitivity of DataLens [227]. Given m teachers, the d-dimensional gradients yielded from each teacher i after applying top-k sign quantization take the following form (refer to Algorithm 2 in [227]):

$$\hat{g}_i \in \{0,1,-1\}^d \quad \text{with} \quad \|\hat{g}_i\|_1 = k \quad \text{and} \quad \|\hat{g}_i\|_2 = \sqrt{k}$$

In other words, g_i contains exactly k non-zero elements, with the non-zero elements taking values of either 1 or -1, depending on the sign of the original upstream gradient.

Consider gradient sets $\{\hat{g}_i\}_{i=1}^m$ and $\{\hat{g}'_i\}_{i=1}^m$ which originate from neighboring datasets $\mathcal{D}$ and $\mathcal{D}'$ respectively. As the influence of each data point is limited to a single teacher model, these gradient sets differ by at most one element. Without loss of generality, let's assume they diverge in the i-th element. The L_2-sensitivity is then computed as follows:

$$\begin{aligned}\Delta^2 &= \max_{\mathcal{D},\mathcal{D}'} \Big\| \sum_{i=1}^m \hat{g}_i - \sum_{i=1}^m \hat{g}'_i \Big\|_2 \\ &= \max_{\hat{g}_i,\hat{g}'_i} \|\hat{g}_i - \hat{g}'_i\|_2 \\ &\leq \|\hat{g}_i\|_2 + \|\hat{g}'_i\|_2 = 2\sqrt{k}\end{aligned}$$

C.2.4 Privacy barrier **B4**

The sensitivity analysis for methods in this category adheres to the DP-SGD framework. While special considerations may be required to ensure the implementation correctly adheres to this framework, these considerations typically do not alter the sensitivity analysis itself.

C.3 Additional Background on Privacy Cost Accumulation

Theorem 4.2.2 (presented in Section 4.2) provides a straightforward method for calculating the aggregated privacy cost when composing multiple (potentially heterogeneous) DP mechanisms. In this section, we present more details regarding determining the accumulated privacy cost over multiple executions of *sampled Gaussian mechanisms* (Definition C.3.1).

Definition C.3.1 (Sampled Gaussian Mechanism (SGM) [1, 143]). Let f be an arbitrary function mapping subsets of $\mathcal{D}$ to $\mathbb{R}^d$. The sampled Gaussian mechanism (SGM) parametrized with the sampling rate $0 < q \leq 1$ and the noise multiplier $\sigma > 0$ is defined as

$$\mathrm{SG}_{q,\sigma} \triangleq f\left(\{\boldsymbol{x} : \boldsymbol{x} \in \mathcal{D} \text{ is sampled with probability } q\}\right) + \mathcal{N}(0, \sigma^2 \boldsymbol{I}_d)$$

where each element of $\mathcal{D}$ is sampled independently at random with probability q without replacement.

The sampled Gaussian mechanism consists of adding i.i.d Gaussian noise with zero mean and variance σ^2 to each coordinate of the true output of f, i.e., $\mathrm{SG}_{q,\sigma}$ injects random vectors from a multivariate isotropic Gaussian distribution $\mathcal{N}(0, \sigma^2 \boldsymbol{I}_d)$ and into the true output, where $\boldsymbol{I}_d$ is written as $\boldsymbol{I}$ if unambiguous in the given context.

Theorem C.3.1. [143] Let $\mathrm{SG}_{q,\sigma}$ be the sampled Gaussian mechanism for some function f with $\Delta_f^2 \leq 1$ for any adjacent $\mathcal{D}, \mathcal{D}'$ under the add-or-remove-one notion. Then $\mathrm{SG}_{q,\sigma}$ satisfies (α, ρ)-RDP if

$$\begin{aligned} & \rho \leq D_\alpha\Big(\mathcal{N}(0,\sigma^2) \,\|\, (1-q)\mathcal{N}(0,\sigma^2) + q\mathcal{N}(1,\sigma^2)\Big) \\ \text{and} \quad & \rho \leq D_\alpha\Big((1-q)\mathcal{N}(0,\sigma^2) + q\mathcal{N}(1,\sigma^2) \,\|\, \mathcal{N}(0,\sigma^2)\Big) \end{aligned}$$

Theorem C.3.1 reduce the problem of proving the RDP bound for $\mathrm{SG}_{q,\sigma}$ to a simple special case of a mixture of one-dimensional Gaussians.

Theorem C.3.2. [143] Let $\mathrm{SG}_{q,\sigma}$ be the sampled Gaussian mechanism for some function f and under the assumption $\Delta_f^2 \leq 1$ for any adjacent $\mathcal{D}, \mathcal{D}'$ under the add-or-remove-one notion. Let μ_0 denote the pdf of $\mathcal{N}(0, \sigma^2)$, μ_1 denote the pdf of $\mathcal{N}(1, \sigma^2)$, and let μ be the mixture of two Gaussians $\mu = (1-q)\mu_0 + q\mu_1$. Then $\mathrm{SG}_{q,\sigma}$ satisfies (α, ρ)-RDP if

$$\rho \leq \frac{1}{\alpha - 1} \log\left(\max\{A_\alpha, B_\alpha\}\right)$$

where

$$\begin{aligned} A_\alpha &\triangleq \mathbb{E}_{z \sim \mu_0}[(\mu(z)/\mu_0(z))^\alpha] \\ B_\alpha &\triangleq \mathbb{E}_{z \sim \mu}[(\mu_0(z)/\mu(z))^\alpha] \end{aligned}$$

Theorem C.3.2 states that applying SGM to a function of sensitivity (Equation 4.2.3) at most 1 satisfies (α, ρ)-RDP if $\rho \leq \frac{1}{\alpha-1}\log(\max\{A_\alpha, B_\alpha\})$. Thus, analyzing RDP properties of SGM is equivalent to upper bounding A_α and B_α.

Corollary C.3.1. [143] $A_\alpha \geq B_\alpha$ for any $\alpha \geq 1$.

This allows reformulation of the RDP bound as

$$\rho \leq \frac{1}{\alpha - 1} \log A_\alpha$$

The A_α can be calculated for a range of α values using the numerically stable computation approach presented in Section 3.3 of [143], which is implemented in standard DP packages such as Opacus[1] and Tensorflow-privacy[2]. Then, the smallest A_α (tightest bound) is used to upper bound ρ and later the RDP privacy cost is converted to (ε, δ)-DP via Theorem 4.2.3. Notably, this approach generalizes previous results such as moment accountant [1] (See Table 1 in [143] for a summary).

[1] https://opacus.ai/

[2] https://github.com/tensorflow/privacy

D Data Forensics in Diffusion Models: A Systematic Analysis of Membership Privacy

This appendix provides additional details to the main paper presented in Chapter 5: we provide additional details on the experimental setup in §D.1 and a range of additional evaluation results and discussion in §D.2.

D.1 Experiment Configuration

D.1.1 Setup

We present the additional details of our experimental setup in this section. Table D.1 summarizes the key hyperparameters that we adopted in training the guided diffusion and improved diffusion models. For the stable diffusion model, we use the official released models with the same hyper-parameters from the Huggingface website[1]. For StyleGAN[2] and PGGAN[3], we use the official open-sourced implementation with the default hyperparameters for training. All experiments were conducted on a single NVIDIA A100 GPU.

Hyperparameters	Guided Diffusion	Improved Diffusion	StyleGAN	PGGAN
channels	128	128	512	128
residual block	3	3	3	-
learn sigma	True	True	-	-
noise scheduler	linear	linear	-	-
batch size	256	256	64	64
learning rate	1e-3	1e-3	1e-3	1e-3
diffusion steps	4000	2000;4000;6000	-	-
dropout	0.3	0.3	0.3	0.3

Table D.1: Summary of the Training Hyperparameters.

D.1.2 Dataset

CelebA [128]. The CelebA is a large-scale face attributes dataset containing 200k RGB images, which are aligned using facial landmarks. To ensure comparability with previous results, we adopt the standard pre-processing procedure when training diffusion models and evaluating attack performance. This involves randomly selecting a maximum of 40k images (corresponding to the more challenging random-split setting in [34]), center-cropping them, and resizing them to a resolution of 64×64 for training the models and evaluating the attacks.

[1] https://huggingface.co/CompVis

[2] https://github.com/NVlabs/stylegan

[3] https://github.com/tkarras/progressive_growing_of_gans

CIFAR-10 [104]. The CIFAR-10 is a dataset of 60k RGB images with shape 32×32×3. Each image is labelled with one of 10 classes, representing the object depicted in the image.

Laion2B-improved-aesthetics[4]. The Laion2B-improved-aesthetics is a curated subset of Laion2B that focuses on images with high-resolution quality and improved aesthetics. It consists of color images with resolutions of 512×512 or higher. Each image has an estimated aesthetics score of >5.0 and an estimated watermark probability of <0.5. Furthermore, the text caption for each image in Laion2B-improved-aesthetics is in English. The dataset includes more than 2.3 million image-caption pairs.

COCO-2017[5]. COCO is a large-scale object detection, segmentation, and captioning dataset. It contains more than 200k images and 80 object categories. Each image is associated with one annotated English text caption. We randomly sample the images from datasets and resize it from the original resolution to 512×512 using the open-source scripts[6].

D.1.3 External Resources

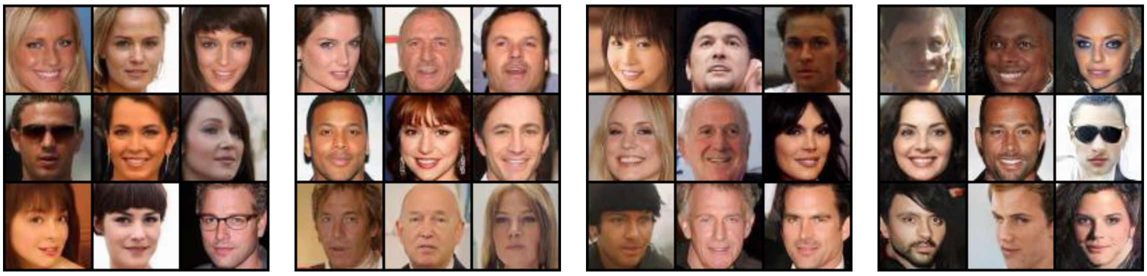
In this section, we list the third-party libraries and tools used to conduct our experiments.

- Hugging Face Stable Diffusion API: https://replicate.com/stability-ai/stable-diffusion.
- Stable-diffusion-V1.4: https://huggingface.co/CompVis/stable-diffusion-v1-4
- stable-diffusion-v1.5: https://huggingface.co/runwayml/stable-diffusion-v1-5

D.2 Additional Results

D.2.1 Generation Quality

We display the generated samples from different generative models in Figure D.1. As can be observed, all models demonstrate a reasonable level of generation quality (practical utility), and none of the models exhibit significant visual differences in their generation quality. This consistency controls the factor of generation quality in their vulnerability to MIAs. The associated quantitative measurements (e.g., FID) are presented in Table 5.8.



(a) Guided Diffusion (b) Improved Diffusion (c) StyleGAN (d) PGGAN

Figure D.1: The synthetic images sampled from Guided Diffusion, Improved Diffusion, StyleGAN and PGGAN trained on CelebA with 5k samples, respectively.

[4] https://huggingface.co/datasets/laion/laion2B-en-aesthetic
[5] https://cocodataset.org/
[6] https://github.com/rom1504/img2dataset

D.2.2 White-box Setting

We present the investigation of various statistic functions f on the loss trajectory on CIFAR-10 dataset in Table D.2. The results confirm the consistency with findings from experiments on the CelebA dataset.

Size	Truncation	Min	Max	Median	Sum
5000	without	0.51	0.54	1.00	0.93
	with	0.50	1.00	0.97	1.00
10000	without	0.49	0.50	0.93	0.74
	with	0.49	0.99	0.74	0.98
15000	without	0.50	0.50	0.91	0.73
	with	0.50	0.99	70	0.98
20000	without	0.49	0.50	0.85	0.67
	with	0.49	0.99	0.64	0.95
30000	without	0.51	0.51	0.73	0.58
	with	0.51	0.92	0.58	0.84
40000	without	0.51	0.51	0.63	0.54
	with	0.51	0.76	0.54	0.70

Table D.2: The **white-box** attack AUCROC when applying different statistic function f (*Min*, *Max*, *Median*, and *Sum*) to the *loss trajectory* $\{\mathcal{L}_t\}_{t=0}^{T}$ with and without truncations. The experiments were conducted on the CIFAR-10 dataset with various training set sizes, as indicated in the first column. For the cases where the truncation technique is applied, we set the truncation step to be the default value with $T_{trun} = 0.75T$.

We present the quantitative results in Table D.3, which is supplementary to Figure 5.4(a) in the main paper.

	CelebA				CIFAR-10			
Truncation	5k	10k	15k	20k	5k	10k	15k	20k
w/o	1.00	0.94	0.80	0.77	1.00	0.93	0.91	0.85
w	1.00	1.00	0.99	0.98	1.00	0.99	0.99	0.99

Table D.3: The **white-box** attack AUCROC across different training set sizes. Without ("w/o") truncation, the statistic function is selected to be "*Median*", while it is set to be "*Max*" with ("w") truncation. The truncation step is set to be $T_{trun} = 0.75T$.

Additionally, we examine in detail the potential factors that may impact the vulnerability of target diffusion models to MIA, such as truncating loss trajectory, statistical functions, training set size, etc. Our extended experiments on the CelebA dataset cover various training configurations, and the results are displayed in Table D.4.

We visualize the results of our white-box attack across various settings of the truncation steps on the CelebA dataset across various training configurations. We present the results in Figure D.2(a). This is supplementary to the results in Table 5.3 in the main paper.

