Methods for Generative Modeling and Interpretable Classification with Strong Differential Privacy

Dissertation
der Mathematisch-Naturwissenschaftlichen Fakultät
der Eberhard Karls Universität Tübingen
zur Erlangung des Grades eines
Doktors der Naturwissenschaften
(Dr. rer. nat.)

vorgelegt von
Frederik Harder
aus Heidelberg

Tübingen
2023

Gedruckt mit Genehmigung der Mathematisch-Naturwissenschaftlichen Fakultät
der Eberhard Karls Universität Tübingen.

Tag der mündlichen Qualifikation:	05.02.2024
Dekan:	Prof. Dr. Thilo Stehle
1. Berichterstatterin:	Assoc. Prof. Dr. Mi Jung Park
2. Berichterstatter:	Prof. Dr. Philipp Hennig

CONTENTS

ZUSAMMENFASSUNG

Von Fahrassistenzsystemen bis hin zu ChatGPT hat Machine Learning (ML), und insbesondere die Unterdisziplin des Deep Learning (DL) die Automatisierung von Aufgaben ermöglicht, welche bis vor kurzem noch nur von Menschen ausgeführt werden konnten. Dieser Fortschritt wird vor allem von der Ansammlung enormer Datenmengen angetrieben, welche zum Training zunehmend größerer ML Modelle dienen. In vielen Bereichen enthalten diese Daten sensible persönliche Informationen über Individuen, wie zum Beispiel in Patientendaten, Einkaufshistorien oder Chat-Logs. Da diese Modelle erwiesenermaßen Informationen über ihre Trainingsdaten preisgeben können, entstehen so Konflikte zwischen dem Schutz der Privatsphäre sensitiver Daten und dem Bedarf an leistungsfähigen ML Modellen. Differential Privacy (DP) ermöglicht den Schutz sensibler Informationen in den Trainingsdaten, aber erfordert, dass der Traininsprozess durch zufälliges Rauschen erschwert wird, was zu schlechteren Ergebnissen führt. So erfordert das Training einen Abwägungsprozess zwischen Privatheit und Nützlichkeit des Modells. Insbesondere im Deep Learning, wo es sich als schwer herausgestellt hat, gute Kompromisse zu finden, hat sich bisherige Forschung oft auf schwache DP-Garantien konzentriert, welche keinen realen Schutz bieten, weil nur so akzeptable Grade der Nützlichkeit erreichbar waren.

Diese Dissertation erforscht DL-Methoden, welche für den Gebrauch mit starken DP-Garantien designt sind. Spezifisch thematisiert sie zwei herausfordernde Probleme in diesem Feld: Interpretierbarkeit und Generative Modelle. Die erste Teil der Arbeit zeigt, dass Methoden, welche die Entscheidungen von DL-Modellen erklären sollen, nicht in der Lage sind, nützliche Erklärungen zu liefern, wenn diese Modelle mit DP-Garantien trainiert wurden. Somit erweitert sich der eben vorgestellte Kompromiss um eine Dimension: Privatheit, Nützlichkeit, und Interpretierbarkeit. Als Alternative zu DL-Modellen stellt die Arbeit das *Locally Linear Maps*-Modell vor, welches bessere Interpretierbarkeit bei gleicher Privatheit und vergleichbarer Nützlichkeit bietet.

Der zweite Teil beschäftigt sich mit der Aufgabe, Datensätze mit DP Garantien zu veröffentlichen, was mithilfe von Generativen DL-Modellen ermöglicht wird. Die vorgestellte Methode *DP Mean Embeddings with Random Features* verwendet Approximationen von Kernel Mean Embeddings, um hochdimensionale Zusammenfassungen von Datensätzen zu erstellen, welche effizient mit DP-Garantien versehen werden können und dann zum Training eines Generativen DL-Modells verwendet werden. An die erste Version dieses Ansatzes, welche Random Fourier Features zur Approximation des Kernels verwendet, wird

in zwei Arbeiten angeschlossen, welche stattdessen Hermite Polynomial Features und gelernte Features aus vortrainierten DL-Modellen verwenden. Diese Methode erreichte neue Bestwerte für DP generative Modellierung mit starken DP-Garantien.

Beide Forschungsbeiträge bringen das Feld näher an dem Punkt, wo Differential Privacy breit in modernen Machine Learning-Methoden eingesetzt werden kann, da es die verlässlichste Methode darstellt, die Privatsphäre sensitiver Trainingsdaten zu schützen.

SUMMARY

From driver assistance in cars to ChatGPT, machine learning, and in particular the sub-field of deep learning, has enabled the automation of tasks which, until recently, could only be performed by humans. These advances are fueled by the collection of vast amounts of data which serve to train increasingly large models. In many domains, this data contains sensitive information about individual people such as patient records, purchasing histories, or logs of online conversations. As these models have been shown to reveal information about the data they have been trained on, this results in a conflict between the need for privacy of sensitive data and the demand for powerful machine learning models. Differential Privacy (DP) provides a way to protect the sensitive information of individuals in the training data but comes at the cost of introducing significant amounts of detrimental noise to the training process and thus induces a trade-off between the levels of privacy and utility that can be achieved in a given model. In deep learning, where finding a good compromise has proven especially difficult, past research has often focused on low levels of DP, which offer no tangible privacy protection, to obtain acceptable levels of utility.

This thesis explores methods for DP deep learning which are designed to function at high levels of DP. In particular, it discusses two challenging problems in this field: interpretability and generative modeling. The contribution presented first shows that methods that provide explanations of deep learning classifiers struggle to yield useful results on models trained with differential privacy, establishing a trade-off between interpretability, privacy, and utility. The work proposes *Locally Linear Maps* as an approach that yields better interpretability under the same privacy constraints while maintaining similar accuracy. The second topic considers the task of DP data release with the help of deep generative models. The proposed method *DP Mean Embeddings with Random Features* uses approximations of kernel mean embeddings to create a high-dimensional summary of a dataset which can be efficiently made DP. We then use this DP summary to train a generative model. The initial work using random Fourier features for kernel approximation is extended in two subsequent works using Hermite polynomial features and perceptual features obtained from pre-trained DL classifiers. This method obtained new state-of-the-art DP generative models for high privacy settings.

Both contributions move the field towards enabling broad application of differential privacy across modern machine learning methods, where it is the safest method for preserving the privacy of sensitive training data.

LIST OF ACRONYMS

ML	machine learning
DL	deep learning
DP	differential privacy *or* differentially private
DNN	deep neural network
CNN	convolutional neural network
SGD	stochastic gradient descent
DP-SGD	differentially private stochastic gradient descent
RDP	Renyi differential privacy
GAN	generative adversarial network
LLM	locally linear maps
RFF	random Fourier features
DP-MERF	DP mean embeddings with random features
DP-HP	DP Hermite polynomials
DP-MEPF	DP mean embeddings with perceptual features

LIST OF PUBLICATIONS AND PERSONAL CONTRIBUTIONS

This cumulative doctoral thesis is based on the following four manuscripts, which are presented in chronological order. All manuscripts have undergone a peer review process and have been published in scientific journals or conference proceedings. For each publication, we outline the individual contributions of the authors involved and use initials in place of full names for the sake of brevity.

MANUSCRIPT 1: **F. Harder**, M. Bauer, & M. Park (2020, April). Interpretable and Differentially Private Predictions. In *Proceedings of the AAAI Conference on Artificial Intelligence* (Vol. 34, No. 04, pp. 4083-4090).

Personal contributions: The DP-LLM method was co-developed, with MP providing the central idea and me contributing modifications, such as the weighting scheme. I implemented the method and designed and performed the experiments on image data, including comparison methods, with input from MP. I then also wrote the corresponding experiments section.

All contributions: MP developed the key ideas for the method in collaboration with FH. FH implemented the method and performed experiments on MNIST and FashionMNIST. MB performed experiments on the hospital dataset. The experiment design was collaborative, led by FH. MP wrote most of the paper, with FH and MB writing the experiments section and performing additional editing. The publication process was managed collaboratively with MP taking the lead.

MANUSCRIPT 2: **F. Harder**, K. Adamczewski, & M. Park (2021, March). DP-MERF: Differentially Private Mean Embeddings with Random Features for Practical Privacy-Preserving Data Generation. In *International Conference on Artificial Intelligence and Statistics* (pp. 1819-1827). PMLR.

Personal contributions: I implemented the method and designed and performed experiments on MNIST, FashionMNIST and the 2d Gaussians dataset, including DP-GAN comparisons. I also wrote the corresponding experiments section.

All contributions: MP provided the key ideas for the method. FH implemented the method and performed experiments on image data and 2d Gaussians data. KA performed experiments on the tabular datasets. MP developed Proposition 3.1 and the associated proof. MP wrote most of the paper, with FH and KA writing the experiments section and performing additional editing. The publication process was managed collaboratively with MP taking the lead.

MANUSCRIPT 3: M. Vinaroz, M. A. Charusaie, **F. Harder**, K. Adamczewski, & M. Park (2022, June). Hermite Polynomial Features for Private Data Generation. In *International Conference on Machine Learning* (pp. 22300-22324). PMLR.

Personal contributions: I performed preliminary experiments with Hermite polynomial features, and helped MV with performing experiments on image data and 2d Gaussians data, which used my code written for DP-MERF. I further created the model and dataset size comparisons shown in Figures 3 and 6 of the paper, along with performing the associated experiments.

All contributions: MP provided the key ideas for the method in collaboration with MAC, MV, and FH. FH performed preliminary experiments, after which MV and MAC developed the full implementation of the method and performed the core experiments. KA assisted in experiments with the tabular datasets. FH assisted in experiments with image data and the 2d Gaussians datasets, MAC developed Propositions $B.1$, $C.3$ and the associated proofs. MP and MAC wrote most of the paper with MV providing the experiments section on image data and with input from the other authors. The publication process was managed collaboratively, with MP and MV taking the lead.

MANUSCRIPT 4: **F. Harder**, M. Jalali, D. J. Sutherland, & M. Park (2023). Pre-trained Perceptual Features Improve Differentially Private Image Generation. In *Transactions on Machine Learning Research*.

Personal contributions: I developed the method in collaboration with MP and DJS and designed and performed all experiments. I wrote several sections, including the experiments section, the broader impact statement, parts of the method, related work, and discussion sections and managed the publication process.

All contributions: MP, DJS and FH developed the method. FH performed all experiments. DJS and JM developed Propositions $4.1 - 4.3$ and the associated proofs. MP, FH, and DJS wrote the paper. The publication process was managed by FH with input from MP and DJS.

ACKNOWLEDGEMENTS

I would like to thank my advisor Mijung Park for her guidance throughout the PhD. I am particularly grateful to her for introducing me to the topic of differential privacy, for her patience with me, as I learned the ropes of machine learning research, for the excellent error culture she maintains in her group, where I always felt able to communicate even some fairly embarrassing mistakes, and for providing many inspiring impulses and ideas to my research.

Further, I would like to thank my fellow PhD students Patrick Putzky, Amin Charusaie, and Kamil Adamczewski for many fruitful technical discussions and good company during uncounted coffee breaks, and almost as many board game sessions.

I am also grateful to Zeynep Akata and Philipp Hennig for the insightful feedback they provided as members of my thesis advisory committee and to both of them and Jakob Macke for being part of my thesis defense committee. I thank Danica Sutherland for her excellent contributions and advice during our DP-MEPF project.

As this thesis marks the end of more than two decades spent as a student, it is also a fitting occasion to thank the teachers whose inspiration and encouragement have helped me become a researcher. In particular, I thank Gisela Schmidt, for fostering my early love for mathematics when I entered elementary school, Eberhardt Hübner, for introducing me to the wonders of philosophy in my last years of high-school, and to Michael Baumgartner, Tarek Besold and Jörn-Henrik Jacobsen for helping me shape my perspective and interests as an aspiring researcher.

Beyond research, I am most deeply thankful to my wife Sylvie, who has made the low points bearable and everything better. After starting to build a life and a family together over the past few years, my pride and joy in finishing this thesis takes a well-earned second place.

And finally, I thank my parents, Karin and Hans, for always being there when I needed support or someone to talk to and for taking a genuine interest in my research.

1

INTRODUCTION

Every time we use our phones, we create data. Every time, our activity is quantified, stored, and sent to the owner of the website we visit, the application we use, or the phone's operating system itself. Most of this data likely contains benign information that we would readily share with anyone. Some of it may be more sensitive. The omnipresence of phones in our lives is emblematic of how commonplace data collection has become, but is only one instance among many. Medical history, financial records, messaging logs, and photographs shared online, all constitute potentially privacy-sensitive data that are nonetheless held by other institutions and companies.

The collection of personal data has the potential for many beneficial effects both in science and in our everyday lives. Medical studies, for instance, would be impossible without patients allowing access to their data. Shared location data allows for route planning and travel time estimation in free services such as Google Maps [39].

The growing demand for data is amplified by the growing capacities of the specialized algorithms used to analyze it. Machine learning and especially its subfield deep learning have made it possible to create models that learn complex patterns from vast quantities of data. Based on this learned information, the models are able to perform various tasks at an unrivaled level of competence, ranging from straightforward classification tasks to image and text generation [19, 131, 133] and autonomous game playing [54, 140]. So, undoubtedly, the collection of *big data* enables better algorithms, but whenever these data contain sensitive information, the question arises of whether the data-holder will use it in a responsible manner. There is always a risk of data leakage to third parties, be it by the data holder's intent or by error.

While ostensibly anonymized data releases have been established to leak sensitive information (see, e.g., Narayanan and Shmatikov [108]), recent research shows that even the publication of trained deep machine learning models can lead to an unintended release of information about training data [24, 25, 27, 57, 139, 144]. In a recent example, Carlini et al. [25] demonstrated that they could retrieve training data from the GPT-2 large language model [125] by, among other strategies, prompting it with partial sentences suspected of being in the dataset and sampling the model's highest confidence prediction. In this way, the model would output text containing items such as individuals' names and

contact information, social insurance number, completed URLs, and log files found in the dataset. In the most successful setting, this method retrieved the training sample verbatim in 67% of attempts, for samples that were verified to be unique in a training set of 40 gigabytes of text data. While the training data of GPT-2 was public prior to training, the same attack method could easily be applied to models trained on privacy-sensitive data.

A second example concerns popular diffusion-based image generation models, such as Stable Diffusion [131] and IMAGEN [133]. With a membership inference attack, Carlini et al. [27] identify which data points were in the training set for a large portion of the dataset (at a true positive rate of over 44% with a negligible false positive rate of 0.1%). Furthermore, the authors extract near-exact matches of several training datapoints by showing that the model occasionally generates these *memorized* datapoints. These findings are particularly impactful privacy breaches, in light of recent controversies around *"generative art"*. Several parties who claim their publicly available image data were used for training without consent are suing the publishers of these models over issues of copyright, after finding that the trained model would accurately copy their art style when prompted, e.g., with their name [21, 73]. Given the findings of Carlini et al. [27], the models may well have memorized parts of their work.

Fortunately, there is a method that can provably prevent these types of privacy violation. The mathematical guarantee of differential privacy (DP) provides an avenue for the privacy-preserving sharing of statistics, including trained ML models, that are based on sensitive data. If the shared statistic is differentially private, this ensures well-defined limits on how much information it conveys about individuals in the underlying dataset. The research area of differentially private machine learning attempts to combine the predictive power of machine learning with the privacy guarantees of differential privacy. Although this work has been quite successful in some areas of classical machine learning, the area of deep learning has proven to be a particularly difficult match for differential privacy. Here, private methods achieve only a fraction of the performance of their excellent nonprivate counterparts.

The remainder of this chapter is divided into two sections. Section 1.1 provides an introduction to the notion of differential privacy and related methods used in this thesis, providing both formal definitions of the key concepts and some intuition about the meaning of the privacy parameters. Following that, we discuss deep learning with DP guarantees in Section 1.2, detailing different approaches to DP deep learning and introducing the most common approach of differentially private stochastic gradient descent. We then present the shortcomings of the current state of DP deep learning models which motivate our research.

1.1 DIFFERENTIAL PRIVACY

The concept of Differential Privacy was first proposed as *ε-indistinguishability* and soon renamed in a series of papers by Cynthia Dwork, Frank McSherry, Kobbi Nissim, Adam D. Smith, and others in 2006 [47–49] and has since become widely accepted as a guarantee of data privacy. Informally, the goal of DP is to limit the amount of information an adversary gains from a published statistic about the underlying dataset. In particular, a DP statistic reveals limited information about each individual's data in the dataset, or even whether said individual is part of the dataset at all. This is achieved, as the initial name suggests, by ensuring that the statistic remains nearly *indistinguishable* under small changes to the dataset. That is, each outcome of the statistic remains about as likely if a single entry in the dataset is changed. The algorithm which accesses the data and releases a differentially private statistic is commonly referred to as a *mechanism* and may range from simple functions such as mean computations across features to rather complex tasks such as the training of a machine learning model on said data.

The amount of information about the data that a DP mechanism reveals depends on the two privacy parameters $\epsilon \geq 0$ and $\delta \in [0, 1]$. When both parameters are set to 0, no information is revealed, ensuring perfect privacy. Unfortunately, this also makes it impossible to learn anything from the data, so small positive values are chosen instead. As the values increase, the privacy guarantee weakens. The choice for ε implies that any set of outputs from the mechanism can only become more likely by a factor of e^{ε}, which, for small values, ensures that all outcomes remain approximately as likely under small changes in the data. δ defines the total probability mass that does not obey the bound defined by ε, and can be understood as the maximum probability that the DP guarantee does not hold. As a result, δ is typically chosen to be very small. (more on this in Section 1.1.5)

1.1.1 Formal definition of differential privacy

DP uses the notion of *neighboring* datasets D and D' to indicate small changes in the data. Two kinds of neighboring relations are typically considered in the DP setting: in the replacement relation, the datasets differ in one data point, that is, $\mathcal{D} \cup \{x\} \setminus \{y\} = \mathcal{D}'$ for some x and some $y \in \mathcal{D}$. In contrast, the inclusion/exclusion relation requires that one dataset be obtained by adding or removing one datapoint from the other, so $\mathcal{D} \cup \{x\} = \mathcal{D}'$ or vice versa. For details, see, e.g. Chatalic et al. [30]. Unless noted otherwise, this work makes use of the replacement relation.

Definition 1 (Differential Privacy). *A mechanism $\mathcal{M} : \mathcal{D} \to o$, which maps from a dataset $\mathcal{D}$ to some output o is (ε, δ)-differentially private iff $\varepsilon \geq 0$, $\delta \in [0, 1]$ and for any subset of outputs S and any neighboring datasets $\mathcal{D}$ and $\mathcal{D}'$, the following holds:*

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

Here, if $\delta = 0$, one speaks of *pure* DP, while for $\delta > 0$ the guarantee is called *approximate* DP. To obtain a meaningful guarantee, both ε and δ should be small. Typical choices are $\delta < 1/N$ and $\varepsilon < 2$, where N is the size of the dataset. The chosen values of (ε, δ) for a dataset are called the *privacy budget* and state the weakest acceptable level of DP the dataset may reach before forbidding further access. Section 1.1.5 will elaborate on how these values should be selected.

1.1.2 Properties of differential privacy

Differential privacy possesses two properties that are particularly important for its application: composability and invariance to post-processing.

Composability

While it is useful to have DP guarantees for individual statistics, it is rare that a dataset is only accessed a single time. DP naturally bounds the incurred privacy loss by accessing the same dataset multiple times with different DP mechanisms. This even holds if one mechanism uses the output of another as auxiliary information, which is referred to as *adaptive composition*. The simplest such composition method is *basic composition* as defined below for illustration. Other more advanced composition methods also exist and may achieve better bounds in specific settings [50, 65, 107].

Theorem 1 (Basic Composition (Thm. 3.16 in [50])). *Let $\mathcal{M}_i : \mathbb{N}^{|\mathcal{X}|} \to \mathcal{R}_i$ be an $(\varepsilon_i, \delta_i)$-DP algorithm for $i \in [k]$. Then if $\mathcal{M}_{[k]}(x) = (\mathcal{M}_i(x), \ldots, \mathcal{M}_k(x))$, then $\mathcal{M}_{[k]}$ is $(\sum_{i=1}^{k} \varepsilon_i, \sum_{i=1}^{k} \delta_i)$-DP.*

Post-processing invariance

Once released, the output of a differentially private mechanism cannot be analyzed in any way to reveal more information about the underlying dataset than the DP guarantee allows. This holds regardless of what possible auxiliary information is available and for any computational capacity. Formally, this is defined as follows:

Theorem 2 (Post-processing (Thm. 2.1 in [50])). *Let $\mathcal{M} : \mathbb{N}^{|\mathcal{X}|} \to \mathcal{R}$ be an (ε, δ)-DP mechanism and let $f : R \to R'$ be an arbitrary randomized mapping. Then $f \circ \mathcal{M} : \mathbb{N}^{|\mathcal{X}|} \to R'$ is (ε, δ)-DP.*

This differentiates DP from other privacy notions such as k-anonymity [149], which provides a degree of anonymity to individuals by ensuring that the values of each datapoint appear at least k times in the data, but may leak information given the right auxiliary information[1]. Differential privacy limits what an adversary can learn about a datapoint even if they precisely know the entire rest of the dataset as side information. This property allows one to use the mechanism output in any context without worrying about unforeseen consequences, as long as the privacy budget under which it has been released is deemed acceptable.

1.1.3 The Gaussian mechanism

Now that several properties of Differential Privacy have been introduced, the question arises as to how this guarantee can actually be achieved. Given that DP is defined in terms of probability distributions, it should come as no surprise that DP mechanisms function by introducing randomness to the otherwise often deterministically computed statistic that will be released. There are several known mechanisms for achieving DP such as randomized response, the exponential mechanism, and the Laplace mechanism (see, e.g. [50] for details). Here, we will focus on the Gaussian mechanism because it is the most commonly used mechanism in DP deep learning.

For a function $f : \mathcal{D} \to \mathbb{R}^d$, the *$L_2$-sensitivity* $\Delta_f = \sup_{D,D'} \|f(D) - f(D')\|_2$ of f describes how much the output of f may change at most between two neighboring datasets D and D' in terms of L_2 distance. By adding Gaussian noise $Z \sim \mathcal{N}(0, \Delta_f^2 \sigma^2 I)$ scaled to sensitivity Δ_f and a chosen noise parameter σ, the Gaussian mechanism $\mathcal{M}(D) = f(D) + Z$ makes the perturbed output of f differentially private.

Theorem 3 (Classical Gaussian Mechanism (e.g. Thm.A.1. in [50])**.** *For any $\varepsilon, \delta \in (0, 1)$, the Gaussian output perturbation mechanism with $\sigma = \sqrt{2 \log(1.25/\delta)}/\varepsilon$ is (ε, δ)-DP.*

This classical analysis of the Gaussian mechanism has the shortcoming of only applying to $\varepsilon < 1$, and has been shown to be generally suboptimal. An improved but more complex version has been proposed by [11]. However, we instead rely on a variant of DP called Renyi differential privacy (RDP) to track the total privacy cost of applying the Gaussian mechanism multiple times, as it allows for simpler composition.

1 For instance, when $k - 1$ identical datapoints are known background information, the final datapoint with the same values has lost all anonymity

1.1.4 Renyi differential privacy

In addition to the notions of pure and approximate DP, there are many related definitions and variants (see Desfontaines and Pejó [41] for an overview). Renyi Differential Privacy (RDP), proposed by Mironov [107], is the one variant we introduce here, as it is used in advanced composition techniques, particularly for the Gaussian mechanism in, e.g., DP stochastic gradient descent (see Section 1.2.1). It is defined in terms of Renyi divergence [128] as follows.

Definition 2 ((α, ε)-Renyi Differential Privacy (Def. 4 in [107]))**.** *A mechanism $\mathcal{M}$ is (α, ε)-RDP with order $\alpha > 1$ if for all neighboring datasets D, D'*

$$D_\alpha(\mathcal{M}(D) \| \mathcal{M}(D')) = \frac{1}{\alpha - 1} \log \mathbb{E}_{x \sim \mathcal{M}(D')} \left(\frac{\mathcal{M}(D)(x)}{\mathcal{M}(D')(x)} \right)^\alpha \leq \varepsilon.$$

RDP is a relaxation of pure DP and approaches ε-DP as $\alpha \to \infty$. Like regular DP, RDP also allows for composition of releases and is invariant to post-processing.

While the Gaussian mechanism provides regular DP for a set of (ε, δ) pairs with a complicated non-linear relation between the parameters, the same mechanism also grants an $(\alpha, \frac{\alpha}{2\sigma^2})$-RDP guarantee for all choices of α (Prop. 7 in [107]), which lends itself better to analysis and optimal composition. The privacy loss in the composition of multiple Gaussian mechanism releases, as we will encounter in DP stochastic gradient descent in Section 1.2.1, is therefore commonly tracked using RDP. After composition, the final privacy loss can still be presented in approximate DP using an existing conversion theorem (Prop. 3 in [107]), as approximate DP is the more common and intuitive privacy notion.

1.1.5 Choosing values for ε and δ

Depending on the choice of privacy parameters (ε, δ), the privacy protection granted by a DP guarantee can range from perfect at $\varepsilon = 0, \delta = 0$ to vacuous at $\varepsilon \to \infty$ or $\delta \to 1$. So what are the appropriate values for these parameters? Although there are no definitive answers to this question, a closer look at the guarantees under specific choices for ε and δ can provide some insight into the range of appropriate values.

Choosing ε

The simplest argument for the choice of ε is to look at the bound on the probability of a successful membership inference attack after a DP release. This has, for instance, been illustrated by Triastcyn and Faltings [160]. In this scenario, the adversary has observed an $(\varepsilon, 0)$-DP release o and is trying to distinguish between two candidate datasets D and D' with a uniform prior

$P(D) = P(D') = 0.5$. Without loss of generality, let D be the correct dataset, then the adversary will infer the probability $P(D|o)$ as

$$P(D|o) = \frac{P(o|D)P(D)}{P(o|D)P(D) + P(o|D')P(D')} \leq \frac{P(o|D)}{P(o|D) + e^{-\varepsilon} \cdot P(o|D)} = \frac{1}{1 + e^{-\varepsilon}},$$

where the inequality follows from the DP guarantee. In Table 1.1 we see that choices of $\varepsilon \geq 1$ allow for a significant improvement of the adversary's chances for accurate selection of D over D', and for near certainty at $\varepsilon \geq 5$. Values around $5 \geq \varepsilon \geq 0.2$ may be said to offer plausible deniability. While the adversary may make the correct inference most of the time, they can never be completely sure. Only for small values around $\varepsilon \leq 0.2$ does the bound ensure that $P(D|o)$ stays close to 50% and little membership information is leaked.

ε choice	5	2	1	0.5	0.2	0.1
$P(D\|o)$ upper bound	99.3%	88.1%	73.1%	62.2%	55.0%	52.5%

Table 1.1: Choices for ε and corresponding confidence bound

Recent work has shown that this worst case estimate can be relaxed when an adversary has to decide between more than two possible candidates. Guo, Sablayrolles, and Sanjabi [67] consider cases where an adversary knows all entries except one in a dataset $\mathcal{D}_{known} = \mathcal{D} \setminus \{x\}$ and is choosing between M different candidates $x \in \{x_1, \ldots, x_M\}$. As part of their results, they show that if the candidates have uniform probability, the attacker's advantage remains constant for a given c when choices for M and ε follow $\varepsilon \approx c \log M$. For instance, an attacker with two options who observes an $(\varepsilon = 1)$-DP release has about the same chance of guessing correctly as an attacker with $e^5 \approx 150$ options will, after observing an $(\varepsilon = 5)$-DP release. In light of this argument, even high values of ε may lend a tangible level of privacy protection, when one can assume that the adversary will have to choose between a large number of options. For example, guessing a 10 digit social security number ($M = 10^{10}$ options) will be prevented even by large choices of ε.

Since well before this argument was developed, large ε values up to $\varepsilon = 10$ are commonly used in works on DP deep learning, where acceptable results at high privacy levels are difficult to obtain (e.g. [33, 37, 45, 59]). In this context of deep learning with DP, we consider $\varepsilon \leq 2$ as a strong privacy guarantee, which offers plausible deniability even in the absolute worst case and is quickly amplified by the considerations in [67] when $M > 2$.

Recommendations for DP ML published by researchers at Google [124] broadly classify $\varepsilon \leq 1$ as a *strong formal privacy guarantee* and $\varepsilon \leq 10$ as a *reasonable privacy guarantee*, acknowledging that what strength is appropriate depends on the details of the application area. Choosing such a loose bound for the second category is justified by several factors, such as the low empirical

success rate of realistic privacy attacks against models with weak DP guarantees [10, 109, 123] and by the fact that many DL models only reach reasonable utility around $\varepsilon = 10$.

In practical applications of the current available models, weak DP guarantees with empirical privacy benefits are preferable to the alternative of not having any privacy protection at all. However, in the research of new DP algorithms, enabling stronger levels of DP with meaningful theoretical guarantees in the future is an important goal. Therefore, it seems sensible to aim for significantly stronger privacy than what [124] suggests as the lowest *reasonable* setting of $\varepsilon = 10$.

Choosing δ

Since δ represents the probability of the bound defined by ε not holding, Dwork, Roth, et al. [50] propose that a *cryptographically small* value should be chosen for it. In DP machine learning, where the dataset size $|D|$ is often in the tens of thousands or even greater, the consensus choice is $\delta < \frac{1}{|D|}$ [1, 40, 59, 68]. Since it usually factors into analyses as $\log \delta$, the exact value of δ is not as critical. The core intuition behind this choice, as Canonne [22] points out, is that the mechanism that releases a single unperturbed datapoint from D at random is $(\varepsilon = 0, \delta = \frac{1}{|D|})$-DP. So in order to ensure that a mechanism can never simply leak a datapoint, a smaller value for δ should be selected.

This concludes the introduction to DP. To complete the foundation of the work presented in this thesis, the following section will provide background on the field of Deep Learning and some relevant subfields, and then explain how DP can be applied there.

1.2 DEEP LEARNING WITH DIFFERENTIAL PRIVACY

The term deep learning (DL) refers to a class of machine learning methods, which are a big contributor to the increasingly widespread use of machine learning applications, particularly in the domains of computer vision and natural language processing. A key feature of DL is the hierarchical computation of learned non-linear features. For a comprehensive introduction, refer to [62]. In a deep learning model, each individual *layer* typically takes the form of

$$g_\theta : \mathbf{x} \to f(\mathbf{W}\mathbf{x} + \mathbf{b})$$

where $\theta = \{\mathbf{W}, \mathbf{b}\}$ are the learned layer parameters and f is a non-linearity such as the sigmoid or ReLU function. Depending on the data domain, θ may exhibit additional structure. The most common examples of this are convolutional layers, which apply the same transformation locally to each region of the

input. Depending on the dimensionality of the data, convolutions may span a single or several dimensions. For audio, image, and MRI data, 1D, 2D, and 3D convolutions are used, respectively. State-of-the-art deep learning architectures further include a variety of other layers, such as pooling, normalization, and attention layers, which will be skipped for this introduction. As several such layers are stacked, the overall model g becomes increasingly *deep* and capable of learning more complex functions.

$$g_\theta = g_{\theta_n} \circ \cdots \circ g_{\theta_3} \circ g_{\theta_2} \circ g_{\theta_1}$$

Given an objective function $\mathcal{L}$, the model g_θ is then typically iteratively trained on small *minibatch* sub-sets of the dataset which are sampled without replacement going through the whole dataset, which is called an *epoch*, and this process is repeated usually until some convergence criterion has been reached. On each training step and given the current minibatch of data, the gradient of the objective with respect to θ is calculated using the backpropagation algorithm [132] and the model is updated using stochastic gradient descent (SGD).

In the following, we introduce two research areas in deep learning that are particularly relevant to our contributions, interpretability and generative modeling, before laying out how deep learning can be made differentially private.

Interpretability in deep learning

Deep learning architectures can model high-dimensional and highly non-linear functions, and it can thus be difficult to put into human-understandable terms how a certain model output arises from the input. It is crucial in many applications to maintain means of understanding how a model arrives at its output, for example, when models are used for medical imaging, autonomous driving, or loan approval. As pointed out in a survey by Zhang et al. [172], a need for interpretability arises in particular when models are used to inform decisions that affect people, as these people often have ethical and legal rights to an explanation [64] and to a guarantee that the decision was made on a fair basis. Since such decisions are often based on sensitive personal data, the application areas of privacy and interpretability methods naturally overlap.

In this work, we focus on passive local attribution methods, which constitute the most actively researched class of interpretability methods for DL according to the taxonomy put forward by [172]. To analyze the importance of individual features, gradient attribution methods compute the gradient of the network output with respect to the network input. By definition, this gradient shows, for each feature, how strongly and in what direction a small change in the feature value would affect the output. Features that elicit a larger change could thus be said to be more important. However, due to the non-linear nature of DL models, the gradient may be highly sensitive to small changes in the input.

More reliable and informative attributions can be obtained through modification of the gradient. For example, SmoothGrad [142] averages gradients of slightly perturbed inputs by adding Gaussian noise, and Integrated Gradient [148] averages gradients from inputs by interpolating linearly between the given datapoint and a chosen *neutral* reference input.

Deep generative models

The objective of generative modeling is to train a model on a finite dataset such that the model can produce samples that resemble the underlying data distribution from which the training set was obtained. Samples should resemble the data both individually and in their distribution. The measure by which the generated distribution should approximate the true distribution is often not clearly defined, as the distributions themselves are typically highly complex, and no analytic definition of their ground truth exists. As a result, various domain-specific quality metrics such as Fréchet Inception Distance [70] for natural images and the BLEU score [117] for language data have been designed as proxies. In the case of DP data release, where the generated data is intended to replace training data for some downstream task, the quality of the generated data is often measured by the performance of a model trained on generated data, tested on real data.

Conditional generative models learn the distribution of data depending on some associated feature. The most common example is conditioning on class labels, but conditioning on continuous parameters such as color [20], or even complex parameters such as picture description [131, 133] in the case of images is equally possible.

Over the past years, some dominant approaches to deep generative modeling have been Variational Auto-Encoders [84], Generative Adversarial Networks (GANs) [63], flow-based models [44, 129] and most recently diffusion models [71, 131, 143, 146, 147].

1.2.1 Making deep learning differentially private

When considering training a DNN in terms of DP, we require a mechanism $\mathcal{M} : \mathcal{D} \to \Theta$ that takes a data set D and releases a set of trained model parameters θ. For $\mathcal{M}$ to be DP, some part the training algorithm must be identified, which constitutes a *bottleneck* such that all information θ receives about D travels through it. This bottleneck must be made DP. Jarin and Eshete [78] distinguish between five potential choices for such bottlenecks in ML algorithms. Of these five, three can be ruled out as infeasible for DNNs and a fourth is only used in a special case. We will briefly discuss these four options and then introduce one technique not mentioned in [78], which is particularly

relevant to this work, before covering the fifth and de facto standard approach to DP model training, DP-SGD, in greater detail.

OUTPUT PERTURBATION (INFEASIBLE) In many simpler mechanisms, including even machine learning algorithms with convex objective functions such as regularized logistic regression (as shown in [31]), the sensitivity of trained weights can be quantified, so noise can be added to the weights directly. This is not the case for DNNs due to the nonconvex optimization taking place.

INPUT PERTURBATION (INFEASIBLE) Adding noise to the input data is also a viable method for convex optimization problems [58, 81], but has not found application in DNNs, as the direct release of the data requires a high noise scale which would erase all useful information.

OBJECTIVE PERTURBATION (INFEASIBLE) Since SGD updates the model parameters with a gradient with respect to some objective function, making the objective itself differentially private is a way of ensuring DP of the trained model. Chaudhuri, Monteleoni, and Sarwate [31] developed this method for convex optimization problems, where the objective itself has a bounded sensitivity. In deep learning, this is generally infeasible due to the nonconvex objective having intractable sensitivity.

PREDICTION PERTURBATION (SPECIAL CASE) The approach of making a nonprivately trained model DP by adding noise to its predictions during deployment is generally infeasible, since, as discussed for output perturbation, the sensitivity of the learned weights is unknown and consequently no sensitivity for the predictions can be found either. An example where prediction perturbation does achieve DP is the PATE framework [115, 116]. Here, an ensemble of models is trained on disparate subsets of the dataset such that only one model is affected by the change of a datapoint and the average prediction of the ensemble has a bounded sensitivity as a result. However, this is a special case and prediction perturbation for DP has not found wider application beyond that.

DATA SUMMARY PERTURBATION In this method, information about the training dataset is collected in some summary statistic (such as a high-dimensional feature embedding), which is released with a DP mechanism. The training objective does not access the data directly, but only through the DP summary, and thus, similar to objective perturbation, the optimization of the objective is DP. This approach is not mentioned in Jarin and Eshete [78] but is central to the methods discussed in Sections 3.2 and 3.3.

GRADIENT PERTURBATION The most common method of DP deep learning is the DP release of the gradient update in each iteration, termed *differentially private stochastic gradient descent* (DP-SGD). By aggregating the privacy loss across all training steps, a DP guarantee is obtained for the final weights. Since the key contributions of this work are motivated by the properties and shortcomings of DP-SGD, we devote the remainder of this section to a detailed introduction.

Differentially private stochastic gradient descent

After some previous work on DP-SGD [13, 145], the work by Abadi et al. [1] added significant improvements in privacy accounting and refined the method to the form which finds common use today. We will begin by explaining the DP training algorithm and then discuss the equally important privacy accounting methods of DP-SGD below.

Releasing the gradient on a given training iteration in a differentially private way requires a bound on the gradient's sensitivity, but no analytic bound is generally available. DP-SGD solves this problem by computing the gradient for each sample in the minibatch separately and clipping them to an L_2-norm of some fixed maximum value C. The sum of these *per-sample gradients* now naturally has an L_2-sensitivity, depending on the neighboring relation used: sensitivity C with the inclusion/exclusion relation (adding or removing a datapoint) and $2C$ with the replacement (exchanging a datapoint) relation. Due to the defined sensitivity, the sum can be released using the Gaussian mechanism. A precondition for DP-SGD is thus that the loss function consists of an aggregation of per-sample losses such that the computation of per-sample gradients makes sense. This condition is frequently satisfied in deep learning, but is violated, for instance, by the use of batch normalization layers [74] and objectives based on the estimation of the maximum mean discrepancy [66]. Algorithm 1 shows the DP-SGD algorithm as given in [1]. Note that sampling and noise addition differ slightly depending on the neighboring relation, as will be explained below.

Algorithm 1 DP-SGD for the inclusion/exclusion relation (Alg 1 in [1])

Require: Dataset $\mathcal{D} = \{x_1, \ldots, x_N\}$, loss function $\mathcal{L}(\theta) = \frac{1}{N}\sum_i \mathcal{L}(\theta, x_i)$. Parameters: learning rate η_t, noise scale σ, group size L, gradient norm bound C.