D.2.3 Gray-box Setting

We provide additional quantitative results for our gray-box attack investigating the selection of truncation steps across various training set sizes on CelebA. The results are presented in

(a) CelebA (5k)

f	T	$0.975T$	$0.875T$	$0.75T$	$0.625T$	$0.5T$
Median	1.00	1.00	0.99	0.94	0.77	0.56
Sum	0.99	0.71	1.00	1.00	1.00	0.97
Min	0.50	0.56	0.50	0.50	0.50	0.50
Max	0.54	1.00	1.00	1.00	1.00	1.00

(b) CelebA (10k)

f	T	$0.975T$	$0.875T$	$0.75T$	$0.625T$	$0.5T$
Median	0.94	0.89	0.83	0.67	0.54	0.50
Sum	0.86	1.00	0.98	0.97	0.92	0.74
Min	0.50	0.50	0.49	0.49	0.49	0.50
Max	0.50	0.97	0.99	1.00	0.99	0.93

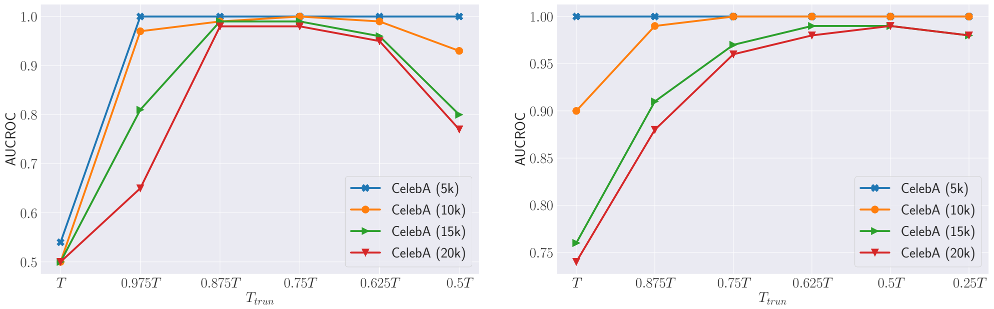
(c) CelebA (15k)

f	T	$0.975T$	$0.875T$	$0.75T$	$0.625T$	$0.5T$
Median	0.80	0.77	0.67	0.56	0.51	0.50
Sum	0.74	0.96	0.98	0.95	0.81	0.62
Min	0.50	0.50	0.50	0.50	0.50	0.50
Max	0.50	0.81	0.99	0.99	0.96	0.80

(d) CelebA (20k)

f	T	$0.975T$	$0.875T$	$0.75T$	$0.625T$	$0.5T$
Median	0.77	0.65	0.65	0.55	0.51	0.50
Sum	0.71	0.84	0.97	0.93	0.79	0.61
Min	0.50	0.49	0.50	0.50	0.50	0.50
Max	0.50	0.65	0.98	0.98	0.95	0.77

Table D.4: The **white-box** attack AUCROC with different statistic function f and truncation steps T_{trun} (shown in each column) on CelebA across various training set sizes (shown in the title of each sub-table). The first column T corresponds to "no truncation".



(a) White-box

(b) Gray-box

Figure D.2: The D.2(a) **white-box** and D.2(b) **gray-box** attack AUCROC for different truncation steps T_{trun} across different dataset sizes on CelebA. The statistic function is selected to be "*Max*" and "*Median*" for the white-box and gray-box attacks, respectively.

Table D.5, with the qualitative illustration being presented in Figure D.2(b).

(a) CelebA (5k)

f	T	$0.875T$	$0.725T$	$0.625T$	$0.5T$	$0.25T$
Median	1.00	1.00	1.00	1.00	1.00	1.00
Sum	0.69	0.80	0.94	1.00	1.00	1.00
Min	0.59	0.55	0.55	0.58	0.55	0.55
Max	0.50	0.50	0.54	0.66	1.00	1.00

(b) CelebA (10k)

f	T	$0.875T$	$0.725T$	$0.625T$	$0.5T$	$0.25T$
Median	0.90	0.99	1.00	1.00	1.00	1.00
Sum	0.59	0.67	0.79	0.94	1.00	0.98
Min	0.50	0.50	0.50	0.51	0.49	0.50
Max	0.49	0.50	0.50	0.55	0.86	0.97

(c) CelebA (15k)

f	T	$0.875T$	$0.725T$	$0.625T$	$0.5T$	$0.25T$
Median	0.76	0.91	0.97	0.99	0.99	0.99
Sum	0.55	0.60	0.70	0.85	0.96	0.99
Min	0.50	0.50	0.50	0.50	0.50	0.50
Max	0.50	0.50	0.50	0.52	0.76	1.00

(d) CelebA (20k)

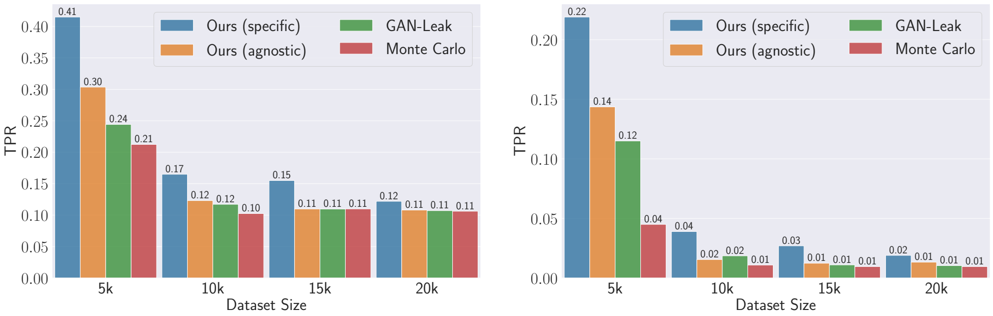
f	T	$0.875T$	$0.75T$	$0.625T$	$0.5T$	$0.25T$
Median	0.74	0.88	0.96	0.98	0.99	0.98
Sum	0.55	0.60	0.69	0.83	0.95	0.98
Min	0.50	0.50	0.50	0.50	0.50	0.50
Max	0.50	0.50	0.50	0.51	0.74	0.99

Table D.5: The **gray-box** attack AUCROC with different statistic function f and truncation steps T_{trun} (shown in each column) on CelebA across various training set sizes (shown in the title of each sub-table). The first column T corresponds to "no truncation".

D.2.4 Black-box Setting

We present the TPR of different black-box attacks at certain levels of low FPR in Figure D.3 and Figure D.4, supplementing the results in Figure 5.6 in the main paper. As can be seen from the plots, the comparison results over TPR are largely consistent with those obtained using AUCROC, though our model-specific attack shows a greater advantage over others when compared at TPR@(0.1 or 0.01)FPR. Consistently across all configurations, the TPR achieved by our attack is higher than the FPR, indicating a successful attack due to its ability to more accurately identify members than incorrectly predict non-members as members. Importantly, an MIA can be regarded as successful if it can reliably identify even a few members.

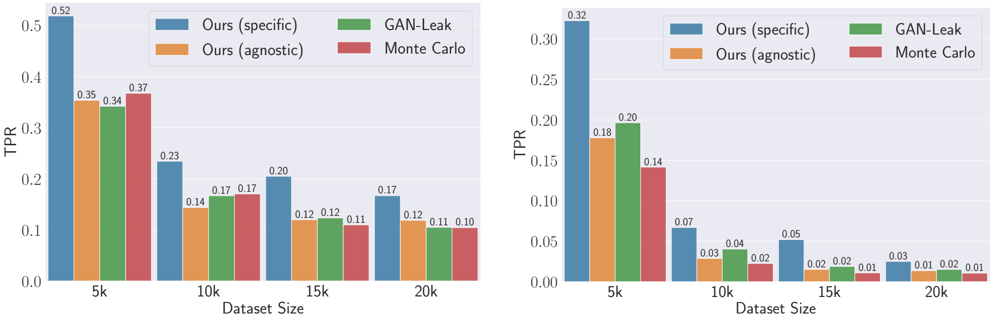
The TPR of different model-agnostic black-box attacks at certain levels of low FPR on various types of generative models is presented in Figure D.5, which supplements the results in Table 5.8 in the main paper. The comparison results are consistent regardless of the adopted metrics (AUCROC or TPR) for evaluating attack performance, all showing that diffusion models generally have a higher vulnerability to MIA than GANs do, even when all of them are trained in the same controlled environment and exhibit similar generation quality (see the quantitative results in Table 5.8).



(a) @0.1 FPR

(b) @0.01 FPR

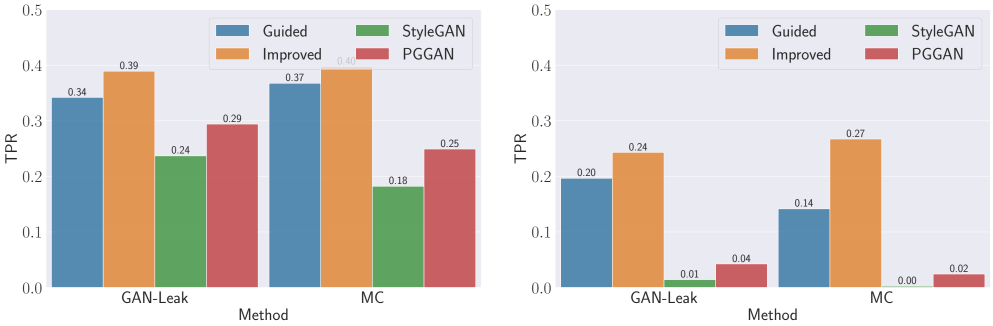
Figure D.3: The black-box attack TPR at low FPR on CIFAR-10.



(a) @0.1 FPR

(b) @0.01 FPR

Figure D.4: The black-box attack TPR at low FPR on CelebA.



(a) @0.1 FPR

(b) @0.01 FPR

Figure D.5: The model-agnostic black-box TPR at low FPR for various generative models on CelebA(5k).

RelaxLoss: Defending Membership Inference Attacks without Losing Utility

E

This appendix provides additional support to the main ideas presented in the submission: §E.1 provides additional theoretical analysis giving rise to insights on the foundations of our method. Moreover, as we have conducted a rigorous and broad experimental analysis that goes beyond the key insights presented in the main paper, we provide additional details on the experimental setup in §E.2 and a range of additional evaluation results and discussion in §E.3.

E.1 Theoretical Analysis

E.1.1 How Does Gradient Ascent Step Increase Loss Variance?

In this section, we show how the gradient ascent step in RelaxLoss increase the loss variance. We write ℓ for the loss instead of $\ell(\boldsymbol{\theta}, \boldsymbol{z}^i)$ for brevity if the dependence is not relevant for our argumentation.

Theorem E.1.1. If $\text{Cov}(\ell, \Delta\ell) > 0$, then the variance of loss distribution $\text{Var}(\ell)$ is increased after a gradient ascent step.

Proof. The variance of loss distribution before and after applying gradient ascent step amounts to $\text{Var}(\ell)$ and $\text{Var}(\ell + \Delta\ell)$, respectively. Following from the fact that

$$\text{Var}(\ell + \Delta\ell) = \text{Var}(\ell) + \text{Var}(\Delta\ell) + 2\text{Cov}(\ell, \Delta\ell)$$

and the non-negativity of $\text{Var}(\Delta\ell)$ as well as $\text{Cov}(\ell, \Delta\ell)$, we conclude $\text{Var}(\ell + \Delta\ell) > \text{Var}(\ell)$, i.e., the loss variance will increase.

We focus on the loss increase after the gradient ascent step (i.e., assuming that $\Delta\ell \geq 0$, which holds for most cases, despite the stochasticity) and interpret $\Delta\ell$ as the rate of loss change. The condition $\text{Cov}(\ell, \Delta\ell) > 0$ can be understood as: the larger the loss value is, the faster it changes, and vice versa. This is a reasonable assumption for most training algorithms for achieving convergence. We reason the exact condition and the related assumptions below.

Condition E.1.1. The gradient magnitude (squared ℓ_2 norm) is positively correlated to the loss value, i.e., $\text{Cov}(\|\nabla\ell\|_2^2, \ell) > 0$. Intuitively, it means the gradient norm tends to decrease as the loss decreases.

We use the cross-entropy loss as an example:

$$\ell_{\text{CE}}(\boldsymbol{\theta}, \boldsymbol{z}_i) = -\sum_{c=1}^{C} y_i^c \log p_i^c \tag{E.1}$$

The gradient is given by:

$$\nabla\ell_{\text{CE}}(\boldsymbol{\theta}, \boldsymbol{z}_i) = J_{\boldsymbol{\theta}i}(\boldsymbol{p}_i - \boldsymbol{y}_i) \tag{E.2}$$

where $J_{\boldsymbol{\theta}i}$ represents the jacobian of the logits (before the final softmax layer) w.r.t. the model parameter $\boldsymbol{\theta}$. Once the loss on sample $\boldsymbol{z}_i$ becomes smaller, we have $\|\boldsymbol{p}_i - \boldsymbol{y}_i\|_2 \rightarrow 0$, i.e., the prediction get closer to the ground-truth label. By the submultiplicativity of matrix norm and the continuity of the squared function, we then have $\|\nabla \ell_{\text{CE}}(\boldsymbol{\theta}, \boldsymbol{z}_i)\|_2^2 \rightarrow 0$, i.e., the gradient norm decreases as the loss value gets smaller. Hence, $\ell_{\text{CE}}(\boldsymbol{\theta}, \boldsymbol{z}_i)$ has the desired property required by Condition E.1.1.

Condition E.1.2. The change in loss after the gradient ascent step $\Delta\ell$ is linear in the squared gradient norm $\|\nabla\ell\|_2^2$, i.e., $\Delta\ell = c_1\|\nabla\ell\|_2^2 + c_2$ with c_1, c_2 the constants quantifying the linear relationship.

Corollary E.1.1. Given the assumption that the gradient of each sample within a batch *(1)* has the same norm and *(2)* has non-negative inner product (i.e., well-aligned) with each other and the gradient alignments remain the same over different batches, the gradient ascent step: $\boldsymbol{\theta}^{(t+1)} = \boldsymbol{\theta}^{(t)} + \tau\nabla\mathcal{L}(\boldsymbol{\theta}^{(t)})$ satisfy the linearity (in Condition E.1.2) with $c_1 > 0$, where the superscript t corresponds to the t-th iteration, and $\nabla\mathcal{L}$ denotes the batch gradient with batchsize $= B$.

Proof. This follows from the nature of the first-order gradient-based optimization method. Applying a first-order Taylor-expansion of the sample loss at $\boldsymbol{\theta}^{(t)}$, we obtain:

$$\begin{aligned}
\ell(\boldsymbol{\theta}^{(t+1)}, \boldsymbol{z}_i) &= \ell(\boldsymbol{\theta}^{(t)}, \boldsymbol{z}_i) + \tau\langle\nabla\ell(\boldsymbol{\theta}^{(t)}), \nabla\mathcal{L}(\boldsymbol{\theta}^{(t)})\rangle + \mathcal{O}(\tau) \\
\Delta\ell &= \ell(\boldsymbol{\theta}^{(t+1)}, \boldsymbol{z}_i) - \ell(\boldsymbol{\theta}^{(t)}, \boldsymbol{z}_i) \qquad \text{(E.3)} \\
&= \frac{\tau}{B}\|\nabla\ell(\boldsymbol{\theta}^{(t)}, \boldsymbol{z}_i)\|_2^2 + \frac{\tau}{B}\sum_{j\neq i}\langle\nabla\ell(\boldsymbol{\theta}^{(t)}, \boldsymbol{z}_i), \nabla\ell(\boldsymbol{\theta}^{(t)}, \boldsymbol{z}_j)\rangle + \mathcal{O}(\tau) \\
&= \frac{\tau}{B}\|\nabla\ell(\boldsymbol{\theta}^{(t)}, \boldsymbol{z}_i)\|_2^2 + \frac{\tau}{B}\sum_{j\neq i}\|\nabla\ell(\boldsymbol{\theta}^{(t)}, \boldsymbol{z}_i)\|_2 \cdot \|\nabla\ell(\boldsymbol{\theta}^{(t)}, \boldsymbol{z}_j)\|_2 \cdot \cos(\alpha_{ij}) + \mathcal{O}(\tau) \\
&= \frac{\tau}{B}\|\nabla\ell(\boldsymbol{\theta}^{(t)}, \boldsymbol{z}_i)\|_2^2(1 + \sum_{j\neq i}\cos(\alpha_{ij})) + \mathcal{O}(\tau) \qquad \text{(E.4)}
\end{aligned}$$

where $\cos(\alpha_{ij})$ is the cosine of the angle between gradients of sample i and j, and $\mathcal{O}(\tau)$ summarizes the higher-order terms and is regarded as a constant (i.e., c_2). Given the assumption that each sample within a batch exhibits well-aligned gradients with the same norm, i.e., $\|\nabla\ell(\boldsymbol{\theta}^{(t)}, \boldsymbol{z}_i)\|_2 = \|\nabla\ell(\boldsymbol{\theta}^{(t)}, \boldsymbol{z}_j)\|_2$ and $\cos(\alpha_{ij}) \geq 0$ for all $j \neq i$, we have Equation E.4 and $\frac{\tau}{B}\big(1 + \sum_{j\neq i}\cos(\alpha_{ij})\big) > 0$. Additionally, given that the gradient alignments remain the same over different batches, i.e., $\big(1 + \sum_{j\neq i}\cos(\alpha_{ij})\big)$ is constant for all i, j, we have $c_1 = \frac{\tau}{B}\big(1 + \sum_{j\neq i}\cos(\alpha_{ij})\big) > 0$.