Initialize θ_0 randomly

for $t \in [T]$ **do**

Take a random sample L_t with sampling probability L/N

Compute gradient

For each $i \in L_t$, compute $\mathbf{g}_t(x_i) \leftarrow \nabla_{\theta_t}\mathcal{L}(\theta_t, x_i)$

Clip gradient

$\bar{\mathbf{g}}_t \leftarrow \mathbf{g}_t(x_i) / \max\left(1, \frac{\|\mathbf{g}_t(x_i)\|_2}{C}\right)$

Add noise

$\tilde{\mathbf{g}}_t \leftarrow \frac{1}{L}\left(\sum_i \bar{\mathbf{g}}_t(x_i) + \mathcal{N}(0, \sigma^2 C^2 \mathbf{I})\right)$

Descend

$\theta_{t+1} \leftarrow \theta_t - \eta_t \tilde{\mathbf{g}}_t$

end for

Output θ_T

Privacy accounting for DP-SGD

DP-SGD performs a differentially private release of the weight gradient via the Gaussian mechanism given a minibatch of data on each iteration. The fact that only a subset of the data is accessed at every step can be used to improve the privacy analysis, as detailed in [9] for (ε, δ)-DP. Thus, an efficient analysis must take advantage of this insight and then compose the resulting privacy loss over the many training steps. Abadi et al. [1] were the first to develop a version of this analysis specific to the Gaussian mechanism termed the *Moments Accountant*. More general methods based on Renyi DP, which we introduced in 1.1.4, have since been developed for both inclusion/exclusion [175] and replacement [162] neighboring relations. Depending on the neighboring relation, the analyses make use of subsampling methods, which differ from regular minibatch sampling. Whereas in the non-DP setting one typically divides the whole dataset into minibatches, which are then iterated over in an epoch, for the inclusion/exclusion relation, each new batch must be sampled independently of the previous ones without replacement. Accounting for the replacement relation, on the other hand, [1, 175] further diverges from non-DP practice and requires *Poisson subsampling*, which includes each datapoint $x \in D$ in each minibatch with some probability $\gamma \in [0, 1]$, leading to varying minibatch sizes during training.

1.2.2 Current shortcomings of differentially private deep learning

Although the theoretical guarantees of DP are a good match for the privacy issues in deep learning, in practice, enforcing the guarantees incurs a heavy loss of utility that has so far prevented the widespread adoption of DP in deep learning. Following their successful extraction attacks on generative diffusion models, Carlini et al. [27] tried to train the models with DP as a countermeasure but were unable to maintain an acceptable level of performance in this setting. In fact, many tasks that are considered solved in nonprivate deep learning are still major challenges in the private setting. For example, in image classification, current state-of-the-art models obtain 99.5% [46] and 91.0% [171] test accuracy on the datasets Cifar10 (10 classes) [86] and ImageNet (1000 classes) [38] respectively. With the remaining error being due in part to false labels in the test set [113], classification in these data sets is largely considered solved. In contrast, without the use of auxiliary public data, De et al. [37] and Sander, Stock, and Sablayrolles [134] obtain the best scores in the DP setting with accuracy 81.4% on Cifar10 and 39.2% on ImageNet, respectively, at a fairly weak level of privacy with $\varepsilon = 8$. The reason for these gaps can be found in several properties of deep learning models that make differentially private training difficult:

LARGE MODEL SIZE DNNs have been found to benefit from a much larger number of learned parameters than other models without encountering the problem of overfitting. The number of model parameters in nonprivate applications frequently ranges from several millions (e.g. AlexNet [86]: 61 million) to billions (GPT-3 [19]: 175 billion). Each of these parameters may store information about the dataset, so all of them must be released privately.

LONG ITERATIVE TRAINING SGD training takes several thousand to millions of steps, where each individual update constitutes a separate access of the data. The rigorous privacy analysis developed for this setting (see Section 1.2.1) makes the high number of DP releases manageable, but does not eliminate the fact that having fewer releases would be more efficient.

LIMITED THEORETICAL UNDERSTANDING As Ye and Shokri [167] point out, DP-SGD provides a stricter DP guarantee than would in many cases be necessary by making each gradient update private, where the only output that actually requires a DP guarantee is the final trained model. Only protecting the final output of the training algorithm should require less effort as less information is revealed, but so far no method of conducting the necessary privacy analysis has been found. Existing approaches that fundamentally improve on the per-iteration privacy cost analysis of DP-SGD [6, 167] do so only under the strong assumption of convex objective functions.

Despite these shortcomings, DP deep learning is the most promising approach to tackling privacy-preserving machine learning on complex and high-dimensional data, such as images or natural language data. It is thus a primary research objective in DP deep learning to improve the privacy-utility trade-off, so it becomes acceptable in practice.

One successful method for improving the trade-off that has recently gained popularity is to use auxiliary public data from the same domain as the private data, often by pretraining a model nonprivately on those public data [28, 37, 94, 95]. A limitation of this approach is that it requires available public data that is similar enough to the private data to contain useful information. This may be true for natural images and text, but it is less likely to be the case for e.g. most tabular datasets. Tramèr, Kamath, and Carlini [157] further raise concerns about using public data scraped from the Web with insufficient curation, as many such datasets have been found to contain sensitive data [16].

Faced with unacceptable privacy-utility trade-offs, another route taken is to simply increase the privacy budget. Increasing ε may seem tempting, since DP provides some theoretical level of privacy for any value of ε and there is no strong consensus on the correct values anyway. In one of the rare cases where the DP implementation of a company was analyzed, Tang et al. [153] found that the privacy budget at $\varepsilon = 16$ per day was set so high that it offered no formal protection. In the DP deep learning literature, similar trends occur on tasks such as generative modeling, where models only yield acceptable results in weak privacy settings, and all results are reported for $\varepsilon \geq 8$ [33, 120, 155, 164]. We caution against this route, as raising ε quickly leads to vacuous guarantees (see section 1.1.5). As we state in the following, our explicit focus for this work lies on developing methods for DP deep learning that still function in the high-privacy regime.

2

RESEARCH AIM

Differentially private deep learning aims to combine the expressive power of deep models with the rigorous privacy guarantees of DP. As explained in Section 1.2.2, the standard approach of training deep models with DP-SGD faces several drawbacks, leading to poor performance in high-privacy settings. For the purpose of this thesis, and in light of the illustration in Section 1.1.5, we define high privacy as (ε, δ)-DP that requires $\varepsilon \leq 2$ and $\delta < 1/N$. The overarching goal of this thesis is the development of differentially private machine learning methods for high-dimensional data that retain good performance even in high-privacy settings. Here, we focus on two important application areas which have proven challenging in the context of DP: interpretable classification and generative modeling.

Objective 1: Interpretable differentially private classification

As the first topic, our aim is to improve the interpretability in DP deep classification models. Probing non-DP deep classifiers with gradient attribution methods yields sensible explanations of their predictions. However, we find that, if the same classifiers are trained under DP constraints, these explanations become hard to interpret due to the added noise in the training process. This poses a problem, as ML applications on sensitive personal data frequently require both preservation of privacy and transparency of their decision-making process.

In manuscript 1 (Appendix A) we present the previously unexplored trade-off between privacy, utility, and interpretability. Our objective is to improve this trade-off by developing *locally linear maps*, a shallow model that preserves better interpretability in DP settings while maintaining a reasonable level of classification accuracy.

Objective 2: DP data release with strong privacy

For an alternative solution to the problem of lacking interpretability, we turn to methods for DP data release as our second topic. These methods create differentially private proxies of sensitive datasets which can be used in their place and ensure privacy prior to model training. These methods can remove

the need for DP model training, and thus the source of the interpretability-privacy trade-off, but only if the released proxy data are a sufficiently good analog for the private data. Our goal is to train a deep generative model, but forego the need for iterative data release present in DP-SGD to obtain stronger privacy guarantees. To this end, we experiment with releasing dataset mean embeddings based on random Fourier features in a privacy-preserving way and use them to train deep generative models. The resulting model is presented in manuscript 2 (Appendix B) and is evaluated on several tabular datasets and small image datasets. We expand on this method with the addition of Hermite polynomial feature embeddings as an improvement on random Fourier features in manuscript 3 (Appendix C).

Objective 3: Leverage auxiliary public data to improve DP data release

Although random Fourier and Hermite polynomial feature embeddings are relatively successful on lower-dimensional image datasets like MNIST and FashionMNIST, they do not scale well to more complex data. Our third objective is therefore to explore how extracting features with a stronger domain prior can improve our method. To this end, in manuscript 4 (Appendix D) we focus on the setting where auxiliary public data are available and leverage features from classifiers trained on these public data in place of random Fourier features. We evaluated our method on the CelebA and Cifar10 image datasets, which up to that point have been too complex for DP data release methods without the help of public data.

3

SUMMARY OF RESULTS

In this chapter, we give an overview of the presented Manuscripts, following the order of publication. We begin with our work in interpretable DP machine learning (Manuscript 1) in Section 3.1. Then we turn to DP generative models (Manuscripts 2 and 3) in Sections 3.2 and 3.2.3, and our follow-up work employing auxiliary public data (Manuscript 4) in Section 3.3 .

3.1 INTERPRETABLE DP PREDICTION

In Manuscript 1, we discovered the detrimental effect of differentially private training on interpretability in deep neural networks. We investigated this phenomenon on MNIST [89], FashionMNIST [163] and the Henan Renmin Hospital Dataset [93, 102]. We trained small DNN classifiers on each dataset, both with and without DP guarantees, and applied the Integrated Gradient [148] and SmoothGrad [142] gradient attribution methods. We compared the feature attributions across models and found that the private models yielded noisy results that were hard to interpret. So, in addition to the privacy-utility trade-off that is always present in the application of DP, we found that interpretability is a third dimension of this trade-off.

3.1.1 Method

In response to this finding, we proposed *locally linear maps* (LLM), a method that, unlike DNNs, could offer good performance along all three axes. As defined below, LLMs consist of a collection of M linear maps g_m^k for each class k. Each linear map corresponds to a part of the input space that belongs to the given class. To classify a datapoint, its similarity with each linear filter is computed to find the best match, and if no differentiability were required, the model would just assign the class $k = \arg\max_k g_m^k(\mathbf{x})$ according to the most highly activated filter. In order to make this process differentiable, we instead aggregate all filters for each class in a class score $f_k(\mathbf{x})$, which computes a softmax weighted sum of all filter activations per class, governed by an inverse temperature parameter β. The differentiable class prediction $s(\mathbf{x})$ is a softmax of these class scores:

$$s_k(\mathbf{x}) = \frac{\exp f_k(\mathbf{x})}{\sum_{k=1}^{K} \exp f_k(\mathbf{x})}, \tag{3.1}$$

$$f_k(\mathbf{x}) = \sum_{m=1}^{M} \sigma_m^k \, g_m^k(\mathbf{x}), \tag{3.2}$$

$$\text{where } g_m^k(\mathbf{x}) = \mathbf{w}_m^{k\top}\mathbf{x} + \mathbf{b}_m^k, \tag{3.3}$$

$$\text{and } \sigma_m^k(\mathbf{x}) = \frac{\exp\left[\beta \cdot g_m^k(\mathbf{x})\right]}{\sum_{m=1}^{M} \exp\left[\beta \cdot g_m^k(\mathbf{x})\right]}. \tag{3.4}$$

The model thus functions like a mixture of experts [75] for each class, where each expert is a linear model, and approximates a Voronoi diagram over the input space around the filter locations.

LLMs can be interpreted as a linearization of DNN classifiers. An LLM could thus also be obtained from an existing classifier by identifying key points in the input space and performing first-order Taylor approximations at those inputs, matching the local linearization which attribution methods perform, which would yield a collection of linear maps. Since it is hard to design this process in a privacy-preserving way, we opted to train the model from scratch instead. However, this perspective illustrates how LLMs are particularly suited to attribution methods, since both perform the same abstraction of a more complex model. Whereas the linearization on DNNs may only hold approximately in a small area around the input space due to the high non-linearity of DNNs, for sufficiently high values of β LLMs are actually nearly linear locally.

In order to obtain better private utility on high-dimensional data, we proposed to use random projections [82] as a data-independent dimensionality reduction method. The projection matrix from the data dimension d to a target dimension d', where each entry is drawn from $\mathcal{N}(0, 1/d')$, is known to nearly preserve the respective distances between points in the data space when mapping them to the embedding space [80]. We find that they significantly improve accuracy at the same levels of privacy by reducing the number of trained weights by a factor d'/d. However, this comes at some cost to the interpretability of the LLM features, introducing another factor in the privacy-utility-interpretability trade-off.

LLM models provide both *local* and *global* attribution-like explanations for classification. Given an input, the most highly activated linear filters for each class constitute an explanation of the model behavior at that point, since it is locally approximately linear. By design, these filters match what a gradient attribution method applied with respect to each class logit would retrieve. Furthermore, the model provides global interpretability, since considering the collection of filters for each class provides a simple overview of what features the

model is sensitive to, as long as the total number of filters remains sufficiently small.

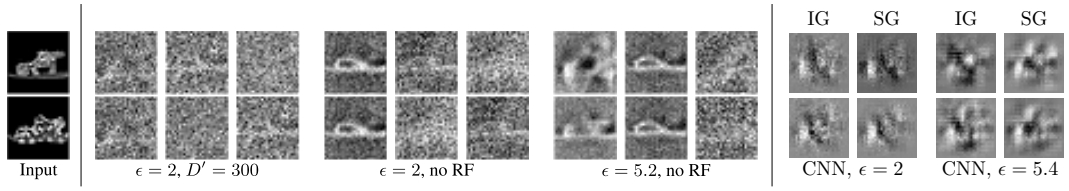


Figure 3.1: For 2 test inputs (left) and 3 different DP-LLM setups (groups of 3 columns), we show the 3 highest activated filters in descending order. We look at the default setting with random filters at $D' = 300$ and $\epsilon = 2$ (center left), the same setting without random filters (center), and in a lower privacy $\epsilon = 5.2$ setting (center right). Attribution plots of DP-CNNs at matching privacy levels with the same input are shown for comparison (right). (Adapted from Fig. 2 in Manuscript 1.)

3.1.2 Evaluation

To evaluate our method, we trained LLMs and reference DNNs on MNIST, FashionMNIST, and a tabular dataset of hospital admissions at varying levels of privacy. We then visually compared the inherent LLM attributions with the gradient attributions of the DP DNN models. Figure 3.1 illustrates our finding that LLM filters retained interpretability at high privacy, while gradient attributions of the DNN classifier were hard to interpret.

We find that for the small image datasets $M = 30$ filters per class struck a good balance between the needs for model expressiveness and low parameter count due to DP. Adding a random projection to reduce the input dimension from 784 to 300 significantly increased the test accuracy of LLMs from 92.2% to 94.2% at $\varepsilon = 2$ but at the cost of diminished interpretability. With a sufficient number of linear maps for each class, the model incurred only a slight loss in accuracy compared to DNN classifiers on the datasets we considered. On MNIST for instance, LLMs achieved a test accuracy of 94.2% with $(2, 10^{-5})$-DP and 91.8% with $(0.5, 10^{-5})$-DP, which constituted a loss of 0.8% and 0.2% compared to state-of-the-art methods [1, 121].

The difference between LLMs and DNNs depends on the complexity of the data, and DNNs would easily outperform LLMs in the non-DP setting. However, since we were only interested in the DP setting, where the capacity of DNNs is a fraction of its non-DP counterparts, DP-LLMs achieved comparable results despite being the simpler model.

The interpretability of DP-LLMs is less strongly affected by the gradient noise of DP-SGD than is the case for DNNs, but it still deteriorates with increasing level of privacy. Therefore, DP-LLMs are not a perfect solution to the privacy-utility-interpretability trade-off in DP training. In the following chapter, we

discuss a separate topic in DP machine learning, which allows one to circumvent the need for DP training entirely by producing training data with built-in DP guarantees.

3.2 DP DATA GENERATION VIA RANDOM FOURIER FEATURE EMBEDDINGS

In Manuscript 2, we turn to the task of differentially private data release, which aims to provide a DP proxy dataset that is similar to the sensitive real data and can be used in its place without violating privacy. Having such a proxy dataset available solves many practical issues in DP machine learning. The data can be shared freely, and there is no need to track any further privacy loss due to DP's post-processing invariance. There is also no constraint on model choice and training time, and no noisy gradient updates, which implies that the problems we observed with attribution methods in the previous section also disappear. In short, DP data release is an attractive concept because it separates out the handling of privacy concerns, and no other part of the workflow needs to change.

We approached DP data release from a deep learning perspective by training a DP deep generative model in order to produce private samples, which could then be used to construct a proxy dataset. In prior research, several deep generative modeling approaches had also been applied in the context of differential privacy but had shown limited success. The initial approaches had used GANs [33, 155, 164] and VAEs [4, 34, 120, 151] trained with DP-SGD to generate tabular data and low-dimensional black-and-white images such as MNIST [89] and FashionMNIST [163]. In low-privacy settings with $\varepsilon \approx 10$, these methods produced recognizable images of digits and clothing items, advancing the state of the art. The major shortcoming of these approaches was that they did not produce reasonable results in high-privacy settings around $\varepsilon \leq 1$, as $\varepsilon = 10$ does not offer a tangible privacy guarantee. Furthermore, the methods left room for improvement both in that they failed to scale to more complex datasets and in the significant gap in quality between DP and non-DP results. The quality of a proxy dataset was measured by training a collection of classifiers on proxy data and reporting their test accuracy on the real data. Whereas in the non-DP setting the proxy data were almost as good for training as using the real data with a 1-2% decrease in accuracy, using DP proxy data led to a significant drop in accuracy of more than 20% [33].[1]

1 Recent approaches based on diffusion models [45, 59] achieve significant improvements on all the problems discussed, but did not exist at the time of this work. They are discussed in Sec. 4.2.

3.2.1 Method

Our goal was to develop a method that could provide useful results at more reasonable DP guarantees, addressing the central weakness of prior approaches. To realize this, we sought a way to avoid DP-SGD altogether, as its iterative DP releases applied to a large generative model led to an unfavorable privacy-utility trade-off.

We achieved this goal by training our generative model with an approximation of a Maximum Mean Discrepancy objective [66]. In this objective, we compute a static high-dimensional mean embedding of the entire data set using random Fourier features [126] as an embedding function, and then match this embedding with an embedding based on generated data. This approximated objective turns out to be particularly suited to DP training, as it neatly divides into one data-dependent term, the dataset embedding $\hat{\boldsymbol{\mu}}_P$, and one model-dependent term, the generated data embedding $\hat{\boldsymbol{\mu}}_{Q_{\boldsymbol{\theta}}}$. The generated data is produced by a generator model with parameters $\boldsymbol{\theta}$. Given that the embedding is an average across the entire data set $\mathcal{D}$, it has significantly lower sensitivity than minibatch-based releases. The random Fourier features have fixed norm $\|\widehat{\phi}(\cdot)\|_2 = 1$ by construction, leading to a sensitivity of $\Delta_{\boldsymbol{\mu}} = 2/|\mathcal{D}|$. After releasing the dataset embedding once with the Gaussian mechanism $\widetilde{\boldsymbol{\mu}}_P = \widehat{\boldsymbol{\mu}}_P + \mathcal{N}(0, \Delta_{\boldsymbol{\mu}}^2 \sigma^2 I)$, we obtain the DP objective shown in Equation 3.5, which we can continuously evaluate with different generated data embeddings for an arbitrary number of times at no further privacy cost to optimize $\boldsymbol{\theta}$. We named this method, summarized in Algorithm 2, *Differentially Private Mean Embedding with Random Features* (DP-MERF).

$$\widetilde{\mathrm{MMD}}^2_{rf}(P, Q_{\boldsymbol{\theta}}) = \|\widetilde{\boldsymbol{\mu}}_P - \widehat{\boldsymbol{\mu}}_{Q_{\boldsymbol{\theta}}}\|_2^2 \tag{3.5}$$

Low sensitivity and the single DP release enable good results under stronger privacy guarantees than were possible for iterative methods. For labeled data, we compute separate embeddings for each class and concatenate them into one large vector. This amounts to computing a kernel product of the data kernel and the label, for which we used a first-order polynomial kernel. While this does not affect the sensitivity $\Delta_{\boldsymbol{\mu}}$ of the release, it does increase the dimensionality of the released vector, leading to the addition of more noise overall, compared to the unlabeled setting. In case of class imbalance, we compute the vector of class counts $\mathbf{m}$ and make it DP in a second release $\widetilde{\mathbf{m}} = \mathbf{m} + \mathcal{N}(0, \Delta_m^2 \sigma_m^2 I)$. We then use this vector to reweigh the objective such that each class contributes equally to the loss, and small classes are not neglected. Since this is a relatively short vector of large counts and the release has low sensitivity $\Delta_m = \sqrt{2}$, it can be released with a large σ_m and thus comes with little additional privacy cost.

DP-MERF falls into the category of data summary perturbation approaches to DP deep learning as described in 1.2.1. In contrast to gradient perturbation

Algorithm 2 DP-MERF

Require: Dataset $\mathcal{D}$, and a privacy level (ϵ, δ), generator parameters $\boldsymbol{\theta}$
Ensure: (ϵ, δ)-DP input output samples for all classes
Step 1. Given (ϵ, δ), compute the privacy parameter σ
Step 2. Release the mean embedding $\widetilde{\boldsymbol{\mu}}_P$
Step 3. Train the generator
$\boldsymbol{\theta}^* = \arg\min_{\boldsymbol{\theta}} \widetilde{\mathrm{MMD}}^2_{rf}(P, Q_{\boldsymbol{\theta}}) = \arg\min_{\boldsymbol{\theta}} \left\| \widetilde{\boldsymbol{\mu}}_P - \widehat{\boldsymbol{\mu}}_{Q_{\boldsymbol{\theta}}} \right\|_F^2$
Step 4. Sample proxy dataset from generator $\mathcal{D}' \sim Q_{\boldsymbol{\theta}^*}$

methods such as DP-SGD, this has the advantage that the choice of generator model and training setup are not constrained by DP. Unlike for DP-SGD, batch norm and regular minibatch sampling may be used for training, and a high number of parameters in the model does not negatively impact the privacy-utility trade-off.

In a proof accompanying the method, we provided a bound on the expected absolute error between the non-private kernel-based MMD estimator and our noisy random Fourier feature approximation. By using the triangle inequality, we obtained two terms which, respectively, account for the error due to the random feature approximation and due to DP release. We note that, varying only the number of features $d > 0$, the random feature error scales in $O(\sqrt{1/d})$, while the DP release error bound scales linearly in $O(d)$. For a given privacy budget and dataset size, the global minimum in this bound provides a heuristic on how to choose d to minimize the overall expected error.

3.2.2 Evaluation

We tested DP-MERF on MNIST and FashionMNIST, on several tabular datasets, and on an illustrative two-dimensional dataset of mixtures of Gaussians. The tabular datasets included both imbalanced and heterogeneous data. To evaluate the quality of our trained models, we had them generate proxy datasets, which we used to train a number of simple classifier models from the scikit-learn library [119]. The trained models were tested on the real test set to indicate how useful the private proxy would be in place of the real data in downstream classification tasks. In terms of this downstream metric, DP-MERF yielded competitive results, especially in the high privacy regime. We illustrate this in Figure 3.3, where we show the average downstream accuracy as a function of the size of the synthetic training set. This not only showed that MP-MERF outperforms GAN-based methods despite a much tighter privacy budget ($\varepsilon = 1$), but also provided insight about the *effective dataset size* for each method, which we defined as the point at which adding more synthetic samples no longer improved accuracy. Although both real data and synthetic data generated

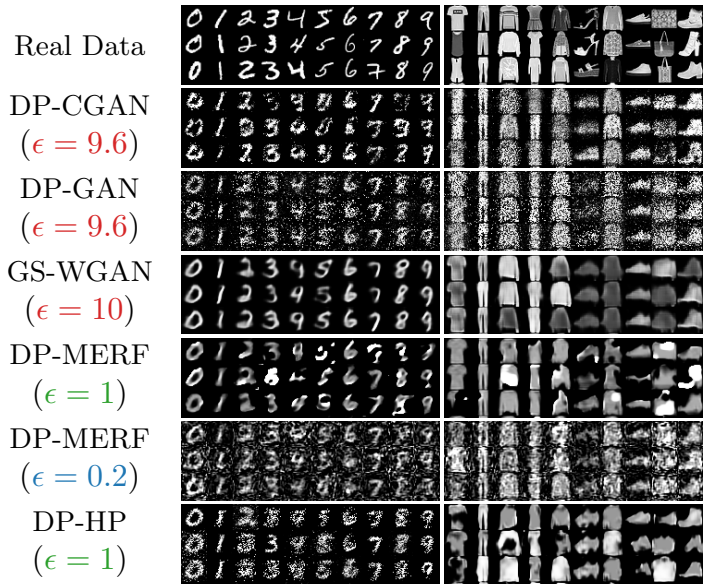
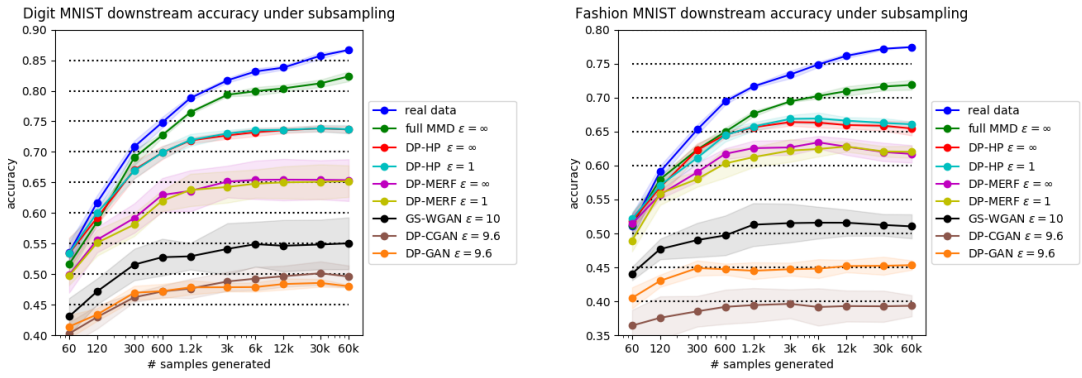


Figure 3.2: Generated MNIST and FashionMNIST samples from DP-MERF, DP-HP and comparison models with different levels of privacy. (Adapted from Fig. 2 in Manuscript 2 and Fig. 4 in Manuscript 3.)



(a) MNIST **(b)** FashionMNIST

Figure 3.3: We compare the real data test accuracy as a function of training set size for models trained on synthetic data from DP-HP and comparison models. The confidence intervals show 1 standard deviation. (Adapted from Fig. 3 in Manuscript 3.)

from a non-DP MMD embedding showed improvements up to the full dataset size of 60,000, DP-MERF and GAN-based methods stopped benefiting from additional data after around 3,000 to 6,000 samples on MNIST, indicating that any additional samples were mostly redundant. On FashionMNIST DP-GAN and DP-CGAN showed an even smaller effective dataset size between 300 and 600 samples, while GS-WGAN and DP-MERF produced redundant data after around 1,200 and 3,000 samples, respectively.

While other models, such as GS-WGAN [33] shown in Figure 3.2 produced samples of higher visual quality than DP-MERF, their downstream utility for classification was lower. This apparent inconsistency is explained by an experiment on 2D Gaussian mixtures data, which we display in Figure 3.4. It shows that our method managed to cover all modes in the distribution, while the GAN-based method DP-CGAN [155] dropped modes in high privacy settings.

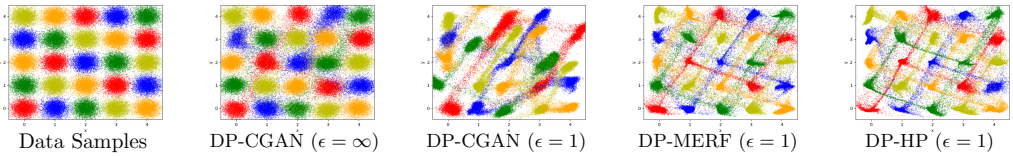


Figure 3.4: From left to right: 1): Data samples drawn from a Gaussian Mixture distribution (each color represents a class) to be learned. NLL denotes the negative log likelihood of the samples given the true data distribution. 2): CGAN ($\epsilon = \infty$) generated data closely matches the real data. 3): DP-CGAN at $\epsilon = 1$, misses some modes, which is reflected in the higher NLL. 4): Synthetic data samples generated by DP-MERF at $\epsilon = 1$. DP-MERF captures all modes at $\epsilon = 1$, which is also reflected in NLL. 5): DP-HP shows a similar behavior as DP-MERF, but improves NLL further. (Adapted from Fig. 1 in Manuscript 2 and Fig. 2 in Manuscript 3.)

3.2.3 Improving DP data generation with Hermite polynomials

As an alternative to random Fourier features, we also explored kernel approximations based on Hermite polynomials. Our approach was based on the *Mehler formula* [104], which allows us to approximate a one-dimensional Gaussian kernel with a weighted sum of Hermite polynomials. These features are ordered and require orders of magnitude fewer features to approximate a kernel to the same quality. Since the number of features directly impacts the noise magnitude required for DP release, HP features are a promising alternative to RFF.

In an evaluation equivalent to the one performed for DP-MERF, DP-HP obtained significant improvements over DP-MERF both on 7 out of 8 tabular datasets and on MNIST and FashionMNIST. For example, on MNIST, the

average downstream accuracy over 12 models improved from 65% for DP-MERF to 73% for DP-HP. In figure 3.3, we see that DP-HP also exhibited a larger effective dataset size than DP-MERF on MNIST, showing improvements up to a dataset size of around 12,000 samples.

The main drawback of the DP-MERF and DP-HP approaches was that while yielding good results on tabular and MNIST type data, they did not scale well to more complex datasets. The data-independent features are a key factor limiting the method, as they do not incorporate any kind of domain prior.

3.3 SCALING DP DATA GENERATION TO COMPLEX DATA THROUGH PUBLIC FEATURES

Past research on DP deep learning has shown that access to auxiliary public data, which is similar to the private data to some extent, but does not require privacy-preserving treatment itself, leads to significantly better performance [1, 37, 87, 94, 105, 156, 169, 170]. The key advantage is that this data may be used to pretrain portions of the machine learning model without exhausting the privacy budget. Through pre-training, the model gains a strong domain prior, which is necessary for complex data domains such as images or natural language. The model can then be fine-tuned on the private data using far fewer iterations than would be necessary when training from scratch.

3.3.1 Method

We used this finding to improve our method based on mean embeddings and scale it to more complex datasets. Rather than using public data to pre-train our deep generative model, we used deep learning classifiers trained on public data as feature extractors in place of the random Fourier features. Since the classifiers were trained to distinguish data from the given domain (e.g. natural images), they extract features from the data, which are relevant for this particular domain.

While requiring available public data in the same domain somewhat limited the scope of our method, we showed that in the domain of natural images, where public data is readily available at a large scale, this approach led to significant improvements. Building on Santos et al. [135], who had previously explored this approach in the non-private setting, we constructed dataset embeddings from pre-trained VGG19 and ResNet18 image classifiers by gathering the first two moments of these features. As in DP-MERF, we privately released this embedding and trained a generator model in a minibatch fashion to match this embedding using an MMD objective. Following [135], we used a moving average loss to improve stability.

After finding that the method tended to overfit to the embedding on some datasets, we designed a DP early stopping method, which uses a second private embedding as a proxy for the FID score and halts training when the proxy loss increases. We allocated a small amount of our privacy budget to this stopping criterion. Since the proposed method used perceptual features instead of random ones, we named it *DP Mean Embeddings with Perceptual Features* (DP-MEPF).

3.3.2 Evaluation

We conducted experiments on the 28×28 pixel grayscale datasets MNIST and FashionMNIST (784 features), as well as the significantly more complex color image datasets Cifar10 at 32×32 resolution (3072 features) and CelebA at 32×32 and 64×64 resolution (12288 features). We evaluated our models based on the accuracy of the downstream classification on the labeled datasets and Fréchet Inception Distance (FID) on the color images. Due to the lack of other published methods that used auxiliary public data for DP generative modeling, we created our own baseline by pretraining a DC-GAN on ImageNet and fine-tuning it with DP-SGD on Cifar10 and CelebA. Furthermore, we compared DP-MEPF with existing methods, which did not use auxiliary data. The latter comparisons were not on equal footing since we accessed auxiliary data, but their purpose was to highlight the leap in scale that using these data enabled. DP-MEPF outperformed all of our baselines, with particularly stark contrast on the more complex color image datasets. On CelebA (32×32) for instance, at $(\varepsilon = 1, \delta = 10^{-6})$-DP, DP-MEPF achieved an FID score of 17.2, a significant improvement over both the 81.3 score of our pre-trained GAN baseline and the 71.8 score of DP Diffusion Models, the highest score from a method that does not use auxiliary data. A comparison of images generated by different methods is shown in Figure 3.6. Cifar10 was the most challenging dataset that we tested, as it was both more diverse and smaller than CelebA. Here, at $(\varepsilon = 1, \delta = 10^{-5})$-DP, our method fared slightly worse with an FID score of 43.0, but it was still significantly better than the GAN baseline with a score of 74.9. On labeled Cifar10, DP-MEPF showed worse scores of 54.0, due to computing a separate embedding per class and thus having to privatize a higher-dimensional overall embedding. The downstream accuracy on Cifar10 also decreased in high privacy settings, falling from 53% at $(\varepsilon = 10, \delta = 10^{-5})$-DP to only 33.2% at $(\varepsilon = 1, \delta = 10^{-5})$-DP. Generated images at different levels of DP are shown in Figure 3.5.

Although we did test several other types of encoder networks, including ResNet [69], ConvNext [98] and EfficientNet [152], VGG [141], and in particular VGG19 significantly outperformed alternative models. The reason for this remains unclear. While larger embeddings tended to perform better within the same model class, ResNet101 and ResNet152 models yielded FID scores that

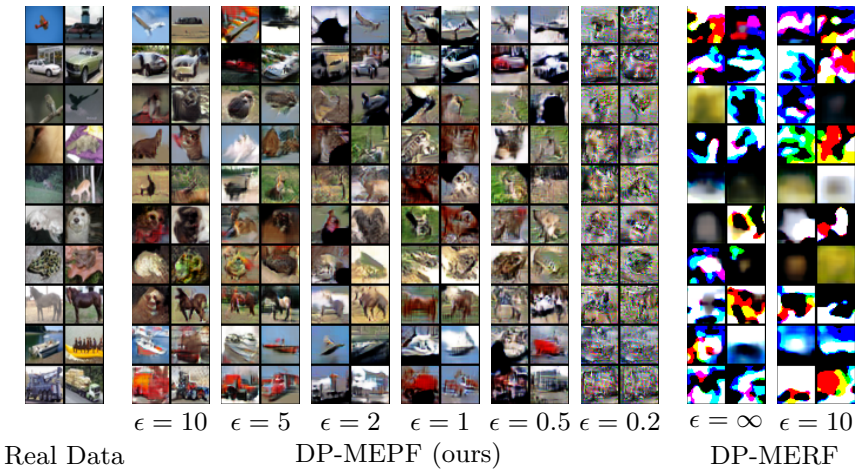


Figure 3.5: Labeled Cifar10 samples from DP-MEPF and DP-MERF. Each row corresponds to a different label. (Adapted from Fig. 5 in Manuscript 4.)

were more than 20 points higher than VGG19 despite having an equal or higher number of features.

Our experiments demonstrated that domain-specific features allowed our method to scale to datasets of higher complexity that were out of reach of generative models that do not use public data. Thanks to the single mean embedding release, we still obtained good results in the high privacy setting, although the linear scaling of embedding size in the number of classes became more of a problem than it was in DP-MERF, as embeddings were about one order of magnitude larger.

We further provided a bound on the approximation error of the MMD loss, which proves that the error introduced by the Gaussian mechanism decays at a rate of at least $O(1/n)$ with increasing number of samples. The approximation error between empirical MMD and the true distribution MMD is known to decrease at a significantly slower rate of $O(1/\sqrt{n})$ [51], indicating that for a sufficient sample size, the approximation error due to privacy becomes negligible. This proof matched our empirical findings, as utility above a certain ε value depending on the dataset was nearly unaffected by privacy and equaled the non-DP utility.

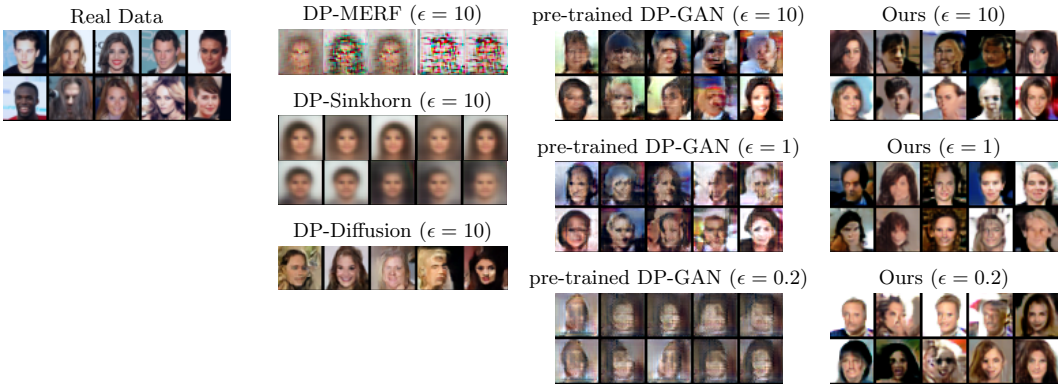


Figure 3.6: Synthetic 32×32 CelebA samples generated at different levels of privacy. Samples for DP-MERF and DP-Sinkhorn are taken from Cao et al. [23] and DP-Diffusion samples are taken from [45]. The pre-trained GAN is our baseline utilizing public data. Even at $\epsilon = 0.2$, DP-MEPF yields samples of higher visual quality than the comparison methods. (Adapted from Fig. 2 in Manuscript 4.)

4

DISCUSSION

In this chapter, we put the presented work into a broader perspective by discussing relevant related and follow-up work, as well as limitations of our approach and potential avenues for future research. Grouping the chapter into two sections according to theme, we begin with our work on interpretability in DP and then discuss our work on DP generative models.

4.1 INTERPRETABLE DP MACHINE LEARNING

4.1.1 Related and follow-up work in context

Besides our work, the trade-off between differentially private training and interpretability in DL models has received little direct attention in research so far. This may stem from the fact that DP deep learning models currently do not see much practical use and therefore the problem of limited interpretability is largely theoretical. If DP deep models are adopted in practice, the problem will likely become more apparent and need to be revisited.

A closely related topic is the question of how model explanations can leak information about the model itself [5, 106] or its training data [61, 138]. In this domain, Shokri, Strobel, and Zick [138] showed that gradient attribution explanations can be used for membership inference attacks even in *black-box* settings where an adversary does not have access to the trained model. In response, Patel, Shokri, and Zick [118] designed a method to make such explanations DP. While they provided guarantees on the quality of their explanation relative to the non-DP counterpart, a limitation of their approach is that it only provides DP for an *explanation dataset*, which may be used to generate explanations. If the model is DP, the DP guarantee with respect to the training data is amplified by their method, but for non-DP models, no concrete guarantee for the actual training data can be provided. This is expected, as the explanations are applied after training and do not quantify the sensitivity of the model with respect to the training data. So even in this setting, where the adversary only accesses the model through explanations, the one reliable protection of the training data remains DP model training.

Similarly to our approach in DP-LLMs, efforts to combine interpretable machine learning with differential privacy are also presented in inherently more interpretable models such as boosting [112] and decision trees [7, 56].

4.1.2 Limitations and future directions

While choosing inherently interpretable DP models or training non-private models on DP proxy data are valid answers to the trade-off between privacy and interpretability in DNNs, there are other avenues worthy of exploration. Since the publication of our study, the state-of-the-art in DP deep learning has advanced significantly, in particular through application to larger datasets and the use of public data, as well as better hyperparameter choices such as extremely large minibatch sizes. Given the improved performance shown, for example in [37, 134], it stands to reason that these new models should learn more useful features than those investigated in our work. It would thus be interesting to study whether the problem of lacking interpretability persists in these models or whether it disappears thanks to higher quality features. If the problem is still present, however, DP-LLMs are unlikely to offer a solution, as its shallow features will not scale well to data of this complexity.

Although gradient attribution methods are the most actively researched approach to DNN explanation according to Zhang et al. [172], they are by no means the only ones. It is not clear from our research how private classifiers differ from non-private ones under feature visualization [100, 114], concept activation vectors (TCAV) [83], or surrogate models such as LIME [130]. Investigating these differences is especially interesting now that DP classifiers have been trained on natural images at ImageNet scale [37, 134], since this is the target domain of much of the existing interpretability literature, and certain methods such as TCAV and LIME are designed for this domain, in particular, making use of natural concepts and image segmentation, respectively.

4.2 DP DEEP GENERATIVE MODELS

4.2.1 Related and follow-up work in context

When DP-MERF was published, it outperformed existing state-of-the-art GAN-based DP generative models [155, 164]. Since then, a large variety of new methods for DP data release have been developed, which we will discuss in the following.

In the one extension to DP-MERF not developed by us, Liew, Takahashi, and Ueno [96] add a critic to the model, which aims to increase the training loss by adversarially re-weighing the importance of the randomly sampled

frequencies. The weighting is parameterized as a Gaussian probability density function over the frequencies with learned parameters μ and σ. Decreasing the weight of frequencies with small losses and increasing the weight of those with large losses is intended to favorably rescale the loss landscape, thus simplifying the optimization. The results in the paper, however, show only small improvements within the margin of error (cf. Table 1 in [96]) and are based on incomplete empirical evaluation,[1] so it is not clear if the method constitutes a real improvement.

Despite the critique raised by us and others [160], DP generative models continue to be presented with large values chosen for ε. For example, Bie, Kamath, and Zhang [14] significantly improve the performance of DP-GAN models and show results for $\varepsilon = 1$ and $\varepsilon = 10$, but still only outperform DP-MERF in the lower privacy setting. Works such as DP-Sinkhorn [23] and DPD-fVAE [120] only publish results for $\varepsilon = 10$. Notably, recent work on DP diffusion models [45] has set a new state-of-the-art for both high and low privacy settings, publishing results for $\varepsilon \in \{0.2, 1, 10\}$ on MNIST, which will hopefully help to place more focus on the high privacy setting in the future. However, on the more challenging CelebA dataset, the same work only reports results for $\varepsilon = 10$. Diffusion models (DM) [147] have surpassed GANs as state-of-the-art generative models in the non-private setting. Dockhorn et al. [45] argue that they are also a particularly good fit for DP training with DP-SGD because DMs already incorporate noisy updates into their training and generate samples over multiple denoising steps, where each step is simple and easy to learn compared to the generative process of, for example, GANs. Through pre-training diffusion models on public data, Ghalebikesabi et al. [59] have further improved on these results and scaled them to the low resolution natural image dataset Cifar10 [86] and the medical histopathology dataset Camelyon17 [12]. In the low privacy setting, this method outperforms DP-MEPF on Cifar10 by a significant margin, obtaining an FID of 15.1 compared to 37.0 for our method at $\varepsilon = 5$. At $\varepsilon = 1$ the difference decreases with FIDs of 25.2 and 43.0, respectively, indicating that the diffusion model is more strongly affected by the necessary increase in noise. Ghalebikesabi et al. [59] do not report results for smaller privacy budgets. On MNIST and Camelyon17, the experiments were performed at $\varepsilon = 10$. Lyu et al. [99] have developed a more compute-efficient alternative to [59] by using a smaller Latent Diffusion Model [131] and only fine-tuning the attention modules and a conditioning embedder. This significantly reduces the number of trained parameters by around 95% and GPU hours by over 98%. While the smaller model performs worse in low DP settings, with FID 19.2 compared to 9.8 for

1 Experiments only show "FID" and "KID" [15, 70] scores for MNIST and FashionMNIST, unconventionally not using features from the Inception DNN [150], but from a LeNet [90] classifier. The visual similarity between the PEARL and DP-MERF samples does not support the claimed improvement in the FID score. Down-stream accuracy evaluation is limited to two tabular datasets, Adult and Credit.

[59] at $\varepsilon = 10$ on Cifar10, reducing the number of parameters pays off in high DP settings, where it matches the performance of [59] at $\varepsilon = 1$ with an FID score of 25.2.

Another promising approach to DP data release is to forego training of a generative model and instead directly create a small dataset of private samples. These methods are based on recent non-DP work on dataset distillation (or condensation) which aims to generate a small *support set* of samples that behave like the full dataset when training specific models. Chen, Kerkouche, and Fritz [32] adopt the dataset condensation approach of [173], which optimizes the support set by matching the gradients between two versions of the same model, where one is trained with real data and the other is trained on the support set. The gradient of the model trained on real data is released privately with DP-SGD. In principle, the training task on which the models should match can be freely chosen, but published experiments use classification.

Vinaroz and Park [161] construct DP-KIP, a differentially private version of the kernel inducing points method (KIP) [110, 111]. This method uses the neural tangent kernel [76] of an untrained encoder network as a proxy for the behavior of the model during training and optimizes the data on a kernel ridge regression objective. In DP-KIP, the gradient update to the support set can be computed per training sample and thus can be clipped to have bounded sensitivity and released with DP-SGD. This makes the dimension of the private release independent of the size of the DNN and instead depends on the size of the support set, which is often significantly smaller. As a result, DP-KIP obtains better results in downstream classification tasks.

A further detail that sets these methods apart from generative models is that the generated datasets are created specifically for a downstream task through the training objective. This means the generated data is trained to perform like the real data on this one specific task, but need not resemble the real data in other aspects. For instance, the images generated by DP-KIP look like noise and yield a bad FID score, but can still be used to train an accurate classifier. This focus on a specific task makes the method more narrow, but may also be desirable in applications where the proxy data is intended to be used for a specific, well-defined task which is known in advance.

Lin et al. [97] also circumvent the need for training a generative model and instead construct a DP dataset by repeatedly querying large *foundation models* such as Stable Diffusion [131] through an API and selecting samples through a DP evolutionary algorithm. Unlike [32, 161], this approach implicitly relies on public data that trained the foundation models. On datasets that contain images close to the training data of the foundation model, such as CIFAR10, the method achieves excellent sample quality at a small privacy budget with $\varepsilon < 1$. However, the method performs significantly worse on samples that require a domain shift away from the training data, such as Camelyon17.

In summary, approaches to DP data release based on training deep generative models via DP-SGD have achieved impressive improvements thanks to the introduction of diffusion models and auxiliary data [45, 59]. However, these methods still fare considerably worse in high DP settings due to the high privacy cost of training a large model. Alternative methods that avoid training a generative model [32, 161] or the use of DP-SGD all-together [97] lose less utility in high DP settings due to more economic DP releases, but face other limitations such as poor scaling to complex data [32, 161] or strong dependence on similar public training data [97].

4.2.2 Limitations and future directions

Our presented line of embedding-based DP generative models faces several challenges. We begin this section by discussing two general unsolved problems in the field of DP machine learning, which manifest in our approach as well, fairness and the role of public data, before going into topics that are specific to our approach.

Research has discovered that there is a trade-off between DP and the fair representation of minorities. This trade-off has since become an active field of study [8, 29, 36, 43, 53, 77, 101, 136, 158, 159, 165, 174]. It has both been shown that fairness in non-DP models can make minorities more susceptible to membership inference attacks [29] and that unfair biases can be amplified when a model is trained with DP guarantees [8]. This relationship follows intuitively from the definition of DP, since the influence of minorities and outliers in the data on the output must be limited in order to preserve their privacy. It nevertheless poses a severe problem for DP, especially in the realm of DP data release, as it interferes with the goal of separating DP release from the ML workflow by providing an accurate proxy dataset. Biases which may be corrected for in the real data are harder to correct in proxy data when the proxy data for minorities is of lower quality. Solutions may instead require greater effort to adequately represent minorities during data collection.

The use of auxiliary public data in DP-MEPF and DP deep learning in general creates several unsolved questions about responsible data use which have recently been formulated in a position paper by Tramèr, Kamath, and Carlini [157], which we briefly touched on in Section 1.2.2 and will elaborate here. One primary concern is that what is generally treated as public data may nonetheless contain sensitive information, as large text or image datasets are typically created by programs scraping the Internet without human oversight. For this reason, they may contain data that was released involuntarily or shared exclusively for a specific context. As such, DP deep learning may still cause harm by revealing information that was never intended to be public. A second concern is that for many application domains public data may not be as readily available as for generic natural images or text. This means that experiments in

these domains may promise better performance than can be obtained in realistic settings, where the shift between a public source domain and the private target domain is considerably larger. Following these arguments, the responsible application and effectiveness of public data in DP face limitations that are not immediately apparent. Progress on this issue requires careful development of consensus data collection practices both for research and for deployment of DP methods using auxiliary data. Irrespective of these hurdles, the authors in [157] agree that public data has played a vital role in advancing DP deep learning. The most recent works on DP generative modeling [45, 59, 97], have started to take some of the criticisms into account by including the Camleyon17 dataset in experiments, which requires a significantly larger domain shift than datasets like Cifar10 or CelebA.

In addition to these two broader issues in DP deep learning, we also now discuss three limitations of our proposed method which warrant further investigation.

First, the use of the product kernel for labeled datasets, which computes a separate embedding for each class, is a conceptually simple way to produce labeled data, but it comes with the downside of creating overly large embeddings and thus a worse signal-to-noise ratio. This became particularly apparent in the Cifar10 experiments with DP-MEPF, where the labeled data yielded significantly worse results in the high-privacy regime due to the ten times larger embedding it required. This prevents the method from scaling to datasets with higher class counts, such as ImageNet (1000 classes). A possible solution to the issue may be found in hybrid embedding approaches, where one part of the embedding is shared across all classes to capture the common features and a second, smaller part of the embedding is computed per class and captures the distinguishing features of each class. In such an approach, only the second part of the embedding would scale in the size of the dataset, leading to a smaller total embedding. How the individual embeddings should be chosen and what weight should be assigned to each in the MMD objective are empirical questions left to future work. A potential starting point with an embedding based on DNN features would be to share lower-layer features across classes, under the assumption that simpler features will be widely shared across classes, and to collect features from higher layers per class, since those features can be expected to be highly differentiated between classes. However, brief preliminary experiments with this setup have not yet yielded improvements.

Second, although our application of DP-MEPF was exclusive to image data because it is the most common data domain for benchmarking DP deep generative models, the method is applicable to other domains as well, as long as public can be utilized to learn a strong domain prior. The study of other such data domains with active work on domain adaptation, for instance, natural language text data [127], audio [88], or social graphs [122, 166] would be a natural extension of our work. For image data, the abundance of pretrained

classifiers made them a natural first choice as feature extractors, but the method is neither limited to the domain of vision nor to the use of classifier models as encoders.

Third, while perceptual features have shown clear improvements over data-agnostic features, such as random Fourier features or Hermite polynomials, it is not clear which of the features are relevant. It is, in fact, likely that some of them are redundant or uninformative, as they were not directly optimized to encode the target dataset. Since the feature embedding must be released privately and the required noise scales with its dimensionality, there is a strong incentive to reduce embedding size by removing all uninformative features. However, estimating the importance of these features, especially in a privacy-preserving way, is a challenging problem that needs to be addressed in future research.

5

CONCLUSION

In this thesis, we have developed machine learning models for interpretable classification and data generation under strong differential privacy guarantees. By questioning the standard approach of training deep models with DP-SGD, we have found methods that achieve better utility at high levels of privacy and demonstrated that the most successful non-DP machine learning methods can be suboptimal choices in differentially private settings.

After identifying the trade-off between utility, privacy, and interpretability in deep classifiers, we developed *DP locally linear maps* as an inherently interpretable alternative to DP DNNs, which we discussed in Section 3.1. With experiments on MNIST, FashionMNIST, and the Henan Renmin Hospital dataset, we demonstrated that our model preserves better interpretability at the same levels of DP and utility. This satisfies our first research objective of improving interpretability in DP deep classification models.

Summarized in Sections 3.2 and 3.2.3, we addressed our second research objective of achieving DP data release without DP-SGD by designing *DP Mean Embeddings with Random Features* and its extension based on *Hermite polynomial features*. Our method is based on releasing dataset mean embeddings, which are then used to train a deep generative model via a maximum mean discrepancy objective. We evaluated our approach on several tabular datasets and small image datasets MNIST and FashionMNIST, measuring the quality of the generated proxy data through its usefulness for training models on downstream tasks. Compared to existing DP deep generative models, our approach yielded state-of-the-art utility at a significantly higher level of DP, maintaining good performance even in the high-DP regime at $\varepsilon \leq 2$.

In pursuit of our third research goal of improving DP data release methods with the help of public data, we used features of trained DNN classifiers to construct dataset embeddings with a strong domain prior, leading us to *DP Mean Embeddings with Perceptual Features*. As summarized in Section 3.3, we evaluated our method on the image datasets CelebA, Cifar10, MNIST and FashionMNIST and reported downstream accuracy and FID score as quality measures. With the help of public data, our method outperforms all comparison methods by a significant margin on the more complex datasets, including our baseline of a pre-trained GAN with DP fine-tuning. Through the completion of these individual objectives, we have made several significant contributions

to our overall research aim of developing DP machine learning methods for high-dimensional data in high-privacy settings.

As a final outlook, we now consider the general direction in which the field of DP deep learning is heading and the positioning and contribution of our work in these developments.

As described in [14], there are two schools of thought in DP deep learning research, based on different intuitions about how much established knowledge about non-DP deep learning still holds true in the private setting. The first approach assumes that the two settings resemble each other enough, such that the most advanced methods from non-DP deep learning will be optimal in the private setting as well, usually with minimal modifications and by employing DP-SGD as a straightforward means of DP release. The opposing perspective posits that introducing DP constraints changes the demands on model training so significantly that other methods, specifically selected for this setting, may be more successful. Motivated by the many apparent problems with DP-SGD, our work falls within this second line of thinking. Our approaches have achieved state-of-the-art results in competition with DP-SGD-based methods, demonstrating that alternative methods are viable. However, recent advances, such as DP diffusion models [45] and ImageNet level classification [37, 134], have shown that the inefficiencies of DP-SGD are not as much of a limiting factor as previous results suggested. Both approaches remain in close competition, but show different strengths. Given these trends, we expect that DP-SGD will ultimately become a strong default option for DP deep learning in settings where large amounts of data are available (both private and public). Tasks with limited data or a particularly small privacy budget will continue to require specialized solutions, such as the methods we have proposed in this work.

Despite its increasing popularity in research and recent breakthroughs, DP is currently only deployed by a handful of companies and institutions which possess both the required technical expertise and the vast scale of data that make it viable, such as Google [17, 52, 103], Apple [153, 154], Microsoft [42], Uber [79] and the US Census [2]. It does not see the widespread adoption that would be necessary to make privacy-preserving machine learning the norm. We believe that the practical application of DP is held back by four key factors: organizational overhead, prohibitive utility trade-off, a missing consensus on the correct choice for ε, and a lack of incentives. In the following, we discuss how each factor may be addressed.

OVERHEAD. The adoption of DP introduces several new steps to the existing machine learning workflow, adding a new layer of complexity to the management of machine learning projects. For each dataset, a privacy budget must be determined and divided among all individual data accesses. This requires careful planning, as each analysis performed on the data will expend valuable privacy budget. DP data release offers a way around many of these added

complexities by providing a DP proxy dataset, which may be used in place of the real data without further privacy cost or the need to change how ML projects are organized.

UTILITY. The perhaps most readily apparent issue holding back DP adoption is the significant drop in utility one incurs compared to the non-DP setting. For several years, the focus of DP ML research has been relegated to toy problems such as MNIST, while non-DP deep learning has tackled problems of ever-increasing complexity. Whether this performance gap will ultimately narrow is unclear and depends, for instance, on how important memorization is for successful deep learning models (as posited by [18, 55]). Nevertheless, the use of public data has enabled significant leaps in scale for DP deep learning to a point where practical application seems much more viable.

CONSENSUS ON ε. Putting DP into practice, requires a consensus on what values for (ε, δ) constitute a reasonable level of privacy for a given application. Despite the centrality of this question, only limited research has been conducted on the matter [3, 72, 91] and the published values in the DP ML literature vary wildly from $\varepsilon = 0.1$ to $\varepsilon = 10$. In light of the associated privacy-utility trade-off, one cannot expect to reach a definitive answer to this question through research alone. We believe that standards will likely develop with the introduction of legal privacy requirements over the coming years. Nevertheless, experts must help inform this debate so that a meaningful level of privacy becomes the norm and practitioners have a reliable baseline when deploying DP methods. This effort begins with the evaluation of new models at reasonable levels of privacy, so research results provide an accurate representation of what is possible in practice. Dedicated research into DP relaxations (e.g. [60, 85, 160]) or guarantees against weaker adversaries (e.g. [67, 92]) may also help in making larger ϵ-values feasible if they can prove additional types of privacy protection that follow from DP.

INCENTIVE. The final obstacle to the adoption of DP is the lack of legal and economic incentives. Ensuring data privacy necessarily comes at some cost of increased workload and diminished utility. Thanks to the aforementioned research, these costs are becoming manageable in some application domains and may decrease further over time, but organizations are unlikely to incur them without a concrete reason. Researchers can raise awareness of existing privacy issues to encourage responsible use of data [24–27, 139, 168], but the most effective incentives are likely to come from regulation. With the *Blueprint for an AI Bill of Rights* [137] in the United States and the *AI Act* [35] in the European Union, both the largest and the third largest economies in the world have submitted proposals for the regulation of AI and Machine Learning, and both target data privacy as a central issue. If these proposals are adopted into

laws with strong data privacy regulations, DP is poised to become a prime technological solution to meet such privacy demands.

As we can see, all four factors which have held DP machine learning back in the past are undergoing rapid changes that promise a partial solution. Our work has contributed both to the goal of reducing overhead through DP generative modeling and to the goal of better utility with improvements in interpretability and generated data quality, including explorations of auxiliary public data. Through our focus on the high-DP setting, we have provided results that will remain viable when a consensus on acceptable ε values is established. As such, DP machine learning is about to transition from a largely academic topic to a deployed technology. This shift will likely reveal a multitude of unexplored problems, promising an exciting future for DP in both research and application.

BIBLIOGRAPHY

[1] Martin Abadi et al. "Deep learning with differential privacy." In: *Proceedings of the 2016 ACM SIGSAC conference on computer and communications security*. 2016, pp. 308–318 (cit. on pp. 8, 12, 13, 21, 27).

[2] John M Abowd. "The US Census Bureau adopts differential privacy." In: *Proceedings of the 24th ACM SIGKDD international conference on knowledge discovery & data mining*. 2018, pp. 2867–2867 (cit. on p. 40).

[3] John M Abowd and Ian M Schmutte. "An economic analysis of privacy protection and statistical accuracy as social choices." In: *American Economic Review* 109.1 (2019), pp. 171–202 (cit. on p. 41).

[4] Gergely Acs, Luca Melis, Claude Castelluccia, and Emiliano De Cristofaro. "Differentially private mixture of generative neural networks." In: *IEEE Transactions on Knowledge and Data Engineering* 31.6 (2018), pp. 1109–1121 (cit. on p. 22).

[5] Ulrich Aïvodji, Alexandre Bolot, and Sébastien Gambs. "Model extraction from counterfactual explanations." In: *arXiv preprint arXiv:2009.01884* (2020) (cit. on p. 31).

[6] Jason Altschuler and Kunal Talwar. "Privacy of noisy stochastic gradient descent: More iterations without more privacy loss." In: *Advances in Neural Information Processing Systems* 35 (2022), pp. 3788–3800 (cit. on p. 14).

[7] Vahid R Asadi, Marco L Carmosino, Mohammadmahdi Jahanara, Akbar Rafiey, and Bahar Salamatian. "Private Boosted Decision Trees via Smooth Re-Weighting." In: *arXiv preprint arXiv:2201.12648* (2022) (cit. on p. 32).

[8] Eugene Bagdasaryan, Omid Poursaeed, and Vitaly Shmatikov. "Differential privacy has disparate impact on model accuracy." In: *Advances in neural information processing systems* 32 (2019) (cit. on p. 35).

[9] Borja Balle, Gilles Barthe, and Marco Gaboardi. "Privacy amplification by subsampling: Tight analyses via couplings and divergences." In: *Advances in Neural Information Processing Systems* 31 (2018) (cit. on p. 13).

[10] Borja Balle, Giovanni Cherubin, and Jamie Hayes. "Reconstructing training data with informed adversaries." In: *2022 IEEE Symposium on Security and Privacy (SP)*. IEEE. 2022, pp. 1138–1156 (cit. on p. 8).

[11] Borja Balle and Yu-Xiang Wang. "Improving the gaussian mechanism for differential privacy: Analytical calibration and optimal denoising." In: *International Conference on Machine Learning*. PMLR. 2018, pp. 394–403 (cit. on p. 5).

[12] Peter Bandi et al. "From detection of individual metastases to classification of lymph node status at the patient level: the camelyon17 challenge." In: *IEEE transactions on medical imaging* 38.2 (2018), pp. 550–560 (cit. on p. 33).

[13] Raef Bassily, Adam Smith, and Abhradeep Thakurta. "Private empirical risk minimization: Efficient algorithms and tight error bounds." In: *2014 IEEE 55th annual symposium on foundations of computer science*. IEEE. 2014, pp. 464–473 (cit. on p. 12).

[14] Alex Bie, Gautam Kamath, and Guojun Zhang. "Private GANs, Revisited." In: *arXiv preprint arXiv:2302.02936* (2023) (cit. on pp. 33, 40).

[15] Mikołaj Bińkowski, Danica J Sutherland, Michael Arbel, and Arthur Gretton. "Demystifying mmd gans." In: *arXiv preprint arXiv:1801.01401* (2018) (cit. on p. 33).

[16] Abeba Birhane and Vinay Uday Prabhu. "Large image datasets: A pyrrhic win for computer vision?" In: *2021 IEEE Winter Conference on Applications of Computer Vision (WACV)*. IEEE. 2021, pp. 1536–1546 (cit. on p. 15).

[17] Andrea Bittau et al. "Prochlo: Strong privacy for analytics in the crowd." In: *Proceedings of the 26th symposium on operating systems principles*. 2017, pp. 441–459 (cit. on p. 40).

[18] Gavin Brown, Mark Bun, Vitaly Feldman, Adam Smith, and Kunal Talwar. "When is memorization of irrelevant training data necessary for high-accuracy learning?" In: *Proceedings of the 53rd annual ACM SIGACT symposium on theory of computing*. 2021, pp. 123–132 (cit. on p. 41).

[19] Tom Brown et al. "Language models are few-shot learners." In: *Advances in neural information processing systems* 33 (2020), pp. 1877–1901 (cit. on pp. 1, 14).

[20] Chris Burgess and Hyunjik Kim. *3D Shapes Dataset*. `https://github.com/deepmind/3dshapes-dataset/`. 2018 (cit. on p. 10).

[21] Matthew Butterick. *Stable Diffusion litigation*. URL: `https://stablediffusionlitigation.com/` (visited on 04/04/2023) (cit. on p. 2).

[22] Clément Canonne. *What is delta, and what difference does it make?* DifferentialPrivacy.org. `https://differentialprivacy.org/flavoursofdelta/`. Mar. 2021 (cit. on p. 8).

[23] Tianshi Cao, Alex Bie, Arash Vahdat, Sanja Fidler, and Karsten Kreis. "Don't generate me: Training differentially private generative models with sinkhorn divergence." In: *Advances in Neural Information Processing Systems* 34 (2021), pp. 12480–12492 (cit. on pp. 30, 33).

[24] Nicholas Carlini, Chang Liu, Úlfar Erlingsson, Jernej Kos, and Dawn Song. "The Secret Sharer: Evaluating and Testing Unintended Memorization in Neural Networks." In: *USENIX Security Symposium*. Vol. 267. 2019 (cit. on pp. 1, 41).

[25] Nicholas Carlini et al. "Extracting Training Data from Large Language Models." In: *USENIX Security Symposium*. Vol. 6. 2021 (cit. on pp. 1, 41).

[26] Nicholas Carlini et al. "Membership inference attacks from first principles." In: *2022 IEEE Symposium on Security and Privacy (SP)*. IEEE. 2022, pp. 1897–1914 (cit. on p. 41).

[27] Nicholas Carlini et al. "Extracting training data from diffusion models." In: *arXiv preprint arXiv:2301.13188* (2023) (cit. on pp. 1, 2, 14, 41).

[28] Yannis Cattan, Christopher A Choquette-Choo, Nicolas Papernot, and Abhradeep Thakurta. "Fine-Tuning with Differential Privacy Necessitates an Additional Hyperparameter Search." In: *arXiv preprint arXiv:2210.02156* (2022) (cit. on p. 15).

[29] Hongyan Chang and Reza Shokri. "On the privacy risks of algorithmic fairness." In: *2021 IEEE European Symposium on Security and Privacy (EuroS&P)*. IEEE. 2021, pp. 292–303 (cit. on p. 35).

[30] Antoine Chatalic et al. "Compressive learning with privacy guarantees." In: *Information and Inference: A Journal of the IMA* 11.1 (2022), pp. 251–305 (cit. on p. 3).

[31] Kamalika Chaudhuri, Claire Monteleoni, and Anand D Sarwate. "Differentially private empirical risk minimization." In: *Journal of Machine Learning Research* 12.3 (2011) (cit. on p. 11).

[32] Dingfan Chen, Raouf Kerkouche, and Mario Fritz. "Private set generation with discriminative information." In: *arXiv preprint arXiv:2211.04446* (2022) (cit. on pp. 34, 35).

[33] 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* 33 (2020), pp. 12673–12684 (cit. on pp. 7, 15, 22, 26).

[34] Qingrong Chen et al. "Differentially private data generative models." In: *arXiv preprint arXiv:1812.02274* (2018) (cit. on p. 22).

[35] European Commission. *Proposal For a Regulation of The European Parliament and of The Council Laying Down Harmonised Rules on Artificial Intelligence (Artificial Intelligence Act) and Amending Certain Union Legislative Acts, COM/2021/206 final, 2021/0106(COD)*. `https://eur-lex.europa.eu/legal-content/EN/TXT/?uri=CELEX%3A52021PC0206`. 2021 (cit. on p. 41).

[36] Rachel Cummings, Varun Gupta, Dhamma Kimpara, and Jamie Morgenstern. "On the compatibility of privacy and fairness." In: *Adjunct Publication of the 27th Conference on User Modeling, Adaptation and Personalization*. 2019, pp. 309–315 (cit. on p. 35).

[37] Soham De, Leonard Berrada, Jamie Hayes, Samuel L Smith, and Borja Balle. "Unlocking high-accuracy differentially private image classification through scale." In: *arXiv preprint arXiv:2204.13650* (2022) (cit. on pp. 7, 14, 15, 27, 32, 40).

[38] Jia Deng et al. "Imagenet: A large-scale hierarchical image database." In: *2009 IEEE conference on computer vision and pattern recognition*. Ieee. 2009, pp. 248–255 (cit. on p. 14).

[39] Austin Derrow-Pinion et al. "Eta prediction with graph neural networks in google maps." In: *Proceedings of the 30th ACM International Conference on Information & Knowledge Management*. 2021, pp. 3767–3776 (cit. on p. 1).

[40] Damien Desfontaines. *Almost differential privacy*. `https://desfontain.es/privacy/almost-differential-privacy.html`. Ted is writing things (personal blog). Feb. 2019 (cit. on p. 8).

[41] Damien Desfontaines and Balázs Pejó. "SoK: Differential Privacies." In: *CoRR* abs/1906.01337 (2019) (cit. on p. 6).

[42] Bolin Ding, Janardhan Kulkarni, and Sergey Yekhanin. "Collecting telemetry data privately." In: *Advances in Neural Information Processing Systems* 30 (2017) (cit. on p. 40).

[43] Jiahao Ding et al. "Differentially private and fair classification via calibrated functional mechanism." In: *Proceedings of the AAAI Conference on Artificial Intelligence*. Vol. 34. 01. 2020, pp. 622–629 (cit. on p. 35).

[44] Laurent Dinh, Jascha Sohl-Dickstein, and Samy Bengio. "Density estimation using real nvp." In: *arXiv preprint arXiv:1605.08803* (2016) (cit. on p. 10).

[45] Tim Dockhorn, Tianshi Cao, Arash Vahdat, and Karsten Kreis. "Differentially private diffusion models." In: *arXiv preprint arXiv:2210.09929* (2022) (cit. on pp. 7, 22, 30, 33, 35, 36, 40).

[46] Alexey Dosovitskiy et al. "An image is worth 16x16 words: Transformers for image recognition at scale." In: *arXiv preprint arXiv:2010.11929* (2020) (cit. on p. 14).

[47] Cynthia Dwork. "Differential Privacy." In: *Automata, Languages and Programming*. Ed. by Michele Bugliesi, Bart Preneel, Vladimiro Sassone, and Ingo Wegener. Berlin, Heidelberg: Springer Berlin Heidelberg, 2006, pp. 1–12 (cit. on p. 3).

[48] Cynthia Dwork, Krishnaram Kenthapadi, Frank McSherry, Ilya Mironov, and Moni Naor. "Our data, ourselves: Privacy via distributed noise generation." In: *Annual international conference on the theory and applications of cryptographic techniques*. Springer. 2006, pp. 486–503 (cit. on p. 3).

[49] Cynthia Dwork, Frank McSherry, Kobbi Nissim, and Adam Smith. "Calibrating noise to sensitivity in private data analysis." In: *Theory of cryptography conference*. Springer. 2006, pp. 265–284 (cit. on p. 3).

[50] Cynthia Dwork, Aaron Roth, et al. "The algorithmic foundations of differential privacy." In: *Foundations and Trends® in Theoretical Computer Science* 9.3–4 (2014), pp. 211–407 (cit. on pp. 4, 5, 8).

[51] Gintare Karolina Dziugaite, Daniel M Roy, and Zoubin Ghahramani. "Training generative neural networks via maximum mean discrepancy optimization." In: *arXiv preprint arXiv:1505.03906* (2015) (cit. on p. 29).

[52] Úlfar Erlingsson, Vasyl Pihur, and Aleksandra Korolova. "Rappor: Randomized aggregatable privacy-preserving ordinal response." In: *Proceedings of the 2014 ACM SIGSAC conference on computer and communications security*. 2014, pp. 1054–1067 (cit. on p. 40).

[53] Maria S Esipova, Atiyeh Ashari Ghomi, Yaqiao Luo, and Jesse C Cresswell. "Disparate Impact in Differential Privacy from Gradient Misalignment." In: *arXiv preprint arXiv:2206.07737* (2022) (cit. on p. 35).

[54] Meta Fundamental AI Research Diplomacy Team (FAIR)† et al. "Human-level play in the game of Diplomacy by combining language models with strategic reasoning." In: *Science* 378.6624 (2022), pp. 1067–1074 (cit. on p. 1).

[55] Vitaly Feldman. "Does learning require memorization? a short tale about a long tail." In: *Proceedings of the 52nd Annual ACM SIGACT Symposium on Theory of Computing*. 2020, pp. 954–959 (cit. on p. 41).

[56] Sam Fletcher and Md Zahidul Islam. "Decision tree classification with differential privacy: A survey." In: *ACM Computing Surveys (CSUR)* 52.4 (2019), pp. 1–33 (cit. on p. 32).

[57] Matt Fredrikson, Somesh Jha, and Thomas Ristenpart. "Model inversion attacks that exploit confidence information and basic countermeasures." In: *Proceedings of the 22nd ACM SIGSAC conference on computer and communications security*. 2015, pp. 1322–1333 (cit. on p. 1).

[58] Kazuto Fukuchi, Quang Khai Tran, and Jun Sakuma. "Differentially private empirical risk minimization with input perturbation." In: *Discovery Science: 20th International Conference, DS 2017, Kyoto, Japan, October 15–17, 2017, Proceedings 20*. Springer. 2017, pp. 82–90 (cit. on p. 11).

[59] Sahra Ghalebikesabi et al. "Differentially Private Diffusion Models Generate Useful Synthetic Images." In: *arXiv preprint arXiv:2302.13861* (2023) (cit. on pp. 7, 8, 22, 33–36).

[60] Badih Ghazi, Noah Golowich, Ravi Kumar, Pasin Manurangsi, and Chiyuan Zhang. "Deep learning with label differential privacy." In: *Advances in neural information processing systems* 34 (2021), pp. 27131–27145 (cit. on p. 41).

[61] Sofie Goethals, Kenneth Sörensen, and David Martens. "The privacy issue of counterfactual explanations: explanation linkage attacks." In: *arXiv preprint arXiv:2210.12051* (2022) (cit. on p. 31).

[62] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep learning*. MIT press, 2016 (cit. on p. 8).

[63] Ian Goodfellow et al. "Generative adversarial networks." In: *Communications of the ACM* 63.11 (2020), pp. 139–144 (cit. on p. 10).

[64] Bryce Goodman and Seth Flaxman. "European Union regulations on algorithmic decision-making and a "right to explanation"." In: *AI magazine* 38.3 (2017), pp. 50–57 (cit. on p. 9).

[65] Sivakanth Gopi, Yin Tat Lee, and Lukas Wutschitz. "Numerical composition of differential privacy." In: *Advances in Neural Information Processing Systems* 34 (2021), pp. 11631–11642 (cit. on p. 4).

[66] Arthur Gretton, Karsten M Borgwardt, Malte J Rasch, Bernhard Schölkopf, and Alexander Smola. "A kernel two-sample test." In: *The Journal of Machine Learning Research* 13.1 (2012), pp. 723–773 (cit. on pp. 12, 23).

[67] Chuan Guo, Alexandre Sablayrolles, and Maziar Sanjabi. "Analyzing privacy leakage in machine learning via multiple hypothesis testing: A lesson from fano." In: *International Conference on Machine Learning*. PMLR. 2023, pp. 11998–12011 (cit. on pp. 7, 41).

[68] Moritz Hardt, Katrina Ligett, and Frank McSherry. "A simple and practical algorithm for differentially private data release." In: *Advances in neural information processing systems* 25 (2012) (cit. on p. 8).

[69] 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*. 2016, pp. 770–778 (cit. on p. 28).

[70] 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* 30 (2017) (cit. on pp. 10, 33).

[71] Jonathan Ho, Ajay Jain, and Pieter Abbeel. "Denoising diffusion probabilistic models." In: *Advances in Neural Information Processing Systems* 33 (2020), pp. 6840–6851 (cit. on p. 10).

[72] Justin Hsu et al. "Differential privacy: An economic method for choosing epsilon." In: *2014 IEEE 27th Computer Security Foundations Symposium*. IEEE. 2014, pp. 398–410 (cit. on p. 41).

[73] Getty Images. *Statement regarding legal proceedings against Stability AI*. URL: `https://newsroom.gettyimages.com/en/getty-images/getty-images-statement` (visited on 04/04/2023) (cit. on p. 2).

[74] Sergey Ioffe and Christian Szegedy. "Batch normalization: Accelerating deep network training by reducing internal covariate shift." In: *International conference on machine learning*. pmlr. 2015, pp. 448–456 (cit. on p. 12).

[75] Robert A Jacobs, Michael I Jordan, Steven J Nowlan, and Geoffrey E Hinton. "Adaptive mixtures of local experts." In: *Neural computation* 3.1 (1991), pp. 79–87 (cit. on p. 20).

[76] Arthur Jacot, Franck Gabriel, and Clément Hongler. "Neural tangent kernel: Convergence and generalization in neural networks." In: *Advances in neural information processing systems* 31 (2018) (cit. on p. 34).

[77] Matthew Jagielski et al. "Differentially private fair learning." In: *International Conference on Machine Learning*. PMLR. 2019, pp. 3000–3008 (cit. on p. 35).

[78] Ismat Jarin and Birhanu Eshete. "DP-UTIL: comprehensive utility analysis of differential privacy in machine learning." In: *Proceedings of the Twelfth ACM Conference on Data and Application Security and Privacy*. 2022, pp. 41–52 (cit. on pp. 10, 11).

[79] Noah Johnson, Joseph P Near, and Dawn Song. "Towards practical differential privacy for SQL queries." In: *Proceedings of the VLDB Endowment* 11.5 (2018), pp. 526–539 (cit. on p. 40).

[80] William B Johnson. "Extensions of Lipschitz mappings into a Hilbert space." In: *Contemp. Math.* 26 (1984), pp. 189–206 (cit. on p. 20).

[81] Yilin Kang et al. "Input perturbation: A new paradigm between central and local differential privacy." In: *arXiv preprint arXiv:2002.08570* (2020) (cit. on p. 11).

[82] Krishnaram Kenthapadi, Aleksandra Korolova, Ilya Mironov, and Nina Mishra. "Privacy via the johnson-lindenstrauss transform." In: *arXiv preprint arXiv:1204.2606* (2012) (cit. on p. 20).

[83] Been Kim et al. "Interpretability beyond feature attribution: Quantitative testing with concept activation vectors (tcav)." In: *International conference on machine learning*. PMLR. 2018, pp. 2668–2677 (cit. on p. 32).

[84] Diederik P Kingma and Max Welling. "Auto-encoding variational bayes." In: *arXiv preprint arXiv:1312.6114* (2013) (cit. on p. 10).

[85] Ios Kotsogiannis, Stelios Doudalis, Sam Haney, Ashwin Machanavajjhala, and Sharad Mehrotra. "One-sided differential privacy." In: *2020 IEEE 36th International Conference on Data Engineering (ICDE)*. IEEE. 2020, pp. 493–504 (cit. on p. 41).

[86] Alex Krizhevsky, Geoffrey Hinton, et al. "Learning multiple layers of features from tiny images." In: (2009) (cit. on pp. 14, 33).

[87] Alexey Kurakin et al. "Toward training at imagenet scale with differential privacy." In: *arXiv preprint arXiv:2201.12328* (2022) (cit. on p. 27).

[88] Siddique Latif et al. "Deep representation learning in speech processing: Challenges, recent advances, and future trends." In: *arXiv preprint arXiv:2001.00378* (2020) (cit. on p. 36).

[89] Yann LeCun. "The MNIST database of handwritten digits." In: *http://yann. lecun. com/exdb/mnist/* (1998) (cit. on pp. 19, 22).

[90] Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. "Gradient-based learning applied to document recognition." In: *Proceedings of the IEEE* 86.11 (1998), pp. 2278–2324 (cit. on p. 33).

[91] Jaewoo Lee and Chris Clifton. "How much is enough? choosing ε for differential privacy." In: *Information Security: 14th International Conference, ISC 2011, Xi'an, China, October 26-29, 2011. Proceedings 14*. Springer. 2011, pp. 325–340 (cit. on p. 41).

[92] Tobias Leemann, Martin Pawelczyk, and Gjergji Kasneci. "Gaussian Membership Inference Privacy." In: *arXiv preprint arXiv:2306.07273* (2023) (cit. on p. 41).

[93] Runzhi Li, Wei Liu, Yusong Lin, Hongling Zhao, and Chaoyang Zhang. "An ensemble multilabel classification for disease risk prediction." In: *Journal of healthcare engineering* 2017 (2017) (cit. on p. 19).

[94] Xuechen Li, Florian Tramer, Percy Liang, and Tatsunori Hashimoto. "Large language models can be strong differentially private learners." In: *arXiv preprint arXiv:2110.05679* (2021) (cit. on pp. 15, 27).

[95] Xuechen Li et al. "When Does Differentially Private Learning Not Suffer in High Dimensions?" In: *arXiv preprint arXiv:2207.00160* (2022) (cit. on p. 15).

[96] Seng Pei Liew, Tsubasa Takahashi, and Michihiko Ueno. "PEARL: Data Synthesis via Private Embeddings and Adversarial Reconstruction Learning." In: *International Conference on Learning Representations* (cit. on pp. 32, 33).