Lemma E.1.1. Given the Condition E.1.1 and E.1.2, we have the desired property $\text{Cov}(\ell, \Delta\ell)$ by linearity.

Proof.

$$\begin{aligned}
\text{Cov}(\ell, \Delta\ell) &= \text{Cov}(\ell, c_1\|\nabla\ell\|_2^2 + c_2) \qquad \text{(E.5)} \\
&= c_1\text{Cov}(\ell, \|\nabla\ell\|_2^2) > 0 \qquad \text{(E.6)}
\end{aligned}$$

where Equation E.5 and E.6 are yielded by using Condition E.1.2 and E.1.1, respectively.

E.1.2 How Does RelaxLoss Affect MIA?

In this section, we show how RelaxLoss affects the optimal MIA $\mathcal{A}_{opt}$ (measured by its AUC value). We exploit the following results for relating the attack AUC to a statistical distance between the loss distributions.

We first regard the MIA as a binary hypothesis testing problem with the null H_0 and alternate hypothesis H_1 defined as follows:

$$\begin{aligned} H_0: &\quad \boldsymbol{z}_i \text{ is a non-member sample, i.e., } \boldsymbol{z}_i \in \overline{\mathcal{D}}_{\text{train}} \\ H_1: &\quad \boldsymbol{z}_i \text{ is a member sample, i.e., } \boldsymbol{z}_i \in \mathcal{D}_{\text{train}} \end{aligned}$$

The attacker need to make a decision on whether the query sample came from $\mathcal{D}_{\text{train}}$ based on a rejection region S_{reject}. As discussed in Section 6.4.1, under a posterior assumption on the model parameter, S_{reject} for the optimal attack $\mathcal{A}_{opt}$ [180] fully depends on the sample loss, i.e., $\mathcal{A}_{opt}$ rejects the null hypothesis if $\ell(\boldsymbol{\theta}, \boldsymbol{z}_i) \in S_{\text{reject}}$. The type I error (i.e., the H_0 is true but rejected) is defined as $\mathcal{P}(\ell(\boldsymbol{\theta}, \overline{\mathcal{D}}_{\text{train}}) \in S_{\text{reject}})$, and the type II error (i.e., the H_0 is false but retained) is defined as $\mathcal{P}(\ell(\boldsymbol{\theta}, \mathcal{D}_{\text{train}}) \in \overline{S}_{\text{reject}})$.

Theorem E.1.2. [91, 123, 124] Let TP and FP denote the true positive rate (1− type II error) and false positive rate (type I error) of $\mathcal{A}_{opt}$ respectively, their relation to the total variation distance between the loss distributions $D_{\text{TV}}(P, Q)$ is quantified as follows (See [91] Appendix A for the derivation):

$$\text{TP} \leq \text{FP} + \min\{D_{\text{TV}}(P, Q), 1 - \text{FP}\} \tag{E.7}$$

where P and Q denote the distribution of the training loss $\ell(\boldsymbol{\theta}, \mathcal{D}_{\text{train}})$ and the testing loss $\ell(\boldsymbol{\theta}, \overline{\mathcal{D}}_{\text{train}})$, respectively.

The ROC curve is obtained by plotting the largest achievable true positive (TP) rate on the vertical axis against the false positive (FP) rate on the horizontal axis, while the AUC value corresponds to a summation over all pairs of TP and FP.

Corollary E.1.2. The AUC value can be upper bounded as follows ([124] Corollary 1):

$$\text{AUC} \leq -\frac{1}{2} D_{\text{TV}}(P, Q)^2 + D_{\text{TV}}(P, Q) + 1/2 \tag{E.8}$$

For ease of analysis, we then upper bound the total variation distance via the Hellinger distance, which is symmetric and has a closed-form expression for common distributions such as Gaussian.

Theorem E.1.3. [202] The total variance distance can be upper bounded by the Hellinger distance:

$$D_{\text{TV}}(P, Q) \leq \sqrt{2} D_{\text{H}}(P, Q) \tag{E.9}$$

where the Hellinger distance satisfies $0 \leq D_{\text{H}}(P, Q) \leq 1$

For Gaussian distributions, the Hellinger distance has a closed form:

$$D_{\text{H}}^2(P, Q) = 1 - \sqrt{\frac{2\sigma_1\sigma_2}{\sigma_1^2 + \sigma_2^2}} \exp\left(-\frac{1}{4}\frac{(\mu_1 - \mu_2)^2}{\sigma_1^2 + \sigma_2^2}\right) \tag{E.10}$$

where μ_1, μ_2 denote the mean and σ_1, σ_2 denote the variance of P and Q.

Let $c = \sigma_2/\sigma_1$ denote the ratio of the training and testing loss variance, we see that $D_{\mathrm{H}}(P, Q)$ is fully characterized by: *(i)* the value of the training loss variance σ_1^2, *(ii)* the squared distance between the mean $(\mu_1 - \mu_2)^2$, and *(iii)* the variance ratio c:

$$D_{\mathrm{H}}^2(P, Q) = 1 - \underbrace{\sqrt{\frac{2c}{1+c^2}}}_{(*)} \underbrace{\exp\left(-\frac{1}{4}\frac{(\mu_1-\mu_2)^2}{(1+c^2)\sigma_1^2}\right)}_{(**)} \tag{E.11}$$

Our approach decreases the $(\mu_1 - \mu_2)^2$ and increases σ_1^2 (Section 6.5) as well, both of which lead to a decrease of the Hellinger distance, and thus decreases the upper bound of the attacker AUC as desired.

It remains to consider how our approach will change c and how the change in c will affect the Hellinger distance. First, we observe that $c \geq 1$, i.e., the testing distribution has larger variance than the training distribution (See Appendix E.3.9). Moreover, c gets closer to 1 when applying our approach (See Figure 6.1 in the main paper). As a result, $(*)$ will increase (Corollary E.1.3) and $(**)$ will decrease (Corollary E.1.4).

Corollary E.1.3. If $c' \geq c \geq 1$, then $\sqrt{\frac{2c}{1+c^2}} \geq \sqrt{\frac{2c'}{1+c'^2}}$

Proof. Let $f(c) = \sqrt{\frac{2c}{1+c^2}}$. We have $f'(c) = \frac{1-c^2}{\sqrt{2c}(c^2+1)^{3/2}}$. It is obvious that f has critical point at $c = 1$, i.e., $f'(1) = 0$ and $f'(c) \leq 0$.

Corollary E.1.4. For fixed value of μ_1, μ_2 and σ_1, if $c' \geq c \geq 1$, then

$$\exp\left(-\frac{1}{4}\frac{(\mu_1-\mu_2)^2}{(1+c^2)\sigma_1^2}\right) \leq \exp\left(-\frac{1}{4}\frac{(\mu_1-\mu_2)^2}{(1+c'^2)\sigma_1^2}\right)$$

Proof. It is obvious as c occurs in the denominator inside the exponential term.

Additionally, we notice that the change in the $(*)$ dominates in most cases: the $(**)$ term commonly has value within $[0.9, 1.0]$, while the $(*)$ term changes from 10^{-3} to 1 when our approach is applied. Therefore, under a Gaussian assumption of the loss distributions, our method can decrease the Hellinger distance between the distributions, thereby reducing an upper bound of the attack AUC.

E.2 Experiment Setup

E.2.1 Datasets

CIFAR-10 [104] is a dataset of 60k color images with shape $32 \times 32 \times 3$. Each image corresponds to a label of 10 classes which categorizes the object inside the image. Following the standard preprocessing procedure [1], we normalize the image pixel value to have zero mean and unit standard deviation.

CIFAR-100 [104] consists of 60k color images of size $32 \times 32 \times 3$ in 100 classes. Same as for the CIFAR-10 dataset, we perform mean-subtraction and standardization.

[1] https://pytorch.org/hub/pytorch_vision_resnet/

CH-MNIST [96] contains 5000 greyscale images of 8 different types of tissues from patients with colorectal cancer. We obtain the preprocessed dataset from Kaggle [2] and use images of size 28×28 for our experiments. All images are normalized to $[-1, 1]$.
Texas100 contains medical records of 67,330 patients published by the Texas Department of State Health Services [3]. Each patient's record contains 6,169 binary features (such as diagnosis, generic information, and procedures the patient underwent) and is labeled by its most suitable procedure (among the 100 most frequent ones). We use the preprocessed data provided by [188, 195][4].
Purchase100 is a dataset of customers' shopping records released by the Kaggle Acquire Valued Shoppers Challenge [5]. We use the preprocessed version provided by [188, 195][4], which contains 197,324 data samples. Each sample, representing one user's purchase history, consists of 600 binary features. Each feature denotes the presence of one product in the corresponding user's purchase history. The data is clustered into 100 classes of different purchase styles. The classification task is to predict the purchase style given the 600 binary features.

We summarize all datasets in details in Table E.1.

Dataset	Data type	Feature type	Feature dimension	N_{total}	$N_{\text{train}}/N_{\text{test}}$ (target model)	$N_{\text{train}}/N_{\text{test}}$ (shadow model)
CIFAR-10	color image	numerical	3072	60000	12000	12000
CIFAR-100	color image	numerical	3072	60000	12000	12000
CH-MNIST	grayscale image	numerical	784	5000	1000	1000
Texas100	medical record	categorical	6169	67330	13466	13466
Purchase100	purchase record	categorical	600	197324	39465	39464

Table E.1: Summary of datasets. N_{total} denotes the total dataset size. N_{train} and N_{test} are the size of the training and testing set, respectively.

E.2.2 Model Architectures

For CIFAR-10 and CIFAR-100 datasets, we use a 20-layer ResNet and an 11-layer VGG architecture [6]. For CH-MNIST, we adopt a 20-layer ResNet. And for the non-image datasets, we adopt the same architecture as used in [148][7]: a 4-layer fully-connected neural network with layer size [1024, 512, 256, 100] for Purchase100, and a 5-layer fully-connected neural network with layer size [2048, 1024, 512, 256, 100] for Texas100.

E.2.3 Implementation details

We apply SGD optimizer with momentum=0.9 and weight-decay=1e-4 by default. We set the initial learning rate $\tau = 0.1$ and drop the learning rate by a factor of 10 at each decay epoch [6]. We list below the decay epochs in square brackets and the total number of training epochs are marked in parentheses: CIFAR-10 and CIFAR-100 [150,225] (300); CH-MNIST [40,60] (80); Texas100 and Purchase100 [50,100] (120). Additionally, we adopt the following techniques for improved performance across heterogeneous data modalities: we restrict the scope of posterior flattening to *incorrect predictions* for natural image datasets (CIFAR-10 and CIFAR-100); and

[2] https://www.kaggle.com/kmader/colorectal-histology-mnist
[3] https://www.dshs.texas.gov/THCIC/Hospitals/Download.shtm
[4] https://github.com/inspire-group/membership-inference-evaluation
[5] https://www.kaggle.com/c/acquire-valued-shoppers-challenge
[6] https://github.com/bearpaw/pytorch-classification
[7] https://github.com/SPIN-UMass/ML-Privacy-Regulization

we further suppress the target posterior scores of *ground-truth* class p^{gt} to small values ($\leq$0.3) during the posterior flattening step on non-image data (Texas100 and Purchase100). By default, no data augmentation is applied.

E.2.4 Required Resources

All our models and methods are implemented in PyTorch. Our experiments are conducted with Nvidia Tesla V100 and Quadro RTX8000 GPUs. Our method introduces minimal changes and negligible additional cost compared with vanilla training and thus can be flexibly integrated into any deep learning framework without imposing specific constraints on the required hardware resources.

E.2.5 Defense Methods

Early-stopping. The basic idea behind Early-stopping is to truncate training before a model starts to overfit. In our experiments, we save target models' checkpoints at varying numbers of training epochs and subsequently evaluate the attack AUC and test accuracy of each model checkpoint. We set checkpoints at the following epochs: $[25, 50, 75, 100, 125, 150, 175, 200, 225, 250, 275]$ for CIFAR-10 and CIFAR-100 datasets; $[5, 10, 15, 20, 25, 30, 40, 50]$ for CH-MNIST dataset; $[10, 20, 30, 40, 50, 60, 70, 80, 90, 100, 110]$ for Texas100 and Purchase100 datasets.

Dropout. Dropout prevents co-adaptation of feature detectors by randomly masking out a set of neurons in the networks, thereby alleviating model overfitting. In our experiments, we apply dropout to the last fully-connected layer of each target model and evaluate across a wide range of dropout rates (over $[0.1, 0.3, 0.5, 0.7, 0.9]$).

Label-smoothing. Label-smoothing prevents overconfident predictions by incorporating a regularization term into the training objective that penalizes the distance (measured by the KL-divergence) between the model predictions and the uniform distribution. The objective is formularized as follows

$$\mathcal{L} = \alpha \cdot D_{\mathrm{KL}}(\mathcal{U} \parallel p_{\boldsymbol{\theta}}(\boldsymbol{y}|\boldsymbol{x})) + (1-\alpha) \cdot \mathcal{L}_{\mathrm{CE}}(\boldsymbol{\theta}, \boldsymbol{z}) \tag{E.12}$$

where D_{KL} is the KL-divergence; $\mathcal{U}$ denotes the uniform distribution; $p_{\boldsymbol{\theta}}(\boldsymbol{y}|\boldsymbol{x})$ denotes the output prediction. α is a hyper-parameter with range $[0, 1]$ that balances the cross-entropy loss $\mathcal{L}_{\mathrm{CE}}$ and the regularization term. We vary the α across its full range for plotting the privacy-utility curves.