[97] Zinan Lin, Sivakanth Gopi, Janardhan Kulkarni, Harsha Nori, and Sergey Yekhanin. "Differentially Private Synthetic Data via Foundation Model APIs 1: Images." In: *arXiv preprint arXiv:2305.15560* (2023) (cit. on pp. 34–36).

[98] Zhuang Liu et al. "A convnet for the 2020s." In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2022, pp. 11976–11986 (cit. on p. 28).

[99] Saiyue Lyu, Margarita Vinaroz, Michael F Liu, and Mijung Park. "Differentially Private Latent Diffusion Models." In: *arXiv preprint arXiv: 2305.15759* (2023) (cit. on p. 33).

[100] Aravindh Mahendran and Andrea Vedaldi. "Understanding deep image representations by inverting them." In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2015, pp. 5188–5196 (cit. on p. 32).

[101] Paul Mangold, Michaël Perrot, Aurélien Bellet, and Marc Tommasi. "Differential Privacy has Bounded Impact on Fairness in Classification." In: (2022) (cit. on p. 35).

[102] Andrew Maxwell et al. "Deep learning architectures for multi-label classification of intelligent health risk prediction." In: *BMC bioinformatics* 18 (2017), pp. 121–131 (cit. on p. 19).

[103] Brendan McMahan and Abhradeep Thakurta. "Federated learning with formal differential privacy guarantees." In: *Google AI Blog* (2022) (cit. on p. 40).

[104] F Gustav Mehler. "Ueber die Entwicklung einer Function von beliebig vielen Variablen nach Laplaceschen Functionen höherer Ordnung." In: (1866) (cit. on p. 26).

[105] Harsh Mehta, Abhradeep Thakurta, Alexey Kurakin, and Ashok Cutkosky. "Large scale transfer learning for differentially private image classification." In: *arXiv preprint arXiv:2205.02973* (2022) (cit. on p. 27).

[106] Smitha Milli, Ludwig Schmidt, Anca D Dragan, and Moritz Hardt. "Model reconstruction from model explanations." In: *Proceedings of the Conference on Fairness, Accountability, and Transparency*. 2019, pp. 1–9 (cit. on p. 31).

[107] Ilya Mironov. "Rényi differential privacy." In: *2017 IEEE 30th computer security foundations symposium (CSF)*. IEEE. 2017, pp. 263–275 (cit. on pp. 4, 6).

[108] Arvind Narayanan and Vitaly Shmatikov. "Robust de-anonymization of large sparse datasets." In: *2008 IEEE Symposium on Security and Privacy (sp 2008)*. IEEE. 2008, pp. 111–125 (cit. on p. 1).

[109] Milad Nasr, Shuang Songi, Abhradeep Thakurta, Nicolas Papernot, and Nicholas Carlin. "Adversary instantiation: Lower bounds for differentially private machine learning." In: *2021 IEEE Symposium on security and privacy (SP)*. IEEE. 2021, pp. 866–882 (cit. on p. 8).

[110] Timothy Nguyen, Zhourong Chen, and Jaehoon Lee. "Dataset meta-learning from kernel ridge-regression." In: *arXiv preprint arXiv:2011.00050* (2020) (cit. on p. 34).

[111] Timothy Nguyen, Roman Novak, Lechao Xiao, and Jaehoon Lee. "Dataset distillation with infinitely wide convolutional networks." In: *Advances in Neural Information Processing Systems* 34 (2021), pp. 5186–5198 (cit. on p. 34).

[112] Harsha Nori, Rich Caruana, Zhiqi Bu, Judy Hanwen Shen, and Janardhan Kulkarni. "Accuracy, interpretability, and differential privacy via explainable boosting." In: *International Conference on Machine Learning*. PMLR. 2021, pp. 8227–8237 (cit. on p. 32).

[113] Curtis G Northcutt, Anish Athalye, and Jonas Mueller. "Pervasive label errors in test sets destabilize machine learning benchmarks." In: *arXiv preprint arXiv:2103.14749* (2021) (cit. on p. 14).

[114] Chris Olah, Alexander Mordvintsev, and Ludwig Schubert. "Feature visualization." In: *Distill* 2.11 (2017), e7 (cit. on p. 32).

[115] Nicolas Papernot, Martín Abadi, Ulfar Erlingsson, Ian Goodfellow, and Kunal Talwar. "Semi-supervised knowledge transfer for deep learning from private training data." In: *arXiv preprint arXiv:1610.05755* (2016) (cit. on p. 11).

[116] Nicolas Papernot et al. "Scalable private learning with pate." In: *arXiv preprint arXiv:1802.08908* (2018) (cit. on p. 11).

[117] Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. "Bleu: a method for automatic evaluation of machine translation." In: *Proceedings of the 40th annual meeting of the Association for Computational Linguistics*. 2002, pp. 311–318 (cit. on p. 10).

[118] Neel Patel, Reza Shokri, and Yair Zick. "Model explanations with differential privacy." In: *2022 ACM Conference on Fairness, Accountability, and Transparency*. 2022, pp. 1895–1904 (cit. on p. 31).

[119] F. Pedregosa et al. "Scikit-learn: Machine Learning in Python." In: *Journal of Machine Learning Research* 12 (2011), pp. 2825–2830 (cit. on p. 24).

[120] Bjarne Pfitzner and Bert Arnrich. "DPD-fVAE: Synthetic Data Generation Using Federated Variational Autoencoders With Differentially-Private Decoder." In: *arXiv preprint arXiv:2211.11591* (2022) (cit. on pp. 15, 22, 33).

[121] NhatHai Phan, Xintao Wu, and Dejing Dou. "Preserving differential privacy in convolutional deep belief networks." In: *Machine learning* 106.9-10 (2017), pp. 1681–1704 (cit. on p. 21).

[122] Mehmet Pilancı and Elif Vural. "Domain adaptation on graphs by learning aligned graph bases." In: *IEEE Transactions on Knowledge and Data Engineering* 34.2 (2020), pp. 587–600 (cit. on p. 36).

[123] Natalia Ponomareva, Jasmijn Bastings, and Sergei Vassilvitskii. "Training text-to-text transformers with privacy guarantees." In: *Findings of the Association for Computational Linguistics: ACL 2022*. 2022, pp. 2182–2193 (cit. on p. 8).

[124] Natalia Ponomareva et al. "How to dp-fy ml: A practical guide to machine learning with differential privacy." In: *Journal of Artificial Intelligence Research* 77 (2023), pp. 1113–1201 (cit. on pp. 7, 8).

[125] Alec Radford et al. "Language models are unsupervised multitask learners." In: *OpenAI blog* 1.8 (2019), p. 9 (cit. on p. 1).

[126] Ali Rahimi and Benjamin Recht. "Random features for large-scale kernel machines." In: *Advances in neural information processing systems* 20 (2007) (cit. on p. 23).

[127] Alan Ramponi and Barbara Plank. "Neural unsupervised domain adaptation in NLP—a survey." In: *arXiv preprint arXiv:2006.00632* (2020) (cit. on p. 36).

[128] Alfréd Rényi. "On measures of entropy and information." In: *Proceedings of the Fourth Berkeley Symposium on Mathematical Statistics and Probability, Volume 1: Contributions to the Theory of Statistics*. Vol. 4. University of California Press. 1961, pp. 547–562 (cit. on p. 6).

[129] Danilo Rezende and Shakir Mohamed. "Variational inference with normalizing flows." In: *International conference on machine learning*. PMLR. 2015, pp. 1530–1538 (cit. on p. 10).

[130] Marco Tulio Ribeiro, Sameer Singh, and Carlos Guestrin. ""Why should i trust you?" Explaining the predictions of any classifier." In: *Proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining*. 2016, pp. 1135–1144 (cit. on p. 32).

[131] 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*. 2022, pp. 10684–10695 (cit. on pp. 1, 2, 10, 33, 34).

[132] David E Rumelhart, Geoffrey E Hinton, and Ronald J Williams. "Learning representations by back-propagating errors." In: *nature* 323.6088 (1986), pp. 533–536 (cit. on p. 9).

[133] Chitwan Saharia et al. "Photorealistic text-to-image diffusion models with deep language understanding." In: *arXiv preprint arXiv:2205.11487* (2022) (cit. on pp. 1, 2, 10).

[134] Tom Sander, Pierre Stock, and Alexandre Sablayrolles. "TAN without a burn: Scaling Laws of DP-SGD." In: *arXiv preprint arXiv:2210.03403* (2022) (cit. on pp. 14, 32, 40).

[135] Cicero Nogueira dos Santos, Youssef Mroueh, Inkit Padhi, and Pierre Dognin. "Learning implicit generative models by matching perceptual features." In: *Proceedings of the IEEE/CVF International Conference on Computer Vision*. 2019, pp. 4461–4470 (cit. on p. 27).

[136] Amartya Sanyal, Yaxi Hu, and Fanny Yang. "How unfair is private learning?" In: *Uncertainty in Artificial Intelligence*. PMLR. 2022, pp. 1738–1748 (cit. on p. 35).

[137] U.S. Office of Science and Technology Policy. *Blueprint for an AI Bill of Rights: Making Automated Systems Work for the American People*. `www.whitehouse.gov/ostp/ai-bill-of-rights/`. 2022 (cit. on p. 41).

[138] Reza Shokri, Martin Strobel, and Yair Zick. "On the privacy risks of model explanations." In: *Proceedings of the 2021 AAAI/ACM Conference on AI, Ethics, and Society*. 2021, pp. 231–241 (cit. on p. 31).

[139] Reza Shokri, Marco Stronati, Congzheng Song, and Vitaly Shmatikov. "Membership inference attacks against machine learning models." In: *2017 IEEE symposium on security and privacy (SP)*. IEEE. 2017, pp. 3–18 (cit. on pp. 1, 41).

[140] David Silver et al. "Mastering chess and shogi by self-play with a general reinforcement learning algorithm." In: *arXiv preprint arXiv:1712.01815* (2017) (cit. on p. 1).

[141] Karen Simonyan and Andrew Zisserman. "Very deep convolutional networks for large-scale image recognition." In: *arXiv preprint arXiv:1409.1556* (2014) (cit. on p. 28).

[142] Daniel Smilkov, Nikhil Thorat, Been Kim, Fernanda Viégas, and Martin Wattenberg. "Smoothgrad: removing noise by adding noise." In: *arXiv preprint arXiv:1706.03825* (2017) (cit. on pp. 10, 19).

[143] Jascha Sohl-Dickstein, Eric Weiss, Niru Maheswaranathan, and Surya Ganguli. "Deep unsupervised learning using nonequilibrium thermodynamics." In: *International Conference on Machine Learning*. PMLR. 2015, pp. 2256–2265 (cit. on p. 10).

[144] Congzheng Song, Thomas Ristenpart, and Vitaly Shmatikov. "Machine learning models that remember too much." In: *Proceedings of the 2017 ACM SIGSAC Conference on computer and communications security*. 2017, pp. 587–601 (cit. on p. 1).

[145] Shuang Song, Kamalika Chaudhuri, and Anand D Sarwate. "Stochastic gradient descent with differentially private updates." In: *2013 IEEE global conference on signal and information processing*. IEEE. 2013, pp. 245–248 (cit. on p. 12).

[146] Yang Song and Stefano Ermon. "Generative modeling by estimating gradients of the data distribution." In: *Advances in neural information processing systems* 32 (2019) (cit. on p. 10).

[147] Yang Song et al. "Score-Based Generative Modeling through Stochastic Differential Equations." In: *International Conference on Learning Representations* (cit. on pp. 10, 33).

[148] Mukund Sundararajan, Ankur Taly, and Qiqi Yan. "Axiomatic attribution for deep networks." In: *International conference on machine learning*. PMLR. 2017, pp. 3319–3328 (cit. on pp. 10, 19).

[149] Latanya Sweeney. "k-anonymity: A model for protecting privacy." In: *International journal of uncertainty, fuzziness and knowledge-based systems* 10.05 (2002), pp. 557–570 (cit. on p. 5).

[150] Christian Szegedy et al. "Going deeper with convolutions." In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2015, pp. 1–9 (cit. on p. 33).

[151] Tsubasa Takahashi, Shun Takagi, Hajime Ono, and Tatsuya Komatsu. "Differentially Private Variational Autoencoders with Term-Wise Gradient Aggregation." In: *arXiv preprint arXiv:2006.11204* (2020) (cit. on p. 22).

[152] Mingxing Tan and Quoc Le. "Efficientnet: Rethinking model scaling for convolutional neural networks." In: *International conference on machine learning*. PMLR. 2019, pp. 6105–6114 (cit. on p. 28).

[153] Jun Tang, Aleksandra Korolova, Xiaolong Bai, Xueqiang Wang, and Xiaofeng Wang. "Privacy loss in apple's implementation of differential privacy on macos 10.12." In: *arXiv preprint arXiv:1709.02753* (2017) (cit. on pp. 15, 40).

[154] Apple Differential Privacy Team. *Learning with Privacy at Scale*. URL: `https://machinelearning.apple.com/research/learning-with-privacy-at-scale` (visited on 09/09/2023) (cit. on p. 40).

[155] Reihaneh Torkzadehmahani, Peter Kairouz, and Benedict Paten. "Dp-cgan: Differentially private synthetic data and label generation." In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*. 2019, pp. 0–0 (cit. on pp. 15, 22, 26, 32).

[156] Florian Tramer and Dan Boneh. "Differentially private learning needs better features (or much more data)." In: *arXiv preprint arXiv:2011.11660* (2020) (cit. on p. 27).

[157] Florian Tramèr, Gautam Kamath, and Nicholas Carlini. "Considerations for Differentially Private Learning with Large-Scale Public Pretraining." In: *arXiv preprint arXiv:2212.06470* (2022) (cit. on pp. 15, 35, 36).

[158] Cuong Tran, My Dinh, and Ferdinando Fioretto. "Differentially private empirical risk minimization under the fairness lens." In: *Advances in Neural Information Processing Systems* 34 (2021), pp. 27555–27565 (cit. on p. 35).

[159] Cuong Tran, Ferdinando Fioretto, and Pascal Van Hentenryck. "Differentially private and fair deep learning: A lagrangian dual approach." In: *Proceedings of the AAAI Conference on Artificial Intelligence*. Vol. 35. 11. 2021, pp. 9932–9939 (cit. on p. 35).

[160] Aleksei Triastcyn and Boi Faltings. "Bayesian differential privacy for machine learning." In: *International Conference on Machine Learning*. PMLR. 2020, pp. 9583–9592 (cit. on pp. 6, 33, 41).

[161] Margarita Vinaroz and Mi Jung Park. "Differentially Private Kernel Inducing Points (DP-KIP) for Privacy-preserving Data Distillation." In: *arXiv preprint arXiv:2301.13389* (2023) (cit. on pp. 34, 35).

[162] Yu-Xiang Wang, Borja Balle, and Shiva Prasad Kasiviswanathan. "Subsampled rényi differential privacy and analytical moments accountant." In: *The 22nd International Conference on Artificial Intelligence and Statistics*. PMLR. 2019, pp. 1226–1235 (cit. on p. 13).

[163] Han Xiao, Kashif Rasul, and Roland Vollgraf. "Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms." In: *arXiv preprint arXiv:1708.07747* (2017) (cit. on pp. 19, 22).

[164] Liyang Xie, Kaixiang Lin, Shu Wang, Fei Wang, and Jiayu Zhou. "Differentially private generative adversarial network." In: *arXiv preprint arXiv:1802.06739* (2018) (cit. on pp. 15, 22, 32).

[165] Depeng Xu, Shuhan Yuan, and Xintao Wu. "Achieving differential privacy and fairness in logistic regression." In: *Companion proceedings of The 2019 world wide web conference*. 2019, pp. 594–599 (cit. on p. 35).

[166] Zihao Xu, Guang-He Lee, Yuyang Wang, Hao Wang, et al. "Graph-relational domain adaptation." In: *arXiv preprint arXiv:2202.03628* (2022) (cit. on p. 36).

[167] Jiayuan Ye and Reza Shokri. "Differentially private learning needs hidden state (or much faster convergence)." In: *arXiv preprint arXiv:2203.05363* (2022) (cit. on p. 14).

[168] Samuel Yeom, Irene Giacomelli, Matt Fredrikson, and Somesh Jha. "Privacy risk in machine learning: Analyzing the connection to overfitting." In: *2018 IEEE 31st computer security foundations symposium (CSF)*. IEEE. 2018, pp. 268–282 (cit. on p. 41).

[169] Da Yu, Huishuai Zhang, Wei Chen, Jian Yin, and Tie-Yan Liu. "Large scale private learning via low-rank reparametrization." In: *International Conference on Machine Learning*. PMLR. 2021, pp. 12208–12218 (cit. on p. 27).

[170] Da Yu et al. "Differentially private fine-tuning of language models." In: *arXiv preprint arXiv:2110.06500* (2021) (cit. on p. 27).

[171] Jiahui Yu et al. "Coca: Contrastive captioners are image-text foundation models." In: *arXiv preprint arXiv:2205.01917* (2022) (cit. on p. 14).

[172] Yu Zhang, Peter Tiňo, Aleš Leonardis, and Ke Tang. "A survey on neural network interpretability." In: *IEEE Transactions on Emerging Topics in Computational Intelligence* 5.5 (2021), pp. 726–742 (cit. on pp. 9, 32).

[173] Bo Zhao, Konda Reddy Mopuri, and Hakan Bilen. "Dataset condensation with gradient matching." In: *arXiv preprint arXiv:2006.05929* (2020) (cit. on p. 34).

[174] Da Zhong, Haipei Sun, Jun Xu, Neil Gong, and Wendy Hui Wang. "Understanding disparate effects of membership inference attacks and their countermeasures." In: *Proceedings of the 2022 ACM on Asia Conference on Computer and Communications Security*. 2022, pp. 959–974 (cit. on p. 35).

[175] Yuqing Zhu and Yu-Xiang Wang. "Poisson subsampled rényi differential privacy." In: *International Conference on Machine Learning*. PMLR. 2019, pp. 7634–7642 (cit. on p. 13).

A

INTERPRETABLE AND DIFFERENTIALLY PRIVATE PREDICTIONS

The following paper has been published in the Proceedings of the 34. AAAI Conference on Artificial Intelligence under the CC-BY-4.0 license[1] and is reprinted without modifications.

1 https://creativecommons.org/licenses/by/4.0/

The Thirty-Fourth AAAI Conference on Artificial Intelligence (AAAI-20)

Interpretable and Differentially Private Predictions

Frederik Harder,[1,2] **Matthias Bauer,**[1,3*] **Mijung Park**[1,2]
[1]Max Planck Institute for Intelligent Systems, Tübingen, Germany
[2]Department of Computer Science, University of Tübingen, Tübingen, Germany
[3]Department of Engineering, University of Cambridge, Cambridge, UK
{fharder, bauer, mpark}@tue.mpg.de

Abstract

Interpretable predictions, which clarify why a machine learning model makes a particular decision, can compromise privacy by revealing the characteristics of individual data points. This raises the central question addressed in this paper: *Can models be interpretable without compromising privacy?* For complex "big" data fit by correspondingly rich models, balancing privacy and explainability is particularly challenging, such that this question has remained largely unexplored. In this paper, we propose a family of simple models with the aim of approximating complex models using several *locally linear maps* per class to provide high classification accuracy, as well as differentially private explanations on the classification. We illustrate the usefulness of our approach on several image benchmark datasets as well as a medical dataset.

1 Introduction

The *General Data Protection Regulation (GDPR)* by the European Union imposes two important requirements on algorithmic design, *interpretability* and *privacy* (Voigt and Bussche 2017). These requirements introduce new standards on future algorithmic techniques, making them of particular concern to the machine learning community (Goodman and Flaxman 2016). This paper addresses these two requirements in the context of classification, and studies the trade-off between privacy, accuracy and interpretability, see Fig. 1.

Broadly speaking, there are two options to take for gaining interpretability: (i) rely on *inherently interpretable models*; and (ii) rely on *post-processing schemes* to probe trained complex models. Inherently interpretable models are often relatively simple and their predictions can be easily analyzed in terms of their respective input features. For instance, in logistic regression classifiers and sparse linear models the coefficients represent the importance of each input feature. However, modern "big" data typically exhibit complex patterns, such that these relatively simplistic models often have lower accuracy than more complex ones.

[*] Matthias Bauer is now at DeepMind

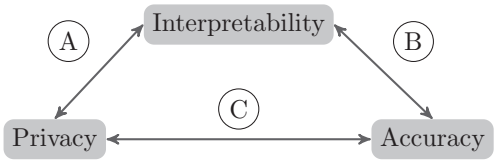


Figure 1: Modern machine learning systems need to trade off accuracy, privacy, and interpretability.

In order to soften this trade-off between interpretability and accuracy (Fig. 1 (B)), many post-processing schemes aim to gain insights from complex models like deep neural networks. One prominent aspect of these approaches is the use of gradient-based attributions (Selvaraju et al. 2016; Ribeiro, Singh, and Guestrin 2016; Smilkov et al. 2017; Sundararajan, Taly, and Yan 2017; Montavon et al. 2015; Bach et al. 2015; Ancona et al. 2017).

In another line of research, many recent papers address the concern that complex models with outstanding predictive performance can expose sensitive information from the dataset they were trained on (Carlini et al. 2018; Song, Ristenpart, and Shmatikov 2017; Shokri and Shmatikov 2015; Fredrikson, Jha, and Ristenpart 2015). To quantify privacy, several approaches adopt the notion of *differential privacy* (DP), which provides a mathematically provable definition of privacy, and can quantify the level of privacy an algorithm or a model provides (Dwork and Roth 2014). In plain English, an algorithm is called differentially private (DP), if its output is random enough to obscure the participation of any single individual in the data. The randomness is typically achieved by injecting noise into the algorithm. The amount of noise is determined by the level of privacy the algorithm guarantees and the sensitivity, a maximum difference in its output depending on a single individual's participation or non-participation in the data (see Sec. 2 for the mathematical definition of DP).

There is, however, a natural trade-off between privacy and accuracy (Fig. 1 (C)): A large amount of added noise provides a high level of privacy but also harms prediction accuracy. When the number of parameters is high, like in deep neural network models, juggling this trade-off is very

challenging, as privatizing high dimensional parameters results in a high privacy loss to meet a good prediction level (Abadi et al. 2016). Therefore, most existing work in differential privacy literature considers relatively small networks or assumes that some relevant data are publicly available to train a significant part of the network without privacy violation to deal with the trade-off (see Sec. 5 for details). However, none of the existing work takes into account the interpretability of the learned models and this is our core contribution described below.

Our contribution. In this paper, we study the trade-off between interpretability, privacy, and accuracy by making the following three contributions.

- **We propose a novel family of interpretable models**: To take into account privacy *and* interpretability (Fig. 1 (A)), we propose a family of inherently interpretable models that can be trained privately. These models approximate the mapping of a complex model from the input data to class score functions, using several *locally linear maps* (LLM) per class. Our formulation for LLM is inspired by the approximation of differentiable functions as a collection of piece-wise linear functions, i.e., the first-order Taylor expansions of the function at a sufficiently large number of input locations. With an adequate number of linear maps, our local models permit a relatively slight loss in accuracy compared to complex model counterparts. The level of loss in prediction accuracy depends on the complexity of data.
- **We provide DP "local" and "global" explanations on classification**: Our model LLM, trained with the DP constraint, provides insights on the key features for classification at a "local" and "global" level. A *local* explanation of a model illustrates how the model behaves at and around a specific input, showing how relevant different features of the input were to the model decision. This is a typical outcome one could obtain by probing a complex model using existing attribution methods. However, our model also provides a *global* explanation, illustrating how the model functions as a whole and, in the case of classification, what types of input the different classes are sensitive to. This is what distinguishes our work from other existing post-processing attribution methods.
- **We propose to use random projections to better deal with privacy and accuracy trade-off**: We propose to adopt the *Johnson-Lindenstrauss transform*, a.k.a., *random projection* (Kenthapadi et al. 2013), to decrease the dimensionality of each LLM and then privatize the resulting lower dimensional quantities. We found that exploiting data-indepdent random projection achieves a significantly better trade-off for high-dimensional image data.

We would like to emphasize that our work is the first to address the interplay between interpretability, privacy, and accuracy. Hence, this work presents not only a novel inherently interpretable model but also an important conceptual contribution to the field that will spur more research on this intersection.

2 Background on Differential Privacy

We start by introducing differential privacy and a composition method that we will use in our algorithm, as well as random projections.

Differential privacy. Consider an algorithm $\mathcal{M}$ and neighboring datasets $\mathcal{D}$ and $\mathcal{D}'$ differing by a single entry, where the dataset $\mathcal{D}'$ is obtained by excluding one datapoint from the dataset $\mathcal{D}$. In DP (Dwork and Roth 2014), the quantity of interest is *privacy loss*, defined by

$$L^{(o)} = \log \frac{\Pr(\mathcal{M}_{(\mathcal{D})} = o)}{\Pr(\mathcal{M}_{(\mathcal{D}')} = o)}, \tag{1}$$

where $\mathcal{M}_{(\mathcal{D})}$ and $\mathcal{M}_{(\mathcal{D}')}$ denote the outputs of the algorithm given $\mathcal{D}$ and $\mathcal{D}'$, respectively. $\Pr(\mathcal{M}_{(\mathcal{D})} = o)$ denotes the probability that $\mathcal{M}$ returns a specific output o. When the two probabilities in Eq. (1) are similar, even a strong adversary, who knows all the datapoints in $\mathcal{D}$ except for one, could not discern the one datapoint by which $\mathcal{D}$ and $\mathcal{D}'$ differ, based on the output of the algorithm alone. On the other hand, when the probabilities are very different, it would be easy to identify the exclusion of the single datapoint in $\mathcal{D}'$. Hence, the privacy loss quantifies how revealing an algorithm's output is about a single entry's presence in the dataset $\mathcal{D}$. Formally, an algorithm $\mathcal{M}$ is called ϵ-DP if and only if $|L^{(o)}| \leq \epsilon, \forall o$ and for all neighbouring datasets $\mathcal{D}, \mathcal{D}'$. A weaker version of the above is (ϵ, δ)-DP, if and only if $|L^{(o)}| \leq \epsilon$, with probability at least $1 - \delta$.

Introducing a noise addition step is a popular way of making an algorithm DP. The *output perturbation* method achieves this by adding noise to the output h, where the noise is calibrated to h's *sensitivity*, defined by

$$S_h = \max_{\mathcal{D},\mathcal{D}',|\mathcal{D}-\mathcal{D}'|=1} \|h(\mathcal{D}) - h(\mathcal{D})\|_2, \tag{2}$$

which is the maximum difference in terms of L2-norm, under the one datapoint's difference in $\mathcal{D}$ and $\mathcal{D}'$. With the sensitivity, we can privatize the output using the *Gaussian mechanism*, which simply adds Gaussian noise of the form: $\tilde{h}(\mathcal{D}) = h(\mathcal{D}) + \mathcal{N}(0, S_h^2\sigma^2\mathbf{I}_p)$, where $\mathcal{N}(0, S_h^2\sigma^2\mathbf{I}_p)$ means the Gaussian distribution with mean 0 and covariance $S_h^2\sigma^2\mathbf{I}_p$. The resulting quantity $\tilde{h}(\mathcal{D})$ is (ϵ, δ)-DP, where $\sigma \geq \sqrt{2\log(1.25/\delta)}/\epsilon$ (see Dwork and Roth (2014) for a proof). In this paper, we use the Gaussian mechanism to achieve differentially private LLM.

Properties of differential privacy. DP has two important properties: (i) post-processing invariance and (ii) composability. *Post-processing invariance* states that applying any *data-independent* mechanism to a DP quantity does not alter the privacy level of the resulting quantity.

Composability states that combining DP quantities degrades privacy in a quantifiable way. The most naïve way is the *linear* composition (Theorem 3.14 in Dwork and Roth (2014)), where the resulting parameter, which is often called *cumulative privacy loss* (cumulative ϵ and δ), are linearly summed up, $\epsilon = \sum_{t=1}^{T} \epsilon_t$ and $\delta = \sum_{t=1}^{T} \delta_t$ after the repeated use of data T times with the per-use privacy loss

(ϵ_t, δ_t). Recently, Abadi et al. (2016) proposed the *moments accountant* method, which provides an efficient way of combining ϵ and δ such that the cumulative privacy loss is significantly smaller than that by other composition methods. The resulting composition provides a better utility, meaning that a smaller amount of noise is required to add for the same privacy guarantee compared to other composition methods.

Random projections in the context of differential privacy A variant of our method involves projecting each input onto a lower-dimensional space using a *Johnson-Lindenstrauss transform* (a.k.a., *random projection*) (Kenthapadi et al. 2013). We construct the projection matrix $\mathbf{R}$ by drawing each entry from $\mathcal{N}(0, 1/D')$ where D' is the dimension of the projected space. This projection nearly preserves the distances between two points in the data space and in the embedding space, as this projection guarantees low-distortion embeddings. Random projections have been used to ensure DP (Blocki et al. 2012). However, here we only utilize them as a convenient method to reduce input dimension to our learnable linear maps. Since the random filters are data-independent, they do not need to be privatized.

3 Our method: Locally Linear Maps (LLM)

Motivation. As mentioned earlier, complex models such as deep neural networks tend to lack interpretability due to their nested feature structure. Gradient-based attribution methods can provide local explanations by computing a linear approximation of the model at a given point in the input space (see Sec. 5 for more details). Such approximations can be seen as sensitivity maps that highlight which parts of the input affect the model decision locally. However, these approaches lack *global* explanations that provide insight on how the model works as a whole, e.g., it is not straightforward to obtain class-wise key features. Furthermore, existing methods in the DP literature do not take into account the interpretability of learned models. In order to satisfy both interpretability and privacy demands, we desire a model with the following properties:

1. It can provide both *local and global explanations*.
2. It has efficient ways to limit in the number of parameters to achieve a *good privacy accuracy trade-off*.
3. It is *more expressive than standard linear models* to capture complex patterns in the data.

Locally Linear Maps (LLM). We introduce a set of local functions f_k for each class k, and parameterize each f_k by a combination of M linear maps denoted by g_m^k. The M linear maps are weighted separately for each class using the weighting coefficients $\sigma(\mathbf{x})_m^k$, which determine how *important* each linear map is for classifying a given input:

$$f_k(\mathbf{x}) = \sum_{m=1}^{M} \sigma(\mathbf{x})_m^k \; g_m^k(\mathbf{x}), \text{ s.t. } g_m^k(\mathbf{x}) = \mathbf{w}_m^{k\top}\mathbf{x} + \mathbf{b}_m^k,$$

$$\text{and } \sigma_m^k(\mathbf{x}) = \frac{\exp\left[\beta \cdot g_m^k(\mathbf{x})\right]}{\sum_{m=1}^{M} \exp\left[\beta \cdot g_m^k(\mathbf{x})\right]}. \tag{3}$$

One way to choose the weighting coefficients is by assigning a probability to each linear map using the softmax function as in Eq. (3). We introduce a global inverse temperature parameter β in the softmax to tune the sensitivity of the relative weighting – large β (small temperature) favors single filters; small β (high temperature) favors several filters. The softmax weighting is useful for avoiding non-identifiability issues of parameters in mixture models. More importantly, the softmax weighting assigns an importance to each map particular to this example. In other words, it provides rankings of filters for different examples even if they are classified as the same class. We revisit this point in Sec. 4. We train the LLM by optimizing the following (standard) cross-entropy loss:

$$\mathcal{L}(\mathbf{W}, \mathcal{D}) = -\sum_{n=1}^{N}\sum_{k=1}^{K} y_{n,k} \log \hat{y}_{n,k}(\mathbf{W}), \tag{4}$$

where we denote the parameters of LLM collectively by $\mathbf{W}$, and we define the predictive class label by the mapping from the pre-activation through another softmax function.

$$\hat{y}_{n,k}(\mathbf{W}) = \exp(f_k(\mathbf{x}_n))/[\sum_{k'=1}^{K} \exp(f_{k'}(\mathbf{x}_n))] \tag{5}$$

When the number of filters per class is one, this reduces to logistic regression; increasing the number of filters adds expressive capacity to each class. The classification is approximately linear at the location of the input, which means that locally each model decision from a certain input can be explained using only the active filters, as we illustrate in the remainder of the paper. In addition the shallow nature of the model lends itself to global interpretability, as the filter-bank for each class is easily accessible and provides an overview of the inputs this class is sensitive to.

LLM as neural network approximations. One interpretation of LLM is linearizations of neural networks. Suppose we trained a neural network model on a K-class classification problem, where the network maps a high dimensional input $\mathbf{x} \in \mathbb{R}^D$ to a class score function $\mathbf{s}(\mathbf{x})$, i.e., the pre-activation before the final softmax, where $\mathbf{s}(\mathbf{x})$ is a K-dimensional vector with entries s_k. Denote the mapping $\boldsymbol{\phi} : \mathbf{x} \mapsto \mathbf{s}(\mathbf{x})$ and the parameters of the network by $\boldsymbol{\theta}$. We would like to find the best approximation to the function $\boldsymbol{\phi}$, which presents interpretable features for classification and also guarantees a certain level of privacy. For this, we take inspiration from gradient-based attribution methods for deep neural networks (Ancona et al. 2017). These methods assume a set of attributions, at which the gradients of a classifier with respect to the input are maximized, *and* that the gradient information provides interpretability as to why the classifier makes a certain prediction. More specifically, these consider a first order Taylor approximation of $\boldsymbol{\phi}$,

$$\boldsymbol{\phi}(\mathbf{x}) \approx \boldsymbol{\phi}(\mathbf{x}_0) + \boldsymbol{\phi}'(\mathbf{x}_0)^\top(\mathbf{x} - \mathbf{x}_0) = \boldsymbol{\phi}'(\mathbf{x}_0)^\top\mathbf{x} + \boldsymbol{\alpha},$$

where $\boldsymbol{\phi}'(\mathbf{x}_0) = \left[\frac{\partial}{\partial \mathbf{x}}\boldsymbol{\phi}(\mathbf{x})\right]_{\mathbf{x}=\mathbf{x}_0}$, and shift term $\boldsymbol{\alpha} = \boldsymbol{\phi}(\mathbf{x}_0) - \boldsymbol{\phi}'(\mathbf{x}_0)^\top\mathbf{x}_0$. Therefore, finding *good* input locations

$\mathbf{x}_0$ to make the first order approximation to the function ϕ and using their gradient information would reveal the discriminative features of a given classifier.

There are two problems in directly using this approach. First, it is challenging to identify which input points (and how many of them) are *informative* to make an interpretable linear approximation of the classifier. Second, directly using ϕ and its gradients violates privacy, as ϕ contains sensitive information about individuals from the training dataset. Privatizing ϕ requires computing the sensitivity, which determines an appropriate amount of noise to add (see Sec. 2). In case of deep neural network models, we cannot analytically identify one datapoint's contribution to the learned function ϕ appeared in Eq. (2). Thus, we cannot use the raw function ϕ and its gradients, unless we privatize the parameters of ϕ[1]. For these reasons, extracting a private approximation of ϕ is difficult and we instead opt to train a model of the same form from scratch, leading us to LLMs, as described above.

3.1 Differentially private LLM

To produce differentially private LLM parameters $\widetilde{\mathbf{W}}$, we adopt the moments accountant method combined with the gradient-perturbation technique, called, *differentially private stochastic gradient descent* (DP-SGD) (Abadi et al. 2016). In our work, we perturb gradients at each learning step when optimizing Eq. (4) for all LLM parameters $\mathbf{W}$; and compute the cumulative privacy loss using the moments accountant method.

When we perturb the gradient, we must ensure to add the right amount of noise. As there is no way of knowing how much change a single datapoint would make in the gradient's L2-norm, we rescale all the datapoint-wise gradients, $\mathbf{h}_t(\mathbf{x}_n) := \nabla_{\mathbf{W}}\mathcal{L}(\mathbf{W}, \mathcal{D}_n)$ for all $n = \{1, \cdots, N\}$, by a pre-defined norm clipping threshold, C, as used in (Abadi et al. 2016), i.e., $\bar{\mathbf{h}}_t(\mathbf{x}_n) \leftarrow \mathbf{h}_t(\mathbf{x}_n)/\max(1, \|\mathbf{h}_t(\mathbf{x}_i)\|_2/C)$. Algorithm 1 summarizes this procedure. We formally state that the resulting LLM is DP in Theorem 3.1.

Algorithm 1 DP-LLM for interpretable classification

Require: Dataset $\mathcal{D}$, norm-clipping threshold C, privacy parameter σ^2, and learning rate η_t

Ensure: (ϵ, δ)-DP locally linear maps for all K classes, $\widetilde{\mathbf{W}}$

for number of training steps $t \leq T$ **do**

1: For each minibatch of size L, we noise up the gradient after clipping the norm of the datapoint-wise gradient via $\tilde{\mathbf{h}}_t \leftarrow \frac{1}{L}\left[\sum_{n=1}^{L} \bar{\mathbf{h}}_t(\mathbf{x}_n) + \mathcal{N}(0, \sigma^2 C^2 \mathbf{I})\right]$.

2: Then, we make a step in the descending direction by $\widetilde{\mathbf{W}}_{t+1} \leftarrow \widetilde{\mathbf{W}}_t - \eta_t \tilde{\mathbf{h}}_t$.

end for

Calculate the cumulative privacy loss (ϵ, δ) using the moments accountant.

Theorem 3.1. *Algorithm 1 produces (ϵ, δ)-DP locally linear maps for all K classes.*

Proof: Given an initial *data-independent* value of $\mathbf{W}_0$, if we add Gaussian noise to the norm-clipped gradient evaluated on the subsampled data with the sampling rate $q = L/N$, then due to the *Gaussian mechanism* (Theorem 3.22 in (Dwork and Roth 2014)), the resulting estimate $\tilde{\mathbf{W}}_1$ from a single gradient step (i.e., the step **2** in Algorithm 1) is (ϵ', δ')-DP, where $\sigma \geq c \cdot q\sqrt{\log(1/\delta')}/\epsilon'$ with some constant c. The cumulative privacy loss after T repetitions of step **2** in Algorithm 1 is (ϵ, δ)-DP, which is a direct consequence of Theorem 1 in (Abadi et al. 2016).

Improving privacy and accuracy trade-off under LLM For high-dimensional inputs such as images, we found that adding noise to the gradient corresponding to the full dimension of $\mathbf{W}$ led to very low accuracies for private training. Therefore, we propose to incorporate the random projection matrix $\mathbf{R}_m \in \mathbb{R}^{D' \times D}$ with $D' \ll D$, which is shared among all classes k, to first decrease the dimensionality of the parameters that need to be privatized. Each LLM is therefore parameterized as $\mathbf{w}_m^k = \mathbf{m}_m^k \mathbf{R}_m$, where the effective parameter for each local linear map is $\mathbf{m}_m^k \in \mathbb{R}^{D'}$. We perturb the gradient of $\mathbf{m}_m^k$ for all k and m in each training step in Algorithm 1 to produce DP linear maps, $\widetilde{\mathbf{w}}_m^k = \widetilde{\mathbf{m}}_m^k \mathbf{R}_m$.

Due to the post-processing invariance property, we can use the differentially private LLM to make predictions on test data. Here we focus on guarding the training data's privacy and assume that the test data do not need to be privatized, which is a common assumption in DP literature.

4 Experiments

In this section we evaluate the trade-off between accuracy, privacy, and interpretability for our LLM model on several datasets and compare to other methods where it is appropriate. Our implementation is available on GitHub[2].

4.1 MNIST Classification

Problem. We consider the classification of MNIST (LeCun and Cortes 2010) and Fashion-MNIST (Xiao, Rasul, and Vollgraf 2017) images with the usual train/test splits and train a CNN[3] as a baseline model, which has two convolutional layers with 5x5 filters and first 20, then 50 channels each followed by max-pooling and finally a fully connected layer with 500 units. The model achieves 99% test accuracy on MNIST and 87% on Fashion-MNIST.