Confidence-penalty. Confidence-penalty regularizes models by penalizing low entropy output distributions. This is achieved via an entropy regularization term in the objective:

$$\mathcal{L} = -\alpha \cdot H(p_{\boldsymbol{\theta}}(\boldsymbol{y}|\boldsymbol{x})) + \mathcal{L}_{\mathrm{CE}}(\boldsymbol{\theta}, \boldsymbol{z}) \tag{E.13}$$

$$\text{where} \quad H(p_{\boldsymbol{\theta}}(\boldsymbol{y}|\boldsymbol{x})) = -\sum_{c=1}^{C} p_{\boldsymbol{\theta}}(y^c|\boldsymbol{x}) \log(p_{\boldsymbol{\theta}}(y^c|\boldsymbol{x})) \tag{E.14}$$

H represents the entropy of the output prediction, and α is a hyper-parameter that controls the importance of the entropy regularization term. Consistent with the original paper [163], we vary the α over $[0.1, 0.3, 0.5, 1.0, 2.0, 4.0, 8.0]$.

Distillation. Knowledge distillation stands for a general process of transferring knowledge from a set of teacher model(s) to a student model. To focus our investigation on the effect of the distillation operation itself, we use self-distillation [255] in our experiments, i.e., we train the student model to match a single teacher model with the same architecture. The objective for training the student model (i.e., the target model) is:

$$\mathcal{L} = \alpha T^2 \cdot D_{\text{KL}}(\widetilde{p}_{\boldsymbol{\theta}_s}(\boldsymbol{y}|\boldsymbol{x}) \parallel \widetilde{p}_{\boldsymbol{\theta}_t}(\boldsymbol{y}|\boldsymbol{x})) + (1-\alpha) \cdot \mathcal{L}_{\text{CE}}(\boldsymbol{\theta}, \boldsymbol{z}) \tag{E.15}$$

$$\text{where} \quad \widetilde{p}_{\boldsymbol{\theta}}(\boldsymbol{y}|\boldsymbol{x})^c = \frac{\exp(f(\boldsymbol{\theta}, \boldsymbol{x})^c / T)}{\sum_{c'} \exp(f(\boldsymbol{\theta}, \boldsymbol{x})^{c'} / T)} \tag{E.16}$$

The KL-divergence term D_{KL} targets at minimizing discrepancy between the softened student $\widetilde{p}_{\boldsymbol{\theta}_s}(\boldsymbol{y}|\boldsymbol{x})$ and teacher prediction $\widetilde{p}_{\boldsymbol{\theta}_t}(\boldsymbol{y}|\boldsymbol{x})$. T denotes the temperature scaling factor that controls the degree of softening. α is a hyper-parameter that balances the KL-divergence and the normal cross-entropy $\mathcal{L}_{\text{CE}}$ term. To determine the hyper-parameter that best describes the privacy-utility trade-off, we conduct preliminary experiments and investigate the effect of α and T independently. By fixing one and changing the other, we observe similar results in terms of the privacy-utility trade-off. Following practical standards, we then fix the α=0.5 and vary the temperature T over $[1, 2, 5, 10, 20, 50, 100]$ for plotting the privacy-utility curves.

DP-SGD. DP-SGD enforces privacy guarantees by modifying the optimization process. It consists of two steps: *(i)* clipping the gradients to have a L_2-norm upper-bounded by C at each training step; *(ii)* injecting random noise to the gradients before performing update steps. We adopt the implementation provided by the Opacus library [8]. We tune the clipping bound C when fixing the noise scale to 0.1, as suggested by the official documents. To plot the privacy-utility curves, we vary the noise scale with a fixed pre-selected clipping bound.

Memguard. Memguard modifies the output predictions of pre-trained target models during test-time, i.e., output predictions are perturbed by adversarial noise to fool a surrogate attack model. Following the official implementation[9], we adopt the same architecture for the surrogate and the real **NN** attack model, and use the complete logits prediction as input to the attack models. Each surrogate attack model is trained on the target model's predictions when inputting the target model's training data (used as member samples) and a separate hold-out set data (used as non-member samples). The privacy-utility trade-off is fully determined by the magnitude of the adversarial noise (measured by L_1-norm). We plot the privacy-utility curves by increasing the noise magnitude from 0 until the **NN** attack has been defended (i.e., attack AUC $\approx$ 0.5).

Adv-Reg. Adv-Reg incorporates an adversarial objective when training the target model: the target model is trained to minimize a weighted sum of the cross-entropy loss and the adversarial loss (obtained from a surrogate attack). Same as in Memguard, each surrogate attack model is trained on the target model's training data and a separate hold-out set data. We follow the official implementation[7] and vary the weight α (=1.0 by default) of the adversarial loss across $[0.8, 1.0, 1.2, 1.4, 1.6, 1.8]$ for plotting the privacy-utility curves.

E.3 Additional Results and Discussion

E.3.1 Limitations and Future Works

Although RelaxLoss is empirically proven effective for improving target models' utility, it is generally hard to explain such improvement, as understanding the generalization ability is still

[8] https://github.com/pytorch/opacus
[9] https://github.com/jjy1994/MemGuard

an open problem. As an attempt, we conduct experiments on toy datasets and attribute the improvement to a flat decision boundary (See Appendix E.3.10). A thorough investigation into how each individual components of our approach affects generalization are left for our future works. In addition, the assumptions of model parameters [180] which are made for the optimal attack demand further validation.

E.3.2 RelaxLoss Vs. Attacks

Supplementary to Table 6.1 in the main paper, Table E.2 summarizes the top-1 *training* and *test* accuracy as well as the *generalization gap* (i.e., difference between the top-1 training and test accuracy) of target models with or without being defended via our method. We observe that our approach, as desired, reduces the generalization gap and is still able to achieve high performance.

	CIFAR-10 (ResNet20)			CIFAR-10 (VGG11)			CIFAR-100 (ResNet20)			CIFAR-100 (VGG11)			CH-MNIST			Texas100			Purchase100		
	train	test	gap	train	test	gap	train	test	gap	train	test	gap	train	test	gap	train	test	gap	train	test	gap
wo defense	100	70.5	29.5	100	73.8	26.2	100	33.2	66.8	100	41.4	58.6	99.0	77.1	21.9	99.9	52.3	47.6	100	89.1	10.9
with defense	87.0	73.8	13.2	99.5	74.4	25.1	52.7	35.1	17.6	99.7	41.4	58.3	90.1	78.4	11.7	67.1	55.3	11.8	99.8	89.1	10.7
Δ	-13.0	3.3	-16.3	-0.5	0.6	-1.1	-47.3	1.9	-49.2	-0.3	0.0	-0.3	-8.9	1.3	-10.2	-32.8	3.0	-35.8	-0.2	0.0	-0.2
selected α	1			0.4			3			0.5			0.2			2.5			0.8		

Table E.2: The top-1 *training* and *test* accuracy as well as the generalization *gap* (in %) of the target models with or without (wo) applying our defense. Δ corresponds to the *absolute* difference after applying our defend method (in %). We also include the selected value of α. This is supplementary to Table 6.1 in the main paper.

		Entropy	M-Entropy	Loss	NN	Grad-x ℓ_1	Grad-x ℓ_2	Grad-w ℓ_1	Grad-w ℓ_2
CIFAR-10 (ResNet20)	wo defense	86.5	87.3	86.9	82.5	87.5	87.5	87.8	87.8
	with defense	50.0	50.0	50.0	49.9	50.0	50.0	49.9	50.0
	Δ	-42.2	-42.7	-42.5	-39.5	-42.9	-42.9	-43.2	-43.1
CIFAR-10 (VGG11)	wo defense	80.1	80.7	80.6	76.1	81.5	81.3	82.7	82.9
	with defense	50.0	50.0	50.0	50.0	50.0	50.0	50.0	50.0
	Δ	-37.6	-38.0	-38.0	-34.3	-38.7	-38.5	-39.5	-39.7
CIFAR-100 (ResNet20)	wo defense	91.8	92.1	92.6	87.0	93.7	93.7	94.6	94.7
	with defense	50.0	50.0	50.0	50.0	50.0	50.0	50.0	50.0
	Δ	-45.5	-45.7	-46.0	-42.5	-46.6	-46.6	-47.1	-47.2
CIFAR-100 (VGG11)	wo defense	97.1	97.5	97.4	98.1	98.5	98.4	98.9	98.9
	with defense	50.0	50.0	50.0	50.6	50.1	50.1	50.6	50.4
	Δ	-48.5	-48.7	-48.7	-48.4	-49.1	-49.1	-48.8	-49.0
CH-MNIST	wo defense	55.5	56.7	56.7	63.6	67.6	67.7	67.1	66.4
	with defense	49.9	52.3	50.9	50.3	52.2	51.7	50.7	49.9
	Δ	-10.1	-7.8	-10.2	-20.9	-22.8	-23.6	-24.4	-24.8
Texas100	wo defense	70.3	79.0	79.0	63.8	78.5	78.5	78.4	78.3
	with defense	50.0	50.0	50.0	50.0	52.0	53.0	51.1	50.0
	Δ	-28.9	-36.7	-36.7	-21.6	-33.8	-32.5	-34.8	-36.1
Purchase100	wo defense	63.9	64.8	64.7	62.6	65.8	65.7	65.8	65.7
	with defense	50.0	50.0	50.0	49.6	52.3	52.2	50.1	50.1
	Δ	-21.8	-22.8	-22.7	-20.8	-20.5	-20.5	-23.9	-23.9

Table E.3: The attacker **accuracy** (in %) evaluated on the target models with and without (wo) applying our defense. Δ corresponds to the *relative* difference after applying our defend method (in %). All the thresholds are selected with undefended shadow models trained with shadow dataset.

In Table E.3, we show the attack **accuracy** values on target models trained with and without (wo) applying our defense method, which is supplementary to Table 6.1 and Figure 6.5 in the main paper. Same as in previous work [195][4], the attack's decision threshold is selected to be the one that yields the best attack accuracy on undefended shadow models. We observe that our approach effectively reduces the attack accuracy to a random-guessing level (around 50%) for most cases. In particular, we find that the selected decision thresholds is highly biased in certain cases s.t. all the queried samples are predicted to be positive (or negative), which leads to exactly 50% accuracy.

E.3.3 Adaptive Attack

In Table E.4, we show the **accuracy** of adaptive (a.) attacks on target models trained with (w/) applying our defense method, which is supplementary to Section 6.6.4 in the main paper. For thresholding-based attacks, the attack's decision threshold is selected to be the one that yields the best attack accuracy on the shadow models (which is trained with exactly the same configuration as our defended target models in Table 6.1). And for the NN-based attack, we use the complete logits prediction from the pre-trained shadow models as features to train the adaptive attack models (modeled as a NN). We observe that our approach consistently reduces the attack accuracy for all cases, though the reduction is less significant compared to non-adaptive attacks shown in Table E.3.

		Entropy	M-Entropy	Loss	NN	Grad-x ℓ_1	Grad-x ℓ_2	Grad-w ℓ_1	Grad-w ℓ_2
CIFAR10 (ResNet20)	w/ defense (a.)	52.5	56.0	56.0	54.2	53.5	53.3	54.0	53.8
	Δ	-39.3	-35.9	-35.6	-34.3	-38.9	-39.1	-38.5	-38.7
CIFAR10 (VGG11)	w/ defense (a.)	64.4	68.2	67.8	66.6	66.2	66.4	67.4	68.0
	Δ	-19.6	-15.5	-15.9	-12.5	-18.8	-18.3	-18.5	-18.0
CIFAR100 (ResNet20)	w/ defense (a.)	52.2	57.8	57.8	53.6	50.1	50.1	50.0	50.1
	Δ	-43.1	-37.2	-37.6	-38.4	-46.5	-46.5	-47.1	-47.1
CIFAR100 (VGG11)	w/ defense (a.)	78.9	84.2	84.0	83.4	78.2	78.2	81.6	82.2
	Δ	-18.7	-13.6	-13.8	-15.0	-20.6	-20.5	-17.5	-16.9
CH-MNIST	w/ defense (a.)	50.7	53.9	53.4	50.9	55.8	55.7	56.6	56.4
	Δ	-8.6	-4.9	-5.8	-20.0	-17.5	-17.7	-15.6	-15.1
Texas100	w/ defense (a.)	51.6	53.8	53.8	52.1	51.9	51.9	51.8	53.4
	Δ	-26.6	-31.9	-31.9	-18.3	-33.9	-33.9	-33.9	-31.8
Purchase100	w/ defense (a.)	54.0	54.8	54.8	54.5	55.4	55.4	55.6	56.0
	Δ	-15.5	-15.4	-15.3	-12.9	-15.8	-15.7	-15.5	-14.8

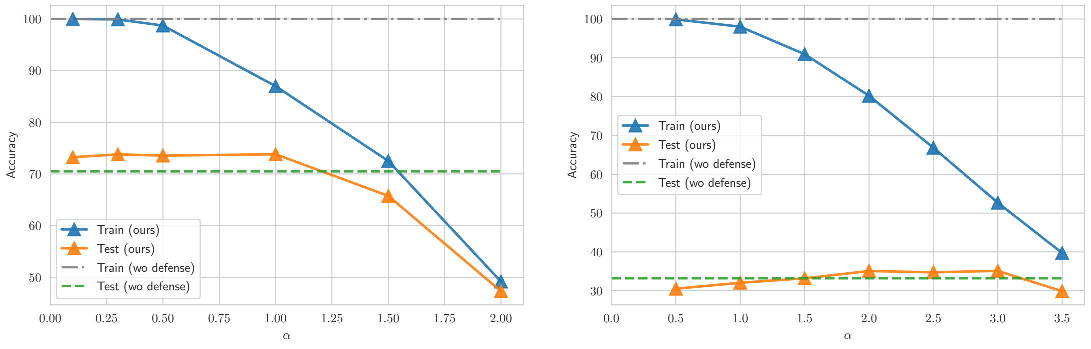
Table E.4: The **accuracy** (in %) of adaptive (a.) attacks evaluated on the target models with (w/) applying our defense. Δ corresponds to the *relative* difference (in %) attack accuracy when applying our defend method compared to vanilla training. The selected target models are the same as in Table 6.1.

E.3.4 Generalization Gap

We show the training and testing accuracy (and the generalization gap) when applying RelaxLoss with varying value of α in Figure E.1. We observe that increasing the value of α will reduce the generalization gap. Moreover, RelaxLoss with a reasonable value of α can even improves the test accuracy of vanilla training.

E.3.5 Compatibility with Data Augmentation

Additionally, we investigate the effectiveness of our approach when data augmentation is applied. Following practical standard, we apply *random cropping* and *random flipping* when



(a) CIFAR-10

(b) CIFAR-100

Figure E.1: Training and testing accuracy (y-axis) with varying value of α (x-axis) on CIFAR-10 (ResNet20) and CIFAR-100 (ResNet20) datasets. We plot the training and testing accuracy of vanilla training (wo defense) in dashed lines for reference.

training the target models on CIFAR-100 dataset. As illustrated in Figure E.2, RelaxLoss is compatible with standard data augmentation techniques: our approach enjoys the performance boost introduced by data augmentation while retaining its effectiveness in defending MIAs.

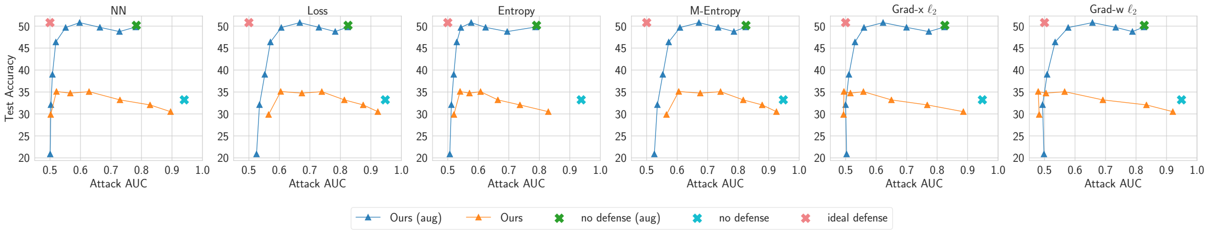


Figure E.2: Effect of data augmentation (denoted as "aug") on CIFAR-100 (ResNet20). When jointly applied with data augmentation, our approach shows consistent effectiveness in improving the MIA resistance and model utility.

E.3.6 Computational Complexity

The additional computation cost of RelaxLoss scales as $\mathcal{O}(BC)$ (B: batch size; C: number of classes), which includes: *(i)* softlabel construction of cost $\mathcal{O}(BC)$; and *(ii)* computation of the cross-entropy loss on the softlabel $\mathcal{O}(BC)$. Note that we reuse the prediction $\mathbf{p}_i$ generated by the previous forward-pass and thus no additional forward (nor backward) pass is required. Compared to the forward and backward pass, which is of magnitude at least $\mathcal{O}(BNL)$ (N: number of neurons per layer; L: number of layers), the additional costs of RelaxLoss are negligible as NL (roughly the total amount of neurons of the whole network) is much larger than C (the number of neurons of the last layer).

E.3.7 Effect on Different Classes of Individuals

To analyze the effect of RelaxLoss on different individuals, we conduct additional experiments on Texas and Purchase datasets which consist of 100 classes with non-uniform class distribution (i.e., the proportion of each class ranges from 0.35% to 4.5% for Texas, and 0.05%-2.6% for Purchase) and evaluate the attack performance on each class separately.

In Table E.5, we show the 10 *highest* **Attack AUC** (in increasing order) among all the classes, which can be regarded as the estimated *worst-case* privacy risk of different classes of individuals. As can be seen from the tables, applying our defense method effectively reduces the Top-10 Attack AUC (i.e., worst-case privacy risk), and the effectiveness is *consistent* on each dataset across different attack methods, with which we conclude that our method, despite the nonuniformity, does defend MIAs for different individuals.

(a) Texas

Atttack methods	with/wo defense	Top-10 Attack AUC
Loss	wo defense	0.985, 0.987, 0.988, 0.989, 0.994, 0.994, 0.995, 0.996, 0.998, 0.999
	with defense	0.717 , 0.719, 0.721, 0.722, 0.724, 0.739, 0.741, 0.743, 0.752, 0.761
Entropy	wo defense	0.846, 0.847, 0.851, 0.851, 0.854, 0.864, 0.865, 0.868, 0.879, 0.880
	with defense	0.617, 0.619, 0.625, 0.625, 0.636, 0.636, 0.648, 0.679, 0.733, 0.747
M-Entropy	wo defense	0.985, 0.987, 0.989, 0.990, 0.992, 0.993, 0.994, 0.996, 0.997, 0.999
	with defense	0.717, 0.718, 0.722, 0.722, 0.724, 0.741, 0.742, 0.742, 0.754, 0.761
Grad-x l2	wo defense	0.837, 0.837, 0.844, 0.848, 0.850, 0.852, 0.859, 0.865, 0.903, 0.943
	with defense	0.644, 0.650, 0.653, 0.655, 0.657, 0.666, 0.679, 0.691, 0.723, 0.744
Grad-w l2	wo defense	0.850, 0.852, 0.853, 0.856, 0.862, 0.862, 0.863, 0.866, 0.867, 0.874
	with defense	0.627, 0.631, 0.632, 0.632, 0.633, 0.634, 0.643, 0.644, 0.661, 0.663

(b) Purchase

Atttack methods	with/wo defense	Top-10 Attack AUC
Loss	wo defense	0.699, 0.709, 0.715, 0.719, 0.727, 0.731, 0.735, 0.738, 0.763, 0.875
	with defense	0.663, 0.666, 0.671, 0.672, 0.684, 0.692, 0.694, 0.701, 0.705, 0.722
Entropy	wo defense	0.692, 0.697, 0.714, 0.714, 0.718, 0.724, 0.725, 0.725, 0.747, 0.868
	with defense	0.651, 0.651, 0.654, 0.657, 0.658, 0.673, 0.678, 0.681, 0.695, 0.711
M-Entropy	wo defense	0.699, 0.711, 0.716, 0.720, 0.727, 0.731, 0.734, 0.739, 0.765, 0.875
	with defense	0.663, 0.666, 0.671, 0.672, 0.684, 0.693, 0.694, 0.701, 0.705, 0.721
Grad-x l2	wo defense	0.708, 0.725, 0.730, 0.733, 0.736, 0.738, 0.742, 0.747, 0.777, 0.897
	with defense	0.653, 0.662, 0.662, 0.667, 0.667, 0.669, 0.672, 0.684, 0.696, 0.697
Grad-w l2	wo defense	0.662, 0.664, 0.665, 0.666, 0.666, 0.668, 0.670, 0.671, 0.681, 0.710
	with defense	0.629, 0.629, 0.632, 0.634, 0.634, 0.638, 0.639, 0.654, 0.655, 0.663

Table E.5: Top-10 Attack AUC among 100 label classes on (a) Texas and (b) Purchase with and without (wo) applying our defense. The AUC values are shown in increasing order.

E.3.8 Privacy-utility Curves

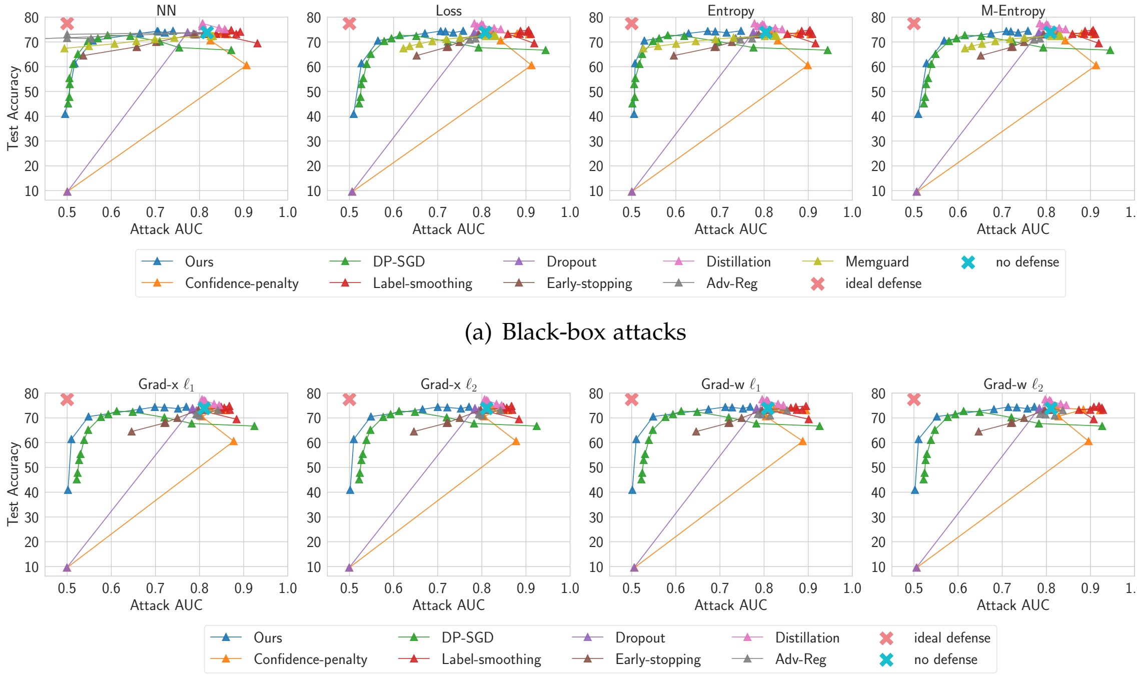
In this section, we include detailed results with regard to various datasets and different target models' architectures: we show results on CIFAR-10 dataset with VGG11 architecture in Figure E.3; CIFAR-100 dataset with VGG11 architecture in Figure E.4; CH-MNIST with ResNet20 architecture in Figure E.5; Texas100 with MLP architecutre in Figure E.6; Purchase100 with MLP architecutre in Figure E.7.

We observe that while baseline approaches are effective for at most one data modality, our approach is the only one that is consistently effective in defending MIAs, across all different datasets and model architectures.

Natural Images. See Figure 6.4(a)-6.4(b) in main paper, and Figure E.3-E.4: for natural image datasets (CIFAR-10 and CIFAR-100), DP-SGD is the best baseline method in terms of defending MIAs and preserving model utility, but is inferior to RelaxLoss as our method consistently achieve better test accuracy (model utility) across the full range of achievable privacy level.

Among all regularization-based defense methods, Early-stopping is the only one that exhibits noticeable effects in reducing attack AUCs. In comparison, our approach can achieve the same level of defense effectiveness with a much better model utility. Moreover, our approach can further decrease the attack AUC to a random-guessing level, which is not achievable by Early-stopping.

Memguard and Adv-Reg, previous state-of-the-art defense mechanisms specifically designed for MIAs, are highly effective in defending NN-based attack but generally lose their effectiveness for other types of attacks. In comparison, our approach shows much better defense effectiveness for all types of attacks while achieving better model utility at the same time.



(a) Black-box attacks

(b) White-box attacks

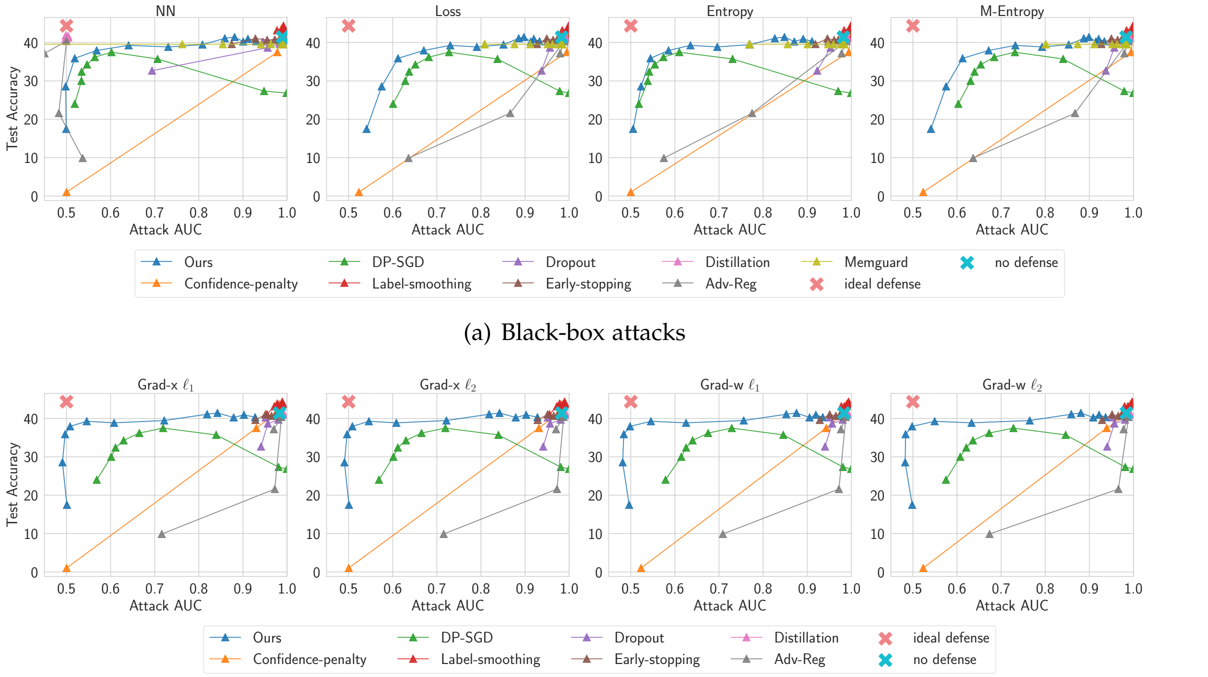
Figure E.3: Comparisons of all defense mechanisms on CIFAR-10 dataset (VGG11). We set the clipping bound $C = 0.5$ and vary the noise scale over 0.01-0.45 for DP-SGD. We vary the noise magnitude across 0-20 for Memguard.

Gray-scale Medical Images See Figure E.5: for gray-scale medical image (CH-MNIST), our approach is comparable with the best baseline methods (i.e., Confidence-penalty), as both approaches are able to reduce the MIAs to a random-guessing level while preserving the model utility.

In comparision, DP-SGD is significantly worse than our approach for this data modality, as it inevitably degrades the model utility.

Memguard and Adv-Reg are still highly effective in defending NN-based attack. Moreover, Memguard is able to defend black-box MIAs to a random-guessing level without degrading the model utility, but is not applicable to white-box attacks. In contrast, our approach is applicable to all attacks, and achieves comparable (or better) effectiveness for black-box attacks.