Setup. We train several LLMs in the private and non-private setting. By default, we use LLM models with $M = 30$ filters per class and random projections to $D' = 300$ dimensions, which are optimized for 20 epochs using the Adam optimizer with learning rate 0.001, decreasing by 20% every 5 epochs. On MNIST the model benefits from a decreased inverse softmax temperature $\beta = 1/30$, while $\beta = 1$ is optimal for Fashion-MNIST. We choose a large

[1] Once ϕ is privatized, we can safely use post-processing methods for interpretability. A comparison to this is shown in Sec. 4.

[2] github.com/frhrdr/dp-llm

[3] github.com/pytorch/examples/blob/master/mnist/main.py

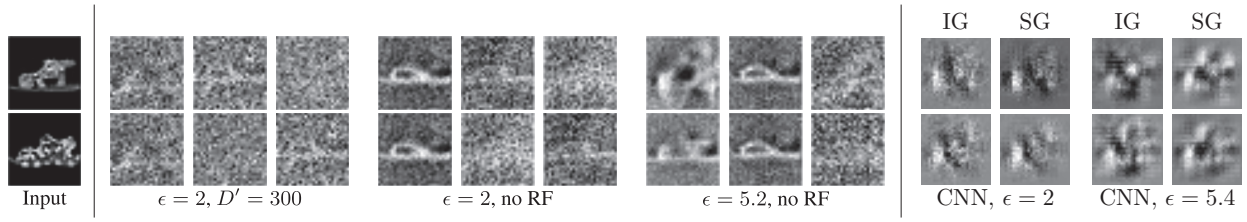


Figure 2: For 2 test inputs (left) and 3 different DP-LLM setups (groups of 3 columns), we show the 3 highest activated filters in desending order. We look at the default setting with random filters at $D' = 300$ and $\epsilon = 2$ (center left), the same setting without random filters (center), and at a lower privacy $\epsilon = 5.2$ setting (center right). Attribution plots from DP-CNNs at matching privacy levels on the same input are shown for comparison (right).

batch size of 500, as this improves the signal-to-noise ratio of our algorithm. In the private setting we clip the per-sample gradient norm to $C = 0.001$ and train with $\sigma = 1.3$, which gives this model an $(\epsilon = 2, \delta = 10^{-5})$-DP guarantee via the moments accountant. For the low privacy regime $\epsilon \geq 4$ we train with a batch size of 1500 and for 60 epochs.

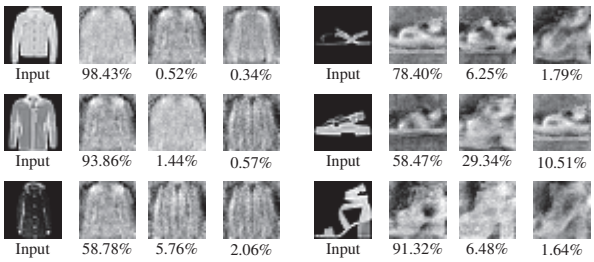


Figure 3: Top 3 filters with associated weightings for test images from two classes. In most cases, a single filter dominates the softmax selection for the class.

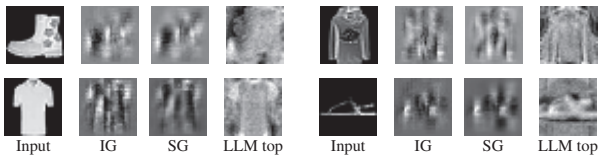


Figure 4: Comparison of CNN and LLM interpretability. *left to right*: input image, integrated gradient (IG) of CNN, smoothed gradient (SG) of CNN, and the top filter of LLM. The CNN attribution match well but aren't as easy to interpret as the simple filter.

Inherent interpretability. In order to highlight the interpretability of the LLM architecture, we compare learned filters of our model to two attribution methods applied to a neural network trained on the same data. We train a simple CNN and an LLM on Fashion-MNIST to matching 87% test accuracy and then visualize the CNN's sensitivity to test images using SmoothGrad (SG) (Smilkov et al. 2017) and integrated gradients (IG) (Sundararajan, Taly, and Yan 2017) and compare these methods to LLM filters in Fig. 4. Note that we do not multiply the integrated gradient with the input image, as Fashion-MNIST images have a mask-like effect which occludes the partial output of the method. We observe that both alternative attribution methods produce similar outputs, which are nonetheless hard to interpret, whereas the LLM filters show simplistic prototype images of the corresponding classes. This is further illustrated in Fig. 3 where we show the three highest weighted filters for test images from three classes. The diversity of filters varies for different class labels, as some are more varied and harder to discriminate than others. For instance, while the sandal class (right) has filters which distinguish between different types of heels, the coat filters (left) are mostly selective in the shoulder region and general silhouette, which is sufficient for classifying a majority of the inputs correctly, but some filters also track other features like arms, collar and zipper. The relevance weights for each filter show that in most cases, the top filter is assigned almost all the weight, indicating that the softmax is a good approximation of the maximum and the class features are indeed approximately linear locally.

Trade-offs with interpretability (Fig. 1 (A) and (B)). We investigate the learned LLM filters under increasing privacy guarantees and increased private utility as shown in Fig. 2. For two test inputs we plot the filters with highest activation in three DP setups. We compare the default setting with random filters at $D' = 300$ and $\epsilon = 2$, the same setting without random filters, and a lower privacy setting trained with $\epsilon = 5.2$. The default setting optimizes privacy and utility at the expense of interpretability. As the figure shows, removing the random projections and reducing the level of privacy gradually restores interpretability of the filters. On the very right, we show attribution plots from CNNs trained with DP-SGD at $\epsilon = 2$ and $\epsilon = 5.4$ on the same input for reference. When comparing to Fig. 4, one can see that the quality of the CNN attributions is diminished by the privacy constraints as well, to the point where it is hard to make out any connection to the input image.

Privacy vs. accuracy. (Fig. 1 (C)) In Fig. 5 (left) we show the trade-off of privacy strength and accuracy in our model. Note that current privatized network methods (Abadi

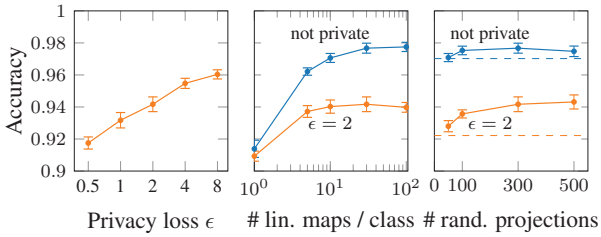


Figure 5: Accuracy of our LLM model on the MNIST testset for different levels of privacy and different model configurations in the private (orange) and non-private (blue) setting. Errorbars are 2 stdev from 10 random restarts; dashed lines on the right denote no random projections.

et al. 2016; Phan, Wu, and Dou 2017) achieve an accuracy of 95% for $\epsilon = 2$ and up to 92% for $\epsilon = 0.5$, which is comparable to our mean accuracy of $94.2 \pm 0.4\%$ and $91.8 \pm 0.4\%$ respectively (on Fashion-MNIST we achieve $80.7 \pm 0.6\%$ and $83.2 \pm 0.4\%$). However, such a privatized network does not provide transparent explanations as opposed to our approach. Another popular reference model is the PATE method (Papernot et al. 2018), which trains a student model to 98% at $\epsilon = 2$ on MNIST using an ensemble of teachers and additional public data. We do not consider this setup here, as the accuracy of the teacher votes alone lies at 94.4% in the nonprivate setting, highlighting the importance of the additional data. In the remainder of Fig. 5 we study the impact of varying the number of filters per class M (center) and the output dimensionality of the random projections D' (right) in private and non-private LLM models. Private LLMs deteriorate beyond a certain number of linear maps due to the increased noise needed to privatize them, whereas non-private models continue to benefit from additional filters. Increasing the dimensionality of the random projections benefits private training.

4.2 Disease classification in a medical dataset

Problem. As a second task we consider disease classification in the Henan Renmin Hospital Data (Li et al. 2017; Maxwell et al. 2017)[4]. It contains 110,300 medical records with 62 input features and 3 binary outputs. The input features are 4 basic examinations (sex, BMI, distolic, systolic), 26 items from blood examinations, 12 items from urine examinations, and 20 items from liver function tests. The three binary outputs denote three medical conditions – hypertension, diabetes, and fatty liver – which can also co-occur. Following (Maxwell et al. 2017) we transform this multi-label task into a multi-class problem by considering the powerset of the three binary choices as eight independent classes. Because these classes are highly imbalanced, we only retain the four most common classes, leaving us with 100,140 records.

Setup. By default, we use an LLM model with $M = 2$ filters per class and no random projections, which is optimized for 20 epochs using the Adam optimizer with learning rate 0.01, decreasing by 20% every 5 epochs. We choose a batch size of 256. In the private setting we clip the per-sample gradient norm to 0.001 and train with $\sigma = 1.25$, which gives this model an $(\epsilon \approx 1.5, \delta = 2 \cdot 10^{-5})$-DP guarantee.

[4]downloaded from http://pinfish.cs.usm.edu/dnn/

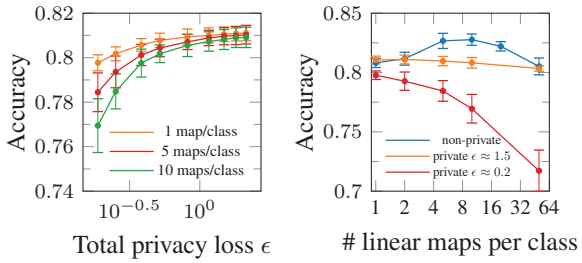


Figure 6: Accuracy of our LLM model on the Henan Renmin Hospital testset for different levels of privacy and different model configurations in the private and non-private setting. Errorbars are 2 stdev for 10 random restarts.

We train a baseline DNN (3 dense hidden layers with 128 units each) as well as several LLMs with varying number of linear filters per class in private and non-private settings. In Fig. 6 we visualize the trade-off between accuracy and privacy for varying privacy losses as well as numbers of linear maps. Like before, the accuracy deteriorates as we decrease the privacy loss (Fig. 6 *top*). As the number of linear maps per class is increased (Fig. 6 *bottom*), the accuracy for the private models also drops due to the privacy budget being spread across more parameters. We attribute the drop in performance for the non-private LLM with number of maps to optimization difficulties and local minima as well as higher sensitivity to hyperparameters. A small number of maps (between 2 and 5) is sufficient for this datasets, especially in the private setting. Our LLMs attain $82.8 \pm 0.5\%$ (non-private), $82.0 \pm 0.4\%$ ($\epsilon \approx 1.5$), and $79.8 \pm 0.4\%$ ($\epsilon \approx 0.2$) compared to $84 \pm 0.5\%$ for a non-private DNN.

In Fig. 7 we consider an example from each class and show the weighted linear maps by the LLM for each example as well as its integrated gradients (IG) (Sundararajan, Taly, and Yan 2017). For our LLM we consider the non-private and two private cases. In general, there is good agreement between all attribution methods; they are relatively sparse and focus on a small set of features. We found that IG varied much more between examples from the same class than our LLM. For strong privacy ($\epsilon < 0.1$), the linear maps are much less sparse, highlighting the trade-off between interpretability and privacy.

5 Related Work

Interpretability. The *saliency* map and *gradient-based attribution* methods are one of the most popular explanation methods that identify relevant regions and assign importance to each input feature (e.g., pixel for image data) (Selvaraju et al. 2016; Ribeiro, Singh, and Guestrin 2016; Smilkov et al. 2017; Sundararajan, Taly, and Yan 2017; Montavon et al. 2015; Bach et al. 2015; Ancona et al. 2017).

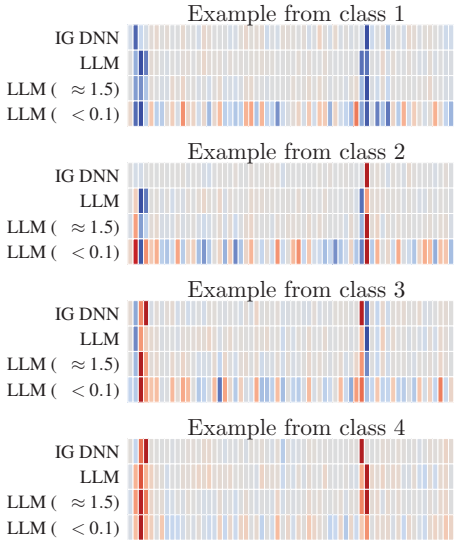


Figure 7: Integrated gradient (IG) and weighted linear filters (LLM; our method) for all 62 feature for one example from each class from the Henan Renmin dataset. For LLM we consider the non-private case (LLM) as well as two private cases with strong ($\epsilon < 0.1$) and weaker ($\epsilon \approx 1.5$) privacy. Entries are normalized and colorcoded between blue $= -1$, gray $= 0$, and red $= 1$.

These methods typically use first-order gradient information of a complex model with respect to inputs, to produce maps that indicate the relative importance of the different input features for the classification. An obvious downside of these approaches is that they provide explanations conditioned on only *a single* input and hence it is necessary to manually assess each input of interest in order to draw a class-wide conclusion. In contrast, our approach can draw class-wide conclusions without manually assessing each input, because it outputs the most relevant explanations in terms of a collection of linear maps for each class. For explanations conditioned on any specific input, our model can provide an input-dependent weighted collection of these features related to that specific input.

Privacy. To privatize complex models, such as deep neural networks, a popular approach is to add noise to the gradients in the stochastic gradient descent (SGD) algorithm (Abadi et al. 2016; Papernot et al. 2017; McMahan et al. 2017). An alternative approach is to directly perturb the objective with additive noise (Zhang et al. 2012; Phan et al. 2016; Phan, Wu, and Dou 2017). In these works, the objective function is approximated by the Taylor expansion, and the resulting coefficients of the polynomials are perturbed before training. We found the latter approach less practical than the former, as we need to choose which order of polynomial degree to use. Typically, adding more layers introduces a more nested-ness in the objective function requiring a higher order polynomial for accurate approximation. A high degree of polynomial approximation, however, increases the privacy loss as the dimensionality of the coefficients grow. From our perspective, the gradient perturbation method is simple to use and model agnostic. Recently proposed methods for private training through ensembles of teacher models (Papernot et al. 2017; 2018) are less useful to us here, as they consider a special setting where some non-private data is available in addition to the private dataset. Our method distinguishes itself by making interpretability a key component of the trained model and does not rely on access to additional public data.

Lastly, LLMs are reminiscent of *Mixture of experts* (ME) models. MEs assign different specialized linear models to different parts of input space in a discriminative task (see Masoudnia and Ebrahimpour (2014) for an overview of existing ME models). In our case, each local expert model is class specific and contributes to a weighted linear map for that class. The weighting provides an input-dependent *significance* for each linear map, and considering more than one map per class increases flexibility to fit the data better. *Mixtures of factor analyzers* (MFA) are also similar to ME models, but for density estimation of high-dimensional real-valued data (Ghahramani and Hinton 1997)

6 Conclusion and Discussion

We proposed a family of simple models which uses several *locally linear maps* (LLM) per class to provide interpretable features in a privacy-preserving manner while maintaining high classification accuracy. Results on two image benchmark datasets as well as a medical dataset indicate that a reasonable trade-off between classification accuracy, privacy *and* interpretability can indeed be struck and tuned by varying the number of linear maps. Nevertheless, several open questions for future research remain. First, the datasets in this paper are still relatively simple, such that it would be intriguing to see the limits of complexity the LLM model can model with a sufficiently high accuracy. Second, the current model does not interact with a larger and richer counterpart, such as a neural network, due to privacy constraints. It would be interesting to investigate if gaining gradient information of a more flexible model at particularly important input points in a DP way would be possible, in order to combine benefits of both models.

Acknowledgments

M. Park and F. Harder are supported by the Max Planck Society and the Gibs Schüle Foundation and the Institutional Strategy of the University of Tübingen (ZUK63). F. Harder thanks the International Max Planck Research School for Intelligent Systems (IMPRS-IS) for its support. M. Bauer gratefully acknowledges partial funding through a Qualcomm studentship in technology as well as by the EPSRC. We thank Maryam Ahmed for useful comments on the manuscript, Patrick Putzky and Wittawat Jitkritum for their scientific insights, and Been Kim and Isabel Valera for their inputs on interpretability research. M. Park is grateful to Gilles Barthe for the inspiration of studying the trade-off between interpretability and privacy.

References

Abadi, M.; Chu, A.; Goodfellow, I.; Brendan McMahan, H.; Mironov, I.; Talwar, K.; and Zhang, L. 2016. Deep learning with differential privacy. *ArXiv e-prints*.

Ancona, M.; Ceolini, E.; Öztireli, A. C.; and Gross, M. H. 2017. A unified view of gradient-based attribution methods for deep neural networks. *CoRR* abs/1711.06104.

Bach, S.; Binder, A.; Montavon, G.; Klauschen, F.; Müller, K.-R.; and Samek, W. 2015. On pixel-wise explanations for non-linear classifier decisions by layer-wise relevance propagation. *PLOS ONE* 10(7):1–46.

Blocki, J.; Blum, A.; Datta, A.; and Sheffet, O. 2012. The johnson-lindenstrauss transform itself preserves differential privacy. *2012 IEEE 53rd Annual Symposium on Foundations of Computer Science* 410–419.

Carlini, N.; Liu, C.; Kos, J.; Erlingsson, Ú.; and Song, D. 2018. The secret sharer: Measuring unintended neural network memorization & extracting secrets. *CoRR* abs/1802.08232.

Dwork, C., and Roth, A. 2014. The algorithmic foundations of differential privacy. *Found. Trends Theor. Comput. Sci.* 9:211–407.

Fredrikson, M.; Jha, S.; and Ristenpart, T. 2015. Model inversion attacks that exploit confidence information and basic countermeasures. In *Proceedings of the 22Nd ACM SIGSAC Conference on Computer and Communications Security*, CCS '15, 1322–1333. New York, NY, USA: ACM.

Ghahramani, Z., and Hinton, G. E. 1997. The em algorithm for mixtures of factor analyzers. Technical report, University of Toronto.

Goodman, B., and Flaxman, S. 2016. European Union regulations on algorithmic decision-making and a "right to explanation". *arXiv e-prints* arXiv:1606.08813.

Kenthapadi, K.; Korolova, A.; Mironov, I.; and Mishra, N. 2013. Privacy via the johnson-lindenstrauss transform. *Journal of Privacy and Confidentiality* 5(1).

LeCun, Y., and Cortes, C. 2010. MNIST handwritten digit database.

Li, R.; Liu, W.; Lin, Y.; Zhao, H.; and Zhang, C. 2017. An ensemble multilabel classification for disease risk prediction. *Journal of healthcare engineering* 2017.

Masoudnia, S., and Ebrahimpour, R. 2014. Mixture of experts: a literature survey. *Artificial Intelligence Review* 42(2):275–293.

Maxwell, A.; Li, R.; Yang, B.; Weng, H.; Ou, A.; Hong, H.; Zhou, Z.; Gong, P.; and Zhang, C. 2017. Deep learning architectures for multi-label classification of intelligent health risk prediction. *BMC Bioinformatics* 18(14):523.

McMahan, H. B.; Ramage, D.; Talwar, K.; and Zhang, L. 2017. Learning differentially private language models without losing accuracy. *CoRR* abs/1710.06963.

Montavon, G.; Bach, S.; Binder, A.; Samek, W.; and Müller, K. 2015. Explaining nonlinear classification decisions with deep taylor decomposition. *CoRR* abs/1512.02479.

Papernot, N.; Abadi, M.; Erlingsson, U.; Goodfellow, I.; and Talwar, K. 2017. Semi-supervised Knowledge Transfer for Deep Learning from Private Training Data. In *Proceedings of the International Conference on Learning Representations (ICLR)*.

Papernot, N.; Song, S.; Mironov, I.; Raghunathan, A.; Talwar, K.; and Erlingsson, U. 2018. Scalable private learning with PATE. In *International Conference on Learning Representations*.

Phan, N.; Wang, Y.; Wu, X.; and Dou, D. 2016. Differential privacy preservation for deep auto-encoders: An application of human behavior prediction. In *Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence*, AAAI'16, 1309–1316. AAAI Press.

Phan, N.; Wu, X.; and Dou, D. 2017. Preserving Differential Privacy in Convolutional Deep Belief Networks. *ArXiv e-prints*.

Ribeiro, M. T.; Singh, S.; and Guestrin, C. 2016. "why should i trust you?": Explaining the predictions of any classifier. In *Proceedings of the 22Nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '16, 1135–1144. New York, NY, USA: ACM.

Selvaraju, R. R.; Das, A.; Vedantam, R.; Cogswell, M.; Parikh, D.; and Batra, D. 2016. Grad-cam: Why did you say that? visual explanations from deep networks via gradient-based localization. *CoRR* abs/1610.02391.

Shokri, R., and Shmatikov, V. 2015. Privacy-preserving deep learning. In *2015 53rd Annual Allerton Conference on Communication, Control, and Computing (Allerton)*, 909–910.

Smilkov, D.; Thorat, N.; Kim, B.; Viégas, F. B.; and Wattenberg, M. 2017. Smoothgrad: removing noise by adding noise. *CoRR* abs/1706.03825.

Song, C.; Ristenpart, T.; and Shmatikov, V. 2017. Machine learning models that remember too much. In *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security*, CCS '17, 587–601. New York, NY, USA: ACM.

Sundararajan, M.; Taly, A.; and Yan, Q. 2017. Axiomatic attribution for deep networks. *CoRR* abs/1703.01365.

Voigt, P., and Bussche, A. v. d. 2017. *The EU General Data Protection Regulation (GDPR): A Practical Guide*. Springer Publishing Company, Incorporated, 1st edition.

Xiao, H.; Rasul, K.; and Vollgraf, R. 2017. Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms. *ArXiv* abs/1708.07747.

Zhang, J.; Zhang, Z.; Xiao, X.; Yang, Y.; and Winslett, M. 2012. Functional Mechanism: Regression Analysis under Differential Privacy. *ArXiv e-prints*.

B

DP-MERF: DIFFERENTIALLY PRIVATE MEAN EMBEDDINGS WITH RANDOM FEATURES FOR PRACTICAL PRIVACY-PRESERVING DATA GENERATION

1 https://creativecommons.org/licenses/by/4.0/

DP-MERF: Differentially Private Mean Embeddings with Random Features for Practical Privacy-Preserving Data Generation

Frederik Harder[*1,2] **Kamil Adamczewski**[*1,3] **Mijung Park**[1,2]
[1] Max Planck Institute for Intelligent Systems, Tübingen, Germany
[2] Department of Computer Science, University of Tübingen, Tübingen, Germany
[3] D-ITET, ETH Zurich, Switzerland
{fharder|kadamczewski|mpark}@tue.mpg.de

Abstract

We propose a differentially private data generation paradigm using random feature representations of kernel mean embeddings when comparing the distribution of true data with that of synthetic data. We exploit the random feature representations for two important benefits. First, we require a minimal privacy cost for training deep generative models. This is because unlike kernel-based distance metrics that require computing the kernel matrix on all pairs of true and synthetic data points, we can detach the data-dependent term from the term solely dependent on synthetic data. Hence, we need to perturb the data-dependent term only once and then use it repeatedly during the generator training. Second, we can obtain an analytic sensitivity of the kernel mean embedding as the random features are norm bounded by construction. This removes the necessity of hyper-parameter search for a clipping norm to handle the unknown sensitivity of a generator network. We provide several variants of our algorithm, differentially-private mean embeddings with random features (DP-MERF) to jointly generate labels and input features for datasets such as heterogeneous tabular data and image data. Our algorithm achieves drastically better privacy-utility trade-offs than existing methods when tested on several datasets.

Proceedings of the 24[th] International Conference on Artificial Intelligence and Statistics (AISTATS) 2021, San Diego, California, USA. PMLR: Volume 130.
* Equal contribution.

1 Introduction

Differential privacy (DP) is a gold standard privacy notion that is widely used in many applications in machine learning. However, due to its composability, every access to data reduces the privacy guarantee, which limits the number of times one can query sensitive data before a desired privacy level is exceeded. Differentially private data generation solves this problem of limited access by creating a synthetic dataset that is *similar* to the true dataset using DP mechanisms. This process also comes at a privacy cost, but afterwards, the synthetic dataset can be used in place of the true one for unlimited time without further loss of privacy.

Classical approaches to differentially private data generation typically assume a certain class of pre-specified queries. These DP algorithms produce a privacy-preserving synthetic database that is similar to the privacy-sensitive original data for that *fixed query class* [17, 34, 13, 40]. However, specifying a query class in advance, significantly limits the flexibility of the synthetic data, if data analysts hope to perform other machine learning tasks.

To overcome this inflexibility, recent papers on DP data generation have utilized deep generative modelling. The majority of these approaches is based on the generative adversarial networks (GAN) [11] framework, where a discriminator and a generator play a min-max form of game to optimize a given distance metric between the true and synthetic data distributions. Most approaches have used either the *Jensen-Shannon divergence* [20, 30, 36], or the Wasserstein distance [35, 9]. For more details on different divergence metrics, see Supplementary Sec. A.

Another popular choice of distance metric for generative modelling is Maximum Mean Discrepancy (MMD). MMD can compare two probability measures in terms of all possible moments. Therefore, there is no information loss due to a selection of a certain set of moments. The MMD estimator is in closed form (eq. 2) and easy to compute by the pair-wise evaluations of a kernel function using the points drawn from the true and the generated data distributions.

In this work, we propose to use a particular form of MMD via *random Fourier feature* representations [22] of kernel mean embeddings for DP data generation. While MMD can be used within a GAN framework as well (see e.g. [14]) we choose a much simpler method, which is particularly suited for training with DP constraints.

In the objective we use (eq. 3), the mean embedding of the true data distribution (data-dependent) is separate from the embedding of the synthetic data distribution (data-independent). Hence, only the data-dependent term requires privatization. Random features provide an analytic sensitivity of the mean embedding, which allows us to release a DP version of this embedding through a DP mechanism as we explain below. With the privatized data embedding and the synthetic data embedding, our objective no longer directly accesses the data and can be optimized freely to train a data generator. Our contributions are summarized below.

(1) We provide a simple algorithm for DP data generation, which improves on existing methods both in privacy and utility.

- *Simple to optimize:* Since the objective of the optimization contains only a specific private release of data, there are no privacy induced constraints on model choice and optimization method due to privacy. In contrast, methods with private releases as part of the training loop are generally constrained in the number of iterations. As a specific example, DP-SGD requires well-defined sample-wise gradients, which prohibits the use of batch-normalization. Further, increasing the number of trained weights raises the sensitivity of DP-SGD [2] and with it the required strength of gradient perturbation, making large networks infeasible. Our method also avoids the cumbersome min-max optimization present in GAN based approaches and requires only a minimal number of hyperparameters[1].
- *Strong privacy:* Computing the *sensitivity* in our method is *analytically tractable* due to its norm-boundedness of random features. In fact, the norm of random features we use is bounded by 1 by construction. The resulting sensitivity is on the order of 1 over the number of training data points. Consequently, a moderate size of training data can significantly reduce the sensitivity. By requiring only a single DP-release with such a low sensitivity, our method can provide strong DP guarantee more easily than methods which access the data on each training iteration.
- *High utility:* We show in our experiments that our method releases private data with higher utility for downstream tasks than comparison methods. This contrast is particularly stark on MNIST, where our model at a strong privacy guarantee of $(0.2, 10^{-5})$-DP outperforms all GAN-based comparison methods, even though they are trained with much weaker privacy of at most $(9.6, 10^{-5})$-DP.
- *Theoretical study:* We provide an error bound on the objective to theoretically quantify the effect of noise added for privacy to the random feature representation of MMD objective. This bound provides an informative way to select the random feature dimension, given a dataset size and a desired privacy level.

(2) Our algorithm accommodates several needs in privacy-preserving data generation.

- *Generating input and output pairs jointly*: We treat both input and output to be privacy-sensitive. This is different from the conditional-GAN type of methods, where the class distribution is treated as non-sensitive, which increases the risk of successful membership inference, particularly in imbalanced datasets where some classes contain only a small number of samples.
- *Generating imbalanced and heterogeneous tabular data*: Real world datasets may exhibit large variation in data types and class sizes. By addressing both of these issues, we ensure that our algorithm is applicable to a wide variety of datasets.

We start by describing relevant background information in Sec. 2 before introducing our method in Sec. 3 and Sec. 4, followed by an overview of related work in Sec. 5 and experiments in Sec. 6.

2 Background

In the following, we describe the kernel mean embeddings with random features and differential privacy, which our model will use in Sec. 3.

2.1 Maximum Mean Discrepancy

Given a positive definite kernel $k\colon \mathcal{X} \times \mathcal{X}$, the MMD between two distributions P, Q is defined as [12]

$$\mathrm{MMD}^2(P,Q) = \mathbb{E}_{x,x'\sim P} k(x,x') + \mathbb{E}_{y,y'\sim Q} k(y,y') - 2\mathbb{E}_{x\sim P}\mathbb{E}_{y\sim Q} k(x,y). \quad (1)$$

According to the Moore–Aronszajn theorem, there exists a unique Hilbert space $\mathcal{H}$ on which k defines an inner product. Hence, we can find a feature map $\phi\colon \mathcal{X} \to \mathcal{H}$ such that $k(x,y) = \langle \phi(x), \phi(y) \rangle_{\mathcal{H}}$, where $\langle \cdot,\cdot \rangle_{\mathcal{H}} = \langle \cdot,\cdot \rangle$ denotes the inner product on $\mathcal{H}$. Using this fact, we can rewrite the MMD in eq. 1 as [12]

$$\mathrm{MMD}(P,Q) = \left\| \mathbb{E}_{x\sim P}[\phi(x)] - \mathbb{E}_{y\sim Q}[\phi(y)] \right\|_{\mathcal{H}},$$

where $\mathbb{E}_{x\sim P}[\phi(x)] \in \mathcal{H}$ is known as the (kernel) mean embedding of P, and exists if $\mathbb{E}_{x\sim P}\sqrt{k(x,x)} < \infty$ [25]. The

[1] Hyperparameters in our method are the number of random features, a kernel parameter, and the learning rate.

MMD can be interpreted as the distance between the mean embeddings of the two distributions. If k is a characteristic kernel [26], then $P \mapsto \mathbb{E}_{x\sim P}[\phi(x)]$ is injective, and MMD forms a metric, implying that $\mathrm{MMD}(P, Q) = 0$, if and only if $P = Q$.

Given the samples drawn from two probability distributions: $X_m = \{x_i\}_{i=1}^m \sim P$ and $X'_n = \{x'_i\}_{i=1}^n \sim Q$, we can estimate[2] the MMD by sample averages [12]:

$$\widehat{\mathrm{MMD}}^2(X_m, X'_n) = \frac{1}{m^2}\sum_{i,j=1}^{m} k(x_i, x_j) + \frac{1}{n^2}\sum_{i,j=1}^{n} k(x'_i, x'_j) - \frac{2}{mn}\sum_{i=1}^{m}\sum_{j=1}^{n} k(x_i, x'_j). \quad (2)$$

However, the total computational cost of $\widehat{\mathrm{MMD}}(X_m, X'_n)$ is $O(mn)$, which is prohibitive for large-scale datasets.

2.2 Random feature mean embeddings

A fast linear-time MMD estimator can be achieved by considering an approximation to the kernel function $k(x, x')$ with an inner product of finite dimensional feature vectors, i.e., $k(x, x') \approx \hat{\phi}(x)^\top \hat{\phi}(x')$ where $\hat{\phi}(x) \in \mathbb{R}^D$ and D is the number of features. The resulting approximation of the MMD estimator given in eq. 2 can be computed in $O(m + n)$, i.e., linear in the sample size:

$$\widehat{\mathrm{MMD}}^2_{rf}(P, Q) = \left\| \frac{1}{m}\sum_{i=1}^{m} \hat{\phi}(x_i) - \frac{1}{n}\sum_{i=1}^{n} \hat{\phi}(x'_i) \right\|_2^2, \quad (3)$$

One popular approach to obtaining such $\hat{\phi}(\cdot)$ is based on random Fourier features [22] which can be applied to any translation invariant kernel, i.e., $k(x, x') = \tilde{k}(x - x')$ for some function $\tilde{k}$. According to Bochner's theorem [23], $\tilde{k}$ can be written as $\tilde{k}(x - x') = \int e^{i\omega^\top (x-x')}\, \mathrm{d}\Lambda(\omega) = \mathbb{E}_{\omega\sim\Lambda} \cos(\omega^\top (x - x'))$, where $i = \sqrt{-1}$ and due to positive-definiteness of $\tilde{k}$, its Fourier transform Λ is non-negative and can be treated as a probability measure. By drawing random frequencies $\{\omega_i\}_{i=1}^D \sim \Lambda$, where Λ depends on the kernel, (e.g., a Gaussian kernel k corresponds to normal distribution Λ), $\tilde{k}(x - x')$ can be approximated with a Monte Carlo average. The vector of random Fourier features is given by

$$\hat{\boldsymbol{\phi}}(x) = (\hat{\phi}_1(x), \ldots, \hat{\phi}_D(x))^\top \quad (4)$$

where each coordinate is defined by

$$\hat{\phi}_j(x) = \sqrt{2/D}\, \cos(\omega_j{}^\top x),$$
$$\hat{\phi}_{j+D/2}(x) = \sqrt{2/D}\, \sin(\omega_j^\top x),$$

for $j = 1, \cdots, D/2$. The approximation error due to these random features was studied in [27].

[2]Note that this particular MMD estimator is biased.

2.3 Differential privacy

Given privacy parameters $\epsilon \geq 0$ and $\delta \geq 0$, a mechanism $\mathcal{M}$ is (ϵ, δ)-DP if and only if for all possible sets of mechanism outputs S and all neighbouring datasets $\mathcal{D}$, $\mathcal{D}'$ differing by a single entry, the following equation holds:

$$\Pr[\mathcal{M}(\mathcal{D}) \in S] \leq e^\epsilon \cdot \Pr[\mathcal{M}(\mathcal{D}') \in S] + \delta \quad (5)$$

A DP mechanism guarantees a limit on the amount of information revealed about any one individual in the dataset. Typically this guarantee is achieved by adding randomness to the algorithms' output. Let a function $h : \mathcal{D} \mapsto \mathbb{R}^p$, which is computed on sensitive data $\mathcal{D}$, output a p-dimensional vector. We can add noise to h for privacy, where the level of noise is calibrated to the *global sensitivity* [8], Δ_h, defined by the maximum difference in terms of L_2-norm $||h(\mathcal{D}) - h(\mathcal{D}')||_2$, for neighbouring $\mathcal{D}$ and $\mathcal{D}'$ (i.e. $\mathcal{D}$ and $\mathcal{D}'$ have one sample difference by replacement). The *Gaussian mechanism* that we will use in this paper outputs $\widetilde{h}(\mathcal{D}) = h(\mathcal{D}) + \mathcal{N}(0, \sigma^2\Delta_h^2\mathbf{I}_p)$. The perturbed function $\widetilde{h}(\mathcal{D})$ is (ϵ, δ)-DP, where σ is a function of ϵ and δ. For a single application of the mechanism, $\sigma \geq \sqrt{2\log(1.25/\delta)}/\epsilon$ holds for $\epsilon \leq 1$. The auto-dp package by [31] computes the relationship between ϵ, δ, σ numerically, which we use in our method.

There are two important properties of DP. The *composability* theorem [8] states that the strength of privacy guarantee degrades in a measurable way with repeated use of DP-algorithms. This allows us to combine the results of different private mechanisms in Sec. 4.2 using the advanced composition methods from [32]. Furthermore, the *post-processing invariance* property [8] tells us that the composition of any data-independent mapping with an (ϵ, δ)-DP algorithm is also (ϵ, δ)-DP. This ensures that no analysis of the released synthetic data can yield more information about the real data than what our choice of ϵ and δ allows.

What comes next describes our proposal for privacy-preserving data generation. We first present the vanilla version of our algorithm called, DP-MERF (differentially private mean embeddings with random features).

3 Vanilla DP-MERF for unlabeled data

We first introduce the basic version of our DP-MERF algorithm to learn the distribution of an unlabeled dataset. In this setting, we obtain a data generator by minimizing the random feature representation of MMD, given by

$$\hat{\boldsymbol{\theta}} = \arg\min_{\boldsymbol{\theta}} \widetilde{\mathrm{MMD}}^2_{rf}(P_{\mathbf{x}}, Q_{\tilde{\mathbf{x}}_{\boldsymbol{\theta}}}) \quad (6)$$

where $P_{\mathbf{x}}$ denotes the true data distribution. The samples from Q denoted by $\tilde{\mathbf{x}}$ are drawn from a generative model $\tilde{\mathbf{x}} = G_{\boldsymbol{\theta}}(\mathbf{z})$. The generative model G_θ is parameterized by θ and takes a sample $\mathbf{z} \sim p(\mathbf{z})$ from a known,

data-independent distribution as input. Using the random Fourier features, we arrive at

$$\widetilde{\mathrm{MMD}}^2_{rf}(P_{\mathbf{x}}, Q_{\tilde{\mathbf{x}}_{\boldsymbol{\theta}}}) = \left\| \widetilde{\boldsymbol{\mu}}_P - \widehat{\boldsymbol{\mu}}_Q \right\|_2^2 \tag{7}$$

where the random feature mean embedding of each distribution is denoted by $\widehat{\boldsymbol{\mu}}_P = \frac{1}{m}\sum_{i=1}^m \hat{\boldsymbol{\phi}}(\mathbf{x}_i)$, and $\widehat{\boldsymbol{\mu}}_Q = \frac{1}{n}\sum_{i=1}^n \hat{\boldsymbol{\phi}}(G_{\boldsymbol{\theta}}(\mathbf{z}_i))$.

Notice that $\widehat{\boldsymbol{\mu}}_P$ is the only data-dependent term. Hence, we privatize this term by applying the Gaussian mechanism, defining $\widetilde{\boldsymbol{\mu}}_P$ by

$$\widetilde{\boldsymbol{\mu}}_P = \widehat{\boldsymbol{\mu}}_P + \mathcal{N}(0, \Delta^2_{\widehat{\boldsymbol{\mu}}_P}\sigma^2 I) \tag{8}$$

where the privacy parameter σ is chosen as a function of the privacy budget (ϵ, δ). The sensitivity of $\widehat{\boldsymbol{\mu}}_P$ is analytically tractable due to the triangle inequality and the fact that $\|\hat{\boldsymbol{\phi}}(\cdot)\|_2 = 1$ by construction of the random feature vector given in eq. 4:

$$\Delta_{\widehat{\boldsymbol{\mu}}_P} = \max_{\mathcal{D},\mathcal{D}'} \left\| \tfrac{1}{m}\sum_{i=1}^m \hat{\boldsymbol{\phi}}(\mathbf{x}_i) - \tfrac{1}{m}\sum_{i=1}^m \hat{\boldsymbol{\phi}}(\mathbf{x}'_i) \right\|_2, \tag{9}$$

$$= \max_{\mathbf{x}_n, \mathbf{x}'_n} \left\| \tfrac{1}{m}\hat{\boldsymbol{\phi}}(\mathbf{x}_n) - \tfrac{1}{m}\hat{\boldsymbol{\phi}}(\mathbf{x}'_n) \right\|_2 \leq \tfrac{2}{m}, \tag{10}$$

Due to the post-processing invariance of DP, we can obtain differentially private generator G, since $\widehat{\boldsymbol{\mu}}_Q$ is data-independent.