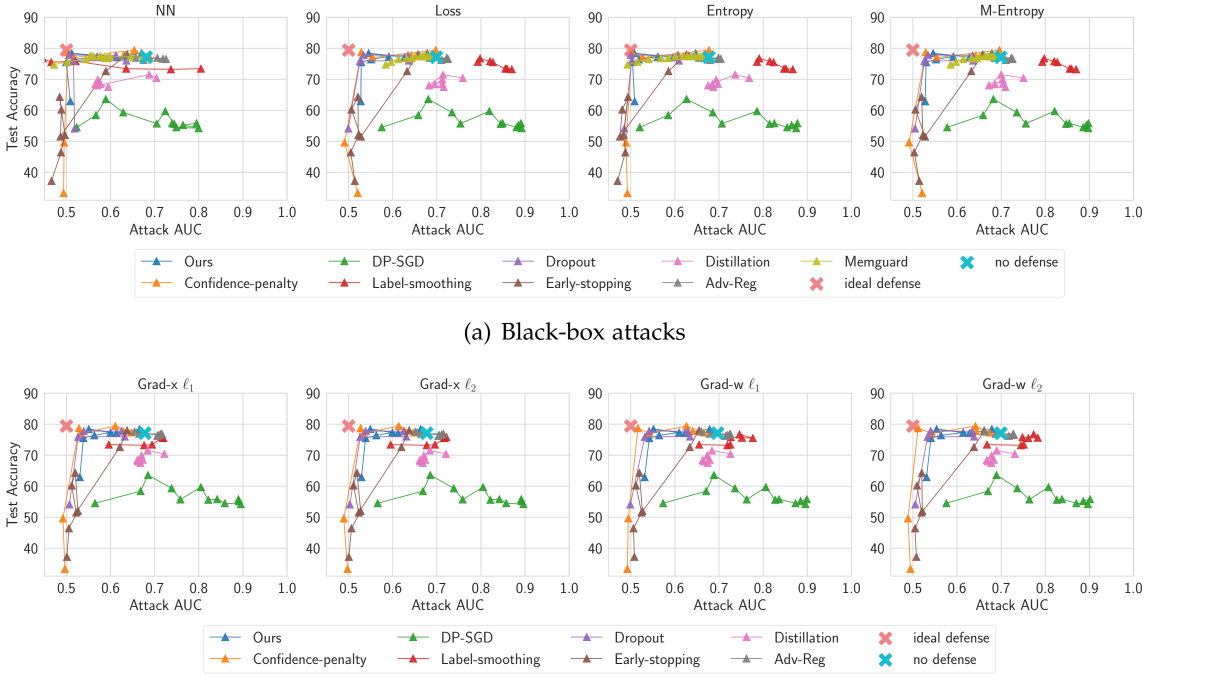
Binary Medical and Transaction Records. See Figure E.6-E.7: for binary medical and transaction records (Texas100 and Purchase100), Label-smoothing performs generally the best among all baseline methods. Compared to our approach, Label-smoothing can achieve superior model utility when the Attack AUC is of range around 0.65-0.8 on Texas100 and 0.55-0.65



(a) Black-box attacks

(b) White-box attacks

Figure E.4: Comparisons of all defense mechanisms on CIFAR-100 dataset (VGG11). We set the clipping bound C=1.0 and vary the noise scale over 0.01-0.3 for DP-SGD. We vary the noise magnitude across 0-400 for Memguard.



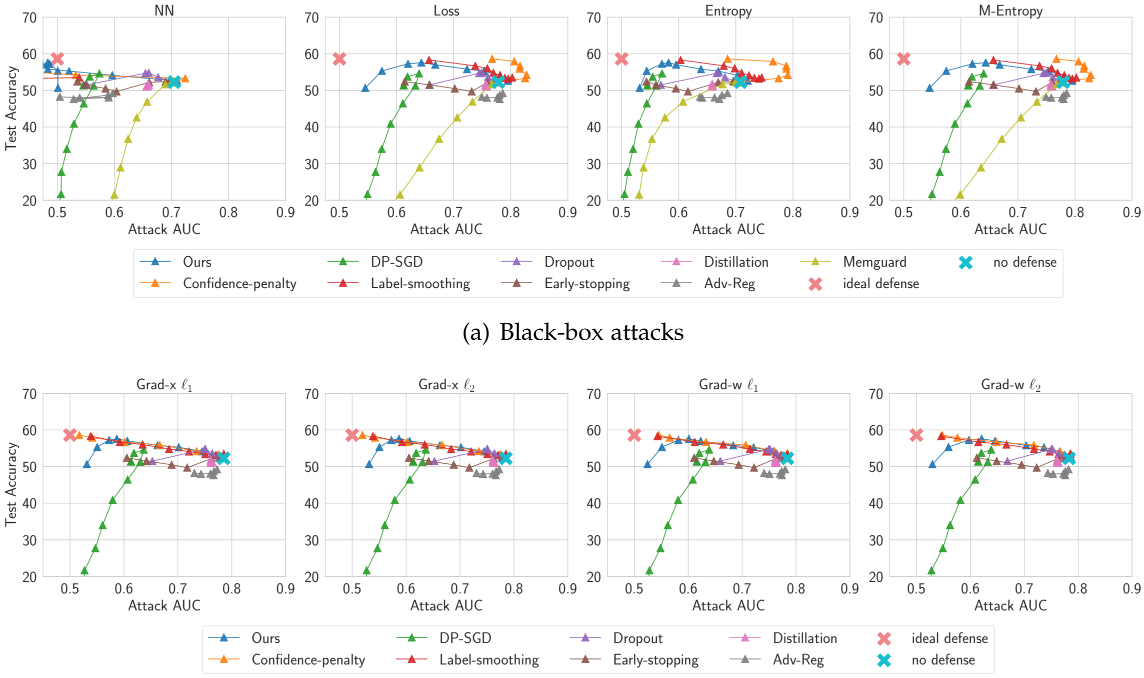
(a) Black-box attacks

(b) White-box attacks

Figure E.5: Comparisons of all defense mechanisms on CH-MNIST dataset. We set the clipping bound C=5.0 and vary the noise scale over 0.001-0.5 for DP-SGD. We vary the noise magnitude across 0-50 for Memguard.

on Purchase100. However, our approach is able to reduce the attack AUC to around the random-guessing level, which is not always possible for Label-smoothing (white-box attacks on Purchase100, black-box attacks on both datasets).

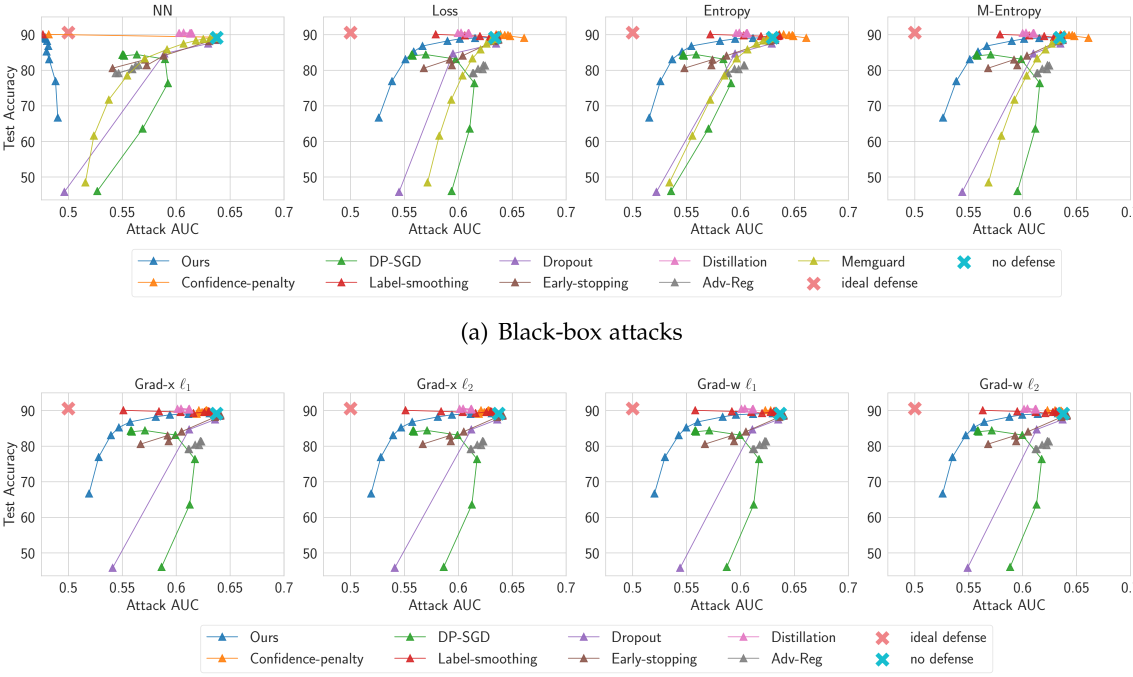
DP-SGD exhibits unexpected results for these two datasets: small noise scale results in both lower Attack AUC and higher model utility, which is contradictory to the common belief that small-scale noise only provides weak privacy guarantee and thus the Attack AUC will remain high. We conjecture that there exists a non-negligible gap between the worst-case privacy guarantee that provided by the theoretical privacy analysis and the real-world attack performance in practice. Especially for the small-scale noise case, the privacy cost ϵ is meaninglessly large and cannot faithfully reflect the risk when facing practical MIAs. We tried varying the noise scale in the experiments: by decreasing the noise scale, we find DP-SGD cannot reduce the Attack AUC to random-guessing level, while by increasing the noise scale, the utility soon drops significantly. In comparison, our approach consistently yields better privacy-utility trade-off.



(a) Black-box attacks

(b) White-box attacks

Figure E.6: Comparisons of all defense mechanisms on Texas100 dataset. We set the clipping bound C=1.0 and vary the noise scale over 0.001-0.5 for DP-SGD. We vary the noise magnitude across 0-500 for Memguard.



(a) Black-box attacks

(b) White-box attacks

Figure E.7: Comparisons of all defense mechanisms on Purchase100 dataset. We set the clipping bound C=1.0 and vary the noise scale over 10^{-4}-0.4 for DP-SGD. We vary the noise magnitude across 0-300 for Memguard.

E.3.9 Loss Histograms

To better understand the effect of each defense method, we additionally plot the loss histograms when applying different defense methods on target models with a ResNet20 architecture trained on CIFAR-10 dataset in Figure E.8-E.14. In the parentheses of each subtitle, we show the hyper-parameter values corresponding to each subfigure from left to right.

We observe that: *(i)* Regularization techniques in general have limited effects in reducing the gap between the training and testing loss distributions. *(ii)* Unlike our approach (See Figure 6.1 in the main paper), baseline methods are generally not able to increase the training loss variance nor closing the gaps between the member and non-member distributions, which explains the superior performance of our approach in defending various types of MIAs. *(iii)* By setting a relatively large noise scale, DP-SGD is able to increase the training loss variance and reducing the gap between the training and testing loss values (See last column of Figure E.11). However, the large scale of noise dampen the learning signal in this case, leading to a non-negligible utility drop.

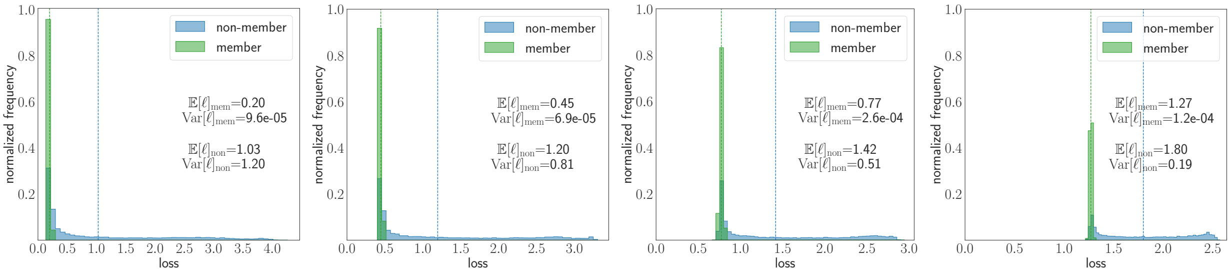


Figure E.8: Loss histograms when applying Label-smoothing (α = 0.2,0.4,0.6,0.8).

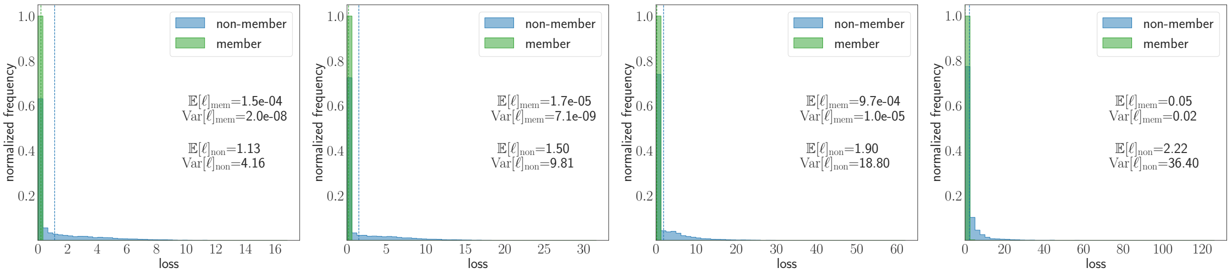


Figure E.9: Loss histograms when applying Dropout (dropout rate = 0.1,0.5,0.7,0.9)

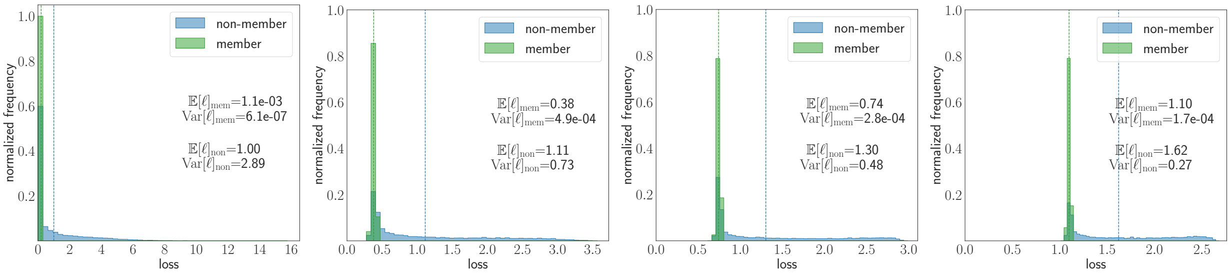


Figure E.10: Loss histograms when applying Confidence-penalty (α = 0.1,0.5,1.0,2.0)

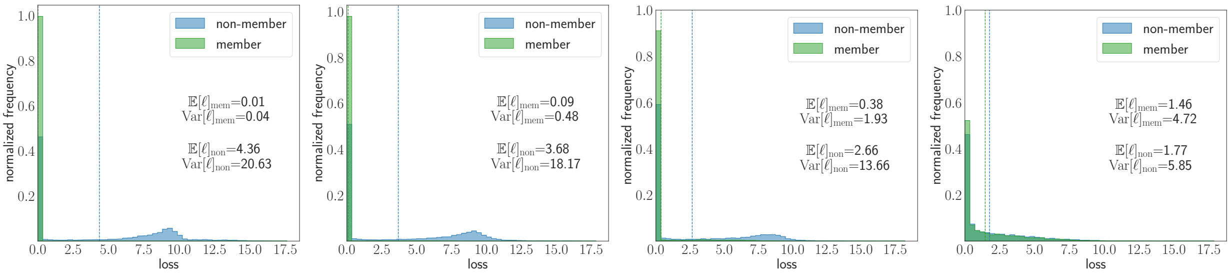


Figure E.11: Loss histograms when applying DP-SGD (noise scale = 0.01,0.05,0.1,0.5)

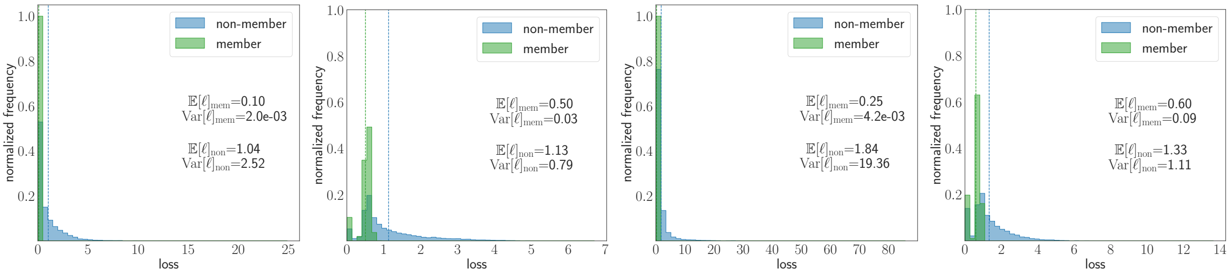


Figure E.12: Loss histograms when applying Adv-Reg (α = 0.8,1.0,1.2,1.4)