3.1 Bound on the expected absolute error

If we add noise to the random-feature mean embedding of the data distribution, what is the effect of that noise on the learned generator? Theoretically quantifying this effect is challenging under an arbitrary neural network-based generator. Instead, we theoretically quantify the effect of noise on the objective function. In particular, given samples $\mathbf{x} = \{x_i\}_{i=1}^m \sim P$ and $\tilde{\mathbf{x}} = \{\tilde{x}_j\}_{j=1}^n \sim Q$, we want to bound the expected absolute error between the noisy random-feature MMD^2 (eq. 7) and the original estimator MMD^2 (eq. 2). Given the samples, the error deals with two types of randomness. The first arises due to the random features, $\hat{\boldsymbol{\phi}}$. The second arises due to the noise, $\mathbf{n}$, that we add to the mean-embedding of the data distribution for privacy. The following proposition formally states the bound to the error (See Supplementary Sec. B for proof).

Proposition 3.1. *Given samples $\mathbf{x} = \{x_i\}_{i=1}^m \sim P$ and $\tilde{\mathbf{x}} = \{\tilde{x}_j\}_{j=1}^n \sim Q$, the expected absolute error between the noisy random-feature MMD^2 given in eq. 7 and the MMD^2 given in eq. 2 is bounded by*

$$\mathbb{E}_{\mathbf{n}}\mathbb{E}_{\hat{\boldsymbol{\phi}}}\left[\left| \widetilde{\mathrm{MMD}}^2_{rf}(\mathbf{x}, \tilde{\mathbf{x}}) - \widehat{\mathrm{MMD}}^2(\mathbf{x}, \tilde{\mathbf{x}}) \right| \right] \tag{11}$$

$$\leq \left(\frac{4D\sigma^2}{m^2} + \frac{8\sqrt{2}\sigma}{m} \frac{\Gamma\big((D+1)/2\big)}{\Gamma\big(D/2\big)} \right) + 8\sqrt{\frac{2\pi}{D}}. \tag{12}$$

where Γ is the Gamma function, σ is the noise scale (inversely proportional to ϵ), m is the number of training datapoints, and D is the number of features.

Remark 1. *To prove Prop. 3.1, we split eq. 11 into two terms using the triangle inequality. The first term involves the expected absolute error between the **noisy** random feature MMD^2 (eq. 7) and random feature MMD^2 (eq. 3), which yields the first term (inside a big parenthesis) in eq. 12. The second term involves the expected absolute error between random feature MMD^2 (eq. 3) and the MMD^2 (eq. 2), which yields the second term in eq. 12. The upper bound is intuitive in that as the number of random features increases, the second term decreases because the random feature MMD is getting closer to MMD, while the first term increases because we add noise to a larger number of random features.*

Remark 2. *This bound provides a guideline on how to choose D given a desired privacy level ϵ and the dataset size m. First, given m, as long as we choose D such that $m > \sqrt{D}$, the error remains relatively small. However, small D can increase the error in the second term (arising from the MMD approximation using random features). Hence, there is a trade-off between these two terms. In our experiments, the datasets we consider have a relatively large m (see Table 2), and so choosing a large D ($D \approx 10,000$) incurred a relatively small error for a small value of ϵ.*

4 Extension of the vanilla DP-MERF

After introducing the core functionality of DP-MERF, we extend the vanilla method to cases for 1) labeled data, 2) class-imbalanced data, and 3) heterogeneous data.

4.1 DP-MERF for labeled data

We begin by extending our method to balanced labeled datasets with input features $\mathbf{x}$ and output labels $\mathbf{y}$. In this case, the generator is conditioned on the label: $G_\theta(\mathbf{z}, \mathbf{y}) \mapsto \tilde{\mathbf{x}}$, where $\mathbf{y}$ is drawn from the uniform distribution over classes.

We encode the class information in the MMD objective, by constructing a kernel from a product of two existing kernels, $k((\mathbf{x}, \mathbf{y}), (\mathbf{x}', \mathbf{y}')) = k_{\mathbf{x}}(\mathbf{x}, \mathbf{x}')k_{\mathbf{y}}(\mathbf{y}, \mathbf{y}')$, where $k_{\mathbf{x}}$ is a kernel for input features and $k_{\mathbf{y}}$ is a kernel for output labels. We choose the Gaussian kernel[3] for $k_{\mathbf{x}}$ and the polynomial kernel with order-1, $k_{\mathbf{y}}(\mathbf{y}, \mathbf{y}') = \mathbf{y}^\top \mathbf{y}' + c$ for one-hot-encoded labels $\mathbf{y}$ and set $c = 0$. In this case, the result-

[3]The optimal choice of kernel requires knowledge on the characteristics of the data (see guidelines in Ch. 4 in [33]). At small data sample sizes, a bad kernel choice will affect the efficiency of the algorithm and can underestimate MMD if the chosen kernel assigns small weights to the "correct" frequencies at which the distributions differ. However, with a large enough sample, any characteristic kernel is able to capture such differences.

ing kernel is also characteristic, forming the corresponding MMD as a metric, as explained in [28]. We represent the mean embeddings using random features by

$$\widehat{\boldsymbol{\mu}}_{P_{\mathbf{x},\mathbf{y}}} = \frac{1}{m}\sum_{i=1}^{m}\hat{\mathbf{f}}(\mathbf{x}_i,\mathbf{y}_i), \text{for true data} \tag{13}$$

$$\widehat{\boldsymbol{\mu}}_{Q_{\mathbf{x},\mathbf{y}}} = \frac{1}{n}\sum_{i=1}^{n}\hat{\mathbf{f}}(G_{\boldsymbol{\theta}}(\mathbf{z}_i,\mathbf{y}_i),\mathbf{y}_i), \text{ for synthetic data}$$

where we define $\hat{\mathbf{f}}(\mathbf{x}_i,\mathbf{y}_i) := \hat{\boldsymbol{\phi}}(\mathbf{x}_i)\mathbf{f}(\mathbf{y}_i)^\top$, where $\mathbf{f}(\mathbf{y}_i) = \mathbf{y}_i$ for the order-1 polynomial kernel and $\mathbf{y}_i$ is one-hot-encoded. See Supplementary Sec. C for derivation. With D random features and C classes, the random feature mean embedding in eq. 13 can also be written as $\widehat{\boldsymbol{\mu}}_{P_{\mathbf{x},\mathbf{y}}} = \begin{bmatrix}\mathbf{u}_1, & \cdots, & \mathbf{u}_C\end{bmatrix} \in \mathbb{R}^{D\times C}$ where c'th column is given by

$$\mathbf{u}_c = \frac{1}{m}\sum_{\mathbf{x}_i\in X_m^{(c)}}\hat{\boldsymbol{\phi}}(\mathbf{x}_i) \tag{14}$$

where $X_m^{(c)}$ is the set of the datapoints that belong to the class c. As in the unlabeled case, $\widehat{\boldsymbol{\mu}}_{P_{\mathbf{x},\mathbf{y}}}$ has sensitivity $\Delta_{\boldsymbol{\mu}_P} = \frac{2}{m}$ and is released with the Gaussian mechanism:

$$\widetilde{\boldsymbol{\mu}}_{P_{\mathbf{x},\mathbf{y}}} = \widehat{\boldsymbol{\mu}}_{P_{\mathbf{x},\mathbf{y}}} + \mathcal{N}(0,\Delta_{\boldsymbol{\mu}_P}^2\sigma^2\mathbf{I}_D) \tag{15}$$

With the released mean embedding $\widetilde{\boldsymbol{\mu}}_{P_{\mathbf{x},\mathbf{y}}}$, we construct the private joint maximum mean discrepancy objective:

$$\widetilde{\mathrm{MMD}}^2_{rf}(P_{\mathbf{x},\mathbf{y}},Q_{\tilde{\mathbf{x}}_{\boldsymbol{\theta}},\tilde{\mathbf{y}}_{\boldsymbol{\theta}}}) = \left\|\widetilde{\boldsymbol{\mu}}_{P_{\mathbf{x},\mathbf{y}}} - \widehat{\boldsymbol{\mu}}_{Q_{\mathbf{x},\mathbf{y}}}\right\|_F^2, \tag{16}$$

where F denotes the Frobenius norm. This kind of objective has been used in the non-private setting [39, 10].

4.2 DP-MERF for imbalanced data

Building on the previous section, notice that in eq. 14 the sum in each column is over m_c, the number of instances that belong to the particular class c, while the divisor is the number of samples in the entire dataset, m. This causes difficulties in learning when classes are highly imbalanced, as for rare classes m can be significantly larger than the sum of the corresponding column. In order to address this problem, we release the vector of class counts, $\mathbf{m} = [m_1,\cdots,m_C]$ using the Gaussian mechanism:

$$\widetilde{\mathbf{m}} = \mathbf{m} + \mathcal{N}(0,\Delta_{\mathbf{m}}^2\sigma^2\mathbf{I}_C) \tag{17}$$

As changing a datapoint affects at most two class counts, $\Delta_{\mathbf{m}} = \sqrt{2}$. We then modify the released mean embedding by appropriately weighting the embedding for each class:

$$\widetilde{\boldsymbol{\mu}}^*_{P_{\mathbf{x},\mathbf{y}}} = \begin{bmatrix}\frac{m}{\widetilde{m}_1}\widetilde{\mathbf{u}}_1, & \cdots, & \frac{m}{\widetilde{m}_C}\widetilde{\mathbf{u}}_C\end{bmatrix} \tag{18}$$

Note that we arrive at this expression of mean embedding if we change the kernel on the labels to a weighted one, i.e., $k_{\mathbf{y}}(\mathbf{y},\mathbf{y}') = \sum_{c=1}^{C}\frac{m}{\widetilde{m}_c}\mathbf{y}_c{}^\top\mathbf{y}'_c$. In the re-weighted mean embedding each class-wise embedding $\frac{m}{\widetilde{m}_c}\widetilde{\mathbf{u}}_c$ has a similar norm, and equally contributes to the objective loss. This ensures that infrequent classes are also modelled accurately.

Algorithm 1 DP-MERF for imbalanced data

Require: Dataset $\mathcal{D}$, and a privacy level (ϵ,δ)

Ensure: (ϵ,δ)-DP input output samples for all classes

Step 1. Given (ϵ,δ), compute the privacy parameter σ by the RDP composition in [32] for the two uses of the Gaussian mechanism in steps 2 and 3.

Step 2. Release the mean embedding $\widetilde{\boldsymbol{\mu}}_{P_{\mathbf{x},\mathbf{y}}}$ via eq. 15

Step 3. Release the class counts $\widetilde{\mathbf{m}}$ using eq. 17.

Step 4. Create the weighted mean embedding $\widetilde{\boldsymbol{\mu}}^*_{P_{\mathbf{x},\mathbf{y}}}$ using eq. 18

Step 5. Train the generator by minimizing $\widetilde{\mathrm{MMD}}^2_{rf}(P_{\mathbf{x},\mathbf{y}},Q_{\tilde{\mathbf{x}}_{\boldsymbol{\theta}},\tilde{\mathbf{y}}_{\boldsymbol{\theta}}}) = \left\|\widetilde{\boldsymbol{\mu}}^*_{P_{\mathbf{x},\mathbf{y}}} - \widehat{\boldsymbol{\mu}}_{Q_{\mathbf{x},\mathbf{y}}}\right\|_F^2$

The total privacy loss results from the composition of the two releases of first $\widetilde{\mathbf{m}}$ and then $\widetilde{\boldsymbol{\mu}}_{P_{\mathbf{x},vy}}$. During training, we sample the generated labels $\tilde{\mathbf{y}}$ proportional to the class sizes in $\widetilde{\mathbf{m}}$. The procedure is summarized in Algorithm 1.

4.3 DP-MERF for heterogeneous data

To handle heterogeneous data consisting of numerical variables denoted by $\mathbf{x}_{num}$ and categorical variables denoted by $\mathbf{x}_{cat}$, we consider the sum of two existing kernels, $k((\mathbf{x}_{num},\mathbf{x}_{cat}),(\mathbf{x}'_{num},\mathbf{x}'_{cat})) = k_{num}(\mathbf{x}_{num},\mathbf{x}'_{num}) + k_{cat}(\mathbf{x}_{cat},\mathbf{x}'_{cat})$, where k_{num} is a kernel for numerical variables and k_{cat} is a kernel for categorical variables. Note that this construction of sum of two kernels does not mean that we implicitly assume independence of the two types of variables, for details see Supplementary Sec. I.

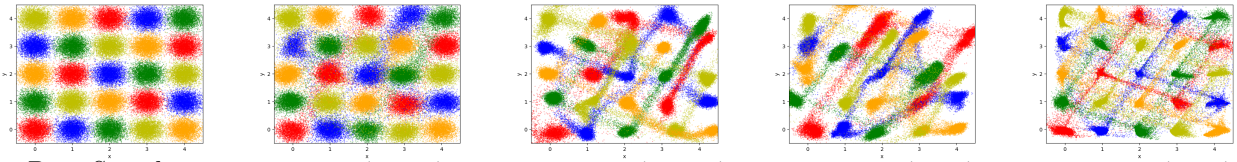
As before, we could use the Gaussian kernel for $k_{num}(\mathbf{x}_{num},\mathbf{x}'_{num}) = \hat{\boldsymbol{\phi}}(\mathbf{x}_{num})^\top\hat{\boldsymbol{\phi}}(\mathbf{x}'_{num})$ and a normalized polynomial kernel with order-1, $k_{cat}(\mathbf{x}_{cat},\mathbf{x}'_{cat}) = \frac{1}{d_{cat}}\mathbf{x}_{cat}{}^\top\mathbf{x}'_{cat}$ for one-hot-encoded values $\mathbf{x}_{cat}$ and the length of $\mathbf{x}_{cat}$ being d_{cat}. This normalization is to match the importance of the two kernels in the resulting mean embeddings. Under these kernels, we define

$$\widehat{\boldsymbol{\mu}}_{P_{\mathbf{x}}} = \frac{1}{m}\sum_{i=1}^{m}\hat{\mathbf{h}}(\mathbf{x}^{(i)}_{num},\mathbf{x}^{(i)}_{cat}), \tag{19}$$

where we define $\hat{\mathbf{h}}(\mathbf{x}^{(i)}_{num},\mathbf{x}^{(i)}_{cat}) := \begin{bmatrix}\hat{\boldsymbol{\phi}}(\mathbf{x}^{(i)}_{num}) \\ \frac{1}{\sqrt{d_{cat}}}\mathbf{x}^{(i)}_{cat}\end{bmatrix}$ based on the definition of kernel k (See Supplementary Sec. D for derivation).

In summary, for generating heterogeneous data, we run Algorithm 1 with three changes:

1. Redefine $\hat{\mathbf{f}}(\mathbf{x},\mathbf{y})$ in eq. 13 as $\hat{\mathbf{h}}(\mathbf{x}_{num},\mathbf{x}_{cat})\mathbf{f}(\mathbf{y})^\top$.



Data Samples NLL $\approx 3.1 \times 10^5$ | CGAN ($\epsilon = \infty$) NLL $\approx 3.8 \times 10^5$ | DP-CGAN ($\epsilon = 9.6$) NLL $\approx 4.2 \times 10^5$ | DP-CGAN ($\epsilon = 1$) NLL $\approx 4.7 \times 10^5$ | DP-MERF ($\epsilon = 1$) NLL $\approx 3.8 \times 10^5$

Figure 1: Simulated example from a Gaussian mixture. **Left**: Data samples drawn from a Gaussian Mixture distribution with 5 classes (each color represents a class). NLL denotes the negative log likelihood of the samples given the true data distribution. **Middle three**: Synthetic data generated by DP-CGANs at different privacy levels. CGAN ($\epsilon = \infty$) performs nearly perfectly. However, at $\epsilon = 1$, some modes are dropped, which is reflected in NLL. **Right**: Synthetic data samples generated by DP-MERF at $\epsilon = 1$. Our method captures all modes accurately at $\epsilon = 1$, which is also reflected in NLL.

2. Redefine $\mathbf{u}_c$ in eq. 14 as $\frac{1}{m}\sum_{i \in X_m^{(c)}} \hat{\mathbf{h}}(\mathbf{x}_i)$.
3. Change the sensitivity of $\mathbf{u}_c$ to $\Delta_{\mathbf{u}_c} = \frac{2\sqrt{2}}{m}$ (see Supplementary Sec. G for proof).

5 Related work

Differentially private data release. The field of DP data release contains several distinct lines of research. As mentioned previously, approaches from a learning theory perspective [17, 34, 13, 40] provide bounds on the utility of the data, but contain either strong assumptions about the types of executed queries or intractable computation, which makes this line of research less relevant to our approach.

Among the query-independent methods, a large body of work on DP data release focuses on discrete or possible to discretize data. This is a relevant sub-problem in which good results can be achieved by releasing carefully selected marginals of feature subsets, as each feature only takes on a finite set of values. Such approaches [38, 21, 6] have been, for instance, been dominant among the winning entries of the NIST 2018 Differential Privacy Synthetic Data Challenge [1], which focused on the task of releasing discrete datasets, utilizing related publicly available data. Although we do not compare to this line of work in the main text, as our method deals with the general setting of DP data release, including continuous data, we show the comparison to [38] in the Supplementary Sec. M.

The recent line of research into GAN-based private data release [35, 30, 9, 36, 5] addresses the same general setting and so we select these models for comparison. GANs are regarded as a promising model for this task because of their great success in non-private generative modelling and thanks to the fact the generator network of a GAN can be trained without direct access to the data. The GAN discriminator must still be trained with privacy constraints. In most cases, this is achieved through gradient perturbation using DP-SGD, with the exception of PATE-GAN [36], which is based on the Private Aggregation of Teacher Ensembles (PATE) [19]. DP-GAN [35] and PATE-GAN [36] generate unlabeled data and thus must train one model per class to obtain a labeled dataset. DP-CGAN [30] and GS-WGAN [5] generate the input features conditioning on the labels, while they do not learn the distribution over the labels. GS-WGAN improves on the basic DP-SGD by alleviating the need for gradient clipping by adapting the loss function and, like PATE-GAN, employs multiple discriminator networks trained on distinct parts of the dataset to amplify privacy by subsampling. We compare these methods with our approach in Sec. 6.

Random feature kernel methods with differential privacy. Some prior work has employed random feature mean embeddings in the context of differential privacy, but not for the purpose of generative modeling. [4] proposed to use the reduced set method in conjunction with random features for sharing DP mean embeddings. This method performs poorly as the dimension of data grows, which is also noted by the authors (see Supplementary Sec. M for comparison to our method). [24] also used the random feature representations of mean embeddings for the DP distributed data summarization to take into account covariate shifts.

6 Experiments

In this section, we show the robustness of our method on a diverse range of data under strong privacy constraints. On each dataset, we train DP-MERF and comparison methods to obtain a set of private *synthetic* data samples and compare, how well these emulate the original dataset. Due to the space limit, we describe all our experimental details (e.g., architecture choices for generators, chosen number of random features, etc.) in the supplementary material. Our code is available at `https://github.com/ParkLabML/DP-MERF`.

Table 1: Performance comparison on tabular datasets, averaged over five runs. DP-MERF achieves the best scores among private models (bold) on the majority of datasets.

	Real		DP-CGAN $(1, 10^{-5})$-DP		DP-GAN $(1, 10^{-5})$-DP		**DP-MERF** $(1, 10^{-5})$-DP		DP-MERF non-DP	
	ROC	PRC	ROC	PRC	ROC	PRC	ROC	PRC	ROC	PRC
adult	0.730	0.639	0.509	0.444	0.511	0.445	**0.650**	**0.564**	0.653	0.570
census	0.747	0.415	0.655	0.216	0.529	0.166	**0.686**	**0.358**	0.692	0.369
cervical	0.786	0.493	0.519	**0.200**	0.485	0.183	**0.545**	0.184	0.896	0.737
credit	0.923	0.874	0.664	0.356	0.435	0.150	**0.772**	**0.637**	0.898	0.774
epileptic	0.797	0.617	0.578	0.241	0.505	0.196	**0.611**	**0.340**	0.616	0.335
isolet	0.893	0.728	0.511	0.198	0.540	0.205	**0.547**	**0.404**	0.733	0.424
	F1		F1		F1		F1		F1	
covtype	0.643		0.285		**0.492**		0.467		0.513	
intrusion	0.959		0.302		0.251		**0.850**		0.856	

2D Gaussian mixtures. We begin our experiments on a simple synthetic distribution of Gaussian mixtures which is aligned on a 5 by 5 grid and assigned to 5 classes as shown in Fig. 1 (left). The dataset is generated by taking 4000 samples from each Gaussian, reserving 10% for the test set, which yields 90000 training samples from the following distribution:

$$p(\mathbf{x}, \mathbf{y}) = \prod_{i}^{N} \sum_{j \in C_{\mathbf{y}_i}} \frac{1}{C} \mathcal{N}(\mathbf{x}_i | \boldsymbol{\mu}_j, \sigma \mathbf{I}_2) \qquad (20)$$

where $N = 90000$, and $\sigma = 0.2$. $C = 25$ is the number of clusters and C_y denotes the set of indices for means $\boldsymbol{\mu}$ assigned to class y. Five Gaussians are assigned to each class, which leads to a uniform distribution over $\mathbf{y}$ and 18000 samples per class.

We choose this dataset because knowing the true data distribution allows us to compute the negative log likelihood (NLL) of the samples under the true distribution as a measure of the generated samples' quality: $\text{NLL}(\mathbf{x}, \mathbf{y}) = -\log p(\mathbf{x}, \mathbf{y})$. Note that this is different from the other common measure of computing the negative log-likelihood of the true data given the learned model parameters.

A high NLL score indicates that many samples lie in low density regions of the data distribution. In cases where models tend to under-fit the data, a lower NLL score can thus be regarded as better. However, a low score does not imply that all modes are covered and may also be the result of low sample variance, although the out-of-distribution samples dominate the score, due to the non-linearity of the $\log$ function.

At different levels of privacy, we train DP-CGAN on this dataset and select the models with the fewest dropped modes and secondarily the lowest NLL. We compare this to a DP-MERF model for balanced datasets in Fig. 1. While DP-CGAN in the non-private setting ($\epsilon = \infty$) fits the data well, more samples fall out of the distribution as privacy is increased and some modes (like the green one in the top right corner) are dropped. DP-MERF on the other hand preserves all modes and places few samples in low density regions as indicated by the low NLL score. This NLL score is particularly low and on par with the non-private DP-CGAN model, despite a slightly worse fit, because DP-MERF seems to underestimate variance.

Table 2: Tabular datasets. num refers to numerical, cat refers to categorical, and ord refers to ordinal variables

dataset	# samps	# classes	# features
isolet	4366	2	617 num
covtype	406698	7	10 num, 44 cat
epileptic	11500	2	178 num
credit	284807	2	29 num
cervical	753	2	11 num, 24 cat
census	199523	2	7 num, 33 cat
adult	22561	2	6 num, 8 cat
intrusion	394021	5	8 cat, 6 ord, 26 num

Real world data evaluation. In the following experiments we do not know the true data distribution and thus require a different method to evaluate the quality of privately generated datasets. Following the common approach used in [36, 30, 5], we use the private datasets to train a selection of 12 *predictive models* (see Table 5 in the Supplementary for the models). We then evaluate these trained models on a test set of *real* data, which indicates how well the models generalize from the synthetic to the real data distribution and thus how useful the private data would be if used in place of the real data. Note that hyper-parameters of the 12 models differ because the exact settings used in [36] were not available to us, which means that their scores are not directly comparable to ours. As comparison models, we test DP-CGAN [30], as well as our own implementation of an ensemble of 10 DP-GANs, where each model generates data for each class. Our version of DP-GAN differs from [35] in that it uses standard DP-SGD [2] with gradient clipping rather than weight clipping. We further

include GS-WGAN [5] on image datasets following their original setup. Note that our DP-GAN implementation and GS-WGAN use the analytical moments accountant [31] via the autodp package. DP-CGAN uses the RDP accountant [16] from the tensorflow-privacy package, which is slightly older but still comparable. The results in [36, 35] could not be reproduced as the released code was incomplete.

As comparison metrics, we use ROC (area under the receiver operating characteristics curve) and PRC (area under the precision recall curve) for binary-labeled data. For multiclass-labeled data we report accuracy for balanced and F1 score for imbalanced data. As a baseline, we also show the performance of the models trained with the real training data. All the numbers shown in the tables are averages over 5 independent runs.

Table 3: Test accuracy on image data experiments. DP-MERF at $\epsilon = 0.2$ outperforms other methods by a significant margin. $\delta = 10^{-5}$ in all private settings.

	MNIST	**FashionMNIST**
Real data	0.87	0.78
DP-CGAN $\epsilon = 9.6$	0.50	0.39
DP-GAN $\epsilon = 9.6$	0.48	0.46
GS-WGAN $\epsilon = 10$	0.53	0.50
DP-MERF $\epsilon = 1$	0.65	0.61
DP-MERF $\epsilon = 0.2$	0.61	0.53

Tabular data. We explore the extensions of DP-MERF for imbalanced and heterogeneous data on a number of real-world tabular datasets. These datasets contain numerical features with both discrete and continuous values as well as categorical features with either two classes (e.g. whether a person smokes or not) or several classes (e.g. country of origin). The output labels are also categorical and we include datasets with both binary and multi-class labels. Table 2 summarizes the datasets. Table 1 shows the average across the 12 predictive models trained by the generated samples from DP-CGAN, DP-GAN and DP-MERF. Results for the individual models can be found in Supplementary Sec. K. Overall, our method achieved higher values on the evaluation metrics compared to other methods at the same privacy level.

As a side note, the reason the non-private MERF on Cervical data outperforms the real data is due to the small size of the dataset, which is prone to overfitting. Hence, the added sample variance in the generated data has a regularizing effect and improves the performance.

Image data Finally, we evaluate our method on the image datasets, MNIST and FashionMNIST, which are common benchmarks used in [30, 35, 5]. We apply DP-MERF for balanced data and include convolutional layers, alternating with bi-linear up-sampling, in the generator network to take advantage of the inherent structure of image data.

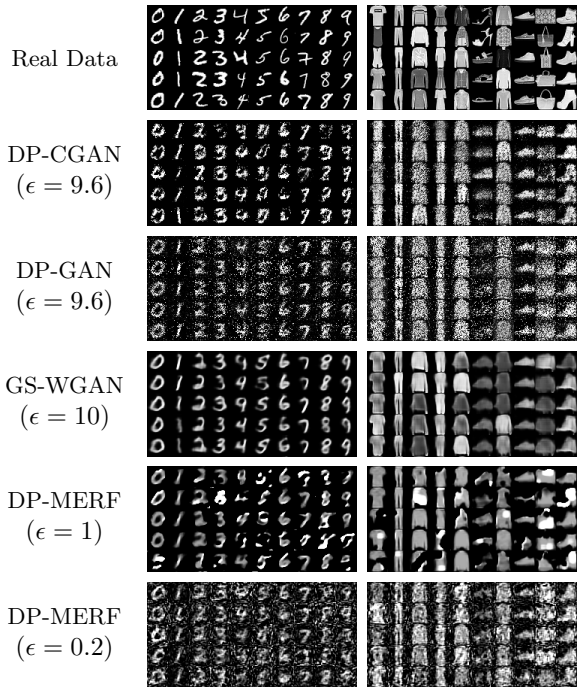


Figure 2: Generated MNIST and FashionMNIST samples from DP-MERF and comparison models with different levels of privacy.

Table 3 compares the test accuracy on real data based on generated samples from DP-CGAN, DP-GAN, GS-WGAN and DP-MERF. Results are averaged over 12 classifiers. For the comparison methods, we use the privacy levels reported in the respective papers, as they do not produce usable samples in the high privacy setting at $\epsilon \leq 1$. It shows that DP-MERF outperforms the GAN based methods by a wide margin and maintains good performance under more meaningful privacy constraints of $(1, 10^{-5})$-DP and $(0.2, 10^{-5})$-DP. Low overall scores are largely due to the Adaboost and decision tree models which over-fit to the generated data while other models like logistic regression and multi-layer-perceptrions generalize much better. Detailed results are shown in Supplementary Sec. L.

In the generated samples of the four tested methods in Fig. 2, we see that the samples from DP-MERF at $\epsilon = 0.2$ are noisier than those of GS-WGAN and DP-CGAN, while still achieving higher downstream accuracy.[4] This indicates that the distinctive features of the data are preserved despite the noisy appearance of the DP-MERF samples. In addition, a loss of sample diversity may explain the worse performance of GS-WGAN and DP-CGAN despite higher perceived sample quality, as we already have observed DP-CGAN dropping modes in the Gaussian data experiment.

[4] As opposed to the version used in [5], the DP-MERF presented here uses an improved generator architecture and privacy analysis, and outperforms GS-WGAN in the classification tasks.

7 Summary and Discussion

We propose a simple and practical algorithm using the random feature representation of kernel mean embeddings for DP data generation. Our method requires a significantly lower privacy budget to produce quality data samples compared to GAN-based approaches, tested on a synthetic dataset, 8 tabular datasets and 2 image datasets. The metrics we use are aimed at supervised learning tasks, but the method is not limited to this application. In the future work, we plan to evaluate our method on a more diverse set of tasks and expand it, to scale to more complex data.

Acknowledgments

We thank Wittawat Jitkrittum, Jia-Jie Zhu, Amin Charusaie and the anonymous reviewers for their valuable time helping us improve our manuscript. All three authors are supported by the Max Planck Society. M. Park and F. Harder are also supported by the Gibs Schüle Foundation and the Institutional Strategy of the University of Tübingen (ZUK63) and the German Federal Ministry of Education and Research (BMBF): Tübingen AI Center, FKZ: 01IS18039B. F. Harder is grateful for the support of the International Max Planck Research School for Intelligent Systems (IMPRS-IS). K. Adamczewski is grateful for the support of the Max Planck ETH Center for Learning Systems.

References

[1] Nist 2018 differential privacy synthetic data challenge. https://www.nist.gov/ctl/pscr/open-innovation-prize-challenges/past-prize-challenges/2018-differential-privacy-synthetic.

[2] Martin Abadi, Andy Chu, Ian Goodfellow, H. Brendan McMahan, Ilya Mironov, Kunal Talwar, and Li Zhang. Deep learning with differential privacy. In *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security*, CCS '16, page 308–318, New York, NY, USA, 2016. Association for Computing Machinery.

[3] Martín Arjovsky, Soumith Chintala, and Léon Bottou. Wasserstein gan. *ArXiv*, abs/1701.07875, 2017.

[4] Matej Balog, Ilya Tolstikhin, and Bernhard Schölkopf. Differentially private database release via kernel mean embeddings. In *Proceedings of the 35th International Conference on Machine Learning (ICML)*, volume 80 of *Proceedings of Machine Learning Research*, pages 423–431. PMLR, July 2018.

[5] 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 33*, 2020.

[6] Rui Chen, Qian Xiao, Yu Zhang, and Jianliang Xu. Differentially private high-dimensional data publication via sampling-based inference. In *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 129–138, 2015.

[7] I. Csiszár and P.C. Shields. Information theory and statistics: A tutorial. *Foundations and Trends® in Communications and Information Theory*, 1(4):417–528, 2004.

[8] Cynthia Dwork, Krishnaram Kenthapadi, Frank McSherry, Ilya Mironov, and Moni Naor. Our data, ourselves: Privacy via distributed noise generation. In *Eurocrypt*, volume 4004, pages 486–503. Springer, 2006.

[9] 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 *ICT Systems Security and Privacy Protection - 34th IFIP TC 11 International Conference, SEC 2019, Lisbon, Portugal, June 25-27, 2019, Proceedings*, pages 151–164, 2019.

[10] Hongchang Gao and Heng Huang. Joint generative moment-matching network for learning structural latent code. In *Proceedings of the Twenty-Seventh International Joint Conference on Artificial Intelligence, IJCAI-18*, pages 2121–2127. International Joint Conferences on Artificial Intelligence Organization, 7 2018.

[11] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. In Z. Ghahramani, M. Welling, C. Cortes, N. D. Lawrence, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems 27*, pages 2672–2680. Curran Associates, Inc., 2014.

[12] Arthur Gretton, Karsten M Borgwardt, Malte J Rasch, Bernhard Schölkopf, and Alexander Smola. A kernel two-sample test. *Journal of Machine Learning Research*, 13(Mar):723–773, 2012.

[13] Moritz Hardt, Katrina Ligett, and Frank Mcsherry. A simple and practical algorithm for differentially private data release. In F. Pereira, C. J. C. Burges, L. Bottou, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems 25*, pages 2339–2347. Curran Associates, Inc., 2012.

[14] Chun-Liang Li, Wei-Cheng Chang, Yu Cheng, Yiming Yang, and Barnabas Poczos. Mmd gan: Towards deeper understanding of moment matching network. In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems 30*, pages 2203–2213. Curran Associates, Inc., 2017.

[15] Ryan McKenna, Daniel Sheldon, and Gerome Miklau. Graphical-model based estimation and inference for differential privacy. *arXiv preprint arXiv:1901.09136*, 2019.

[16] Ilya Mironov. Rényi differential privacy. In *2017 IEEE 30th Computer Security Foundations Symposium (CSF)*, pages 263–275. IEEE, 2017.

[17] Noman Mohammed, Rui Chen, Benjamin C.M. Fung, and Philip S. Yu. Differentially private data release for data mining. In *Proceedings of the 17th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '11, pages 493–501, New York, NY, USA, 2011. ACM.

[18] Sebastian Nowozin, Botond Cseke, and Ryota Tomioka. f-gan: Training generative neural samplers using variational divergence minimization. In *Proceedings of the 30th International Conference on Neural Information Processing Systems*, NIPS'16, pages 271–279, USA, 2016. Curran Associates Inc.

[19] Nicolas Papernot, Martín Abadi, Úlfar Erlingsson, Ian Goodfellow, and Kunal Talwar. Semi-supervised Knowledge Transfer for Deep Learning from Private Training Data. In *Proceedings of the International Conference on Learning Representations (ICLR)*, April 2017.

[20] Noseong Park, Mahmoud Mohammadi, Kshitij Gorde, Sushil Jajodia, Hongkyu Park, and Youngmin Kim. Data synthesis based on generative adversarial networks. *Proc. VLDB Endow.*, 11(10):1071–1083, June 2018.

[21] Wahbeh Qardaji, Weining Yang, and Ninghui Li. Priview: practical differentially private release of marginal contingency tables. In *Proceedings of the 2014 ACM SIGMOD international conference on Management of data*, pages 1435–1446, 2014.

[22] Ali Rahimi and Benjamin Recht. Random features for large-scale kernel machines. In *Advances in neural information processing systems*, pages 1177–1184, 2008.

[23] Walter Rudin. *Fourier Analysis on Groups: Interscience Tracts in Pure and Applied Mathematics, No. 12*. Literary Licensing, LLC, 2013.

[24] Kanthi Sarpatwar, Karthikeyan Shanmugam, Venkata Sitaramagiridharganesh Ganapavarapu, Ashish Jagmohan, and Roman Vaculin. Differentially private distributed data summarization under covariate shift. In *Advances in Neural Information Processing Systems*, pages 14432–14442, 2019.

[25] A. Smola, A. Gretton, L. Song, and B. Schölkopf. A Hilbert space embedding for distributions. In *ALT*, pages 13–31, 2007.

[26] Bharath K Sriperumbudur, Kenji Fukumizu, and Gert RG Lanckriet. Universality, characteristic kernels and rkhs embedding of measures. *Journal of Machine Learning Research*, 12(7), 2011.

[27] Dougal J. Sutherland and Jeff Schneider. On the error of random fourier features. In *Proceedings of the Thirty-First Conference on Uncertainty in Artificial Intelligence*, UAI'15, page 862–871, Arlington, Virginia, USA, 2015. AUAI Press.

[28] Zoltán Szabó and Bharath K. Sriperumbudur. Characteristic and universal tensor product kernels. *Journal of Machine Learning Research*, 18(233):1–29, 2018.

[29] Ilya Tolstikhin, Olivier Bousquet, Sylvain Gelly, and Bernhard Schoelkopf. Wasserstein auto-encoders. In *International Conference on Learning Representations*, 2018.

[30] Reihaneh Torkzadehmahani, Peter Kairouz, and Benedict Paten. Dp-cgan: Differentially private synthetic data and label generation. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR) Workshops*, June 2019.

[31] Yu-Xiang Wang, Borja Balle, and Shiva Prasad Kasiviswanathan. Subsampled rényi differential privacy and analytical moments accountant. PMLR, 2019.

[32] Yu-Xiang Wang, Borja Balle, and Shiva Prasad Kasiviswanathan. Subsampled renyi differential privacy and analytical moments accountant. In Kamalika Chaudhuri and Masashi Sugiyama, editors, *Proceedings of Machine Learning Research*, volume 89 of *Proceedings of Machine Learning Research*, pages 1226–1235. PMLR, April 2019.

[33] Christopher KI Williams and Carl Edward Rasmussen. *Gaussian processes for machine learning*, volume 2. MIT press Cambridge, MA, 2006.

[34] Yonghui Xiao, Li Xiong, and Chun Yuan. Differentially private data release through multidimensional partitioning. In Willem Jonker and Milan Petković, editors, *Secure Data Management*, pages 150–168, Berlin, Heidelberg, 2010. Springer Berlin Heidelberg.

[35] Liyang Xie, Kaixiang Lin, Shu Wang, Fei Wang, and Jiayu Zhou. Differentially private generative adversarial network. *CoRR*, abs/1802.06739, 2018.

[36] Jinsung Yoon, James Jordon, and Mihaela van der Schaar. PATE-GAN: Generating synthetic data with differential privacy guarantees. In *International Conference on Learning Representations*, 2019.

[37] Dan Zhang, Ryan McKenna, Ios Kotsogiannis, Michael Hay, Ashwin Machanavajjhala, and Gerome Miklau. Ektelo: A framework for defining differentially-private computations. SIGMOD, 2018.

[38] 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):1–41, 2017.

[39] Yi-Ying Zhang, Chao-Min Shen, Hao Feng, Preston Thomas Fletcher, and Gui-Xu Zhang. Generative adversarial networks with joint distribution moment matching. *Journal of the Operations Research Society of China*, 7(4):579–597, December 2019.

[40] T. Zhu, G. Li, W. Zhou, and P. S. Yu. Differentially private data publishing and analysis: A survey. *IEEE Transactions on Knowledge and Data Engineering*, 29(8):1619–1638, August 2017.

Supplementary Material: Differentially Private Random Feature Mean Embeddings for Synthetic Data Generation

A Background on distance measures for DP data generation

Many recent papers on DP data generation have utilized the generative adversarial networks (GAN) [11] framework, where a discriminator and a generator play a min-max form of game to optimize for the *Jensen-Shannon divergence* between the true and synthetic data distributions [20, 30, 36]. The Jensen-Shannon divergence belongs to the family of divergences, known as *Ali-Silvey distance*, *Csiszár's ϕ-divergence* [7], defined as $D_\phi(P,Q) = \int_M \phi\left(\frac{P}{Q}\right) dQ$ where M is a measurable space and P, Q are probability distributions. Depending on the form of ϕ, $D_\phi(P,Q)$ recovers popular divergences[5] such as the Kullback-Liebler (KL) divergence ($\phi(t) = t\log t$).

Another popular family of distance measure is *integral probability metrics (IPMs)*, which is defined by $D(P,Q) = \sup_{f\in\mathcal{F}} \left|\int_M f dP - \int_M f dQ\right|$ where $\mathcal{F}$ is a class of real-valued bounded measurable functions on M. Depending on the class of functions, there are several popular choices of IPMs. For instance, when $\mathcal{F} = \{f : \|f\|_L \leq 1\}$, where $\|f\|_L := \sup\{|f(x) - f(y)|/\rho(x,y) : x \neq y \in M\}$ for a metric space (M,ρ), $D(P,Q)$ yields the *Kantorovich* metric, and when M is separable, the Kantorovich metric recovers the *Wasserstein* distance, a popular choice for generative modelling such as Wasserstein-GAN and Wasserstein-VAE [3, 29]. The GAN framework with the Wasserstein distance was also used for DP data generation [35, 9].

As another example of IPMs, when $\mathcal{F} = \{f : \|f\|_{\mathcal{H}} \leq 1\}$, i.e., the function class is a unit ball in reproducing kernel Hilbert space (RKHS) $\mathcal{H}$ associated with a positive-definite kernel k, $D(P,Q)$ yields the *maximum mean discrepancy* (MMD), $MMD(P,Q) = \sup_{f\in\mathcal{F}} \left|\int_M f dP - \int_M f dQ\right|$. In this case finding a supremum is analytically tractable and the solution is represented by the difference in the mean embeddings of each probability measure: $MMD(P,Q) = \|\mu_P - \mu_Q\|_H$, where $\mu_P = \mathbb{E}_{\mathbf{x}\sim\mathbb{P}}[k(\mathbf{x},\cdot)]$ and $\mu_{\mathbb{Q}} = \mathbb{E}_{\mathbf{y}\sim\mathbb{Q}}[k(\mathbf{y},\cdot)]$. For a characteristic kernel k, the squared MMD forms a metric, i.e., $MMD^2 = 0$, if and only if $P = Q$. MMD is also a popular choice for generative modelling in the GAN frameworks [14], as MMD compares two probability measures in terms of all possible moments (no information loss due to a selection of a certain set of moments); and the MMD estimator is in closed form (eq. 2) and easy to compute by the pair-wise evaluations of a kernel function using the points drawn from P and Q.

In this work, we propose to use a particular form of MMD via *random Fourier feature* representations [22] of kernel mean embeddings for DP data generation.

B Derivation of the bound on the expected absolute error

Given the samples drawn from two probability distributions: $X_m = \{x_i\}_{i=1}^m \sim P$ and $X'_n = \{x'_i\}_{i=1}^n \sim Q$, the biased MMD estimator is given by [12]:

$$\widehat{\mathrm{MMD}}^2(X_m, X'_n) = \frac{1}{m^2}\sum_{i,j=1}^m k(x_i,x_j) + \frac{1}{n^2}\sum_{i,j=1}^n k(x'_i,x'_j) - \frac{2}{mn}\sum_{i=1}^m\sum_{j=1}^n k(x_i,x'_j). \tag{21}$$

The MMD estimator using the D-dimensional random Fourier features $\hat{\boldsymbol{\phi}}$ for the mean embeddings $\widehat{\boldsymbol{\mu}}_P = \frac{1}{m}\sum_{i=1}^m \hat{\boldsymbol{\phi}}(\mathbf{x}_i)$ and $\widehat{\boldsymbol{\mu}}_Q = \frac{1}{n}\sum_{i=1}^n \hat{\boldsymbol{\phi}}(G_{\boldsymbol{\theta}}(\mathbf{z}_i))$ is defined as

$$\widehat{\mathrm{MMD}}^2_{rf}(P,Q) = \left\|\widehat{\boldsymbol{\mu}}_P - \widehat{\boldsymbol{\mu}}_Q\right\|_2^2. \tag{22}$$

The noisy MMD is given by

$$\widetilde{\mathrm{MMD}}^2_{rf}(P_{\mathbf{x}}, Q_{\tilde{\mathbf{x}}_{\boldsymbol{\theta}}}) = \left\|\widetilde{\boldsymbol{\mu}}_P - \widehat{\boldsymbol{\mu}}_Q\right\|_2^2, \tag{23}$$

[5] See Table 1 in [18] for various ϕ divergences in the context of GANs.

where $\widetilde{\boldsymbol{\mu}}_P$ is given by

$$\widetilde{\boldsymbol{\mu}}_P = \widehat{\boldsymbol{\mu}}_P + \mathbf{n} \tag{24}$$

where $\mathbf{n}$ is a draw from a Gaussian distribution $\mathbf{n} \sim \mathcal{N}(0, \Delta^2_{\widehat{\boldsymbol{\mu}}_P}\sigma^2 I)$. Note that for the bounded kernels with bound 1, $\Delta_{\widehat{\boldsymbol{\mu}}_P} = \frac{2}{m}$.

Now the proposition is given as follows.

Proposition B.1. *Given samples $\mathbf{x} = \{x_i\}_{i=1}^m \sim P$ and $\tilde{\mathbf{x}} = \{\tilde{x}_j\}_{j=1}^n \sim Q$, the expected absolute error between the noisy random-feature (squared) MMD defined in eq. 7 and the squared MMD eq. 2 is bounded by*

$$\mathbb{E}_{\mathbf{n}}\mathbb{E}_{\hat{\boldsymbol{\phi}}}\left[\left|\widetilde{\mathrm{MMD}}^2_{rf}(\mathbf{x},\tilde{\mathbf{x}}) - \widehat{\mathrm{MMD}}^2(\mathbf{x},\tilde{\mathbf{x}})\right|\right], \tag{25}$$

$$\leq \left(\frac{4D\sigma^2}{m^2} + \frac{8\sqrt{2}\sigma}{m}\frac{\Gamma\big((D+1)/2\big)}{\Gamma\big(D/2\big)}\right) + 8\sqrt{\frac{2\pi}{D}} \tag{26}$$

where Γ is the Gamma function.

To prove this proposition, we first rewrite the absolute error in terms of two terms due to the triangle inequality:

$$\begin{aligned}&\mathbb{E}_{\mathbf{n}}\mathbb{E}_{\hat{\boldsymbol{\phi}}}\left[\left|\widetilde{\mathrm{MMD}}^2_{rf}(\mathbf{x},\tilde{\mathbf{x}}) - \widehat{\mathrm{MMD}}^2(\mathbf{x},\tilde{\mathbf{x}})\right|\right]\\ &\leq \mathbb{E}_{\mathbf{n}}\mathbb{E}_{\hat{\boldsymbol{\phi}}}\left[\left|\widetilde{\mathrm{MMD}}^2_{rf}(\mathbf{x},\tilde{\mathbf{x}}) - \widehat{\mathrm{MMD}}^2_{rf}(\mathbf{x},\tilde{\mathbf{x}})\right|\right] + \mathbb{E}_{\hat{\boldsymbol{\phi}}}\left[\left|\widehat{\mathrm{MMD}}^2_{rf}(\mathbf{x},\tilde{\mathbf{x}}) - \widehat{\mathrm{MMD}}^2(\mathbf{x},\tilde{\mathbf{x}})\right|\right].\end{aligned} \tag{27}$$

What follows next proves each of these terms.

B.1 Randomness due to random features

We restate the result of [Sec. 3.3 of Sutherland and Schneider 2016].

Lemma B.1 (Sec. 3.3 of Sutherland and Schneider 2016)**.** *Given samples $\mathbf{x} = \{x_i\}_{i=1}^n \sim P$ and $\tilde{\mathbf{x}} = \{\tilde{x}_j\}_{j=1}^m \sim Q$, the probabilistic bound between the approximate MMD with random features, denoted by $\widehat{\mathrm{MMD}}_{rf}(\mathbf{x},\tilde{\mathbf{x}})$ and the original MMD, denoted by $\widehat{\mathrm{MMD}}(\mathbf{x},\tilde{\mathbf{x}})$, holds*

$$\mathbb{P}\left[\left|\widehat{\mathrm{MMD}}^2_{rf}(\mathbf{x},\tilde{\mathbf{x}}) - \widehat{\mathrm{MMD}}^2(\mathbf{x},\tilde{\mathbf{x}})\right| \geq t_1\right] \leq 2\exp\left(-\frac{1}{128}Dt_1^2\right) := U_1, \tag{28}$$

where the randomness comes from the random features, and $\mathbb{E}_{\hat{\boldsymbol{\phi}}}[\widehat{\mathrm{MMD}}_{rf}(\mathbf{x},\tilde{\mathbf{x}})] = \mathrm{MMD}(\mathbf{x},\tilde{\mathbf{x}})$.

Proof. To prove the proposition, we first consider the mean map kernel (MMK) defined by

$$\mathrm{MMK}(\mathbf{x},\tilde{\mathbf{x}}) = \frac{1}{nm}\sum_{i=1}^n\sum_{j=1}^m k(x_i,\tilde{x}_j) \approx \mathrm{MMK}_{\hat{\boldsymbol{\phi}}}(\mathbf{x},\tilde{\mathbf{x}}) := \hat{\boldsymbol{\phi}}(\mathbf{x})^\top\hat{\boldsymbol{\phi}}(\tilde{\mathbf{x}}), \tag{29}$$

which can be approximated by the random feature representations, denoted by $\mathrm{MMK}_{\hat{\boldsymbol{\phi}}}(\mathbf{x},\tilde{\mathbf{x}})$. The random feature mean-embedding of P is denoted by $\hat{\boldsymbol{\phi}}(\mathbf{x})$. Similarly, we can define $\mathrm{MMK}(\mathbf{x},\mathbf{x})$ and $\mathrm{MMK}(\tilde{\mathbf{x}},\tilde{\mathbf{x}})$, and define MMD in terms of MMKs

$$\widehat{\mathrm{MMD}}^2(\mathbf{x},\tilde{\mathbf{x}}) = \mathrm{MMK}(\mathbf{x},\mathbf{x}) + \mathrm{MMK}(\tilde{\mathbf{x}},\tilde{\mathbf{x}}) - 2\mathrm{MMK}(\mathbf{x},\tilde{\mathbf{x}}). \tag{30}$$

Notice that when we use the cosine/sine representation of random features, changing the frequency ω_k to $\hat{\omega}_k$ causes a bounded difference in the kth coordinate of the MMK estimate, $\mathrm{MMK}_{\boldsymbol{\phi}}(\mathbf{x},\tilde{\mathbf{x}})$:

$$\left|\frac{1}{nm}\sum_{i=1}^n\sum_{j=1}^m\frac{2}{D}\left[\cos((\omega_k{}^\top(x_i-\tilde{x}_j)) - \cos((\omega_k'{}^\top(x_i-\tilde{x}_j))\right]\right| \leq \frac{4}{D}. \tag{31}$$

Due to this bounded difference in each coordinate of random feature MMK, we can compute the tail bound using the McDiarmid's inequality,

$$\Pr\left[\left|\mathrm{MMK}_{\hat{\boldsymbol{\phi}}}(\mathbf{x},\tilde{\mathbf{x}}) - \mathrm{MMK}(\mathbf{x},\tilde{\mathbf{x}})\right| \geq t_1\right] \leq 2\exp\left(-\frac{1}{8}Dt_1^2\right). \tag{32}$$

Now using the definition of MMD^2 given in eq. 30, we obtain the tail bound.

$$\Pr\left[\left|\widehat{\mathrm{MMD}}^2_{rf}(\mathbf{x},\tilde{\mathbf{x}}) - \widehat{\mathrm{MMD}}^2(\mathbf{x},\tilde{\mathbf{x}})\right| \geq t_1\right] \leq 2\exp\left(-\frac{1}{128}Dt_1^2\right). \tag{33}$$

□

As a result of Lemma. B.1, the expected absolute error of the random-feature MMD is bounded by

Lemma B.2 (Sec. 3.3 of Sutherland and Schneider 2016)**.** *Given samples $\mathbf{x} = \{x_i\}_{i=1}^n \sim P$ and $\tilde{\mathbf{x}} = \{\tilde{x}_j\}_{j=1}^m \sim Q$, the probabilistic bound between the approximate MMD with random features, denoted by $\widehat{\mathrm{MMD}}_{rf}(\mathbf{x},\tilde{\mathbf{x}})$ and the original MMD, denoted by $\mathrm{MMD}(\mathbf{x},\tilde{\mathbf{x}})$, holds*

$$\mathbb{E}_{\hat{\boldsymbol{\phi}}}\left[\left|\widehat{\mathrm{MMD}}^2_{rf}(\mathbf{x},\tilde{\mathbf{x}}) - \widehat{\mathrm{MMD}}^2(\mathbf{x},\tilde{\mathbf{x}})\right|\right] \leq 8\sqrt{2\pi/D}. \tag{34}$$

Proof. For a non-negative random variable, $\left|\widehat{\mathrm{MMD}}^2_{rf}(P,Q) - \widehat{\mathrm{MMD}}^2(P,Q)\right|$

$$\mathbb{E}_{\hat{\boldsymbol{\phi}}}\left[\left|\widehat{\mathrm{MMD}}^2_{rf}(\mathbf{x},\tilde{\mathbf{x}}) - \widehat{\mathrm{MMD}}^2(\mathbf{x},\tilde{\mathbf{x}})\right|\right] = \int_0^\infty \Pr\left[\left|\widehat{\mathrm{MMD}}^2_{rf}(\mathbf{x},\tilde{\mathbf{x}}) - \widehat{\mathrm{MMD}}^2(\mathbf{x},\tilde{\mathbf{x}})\right| \geq t_1\right] dt_1, \tag{35}$$

$$\leq 2\int_0^\infty \exp\left(-\frac{1}{128}Dt_1^2\right) dt_1, \text{due to Lemma. B.1,} \tag{36}$$

$$= 8\sqrt{\frac{2\pi}{D}}, \text{due to the Gaussian integral.} \tag{37}$$

□

B.2 Randomness due to noise for privacy

The following remark bound the first moment of the privatized MMD proxy $\widetilde{\mathrm{MMD}}_{rf}$ and the MMD proxy $\widehat{\mathrm{MMD}}_{rf}$.

Lemma B.3. *Let $\widetilde{\mathrm{MMD}}_{rf}(\mathbf{x},\tilde{\mathbf{x}}) := \|\widehat{\boldsymbol{\mu}}_P(\mathbf{x}) + \mathbf{n} - \widehat{\boldsymbol{\mu}}_Q(\tilde{\mathbf{x}})\|_2$, where $\mathbf{n} \sim \mathcal{N}(0, \sigma^2\Delta^2_{\widehat{\boldsymbol{\mu}}_P} I_D)$. Also, let $\widehat{\mathrm{MMD}}_{rf}(\mathbf{x},\tilde{\mathbf{x}}) := \|\widehat{\boldsymbol{\mu}}_P(\mathbf{x}) - \widehat{\boldsymbol{\mu}}_Q(\tilde{\mathbf{x}})\|_2$. Then,*

$$\mathbb{E}_{\mathbf{n}}\mathbb{E}_{\hat{\boldsymbol{\phi}}}\left[\left|\widetilde{\mathrm{MMD}}^2_{rf}(\mathbf{x},\tilde{\mathbf{x}}) - \widehat{\mathrm{MMD}}^2_{rf}(\mathbf{x},\tilde{\mathbf{x}})\right|\right] \leq \frac{D\sigma^2}{m^2} + 4\sqrt{2}\sigma\frac{\Gamma\big((D+1)/2\big)}{m\Gamma\big(D/2\big)} \tag{38}$$

Proof.

$$\mathbb{E}_{\mathbf{n}}\mathbb{E}_{\hat{\boldsymbol{\phi}}}\left[\left|\widetilde{\mathrm{MMD}}^2_{rf}(\mathbf{x},\tilde{\mathbf{x}}) - \widehat{\mathrm{MMD}}^2_{rf}(\mathbf{x},\tilde{\mathbf{x}})\right|\right] \overset{(a)}{=} \mathbb{E}_{\hat{\boldsymbol{\phi}}}\left[\mathbb{E}_{\mathbf{n}}\left[\left|\mathbf{n}^\top\mathbf{n} + 2\mathbf{n}^\top(\widehat{\boldsymbol{\mu}}_P(\mathbf{x}) - \widehat{\boldsymbol{\mu}}_Q(\tilde{\mathbf{x}}))\right|\right]\right], \tag{39}$$

$$\overset{(b)}{\leq} \mathbb{E}_{\hat{\boldsymbol{\phi}}}\left[\mathbb{E}_{\mathbf{n}}\left[\mathbf{n}^\top\mathbf{n}\right] + 2\mathbb{E}_{\mathbf{n}}\left[\left|\mathbf{n}^\top(\widehat{\boldsymbol{\mu}}_P(\mathbf{x}) - \widehat{\boldsymbol{\mu}}_Q(\tilde{\mathbf{x}}))\right|\right]\right],$$

$$\overset{(c)}{=} D\sigma^2\Delta^2_{\widehat{\boldsymbol{\mu}}_P} + 2\sqrt{2}\mathbb{E}_{\hat{\boldsymbol{\phi}}}\left[\|\widehat{\boldsymbol{\mu}}_P(\mathbf{x}) - \widehat{\boldsymbol{\mu}}_Q(\mathbf{x})\|_2\right]\sigma\Delta_{\widehat{\boldsymbol{\mu}}_P}\frac{\Gamma\big((D+1)/2\big)}{\Gamma\big(D/2\big)}, \tag{40}$$

$$\overset{(d)}{=} \frac{D\sigma^2}{m^2} + 4\sqrt{2}\sigma\frac{\Gamma\big((D+1)/2\big)}{m\Gamma\big(D/2\big)}, \tag{41}$$

□

where (a) is by expanding two terms following their definitions: $\widetilde{\text{MMD}}^2_{rf}(\mathbf{x}, \tilde{\mathbf{x}}) - \widehat{\text{MMD}}^2_{rf}(\mathbf{x}, \tilde{\mathbf{x}}) = \mathbf{n}^\top \mathbf{n} + 2\mathbf{n}^\top (\widehat{\boldsymbol{\mu}}_P(\mathbf{x}) - \widehat{\boldsymbol{\mu}}_Q(\tilde{\mathbf{x}}))$. (b) is followed by triangle inequality. (c) is followed by the second moment of the chi-square random variable (first term) and the first moment of the chi distribution (second term). (d) is by taking the maximum over random features. Under the random feature representation we use in our paper, the L2-norm of random features is bounded by 1. Hence, $\mathbb{E}_{\hat{\boldsymbol{\phi}}} [\|\widehat{\boldsymbol{\mu}}_P(\mathbf{x}) - \widehat{\boldsymbol{\mu}}_Q(\mathbf{x})\|_2] \leq \max_{\hat{\boldsymbol{\phi}}} [\|\widehat{\boldsymbol{\mu}}_P(\mathbf{x}) - \widehat{\boldsymbol{\mu}}_Q(\mathbf{x})\|_2] \leq \max_{\hat{\boldsymbol{\phi}}} [\|\widehat{\boldsymbol{\mu}}_P(\mathbf{x})\|_2 + \|\widehat{\boldsymbol{\mu}}_Q(\mathbf{x})\|_2] \leq 1 + 1 = 2$.

C Derivation of feature maps for a product of two kernels

Under our assumption, we decompose the kernel below into two kernels:

$$
\begin{aligned}
&k((\mathbf{x}, \mathbf{y}), (\mathbf{x}', \mathbf{y}')) \\
&= k_{\mathbf{x}}(\mathbf{x}, \mathbf{x}') k_{\mathbf{y}}(\mathbf{y}, \mathbf{y}'), \text{ product of two kernels} \\
&\approx \left[\hat{\boldsymbol{\phi}}(\mathbf{x}')^\top \hat{\boldsymbol{\phi}}(\mathbf{x})\right] \left[\mathbf{f}(\mathbf{y})^\top \mathbf{f}(\mathbf{y}')\right], \text{ random features for kernel } k_{\mathbf{x}} \\
&= \text{Tr}\left(\hat{\boldsymbol{\phi}}(\mathbf{x}')^\top \hat{\boldsymbol{\phi}}(\mathbf{x}) \mathbf{f}(\mathbf{y})^\top \mathbf{f}(\mathbf{y}')\right), \\
&= \text{vec}(\hat{\boldsymbol{\phi}}(\mathbf{x}') \mathbf{f}(\mathbf{y}')^\top)^\top \text{vec}(\hat{\boldsymbol{\phi}}(\mathbf{x}) \mathbf{f}(\mathbf{y})^\top) = \hat{\mathbf{f}}(\mathbf{x}', \mathbf{y}')^\top \hat{\mathbf{f}}(\mathbf{x}, \mathbf{y})
\end{aligned}
$$

D Derivation of feature maps for a sum of two kernels

Under our assumption, we compose the kernel below from the sum of two kernels:

$$
\begin{aligned}
&k((\mathbf{x}_{num}, \mathbf{x}_{cat}), (\mathbf{x}'_{num}, \mathbf{x}'_{cat})) \\
&= k_{num}(\mathbf{x}_{num}, \mathbf{x}'_{num}) + k_{cat}(\mathbf{x}_{cat}, \mathbf{x}'_{cat}), \\
&\approx \hat{\boldsymbol{\phi}}(\mathbf{x}_{num})^\top \hat{\boldsymbol{\phi}}(\mathbf{x}'_{num}) + \tfrac{1}{\sqrt{d_{cat}}} \mathbf{x}_{cat}{}^\top \mathbf{x}'_{cat}, \\
&= \begin{bmatrix} \hat{\boldsymbol{\phi}}(\mathbf{x}_{num}) \\ \frac{1}{\sqrt{d_{cat}}} \mathbf{x}_{cat} \end{bmatrix}^T \begin{bmatrix} \hat{\boldsymbol{\phi}}(\mathbf{x}_{num}) \\ \frac{1}{\sqrt{d_{cat}}} \mathbf{x}_{cat} \end{bmatrix} \\
&= \hat{\mathbf{h}}(\mathbf{x}_{num}, \mathbf{x}_{cat})^T \hat{\mathbf{h}}(\mathbf{x}_{num}, \mathbf{x}_{cat}).
\end{aligned}
$$

E Sensitivity of class counts

Consider the vector of class counts $\mathbf{m} = [m_1, \cdots, m_C]$, where each element m_c is the number of samples with class c in the dataset. The class counts of two neighbouring datasets $\mathcal{D}$ and $\mathcal{D}' = (\mathcal{D} \setminus \{\mathbf{x}\}) \cup \{\mathbf{x}'\}$ can differ in at most two entries k, l and at most by 1 in either entry. Assuming $\mathbf{y} \neq \mathbf{y}'$, then for $\mathbf{y}_k = 1$, $m_k = m'_k + 1$ and for $\mathbf{y}'_l = 1$, $m'_l = m_l + 1$ and $m_i = m'_i$ in all other cases. If $\mathbf{y} = \mathbf{y}'$, then $\mathbf{m} = \mathbf{m}'$. Letting $\mathbf{m}$ and $\mathbf{m}'$ denote the class counts of $\mathcal{D}$ and $\mathcal{D}'$ respectively, we get the following:

$$
\Delta_{\mathbf{m}} = \max_{\mathcal{D}, \mathcal{D}'} \|\mathbf{m} - \mathbf{m}'\|_2 = \max_{\mathcal{D}, \mathcal{D}'} \sqrt{\sum_{i=1}^{C} m_i - m'_i} = \sqrt{2} \tag{42}
$$

F Sensitivity of $\hat{\boldsymbol{\mu}}_P$ with homogeneous data

Below, we show that the sensitivity of the data mean embedding for homogeneous labeled data is the same as for unlabeled data. In order, we first use the fact that $\mathcal{D}$ and $\mathcal{D}'$ are neighbouring, which implies that $m - 1$ of the summands on each side cancel and we are left with the only distinct datapoints, which we denote as $(\mathbf{x}, \mathbf{y})$ and $(\mathbf{x}', \mathbf{y}')$. We then apply the triangle inequality and the definition of $\mathbf{f}$. As $\mathbf{y}$ is a one-hot vector, all but one column of $\hat{\boldsymbol{\phi}}(\mathbf{x})\mathbf{y}^\top$ are 0, so we omit them in the next step and finally use that $\|\hat{\boldsymbol{\phi}}(\mathbf{x})\|_2 = 1$.

$$\Delta_{\hat{\boldsymbol{\mu}}_P} = \max_{\mathcal{D},\mathcal{D}'} \left\| \frac{1}{m} \sum_{(\mathbf{x}_i,\mathbf{y}_i)\in\mathcal{D}} \hat{\mathbf{f}}(\mathbf{x}_i,\mathbf{y}_i) - \frac{1}{m} \sum_{(\mathbf{x}'_i,\mathbf{y}'_i)\in\mathcal{D}'} \hat{\mathbf{f}}(\mathbf{x}'_i,\mathbf{y}'_i) \right\|_F \tag{43}$$

$$= \max_{(\mathbf{x},\mathbf{y}),(\mathbf{x}',\mathbf{y})} \left\| \frac{1}{m}\hat{\mathbf{f}}(\mathbf{x},\mathbf{y}) - \frac{1}{m}\hat{\mathbf{f}}(\mathbf{x}',\mathbf{y}') \right\|_F \tag{44}$$

$$\leq \max_{(\mathbf{x},\mathbf{y})} \frac{2}{m} \left\| \hat{\mathbf{f}}(\mathbf{x},\mathbf{y}) \right\|_F \tag{45}$$

$$= \max_{(\mathbf{x},\mathbf{y})} \frac{2}{m} \left\| \hat{\boldsymbol{\phi}}(\mathbf{x})\mathbf{y}^\top \right\|_F \tag{46}$$

$$= \max_{\mathbf{x}} \frac{2}{m} \left\| \hat{\boldsymbol{\phi}}(\mathbf{x}) \right\|_2 \tag{47}$$

$$= \frac{2}{m} \tag{48}$$

G Sensitivity of $\boldsymbol{\mu}_P$ with heterogeneous data

In the case of heterogeneous data, recall that $\hat{\mathbf{h}}(\mathbf{x}^{(i)}_{num}, \mathbf{x}^{(i)}_{cat}) = \begin{bmatrix} \hat{\boldsymbol{\phi}}(\mathbf{x}^{(i)}_{num}) \\ \frac{1}{\sqrt{d_{cat}}}\mathbf{x}^{(i)}_{cat} \end{bmatrix}$ and $\boldsymbol{\mu}_P = \frac{1}{m}\sum_{(\mathbf{x}_i,\mathbf{y}_i)\in\mathcal{D}} \hat{\mathbf{h}}(\mathbf{x}_i)\mathbf{y}_i{}^\top$ where $\mathbf{x}_i$ is the concatenation of $\mathbf{x}^{(i)}_{num}$ and $\mathbf{x}^{(i)}_{cat}$. Analogous to the homogeneous case, we first derive that the labeled and unlabeled embedding have the same sensitivity (in eq. 52). We apply the definition of $\hat{\mathbf{h}}$ and analyze the numerical and categorical parts separately, using the facts that $\|\hat{\boldsymbol{\phi}}(\mathbf{x})\|_2 = 1$ and, since $\mathbf{x}_{cat}$ is binary, $\|\mathbf{x}_{cat}\|_2 \leq \sqrt{d_{cat}}$.

$$\Delta_{\boldsymbol{\mu}_P} = \max_{\mathcal{D},\mathcal{D}'} \left\| \frac{1}{m} \sum_{(\mathbf{x}_i,\mathbf{y}_i)\in\mathcal{D}} \hat{\mathbf{h}}(\mathbf{x}_i)\mathbf{y}_i{}^\top - \frac{1}{m} \sum_{(\mathbf{x}'_i,\mathbf{y}'_i)\in\mathcal{D}'} \hat{\mathbf{h}}(\mathbf{x}'_i)\mathbf{y}'_i{}^\top \right\|_F \tag{49}$$

$$= \max_{(\mathbf{x},\mathbf{y}),(\mathbf{x}',\mathbf{y}')} \left\| \frac{1}{m}\hat{\mathbf{h}}(\mathbf{x}_i)\mathbf{y}_i{}^\top - \frac{1}{m}\hat{\mathbf{h}}(\mathbf{x}'_i)\mathbf{y}'_i{}^\top \right\|_F \tag{50}$$

$$\leq \max_{(\mathbf{x},\mathbf{y})} \frac{2}{m} \left\| \hat{\mathbf{h}}(\mathbf{x})\mathbf{y}^\top \right\|_F \tag{51}$$

$$= \max_{\mathbf{x}} \frac{2}{m} \left\| \hat{\mathbf{h}}(\mathbf{x}) \right\|_2 \tag{52}$$

$$= \max_{\mathbf{x}} \frac{2}{m} \left\| \begin{bmatrix} \hat{\boldsymbol{\phi}}(\mathbf{x}_{num}) \\ \frac{1}{\sqrt{d_{cat}}}\mathbf{x}_{cat} \end{bmatrix} \right\|_2 \tag{53}$$

$$= \max_{\mathbf{x}} \frac{2}{m} \sqrt{\|\hat{\boldsymbol{\phi}}(\mathbf{x}_{num})\|_2^2 + \|\frac{1}{\sqrt{d_{cat}}}\mathbf{x}_{cat}\|_2^2} \tag{54}$$

$$= \frac{2}{m}\sqrt{1 + \frac{d_{cat}}{d_{cat}}} \tag{55}$$

$$= \frac{2\sqrt{2}}{m} \tag{56}$$

I Variables in heterogeneous data are not treated as independent

While the impression may arise, our method does not assume independence between the continuous and the discrete variables, but models correlations between the two types of variables implicitly. With the sum of two kernels, the embedding is a concatenation of the two: $[E_x\phi_x(x), E_y\phi_y(y)]$, where E_x means expectation wrt $p(x)$ and E_y is wrt $p(y)$. To compute $p(x)$, we need $p(y)$ with which we marginalize out y, as $p(x) = \int p(x,y)dy$. This marginalization implicitly takes into account the correlation between the two. This is less explicit than the case using the product of two kernels. However, the sum kernel is chosen for computational tractability: a sum kernel in Fourier representation has $d_x + d_y$ features while a product kernel has $d_x \cdot d_y$.

K Heterogeneous and homogenous tabular data

In this section we describe the tabular datasets we have used in our experiments with their respective sources. We include the details of data preprocessing in case it was performed on a dataset. The datasets in this form were used in all our experiments as well as the experiments on the benchmark methods.

Credit

Credit card fraud detection dataset contains the categorized information of credit card transactions which were either fraudelent or not. Ten dataset comes from a Kaggle competition and is available at the source, `https://www.kaggle.com/mlg-ulb/creditcardfraud`. The original data has 284807 examples, of which negative samples are 284315 and positive 492. The dataset has 31 categories, 30 numerical features and a binary label. We used all but the first feature (Time).

Epileptic

Epileptic dataset describes brain activity with numerical features being EEG recording at a different point in time. The dataset comes from the UCI database, `https://archive.ics.uci.edu/ml/datasets/Epileptic+Seizure+Recognition`. It contains 11500 data points, and 179 categories, 178 features and a label. The original dataset contains five different labels which we binarize into two states, seizure or no seizure. Thus, there are 9200 negative samples and 2300 positive samples.

Census

The dataset can be downloaded by means of SDGym package, `https://pypi.org/project/sdgym/`. The dataset has 199523 examples, 187141 are negative and 12382 are positive. There are 40 categories and a binary label. This dataset contains 7 numerical and 33 categorical features.

Intrusion

The dataset was used for The Third International Knowledge Discovery and Data Mining Tools Competition held at the Conference on Knowledge Discovery and Data Mining, 1999, and can be found at `http://kdd.ics.uci.edu/databases/kddcup99/kddcup99.html`. We used the file, kddcup.data_10_percent.gz. It is a multi-class dataset with five labels describing different types of connection intrusions. The labels were first grouped into five categories and due to few examples, we restricted the data to the top four categories.

Adult

The dataset contains information about people's attributes and their respective income which has been thresholded and binarized. It has 22561 examples, and 14 features and a binary label. The dataset can be downloaded by means of SDGym package,`https://pypi.org/project/sdgym/`.

Isolet

The dataset contains sound features to predict a spoken letter of alphabet. The inputs are sound features and the output is a latter. We binaried the labels into two classes, consonants and vowels. The dataset can be found at `https://archive.ics.uci.edu/ml/datasets/isolet`

Cervical

This dataset is created with the goal to identify the risk factors associated with cervical cancer. It is the smallest dataset with 858 instances, and 35 attributes, of which The data can be found at 15 are numerical 24 are categorical (binary). The dataset can be found at `https://archive.ics.uci.edu/ml/datasets/Cervical+cancer+%28Risk+Factors%29`. The data, however, contains missing data. We followed the pre-processing suggested at `https://www.kaggle.com/saflynn/cervical-cancer-lynn` and further removed the data with the most missing values and replaced the rest with the category mean value.

Covtype

The dataset describes forest cover type from cartographic variables. The data can be found at `https://archive.ics.uci.edu/ml/datasets/covertype`. It contains 53 attributes and a multi-class label with 7 classes of forest cover types.

K.1 The training

We provide here the details of training procedure. Some of the datasets are very imbalanced, that is they contain much more examples with one label over the others. In attempt of making categories more balanced, we undersampled the class with the largest number of samples. The complexity of a dataset also determined the number of Fourier features we used. We also varied the batch size (we include the fraction of dataset used in a batch), and the number of epochs in the training. We provide the detailed parameter settings for each of the dataset in the following table.

Table 4: Parameters settings for training tabular datasets

	non-private			**private**			
	# epochs	mini-batch size	# Fourier features	# epochs	mini-batch size	# Fourier features	undersampling rate
adult	8000	0.1	50000	8000	0.1	1000	0.4
census	200	0.5	10000	2000	0.5	10000	0.4
cervical	2000	0.6	2000	200	0.5	2000	1
credit	4000	0.6	50000	4000	0.5	5000	0.005
epileptic	6000	0.5	100000	6000	0.5	80000	1
isolet	4000	0.6	100000	4000	0.5	500	1
covtype	6000	0.05	1000	6000	0.05	1000	0.03
intrusion	10000	0.03	2000	10000	0.03	2000	0.1

K.2 Detailed results for binary class dataset

In the main text we included the details for a multi-class dataset and here we also include the results across all the classification methods for a binary dataset in Table 5 and Table 6. We also include the best and average F1-score over five runs for the respective classification methods in Table 7 and Table 8. Notice that this average corresponds to the average reported in Table 1 in the main text.

Table 5: Performance comparison on Credit dataset. The highest performance in five runs.

	Real		DP-CGAN (non-priv)		DP-MERF (non-priv)		DP-CGAN $(1, 10^{-5})$-DP		**DP-MERF** $(1, 10^{-5})$-DP	
	ROC	PRC	ROC	PRC	ROC	PRC	ROC	PRC	ROC	PRC
Logistic Regression	0.95	0.91	0.83	0.37	0.92	0.79	0.74	0.52	0.78	0.61
Gaussian Naive Bayes	0.90	0.80	0.85	0.39	0.92	0.76	0.80	0.55	0.65	0.48
Bernoulli Naive Bayes	0.89	0.84	0.58	0.19	0.89	0.82	0.67	0.42	0.90	0.74
Linear SVM	0.92	0.89	0.84	0.48	0.91	0.65	0.78	0.45	0.64	0.38
Decision Tree	0.91	0.82	0.74	0.32	0.92	0.69	0.58	0.22	0.72	0.58
LDA	0.87	0.82	0.86	0.53	0.82	0.68	0.58	0.24	0.69	0.51
Adaboost	0.94	0.89	0.83	0.51	0.93	0.85	0.62	0.32	0.75	0.63
Bagging	0.91	0.84	0.79	0.42	0.91	0.79	0.57	0.21	0.74	0.61
Random Forest	0.93	0.90	0.82	0.54	0.92	0.86	0.63	0.31	0.75	0.62
GBM	0.94	0.89	0.85	0.54	0.94	0.85	0.58	0.22	0.74	0.61
Multi-layer perceptron	0.92	0.89	0.83	0.47	0.91	0.74	0.78	0.55	0.66	0.44
XGBoost	0.94	0.91	0.81	0.49	0.94	0.87	0.70	0.53	0.72	0.59
Average	0.91	0.86	0.80	0.44	0.91	0.78	0.67	0.38	0.73	0.57

Table 6: Performance comparison on Credit dataset. The average performance over five runs.

	DP-MERF (non-private)		DP-MERF (private)	
	ROC	PRC	ROC	PRC
Logistic Regression	0.919	0.808	0.796	0.665
Gaussian Naive Bayes	0.898	0.725	0.729	0.582
Bernoulli Naive Bayes	0.879	0.791	0.752	0.586
Linear SVM	0.876	0.667	0.742	0.549
Decision Tree	0.901	0.700	0.775	0.650
LDA	0.838	0.697	0.725	0.544
Adaboost	0.912	0.828	0.787	0.689
Bagging	0.909	0.805	0.811	0.709
Random Forest	0.911	0.840	0.786	0.686
GBM	0.917	0.812	0.807	0.707
Multi-layer perceptron	0.905	0.777	0.747	0.570
XGBoost	0.915	0.837	0.812	0.716
Average	0.898	0.774	0.772	0.638

Table 7: Performance comparison on Intrusion dataset. The highest performance in five runs.

	Real	DP-CGAN (non-priv)	DP-MERF (non-priv)	DP-CGAN $(1, 10^{-5})$-DP	DP-MERF $(1, 10^{-5})$-DP
Logistic Regression	0.948	0.710	0.926	0.567	0.940
Gaussian Naive Bayes	0.757	0.503	0.804	0.215	0.736
Bernoulli Naive Bayes	0.927	0.693	0.822	0.475	0.755
Linear SVM	0.983	0.639	0.922	0.915	0.937
Decision Tree	0.999	0.496	0.862	0.153	0.952
LDA	0.990	0.224	0.910	0.652	0.950
Adaboost	0.947	0.898	0.924	0.398	0.503
Bagging	1.000	0.499	0.914	0.519	0.956
Random Forest	1.000	0.497	0.941	0.676	0.943
GBM	0.999	0.501	0.924	0.255	0.933
Multi-layer perceptron	0.997	0.923	0.933	0.733	0.957
XGBoost	0.999	0.886	0.921	0.751	0.933
Average	0.962	0.622	0.900	0.526	0.875

Table 8: Performance comparison on Intrusion dataset. The average performance as F1 score over five runs.

	DP-MERF (non-private)	DP-MERF (private)
Logistic Regression	0.891	0.928
Gaussian Naive Bayes	0.845	0.792
Bernoulli Naive Bayes	0.454	0.508
Linear SVM	0.890	0.917
Decision Tree	0.911	0.907
LDA	0.859	0.925
Adaboost	0.899	0.592
Bagging	0.926	0.922
Random Forest	0.904	0.923
GBM	0.901	0.926
Multi-layer perceptron	0.898	0.941
XGBoost	0.891	0.921
Average	0.856	0.850

L Image data

L.1 Datasets

Both digit and fashion MNIST datasets are loaded through the torchvision package and used without further preprocessing. Both datasets of size 60000 consist of samples from 10 classes, which are close to perfectly balanced. Each sample is a 28x28 pixel image and thus of significantly higher dimensionality than the tabular data we tested.

L.2 Detailed results

A detailed version of the results summarized in Table 3 of the paper are shown below, for digit MNIST is Table 9 and fashion MNIST in Table 10. All scores are the average of 5 independent runs of training a generator and evaluating the synthetic data it produced. The tables show that DP-MERF consistently outperforms the other approaches across models. The only exceptions are Gaussian Naive Bayes and XGBoost on MNIST, where GS-WGAN and DP-CGAN respectively perform slightly better.

Table 9: Test accuracy on digit MNIST data. Average over 5 runs (data generation & model training). Best scores among private models are bold.