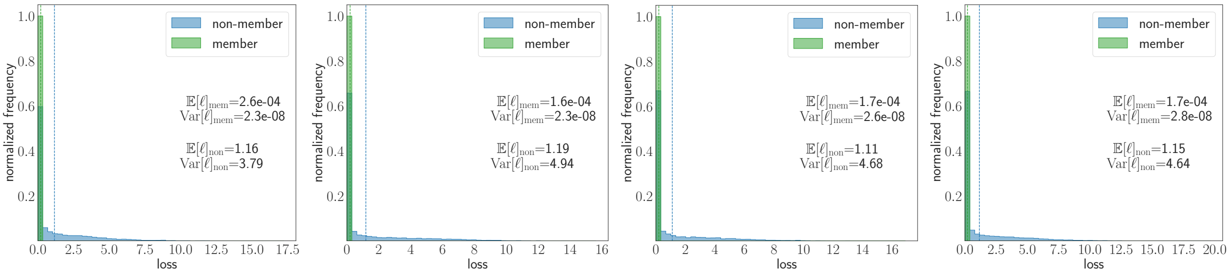


Figure E.13: Loss histograms when applying Distillation (T = 1,10,50,100)

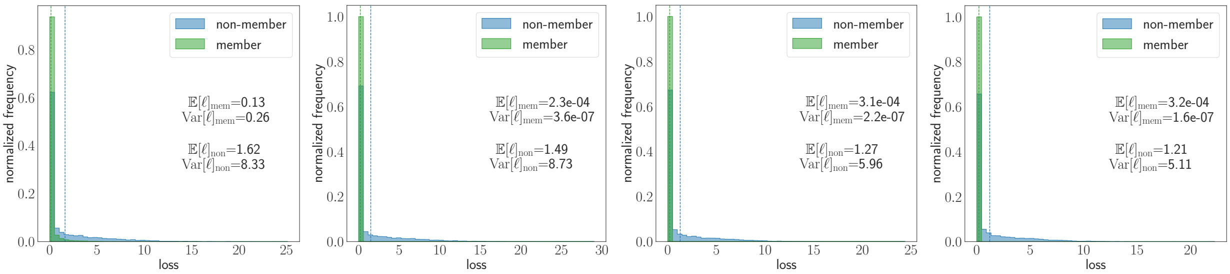


Figure E.14: Loss histograms when applying Early-stopping (ep = 25,50,75,100).

E.3.10 Analysis of Model Generalization

As supplementary to Section 6.7 of our main paper, we show results of toy experiments that investigate the impact of our approach on model generalization. We visualize the prediction scores and the decision boundaries in Figure E.15. In contrast to vanilla training that assigns high confidence scores on hard examples near the decision boundary, our approach can soften the decision boundaries, leading to a large area with low (and well-calibrated) predicted confidence scores. In line with [253, 163], we conjecture that the flatness of decision boundaries improves model generalization, while an in-depth analysis is left as future work.

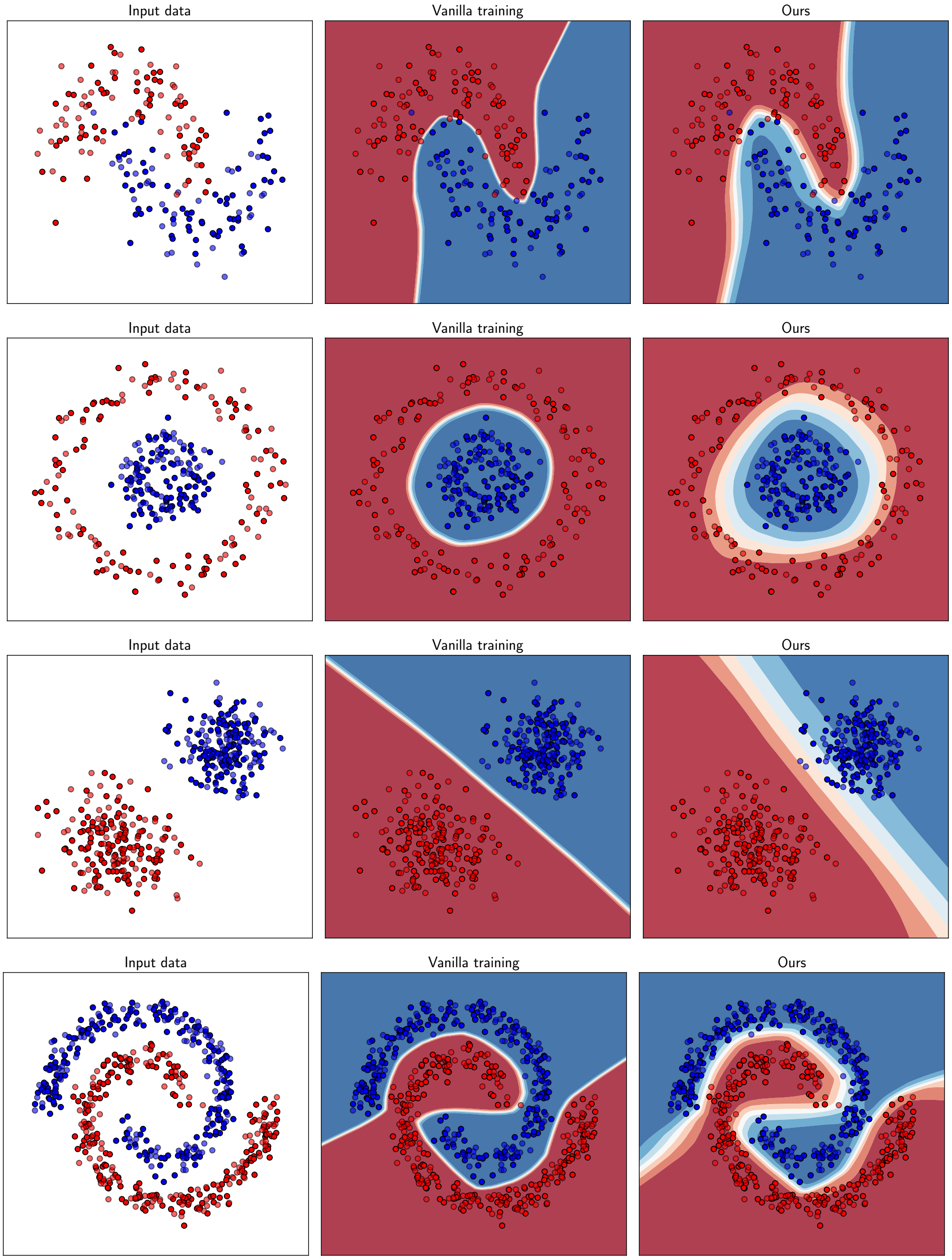


Figure E.15: Visualization of target models' prediction scores and decision boundaries. The training samples are shown in solid colors and testing points are semi-transparent.

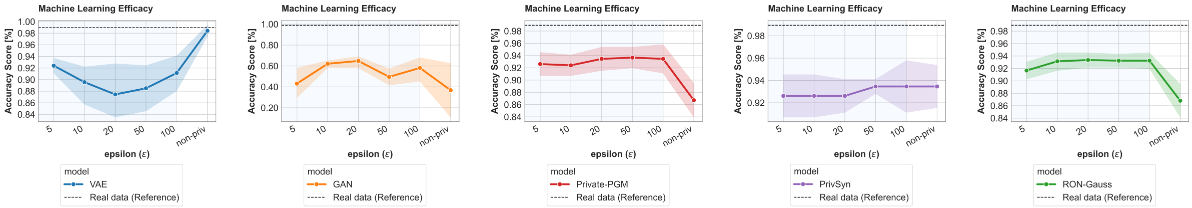
Towards Biologically Plausible and Private Gene Expression Data Generation

F

This appendix provides additional details to the main paper presented in Chapter 7: we provide additional results regarding the utility and statistical evaluation in §F.1, and co-expression evaluation in §F.2.

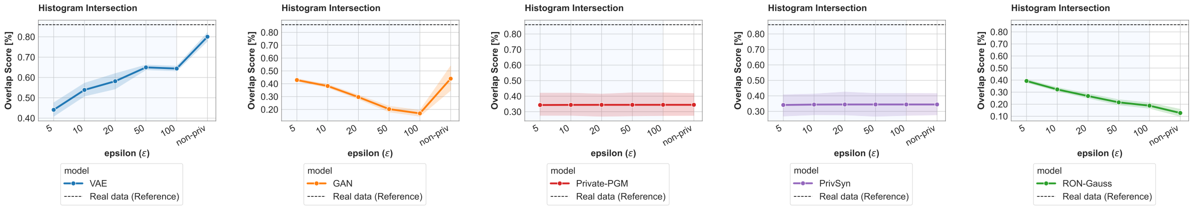
F.1 Utility and Statistical Evaluation

F.1.1 Plot of Evaluation Metrics Across Individual Models



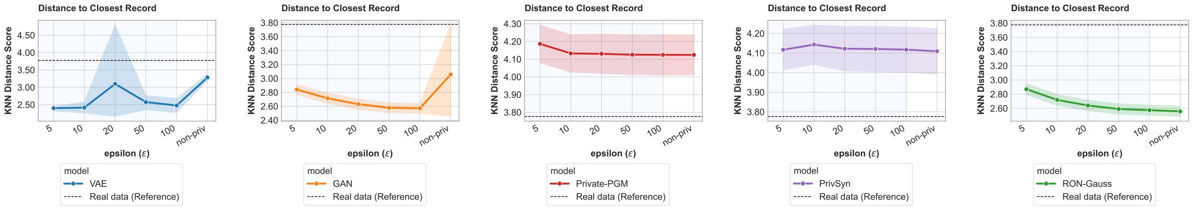
(a) VAE (b) GAN (c) Private-PGM (d) PrivSyn (e) RON-Gauss

Figure F.1: Utility Evaluation by Machine Learning Efficacy.



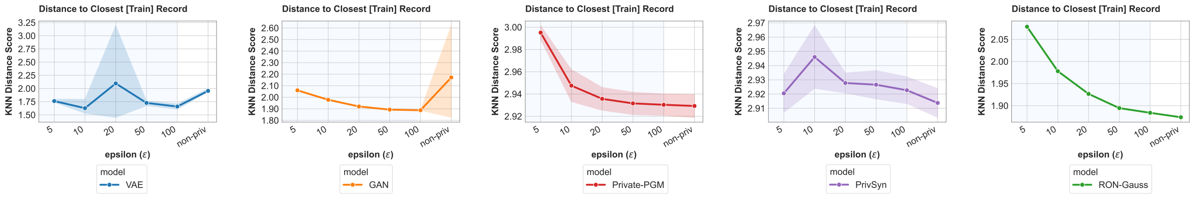
(a) VAE (b) GAN (c) Private-PGM (d) PrivSyn (e) RON-Gauss

Figure F.2: Statistical Evaluation by Histogram Intersection.



(a) VAE (b) GAN (c) Private-PGM (d) PrivSyn (e) RON-Gauss

Figure F.3: Distance to Closest Record.



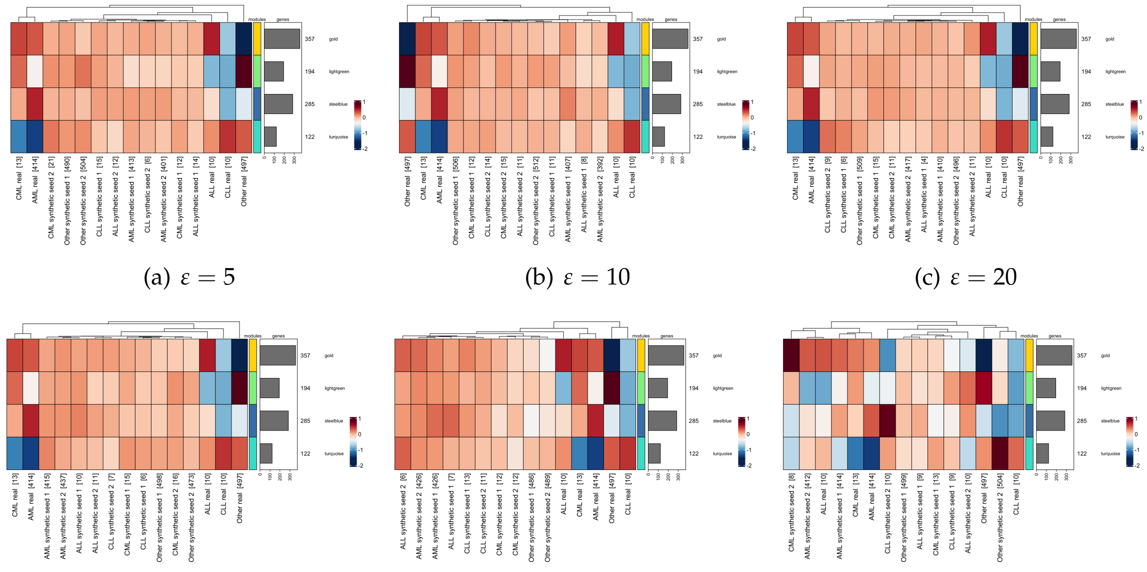
(a) VAE (b) GAN (c) Private-PGM (d) PrivSyn (e) RON-Gauss

Figure F.4: Distance to Closest (Train) Record.

F.2 Additional Plots for Gene Co-expression Evaluation

F.2.1 $r > 0$ (Default Setting)

We present in Figure F.5 the co-expressed gene module evaluated on the **GAN** generated samples (with $r > 0$). It is evident that there is a notable degradation of structural integrity in the module activation patterns within the synthetic data. This phenomenon persists across various privacy budget levels, including the non-private scenario.



(a) $\varepsilon = 5$ (b) $\varepsilon = 10$ (c) $\varepsilon = 20$

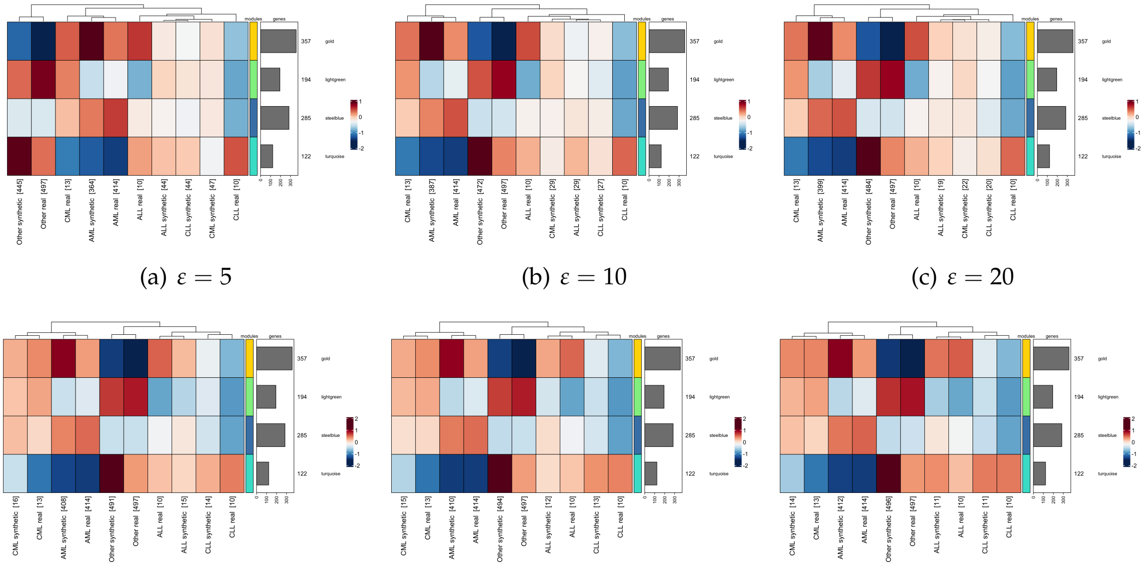
(d) $\varepsilon = 50$ (e) $\varepsilon = 100$ (f) non-priv

Figure F.5: Activation patterns of co-expressed gene modules in **GAN** after filtering co-expressions for $r > 0$. Shown are the Group Fold Changes (GFCs) of gene modules (rows) in the real and the synthetic data sampled with two different seeds. Numbers on the right indicate the number of genes per module. Darker shades of red imply activation of the gene module, while darker shades of blue indicate deactivation. The dendrograms show the hierarchical clustering of the classes in the different data sets.

We present in Figure F.6 the co-expressed gene module evaluated on the **PGM** generated samples (with $r > 0$). As can be observed, there is a noticeable decline in the activation patterns of the detected co-expression module when $\varepsilon \leq 20$.

We present in Figure F.7 the co-expressed gene module evaluated on the **PrivSyn** generated samples (with $r > 0$). A loss of module activation patterns can be observed for all shown privacy budgets, becoming increasingly prominent with decreasing ε.

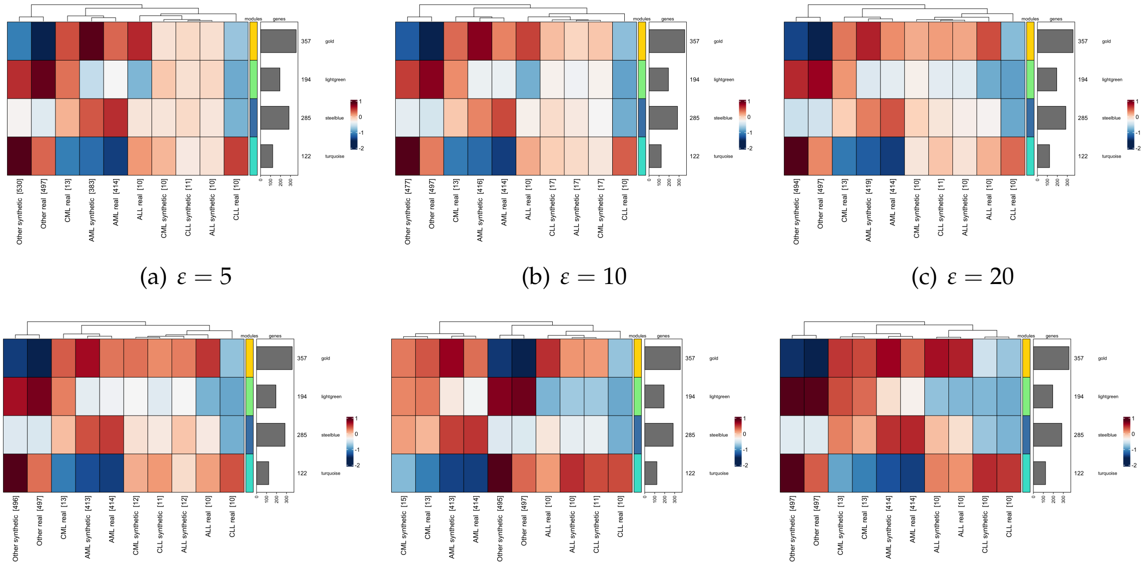
We present in Figure F.8 the co-expressed gene module evaluated on the **Ron-Gauss**



(a) $\varepsilon = 5$ (b) $\varepsilon = 10$ (c) $\varepsilon = 20$

(d) $\varepsilon = 50$ (e) $\varepsilon = 100$ (f) non-priv

Figure F.6: Activation patterns of co-expressed gene modules in **PGM** after filtering co-expressions for $r > 0$. Shown are the Group Fold Changes (GFCs) of gene modules (rows) in the real and the synthetic data sampled with two different seeds. Numbers on the right indicate the number of genes per module. Darker shades of red imply activation of the gene module, while darker shades of blue indicate deactivation. The dendrograms show the hierarchical clustering of the classes in the different data sets.

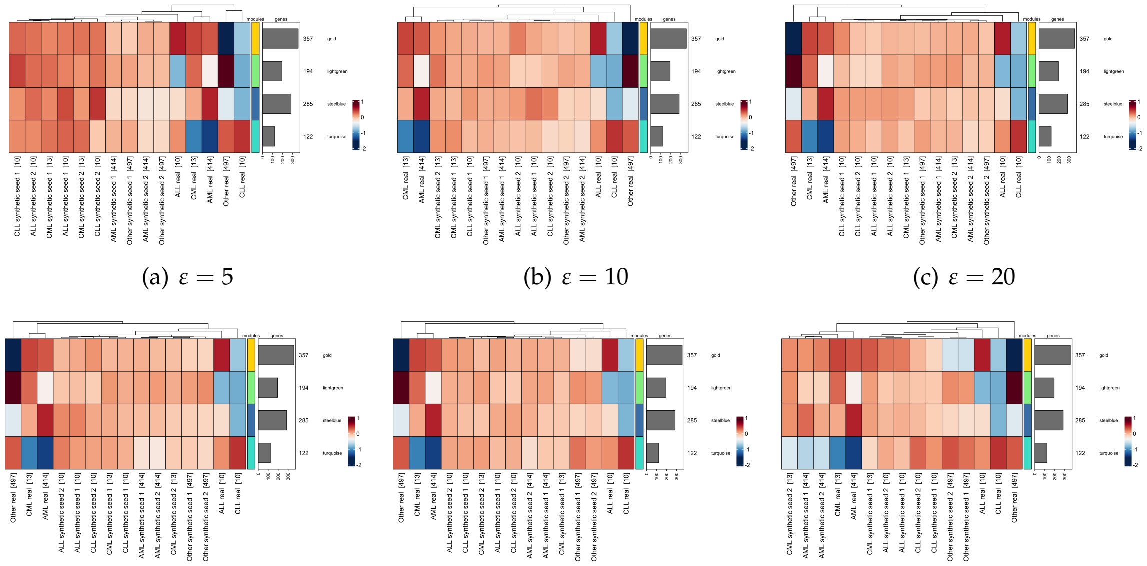


(a) $\varepsilon = 5$ (b) $\varepsilon = 10$ (c) $\varepsilon = 20$

(d) $\varepsilon = 50$ (e) $\varepsilon = 100$ (f) non-priv

Figure F.7: Activation patterns of co-expressed gene modules in **PrivSyn** after filtering co-expressions for $r > 0$. Shown are the Group Fold Changes (GFCs) of gene modules (rows) in the real and the synthetic data sampled with two different seeds. Numbers on the right indicate the number of genes per module. Darker shades of red imply activation of the gene module, while darker shades of blue indicate deactivation. The dendrograms show the hierarchical clustering of the classes in the different data sets.

generated samples (with $r > 0$). Already in the non-private setting, the synthetic data exhibits mostly uniform activation of the different gene modules, maintaining almost none of the structure present in the real data.



(a) $\varepsilon = 5$

(b) $\varepsilon = 10$

(c) $\varepsilon = 20$

(d) $\varepsilon = 50$

(e) $\varepsilon = 100$

(f) non-priv

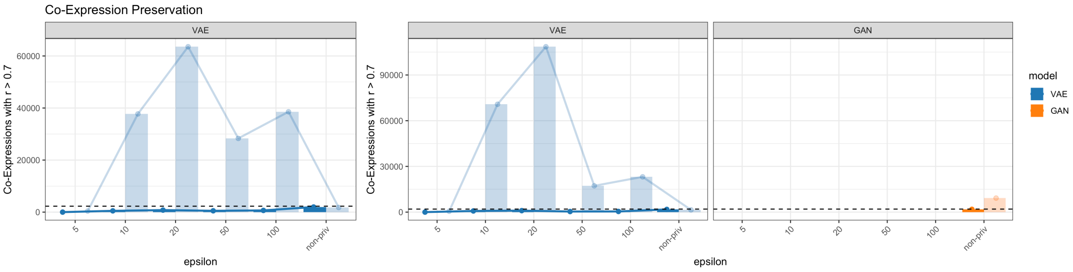
Figure F.8: Activation patterns of co-expressed gene modules in **RON-Gauss** after filtering co-expressions for $r > 0$. Shown are the Group Fold Changes (GFCs) of gene modules (rows) in the real and the synthetic data sampled with two different seeds. Numbers on the right indicate the number of genes per module. Darker shades of red imply activation of the gene module, while darker shades of blue indicate deactivation. The dendrograms show the hierarchical clustering of the classes in the different data sets.

F.2.2 $r > 0.7$

We evaluate the co-expression preservation for r>0.7 which is typically considered as biologically meaningful and present the results in Figure F.9. Note that in the first split (left) only the VAE model was capable of generating significant co-expressions with $r > 0.7$, while in the second seed (right) also the GAN trained without DP yielded some co-expressions. Within the VAE, co-expressions that were incorrectly introduced in the synthetic data greatly exceed the correctly reconstructed ones. Meanwhile, the GAN model struggles to generate any co-expressions above 0.7, regardless of their correctness.

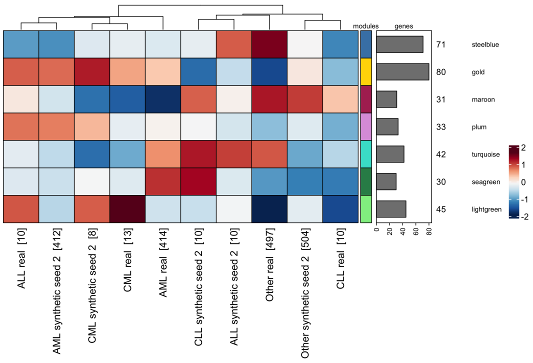
We present in Figure F.10 the co-expressed gene module evaluated on the **GAN** generated samples (with $r > 0.7$). The results indicate that the activation patterns of co-expression modules are poorly maintained in the synthetic data. This is evidenced by the incorrect clustering of disease classes when comparing the real and synthetic data.

We present in Figure F.11 the co-expressed gene module evaluated on the **VAE** generated samples (with $r > 0.7$). We observe a general decline in module activation with the reduction of privacy budgets, which suggests a degradation in the accuracy of reconstructed module activation patterns within the synthetic data.



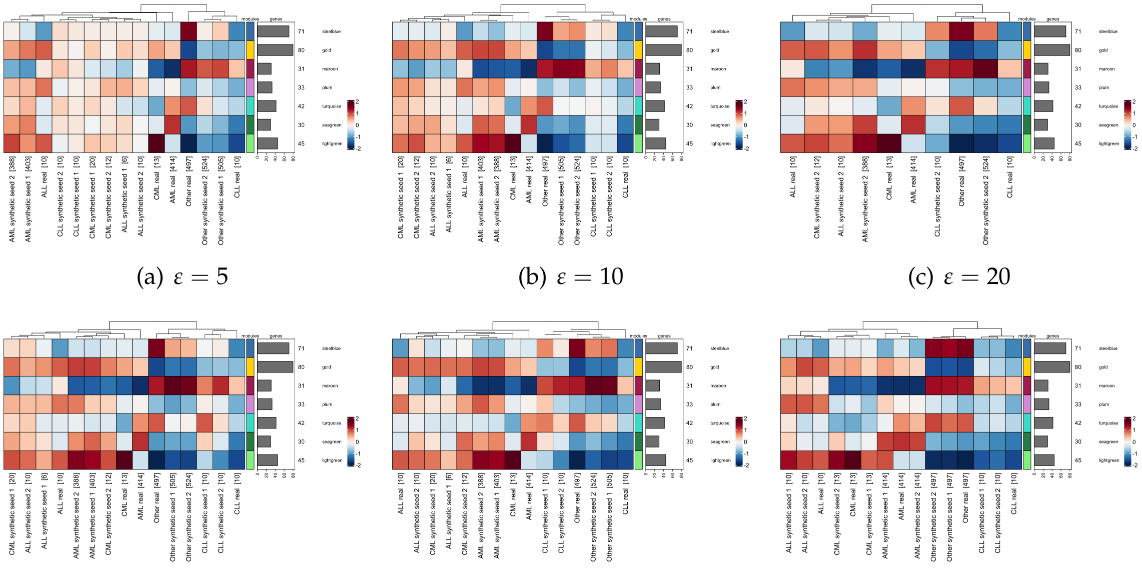
(a) Co-Expression Preservation

Figure F.9: Biological Evaluation by Co-Expression Preservation for $r > 0.7$. Shown is the co-expression preservation across the tested models for different values of ε as well as the non-private case for two different seeds used for creating the training split (left and right plot). Non-transparent bars give the number of correctly reconstructed co-expressions with Pearson Correlation Coefficient $r > 0.7$ and an associated p-value < 0.05, while semi-transparent bars give the number of co-expressions introduced by the model that did not exist in the real data. The dashed black line indicates the number of co-expressions in the real data. All values shown are means across two different seeds set for generating the data.



(a) non-priv

Figure F.10: Activation patterns of co-expressed gene modules in **GAN** after filtering co-expressions for $r > 0.7$. Shown are the Group Fold Changes (GFCs) of gene modules (rows) in the real and the synthetic data sampled with two different seeds. Numbers on the right indicate the number of genes per module. Darker shades of red imply activation of the gene module, while darker shades of blue indicate deactivation. The dendrograms show the hierarchical clustering of the classes in the different data sets.



(a) $\varepsilon = 5$

(b) $\varepsilon = 10$

(c) $\varepsilon = 20$

(d) $\varepsilon = 50$

(e) $\varepsilon = 100$

(f) non-priv

Figure F.11: Activation patterns of co-expressed gene modules in **VAE** after filtering co-expressions for $r > 0.7$. Shown are the Group Fold Changes (GFCs) of gene modules (rows) in the real and the synthetic data sampled with two different seeds. Numbers on the right indicate the number of genes per module. Darker shades of red imply activation of the gene module, while darker shades of blue indicate deactivation. The dendrograms show the hierarchical clustering of the classes in the different data sets.