	Real	DP-CGAN $\epsilon = 9.6$	DP-GAN $\epsilon = 9.6$	GS-WGAN $\epsilon = 10$	DP-MERF $\epsilon = \infty$	DP-MERF $\epsilon = 1$	DP-MERF $\epsilon = 0.2$
Logistic Regression	0.930	0.600	0.702	0.741	0.772	**0.769**	0.772
Random Forest	0.969	0.638	0.538	0.460	0.714	**0.685**	0.702
Gaussian Naive Bayes	0.560	0.310	0.364	**0.576**	0.527	0.545	0.539
Bernoulli Naive Bayes	0.840	0.610	0.702	0.699	0.746	**0.750**	0.780
Linear SVM	0.920	0.550	0.700	0.704	0.756	**0.746**	0.726
Decision Tree	0.880	0.340	0.255	0.326	0.443	**0.456**	0.346
LDA	0.879	0.590	0.694	0.732	0.789	**0.793**	0.753
Adaboost	0.729	0.254	0.159	0.170	0.441	**0.456**	0.362
MLP	0.978	0.564	0.652	0.744	0.807	**0.807**	0.768
Bagging	0.928	0.430	0.282	0.387	0.624	**0.602**	0.508
GBM	0.909	0.460	0.205	0.362	0.678	**0.659**	0.552
XGBoost	0.912	**0.614**	0.459	0.408	0.525	0.555	0.509
Average	0.870	0.500	0.476	0.526	0.652	**0.652**	0.610

Table 10: Test accuracy on fashion MNIST data. Average over 5 runs (data generation & model training). Best scores among private models are bold.

	Real	DP-CGAN $\epsilon = 9.6$	DP-GAN $\epsilon = 9.6$	GS-WGAN $\epsilon = 10$	DP-MERF $\epsilon = \infty$	DP-MERF $\epsilon = 1$	DP-MERF $\epsilon = 0.2$
Logistic Regression	0.844	0.461	0.626	0.674	0.725	**0.728**	0.714
Random Forest	0.875	0.482	0.573	0.498	0.657	**0.684**	0.553
Gaussian Naive Bayes	0.585	0.286	0.149	0.505	0.598	**0.575**	0.467
Bernoulli Naive Bayes	0.648	0.497	0.592	0.558	0.602	**0.604**	0.629
Linear SVM	0.839	0.389	0.613	0.639	0.685	**0.684**	0.697
Decision Tree	0.790	0.315	0.317	0.389	0.433	**0.462**	0.352
LDA	0.799	0.490	0.638	0.653	0.735	**0.733**	0.701
Adaboost	0.561	0.217	0.224	0.275	0.291	**0.359**	0.258
MLP	0.879	0.459	0.601	0.647	0.739	**0.738**	0.696
Bagging	0.841	0.309	0.410	0.413	0.576	**0.593**	0.372
GBM	0.834	0.331	0.254	0.352	0.626	**0.624**	0.429
XGBoost	0.826	0.489	0.478	0.427	0.596	**0.610**	0.445
Average	0.780	0.390	0.457	0.502	0.605	**0.616**	0.526

M Comparison with other methods

M.1 Comparison with [4].

Algorithm 2 in [4] uses the random features similar to ours, while it releases the privatized mean embedding in terms of a weighted sum of feature maps evaluated at synthetic datapoints. The challenge is that optimizing for the synthetic datapoints using the reduced-set method becomes harder in high dimensions. To illustrate this point, we took the simulated data generated from *5-dimensional* mixture of Gaussians (the dataset [4] used). Unlike [4], our method directly trains a neural-net based generator, which can effectively approximate the privatized kernel mean embedding of the data. As a result, our method reduces the distance (this metric [4] used) between between the true kernel mean embedding $\hat{\mu}_x$ and that of the released dataset as we increase the number of synthetic datapoints, as shown in Fig. 3.

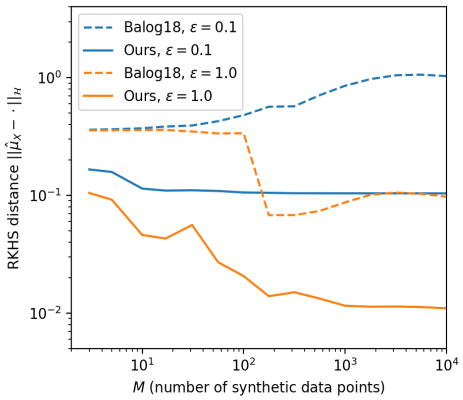


Figure 3: Comparison to [4].

M.2 Comparison with PrivBayes [38].

We compare our method to PrivBayes [38] using the published code from [15], which builds on the original code with [37] as a wrapper. We test the model on the Adult and Census datasets used in our paper by creating a version $\mathcal{D}$ of the dataset where all continuous features are discretized, and a version $\mathcal{D}^*$ where the domain of all features is reduced to a max of 15 to reduce complexity. Following [38], we measure α-way marginals for varying levels of ϵ-DP and compare them to DP-MERF at (ϵ, δ)-DP with $\delta = 10^{-5}$. Optimizing the "usefulness" parameter θ, we find, as in [38], that $\theta = 4$ is close to optimal in most settings. Results for the best θ are shown. We observe that PrivBayes performs better at $\epsilon = 1$, but is more affected by increased noise, so at $\epsilon = 0.3$ the methods are roughly tied and at $\epsilon = 0.1$ DP-MERF has lower error.

2*Adult		PrivBayes			DP-MERF			2*Census		PrivBayes			DP-MERF		
		ϵ=1	ϵ=0.3	ϵ=0.1	ϵ=1	ϵ=0.3	ϵ=0.1			ϵ=1	ϵ=0.3	ϵ=0.1	ϵ=1	ϵ=0.3	ϵ=0.1
2*$\mathcal{D}$	α=3	0.275	0.446	0.577	0.348	0.405	0.480	2*$\mathcal{D}$	α=2	0.131	0.180	0.291	0.172	0.190	0.222
	α=4	0.377	0.547	0.673	0.468	0.508	0.590		α=3	0.264	0.323	0.429	0.291	0.302	0.337
2*$\mathcal{D}^*$	α=3	0.182	0.284	0.317	0.235	0.287	0.352	2*$\mathcal{D}^*$	α=2	0.111	0.136	0.199	0.139	0.140	0.176
	α=4	0.257	0.371	0.401	0.301	0.363	0.453		α=3	0.199	0.258	0.325	0.228	0.234	0.269

It is important to stress that our approach is more general than PrivBayes in that *(i)* it does not require discretization of the data and *(ii)* scales to higher dimensionality and arbitrary domains. Bayesian network construction in PrivBayes for a

k-degree graph with d nodes (i.e. features) compares up to $\binom{d}{k}$ options on each iteration, which restricts k to small values if d is large. This means, e.g., testing PrivBayes on binarized MNIST ($d = 784$) with any $k > 2$ is infeasible.

C

HERMITE POLYNOMIAL FEATURES FOR PRIVATE DATA GENERATION

The following paper has been published in the Proceedings of the 39th International Conference on Machine Learning under the CC-BY-4.0 license[1] and is reprinted without modifications.

1 https://creativecommons.org/licenses/by/4.0/

Hermite Polynomial Features for Private Data Generation

Margarita Vinaroz [*1] **Mohammad-Amin Charusaie** [*1] **Frederik Harder** [1] **Kamil Adamczewski** [1]
Mi Jung Park [2]

Abstract

Kernel mean embedding is a useful tool to represent and compare probability measures. Despite its usefulness, kernel mean embedding considers infinite-dimensional features, which are challenging to handle in the context of *differentially private* data generation. A recent work (Harder et al., 2021) proposes to approximate the kernel mean embedding of data distribution using *finite-dimensional random features*, which yields analytically tractable sensitivity. However, the number of required random features is excessively high, often ten thousand to a hundred thousand, which worsens the privacy-accuracy trade-off. To improve the trade-off, we propose to replace random features with *Hermite polynomial* features. Unlike the random features, the Hermite polynomial features are *ordered*, where the features at the low orders contain more information on the distribution than those at the high orders. Hence, a relatively low order of Hermite polynomial features can more accurately approximate the mean embedding of the data distribution compared to a significantly higher number of random features. As demonstrated on several tabular and image datasets, Hermite polynomial features seem better suited for private data generation than random Fourier features.

1. Introduction

One of the popular distance metrics for generative modelling is *Maximum Mean Discrepancy* (MMD) (Gretton et al., 2012). MMD computes the average distance between the realizations of two distributions mapped to a reproducing kernel Hilbert space (RKHS). Its popularity is due to several facts: (a) MMD can compare two probability measures in terms of all possible moments (i.e., infinite-dimensional features), resulting in no information loss due to a particular selection of moments; and (b) estimating MMD does not require the knowledge of the probability density functions. Rather, MMD estimators are in closed form, which can be computed by pair-wise evaluations of a kernel function using the points drawn from two distributions.

However, using the MMD estimators for training a generator is not well suited when *differential privacy* (DP) of the generated samples is taken into consideration. In fact, the generated points are updated in every training step and the pair-wise evaluations of the kernel function on generated and true data points require accessing data multiple times. One of the key properties of DP is composability that implies each access of data causes privacy loss. Hence, privatizing the MMD estimator in every training step – which is necessary to ensure the resulting generated samples are differentially private – incurs a large privacy loss.

A recent work (Harder et al., 2021), called *DP-MERF*, uses a particular form of MMD via a *random Fourier feature* representation (Rahimi & Recht, 2008) of kernel mean embeddings for DP data generation. Under this representation, one can approximate the MMD in terms of two finite-dimensional mean embeddings (as in eq. 3), where the approximate mean embedding of the true data distribution (data-dependent) is detached from that of the synthetic data distribution (data-independent). Thus, the data-dependent term needs privatization only once and can be re-used repeatedly during training of a generator. However, DP-MERF requires an excessively high number of random features to approximate the mean embedding of data distributions.

We propose to replace[1] the random feature representation of the kernel mean embedding with the *Hermite polynomial* representation. We observe that Hermite polynomial features are ordered where the features at the low orders contain more information on the distribution than those at the high orders. Hence, the required order of Hermite polynomial features is significantly lower than the required number of

[*]Equal contribution [1]Max Planck Institute for Intelligent Systems, Tuebingen, Germany [2]University of British Columbia, Vancouver, Canada. CIFAR AI Chair at AMII. Correspondence to: Mi Jung Park <mijungp@cs.ubc.ca>.

Proceedings of the 39^{th} International Conference on Machine Learning, Baltimore, Maryland, USA, PMLR 162, 2022.

[1]There are efforts on improving the efficiency of randomized Fourier feature maps, e.g., by using quasi-random points in (Avron et al., 2016).

random features, for the similar quality of the kernel approximation (see Fig. 1). This is useful in reducing the *effective sensitivity* of the data mean embedding. Although the sensitivity is $\frac{1}{m}$ in both cases with the number of data samples m (see Sec. 3), adding noise to a vector of longer length (when using random features) has a worse signal-to-noise ratio, as opposed to adding noise to a vector of shorter length (when using Hermite polynomial features), if we require the norms of these vectors to be the same (for a limited sensitivity). Furthermore, the Hermite polynomial features maintain a better signal-to-noise ratio as it contains more information on the data distribution, even when Hermite polynomial features are the same length as the random Fourier features

To this end, we develop a private data generation paradigm, called *differentially private Hermite polynomials* (DP-HP), which utilizes a novel kernel which we approximate with Hermite polynomial features in the aim of effectively tackling the privacy-accuracy trade-off. In terms of three different metrics we use to quantify the quality of generated samples, our method outperforms the state-of-the-art private data generation methods at the same privacy level. What comes next describes relevant background information before we introduce our method.

2. Background

In the following, we describe the background on kernel mean embeddings and differential privacy.

2.1. Maximum Mean Discrepancy

Given a positive definite kernel $k\colon \mathcal{X} \times \mathcal{X}$, the MMD between two distributions P, Q is defined as (Gretton et al., 2012): $\mathrm{MMD}^2(P,Q) = \mathbb{E}_{x,x'\sim P}k(x,x') + \mathbb{E}_{y,y'\sim Q}k(y,y') - 2\mathbb{E}_{x\sim P}\mathbb{E}_{y\sim Q}k(x,y)$. According to the Moore–Aronszajn theorem (Aronszajn, 1950), there exists a unique reproducing kernel Hilbert space of functions on $\mathcal{X}$ for which k is a reproducing kernel, i.e., $k(x,\cdot) \in \mathcal{H}$ and $f(x) = \langle f, k(x,\cdot)\rangle_{\mathcal{H}}$ for all $x \in \mathcal{X}$ and $f \in \mathcal{H}$, where $\langle\cdot,\cdot\rangle_{\mathcal{H}} = \langle\cdot,\cdot\rangle$ denotes the inner product on $\mathcal{H}$. Hence, we can find a *feature map*, $\phi\colon \mathcal{X} \to \mathcal{H}$ such that $k(x,y) = \langle\phi(x),\phi(y)\rangle_{\mathcal{H}}$, which allows us to rewrite MMD as (Gretton et al., 2012):

$$\mathrm{MMD}^2(P,Q) = \left\|\mathbb{E}_{x\sim P}[\phi(x)] - \mathbb{E}_{y\sim Q}[\phi(y)]\right\|^2_{\mathcal{H}}, \quad (1)$$

where $\mathbb{E}_{x\sim P}[\phi(x)] \in \mathcal{H}$ is known as the (kernel) mean embedding of P, and exists if $\mathbb{E}_{x\sim P}\sqrt{k(x,x)} < \infty$ (Smola et al., 2007). If k is *characteristic* (Sriperumbudur et al., 2011), then $P \mapsto \mathbb{E}_{x\sim P}[\phi(x)]$ is injective, meaning $\mathrm{MMD}(P,Q) = 0$, if and only if $P = Q$. Hence, the MMD associated with a characteristic kernel (e.g., Gaussian kernel) can be interpreted as a distance between the mean embeddings of two distributions.

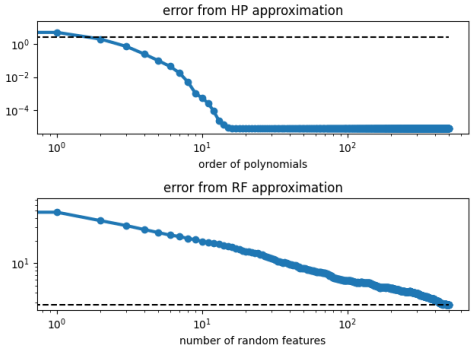


Figure 1. **HP VS. RF features.** Dataset X contains $N = 100$ samples drawn from $\mathcal{N}(0,1)$ and X' contains $N = 100$ samples drawn from $\mathcal{N}(1,1)$. The error is defined by: $\frac{1}{N^2}\sum_{i=1}^{N}\sum_{j=1}^{N}|k(x_i,x'_j) - \hat{\phi}(x_i)^\top\hat{\phi}(x'_j)|$ where $\hat{\phi}$ is either RF or HP features. **Top**: The error decays fast when using HP features (eq. 6). **Bottom**: The plot shows the average error over 100 independent draws of RF features (eq. 4). The error decays slowly when using RF features. The best error (black dotted line) using 500 RF features coincides with the error using HP features with order 2 only.

Given the samples drawn from two distributions: $X_m = \{x_i\}_{i=1}^m \sim P$ and $X'_n = \{x'_i\}_{i=1}^n \sim Q$, we can estimate[2] the MMD by sample averages (Gretton et al., 2012):

$$\widehat{\mathrm{MMD}}^2(X_m, X'_n) = \frac{1}{m^2}\sum_{i,j=1}^{m} k(x_i,x_j) + \frac{1}{n^2}\sum_{i,j=1}^{n} k(x'_i,x'_j) - \frac{2}{mn}\sum_{i=1}^{m}\sum_{j=1}^{n} k(x_i,x'_j). \quad (2)$$

However, at $O(mn)$ the computational cost of $\widehat{\mathrm{MMD}}(X_m, X'_n)$ is prohibitive for large-scale datasets.

2.2. Kernel approximation

By approximating the kernel function $k(x,x')$ with an inner product of finite dimensional feature vectors, i.e., $k(x,x') \approx \hat{\phi}(x)^\top\hat{\phi}(x')$ where $\hat{\phi}(x) \in \mathbb{R}^A$ and A is the number of features, the MMD estimator given in eq. 2 can be computed in $O(m+n)$, i.e., linear in the sample size:

$$\widehat{\mathrm{MMD}}^2(P,Q) = \left\|\frac{1}{m}\sum_{i=1}^{m}\hat{\phi}(x_i) - \frac{1}{n}\sum_{i=1}^{n}\hat{\phi}(x'_i)\right\|_2^2. \quad (3)$$

This approximation is also beneficial for private data generation: assuming P is a data distribution and Q is a synthetic data distribution, we can summarize the data distribution in terms of its kernel mean embedding (i.e., the first term on the right-hand side of eq. 3), which can be privatized only

[2]This particular MMD estimator is biased.

once and used repeatedly during training of the generator which produces samples from Q.

2.3. Random Fourier features.

As an example of $\hat{\phi}(\cdot)$, the random Fourier features (Rahimi & Recht, 2008) are derived from the following. Bochner's theorem (Rudin, 2013) states that for any translation invariant kernel, the kernel can be written as $k(x, x') = \tilde{k}(x - x') = \mathbb{E}_{\omega\sim\Lambda} \cos(\omega^\top (x - x'))$. By drawing random frequencies $\{\omega_i\}_{i=1}^A \sim \Lambda$, where Λ depends on the kernel, (e.g., a Gaussian kernel k corresponds to normal distribution Λ), $\tilde{k}(x - x')$ can be approximated with a Monte Carlo average. The resulting vector of random Fourier features (of length A) is given by

$$\hat{\boldsymbol{\phi}}_{RF,\boldsymbol{\omega}}(x) = (\hat{\phi}_{1,\boldsymbol{\omega}}(x), \ldots, \hat{\phi}_{A,\boldsymbol{\omega}}(x))^\top \tag{4}$$

where $\hat{\phi}_{j,\boldsymbol{\omega}}(x) = \sqrt{2/A}\ \cos({\omega_j}^\top x)$, $\hat{\phi}_{j+A/2,\boldsymbol{\omega}}(x) = \sqrt{2/A}\sin(\omega_j^\top x)$, for $j = 1, \cdots, A/2$.

DP-MERF (Harder et al., 2021) uses this very representation of the feature map given in eq. 4, and minimizes eq. 3 with a privatized data mean embedding to train a generator.

2.4. Hermite polynomial features.

For another example of $\hat{\phi}(\cdot)$, one could also start with the *Mercer's theorem* (See Appendix Sec. C), which allows us to express a positive definite kernel k in terms of the eigen-values λ_i and eigen-functions f_i: $k(x, x') = \sum_{i=1}^\infty \lambda_i f_i(x) f_i^*(x')$, where $\lambda_i > 0$ and complex conjugate is denoted by $*$. The resulting *finite-dimensional* feature vector is simply $\hat{\phi}(x) = \hat{\phi}_{HP}(x) = [\sqrt{\lambda_0} f_0(x), \sqrt{\lambda_1} f_1(x), \cdots, \sqrt{\lambda_C} f_C(x)]$, where the cut-off is made at the C-th eigen-value and eigen-function. For the commonly-used Gaussian kernel, $k(x, x') = \exp(-\frac{1}{2l^2}(x - x')^2)$, where l is the length scale parameter, an analytic form of eigen-values and eigen-functions are available, where the eigen-functions are represented with Hermite polynomials (See Sec. 3 for definition). This is the approximation we will use in our method.

2.5. Differential privacy

Given privacy parameters $\epsilon \geq 0$ and $\delta \geq 0$, a mechanism $\mathcal{M}$ is (ϵ, δ)-DP if the following equation holds: $\Pr[\mathcal{M}(\mathcal{D}) \in S] \leq e^\epsilon \cdot \Pr[\mathcal{M}(\mathcal{D}') \in S] + \delta$, for all possible sets of the mechanism's outputs S and all neighbouring datasets $\mathcal{D}$, $\mathcal{D}'$ differing by a single entry. In this paper, we use the *Gaussian mechanism* to ensure the output of our algorithm is DP. Consider a function $h : \mathcal{D} \mapsto \mathbb{R}^p$, where we add noise for privacy and the level of noise is calibrated to the *global sensitivity* (Dwork et al., 2006), Δ_h, defined by the maximum difference in terms of L_2-norm $||h(\mathcal{D}) - h(\mathcal{D}')||_2$, for neighbouring $\mathcal{D}$ and $\mathcal{D}'$ (i.e. $\mathcal{D}$ and $\mathcal{D}'$ have one sample difference by replacement). where the output is denoted by $\widetilde{h}(\mathcal{D}) = h(\mathcal{D}) + n$, where $n \sim \mathcal{N}(0, \sigma^2 \Delta_h^2 \mathbf{I}_p)$. The perturbed function $\widetilde{h}(\mathcal{D})$ is (ϵ, δ)-DP, where σ is a function of ϵ and δ and can be numerically computed using, e.g., the auto-dp package by (Wang et al., 2019).

3. Our method: DP-HP

3.1. Approximating the Gaussian kernel using Hermite polynomials (HP)

Using the *Mehler formula*[3] (Mehler, 1866), for $|\rho| < 1$, we can write down the Gaussian kernel[4] as a weighted sum of Hermite polynomials

$$\exp\left(-\frac{\rho}{1-\rho^2}(x-y)^2\right) = \sum_{c=0}^{\infty} \lambda_c f_c(x) f_c(y) \tag{5}$$

where the c-th eigen-value is $\lambda_c = (1-\rho)\rho^c$ and the c-th eigen-function is defined by f_c, where $f_c(x) = \frac{1}{\sqrt{N_c}} H_c(x) \exp\left(-\frac{\rho}{1+\rho}x^2\right)$, and $N_c = 2^c c! \sqrt{\frac{1-\rho}{1+\rho}}$. Here, $H_c(x) = (-1)^c \exp(x^2) \frac{d^c}{dx^c} \exp(-x^2)$ is the c-th order physicist's Hermite polynomial.

As a result of the Mehler formula, we can define a C-th order Hermite polynomial features as a feature map (a vector of length $C + 1$):

$$\hat{\phi}_{HP}^{(C)}(x) = \left[\sqrt{\lambda_0} f_0(x), \cdots, \sqrt{\lambda_C} f_C(x)\right], \tag{6}$$

and approximate the Gaussian kernel via $\exp\left(-\frac{\rho}{1-\rho^2}(x-y)^2\right) \approx \hat{\phi}_{HP}^{(C)}(x)^\top \hat{\phi}_{HP}^{(C)}(y)$.

This feature map provides us with a uniform approximation to the MMD in eq. 1, for every pair of distributions P and Q (see Theorem C.1 and Lemma C.1 in Appendix Sec. C).

We compare the accuracy of this approximation with random features in Fig. 1, where we fix the length scale to the median heuristic value[5] in both cases. Note that the bottom plot shows the average error across 100 independent draws of random Fourier features. We observe that the error decay is significantly faster when using HPs than using RFs. For completeness, we derive the kernel approximation error under HP features and random features for 1-dimensional data in Appendix Sec. B. Additionally, we visualize the effect of length scale on the error further in Appendix Sec. A.

Computing the Hermite polynomial features. Hermite polynomials follow the recursive definition: $H_{c+1}(x) =$

[3] This formula can be also derived from the Mercer's theorem as shown in (Zhu et al., 1997; Rasmussen & Williams, 2005).

[4] The length scale l in terms of ρ is $\frac{1}{2l^2} = \frac{\rho}{1-\rho^2}$.

[5] Median heuristic is a commonly-used heuristic to choose a length scale, which picks a value in the middle range (i.e., median) of $\|x_i - x_j\|$ for $1 \leq i, j \leq n$ for the dataset of n samples.

$2xH_c(x) - 2cH_{c-1}(x)$. At high orders, the polynomials take on large values, leading to numerical instability. So we compute the re-scaled term $\phi_c = \sqrt{\lambda_c} f_c$ iteratively using a similar recursive expression given in Appendix Sec. E.

3.2. Handling multi-dimensional inputs

3.2.1. Tensor (or outer) product kernel

The Mehler formula holds for 1-dimensional input space. For D-dimensional inputs $\mathbf{x}, \mathbf{x}' \in \mathbb{R}^D$, where $\mathbf{x} = [x_1, \cdots, x_D]$ and $\mathbf{x}' = [x'_1, \cdots, x'_D]$, the *generalized Hermite Polynomials* (Proposition C.3 and Remark 1 in Appendix Sec. C) allows us to represent the multivariate Gaussian kernel $k(\mathbf{x}, \mathbf{x}')$ by a tensor (or outer) products of the Gaussian kernel defined on each input dimension, where the coordinate-wise Gaussian kernel is approximated with Hermite polynomials:

$$k(\mathbf{x}, \mathbf{x}') = k_{X_1} \otimes k_{X_2} \cdots \otimes k_{X_D} = \prod_{d=1}^{D} k_{X_d}(x_d, x'_d),$$
$$\approx \prod_{d=1}^{D} \hat{\boldsymbol{\phi}}^{(C)}_{HP}(x_d)^\top \hat{\boldsymbol{\phi}}^{(C)}_{HP}(x_d), \tag{7}$$

where $\hat{\boldsymbol{\phi}}^{(C)}_{HP}(.)$[6] is defined in eq. 6. The corresponding feature map, from $k(\mathbf{x}, \mathbf{x}') \approx \mathbf{h}_p(\mathbf{x})^\top \mathbf{h}_p(\mathbf{x}')$, is written as

$$\mathbf{h}_p(\mathbf{x})$$
$$= \text{vec}\left[\hat{\boldsymbol{\phi}}^{(C)}_{HP}(x_1) \otimes \hat{\boldsymbol{\phi}}^{(C)}_{HP}(x_2) \otimes \cdots \hat{\boldsymbol{\phi}}^{(C)}_{HP}(x_D)\right] \tag{8}$$

where $\otimes$ denotes the tensor (outer) product and vec is an operation that vectorizes a tensor. The size of the feature map is $(C+1)^D$, where D is the input dimension of the data and C is the chosen order of the Hermite polynomials. This is prohibitive for the datasets we often deal with, e.g., for MNIST ($D = 784$) with a relatively small order (say $C = 10$), the size of feature map is 11^{784}, impossible to fit in a typical size of memory.

In order to handle high-dimensional data in a computationally feasible manner, we propose the following approximation. First we subsample input dimensions where the size of the selected input dimensions is denoted by D_{prod}. We then compute the feature map only on those selected input dimensions denoted by $\mathbf{x}^{D_{prod}}$. We repeat these two steps during training. The size of the feature map becomes $(C+1)^{D_{prod}}$, significantly lower than $(C+1)^D$ if $D_{prod} \ll D$. What we lose in return is the injectivity of the Gaussian kernel on the full input distribution, as we compare two distributions in terms of selected input dimensions. We need a quantity that is more computationally tractable and also helps distinguishing two distributions, which we describe next.

[6] One can let each coordinate's Hermite Polynomials $\phi^{(C)}_{HP,d}(x_d)$ take different values of ρ, which determine a different level of fall-offs of the eigen-values and a different range of values of the eigen-functions. Imposing a different cut-off C for each coordinate is also possible.

3.2.2. Sum kernel

Here, we define another kernel on the joint distribution over $(x_1, \cdots, x_D)$. The following kernel is formed by defining a 1-dimensional Gaussian kernel on each of the input dimensions:

$$\tilde{k}(\mathbf{x}, \mathbf{x}') = \tfrac{1}{D}\left[k_{X_1}(x_1, x'_1) + \cdots + k_{X_D}(x_D, x'_D)\right],$$
$$= \tfrac{1}{D}\sum_{d=1}^{D} k_{X_d}(x_d, x'_d),$$
$$\approx \tfrac{1}{D}\sum_{d=1}^{D} \hat{\boldsymbol{\phi}}^{(C)}_{HP}(x_d)^\top \hat{\boldsymbol{\phi}}^{(C)}_{HP}(x_d), \tag{9}$$

where $\hat{\boldsymbol{\phi}}^{(C)}_{HP,d}(.)$ is given in eq. 6. The corresponding feature map, from $\tilde{k}(\mathbf{x}, \mathbf{x}') \approx \mathbf{h}_s(\mathbf{x})^\top \mathbf{h}_s(\mathbf{x}')$, is represented by

$$\mathbf{h}_s(\mathbf{x}) = \begin{bmatrix} \hat{\boldsymbol{\phi}}^{(C)}_{HP,1}(x_1)/\sqrt{D} \\ \hat{\boldsymbol{\phi}}^{(C)}_{HP,2}(x_2)/\sqrt{D} \\ \vdots \\ \hat{\boldsymbol{\phi}}^{(C)}_{HP,D}(x_D)/\sqrt{D} \end{bmatrix} \in \mathbb{R}^{((C+1)\cdot D)\times 1}, \tag{10}$$

where the features map is the size of $(C+1)D$. For the MNIST digit data ($D = 784$), with a relatively small order, say $C = 10$, the size of the feature map is $11 \times 784 = 8624$ dimensional, which is manageable compared to the size (11^{784}) of the feature map under the generalized Hermite polynomials.

Note that the sum kernel does not approximate the Gaussian kernel defined on the joint distribution over all the input dimensions. Rather, the assigned Gaussian kernel *on each dimension is characteristic*. The Lemma D.1 in Appendix Sec. D shows that by minimizing the approximate MMD between the real and synthetic data distributions based on feature maps given in eq. 10, we assure that the marginal probability distributions of the synthetic data converges to those of the real data.

3.2.3. Combined Kernel

Finally we arrive at a new kernel, which comes from a sum of the two fore-mentioned kernels:

$$k_c(\mathbf{x}, \mathbf{x}') = k(\mathbf{x}, \mathbf{x}') + \tilde{k}(\mathbf{x}, \mathbf{x}'), \tag{11}$$

where $k(\mathbf{x}, \mathbf{x}') \approx \mathbf{h}_p(\mathbf{x}^{D_{prod}})^\top \mathbf{h}_p(\mathbf{x}'^{D_{prod}})$ and $\tilde{k}(\mathbf{x}, \mathbf{x}') \approx \mathbf{h}_s(\mathbf{x})^\top \mathbf{h}_s(\mathbf{x}')$, and consequently the corresponding feature map is given by

$$\mathbf{h}_c(\mathbf{x}) = \begin{bmatrix} \mathbf{h}_p(\mathbf{x}^{D_{prod}}) \\ \mathbf{h}_s(\mathbf{x}) \end{bmatrix} \tag{12}$$

where the size of the feature map is $\mathbb{R}^{\left((C+1)^{D_{prod}}+(C+1)\cdot D\right)\times 1}$.

Why this kernel? When D_{prod} goes to D, the product kernel itself in eq. 11 becomes characteristic, which allows us to reliably compare two distributions. However, for computational tractability, we are restricted to choose a relatively small D_{prod} to subsample the input dimensions, which forces us to lose information on the distribution over the un-selected input dimensions. The use of sum kernel is to provide extra information on the un-selected input dimensions at a particular training step. Under our kernel in eq. 11, every input dimension's marginal distributions are compared between two distributions in all the training steps due to the sum kernel, while some of the input dimensions are chosen to be considered for more detailed comparison (e.g., high-order correlations between selected input dimensions) due to the outer product kernel.

3.3. Approximate MMD for classification

For classification tasks, we define a mean embedding for the joint distribution over the input and output pairs $(\mathbf{x}, \mathbf{y})$, with the particular feature map given by $\mathbf{g}$

$$\widehat{\boldsymbol{\mu}}_{P_{\mathbf{x},\mathbf{y}}}(\mathcal{D}) = \frac{1}{m}\sum_{i=1}^{m}\mathbf{g}(\mathbf{x}_i, \mathbf{y}_i). \tag{13}$$

Here, we define the feature map as an outer product between the input features represented by eq. 12 and the output labels represented by one-hot-encoding $\mathbf{f}(\mathbf{y}_i)$:

$$\mathbf{g}(\mathbf{x}_i, \mathbf{y}_i) = \mathbf{h}_c(\mathbf{x}_i)\mathbf{f}(\mathbf{y}_i)^T. \tag{14}$$

Given eq. 14, we further decompose eq. 13 into two, where the first term corresponds to the outer product kernel denoted by $\widehat{\boldsymbol{\mu}}_P^p$ and the second term corresponds to the sum kernel denoted by $\widehat{\boldsymbol{\mu}}_P^s$:

$$\widehat{\boldsymbol{\mu}}_{P_{\mathbf{x},\mathbf{y}}} = \begin{bmatrix}\widehat{\boldsymbol{\mu}}_P^p \\ \widehat{\boldsymbol{\mu}}_P^s\end{bmatrix} = \begin{bmatrix}\frac{1}{m}\sum_{i=1}^{m}\mathbf{h}_p(\mathbf{x}_i^{D_{prod}})\mathbf{f}(\mathbf{y}_i)^T \\ \frac{1}{m}\sum_{i=1}^{m}\mathbf{h}_s(\mathbf{x}_i)\mathbf{f}(\mathbf{y}_i)^T\end{bmatrix}. \tag{15}$$

Similarly, we define an approximate mean embedding of the synthetic data distribution by $\widehat{\boldsymbol{\mu}}_{Q_{\mathbf{x}',\mathbf{y}'}}(\mathcal{D}'_{\boldsymbol{\theta}}) = \frac{1}{n}\sum_{i=1}^{n}\mathbf{g}(\mathbf{x}'_i(\boldsymbol{\theta}), \mathbf{y}'_i(\boldsymbol{\theta}))$, where $\boldsymbol{\theta}$ denotes the parameters of a synthetic data generator. Then, the approximate MMD is given by: $\widehat{\mathrm{MMD}}^2_{HP}(P, Q) = ||\widehat{\boldsymbol{\mu}}_{P_{\mathbf{x},\mathbf{y}}}(\mathcal{D}) - \widehat{\boldsymbol{\mu}}_{Q_{\mathbf{x}',\mathbf{y}'}}(\mathcal{D}'_{\boldsymbol{\theta}})||_2^2 = ||\widehat{\boldsymbol{\mu}}_P^p - \widehat{\boldsymbol{\mu}}_{Q_{\boldsymbol{\theta}}}^p||_2^2 + ||\widehat{\boldsymbol{\mu}}_P^s - \widehat{\boldsymbol{\mu}}_{Q_{\boldsymbol{\theta}}}^s||_2^2$. In practice, we minimize the augmented approximate MMD:

$$\min_{\boldsymbol{\theta}} \gamma||\widehat{\boldsymbol{\mu}}_P^p - \widehat{\boldsymbol{\mu}}_{Q_{\boldsymbol{\theta}}}^p||_2^2 + ||\widehat{\boldsymbol{\mu}}_P^s - \widehat{\boldsymbol{\mu}}_{Q_{\boldsymbol{\theta}}}^s||_2^2. \tag{16}$$

where γ is a positive constant (a hyperparameter) that helps us to deal with the scale difference in the two terms (depending on the selected HP orders and subsampled input dimensions) and also allows us to give a different importance on one of the two terms. We provide the details on how γ plays a role and whether the algorithm is sensitive to γ in Sec. 5. Minimizing eq. 16 yields a synthetic data distribution over the input and output, which minimizes the discrepancy in terms of the particular feature map eq. 15 between synthetic and real data distributions.

3.4. Differentially private data samples

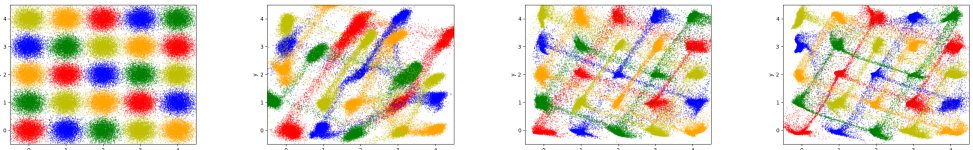
For obtaining privacy-preserving synthetic data, all we need to do is privatizing $\widehat{\boldsymbol{\mu}}_P^p$ and $\widehat{\boldsymbol{\mu}}_P^s$ given in eq. 15, then training a generator. We use the Gaussian mechanism to privatize both terms. See Appendix Sec. F for sensitivity analysis. Unlike $\widehat{\boldsymbol{\mu}}_P^s$ that can be privatized only and for all, we need to privatize $\widehat{\boldsymbol{\mu}}_P^p$ every time we redraw the subsampled input dimensions. We split a target ϵ into two such that $\epsilon = \epsilon_1 + \epsilon_2$ (also the same for δ), where ϵ_1 is used for privatizing $\widehat{\boldsymbol{\mu}}_P^s$ and ϵ_2 is used for privatizing $\widehat{\boldsymbol{\mu}}_P^p$. We further compose the privacy loss incurred in privatizing $\widehat{\boldsymbol{\mu}}_P^p$ during training by the analytic moments accountant (Wang et al., 2019), which returns the privacy parameter σ as a function of (ϵ_2, δ_2). In the experiments, we subsample the input dimensions for the outer product kernel in every epoch as opposed to in every training step for an economical use of ϵ_2.

4. Related Work

Approaches to differentially private data release can be broadly sorted into three categories. One line of prior work with background in learning theory aims to provide theoretical guarantees on the utility of released data (Snoke & Slavković, 2018; Mohammed et al., 2011; Xiao et al., 2010; Hardt et al., 2012; Zhu et al., 2017). This usually requires strong constraints on the type of data and the intended use of the released data.

A second line of work focuses on the sub-problem of discrete data with limited domain size, which is relevant to tabular datasets (Zhang et al., 2017; Qardaji et al., 2014; Chen et al., 2015; Zhang et al., 2021). Such approaches typically approximate the structure in the data by identifying small sub-sets of features with high correlation and releasing these lower order marginals in a private way. Some of these methods have also been successful in the recent NIST 2018 Differential Privacy Synthetic Data Challenge (nis), while these methods often require discretization of the data and do not scale to higher dimensionality in arbitrary domains.

The third line of work aims for broad applicability without constraints on the type of data or the kind of downstream tasks to be used. Recent approaches attempt to leverage the modeling power of deep generative models in the private setting. While work on VAEs exists (Acs et al., 2018), GANs are the most popular model (Xie et al., 2018; Torkzadehma-



Data Samples
NLL $\approx 3.1 \times 10^5$

DP-CGAN ($\epsilon = 1$)
NLL $\approx 4.7 \times 10^5$

DP-MERF ($\epsilon = 1$)
NLL $\approx 4.1 \times 10^5$

DP-HP ($\epsilon = 1$)
NLL $\approx 3.7 \times 10^5$

Figure 2. Simulated example from a Gaussian mixture. **Left**: Data samples drawn from a Gaussian Mixture distribution with 5 classes (each color represents a class). NLL denotes the negative log likelihood of the samples given the true data distribution. **Middle-Left**: Synthetic data generated by DP-CGANs at $\epsilon = 1$, where some modes are dropped, which is reflected in poor NLL. **Middle-Right**: Synthetic data samples generated by DP-MERF at $\epsilon = 1$. **Right**: Synthetic data samples generated by DP-HP at $\epsilon = 1$. Our method captures all modes accurately at $\epsilon = 1$, and achieves better NLL thanks to a smaller size of feature map than that of DP-MERF (see text).

hani et al., 2019; Frigerio et al., 2019; Yoon et al., 2019; Chen et al., 2020), where most of these utilize a version of DP-SGD (Abadi et al., 2016) to accomplish this training, while PATE-GAN is based on the private aggregation of teacher ensembles (PATE) (Papernot et al., 2017).

The closest prior work to the proposed method is DP-MERF (Harder et al., 2021), where kernel mean embeddings are approximated with random Fourier features (Rahimi & Recht, 2008) instead of Hermite polynomials. Random feature approximations of MMD have also been used with DP (Balog et al., 2018; Sarpatwar et al., 2019). A recent work utilizes the Sinkhorn divergence for private data generation (Cao et al., 2021), which more or less matches the results of DP-MERF when the regularizer is large and the cost function is the L2 distance. To our knowledge, ours is the first work using Hermite polynomials to approximate MMD in the context of differentially private data generation.

5. Experiments

Here, we show the performance of our method tested on several real world datasets. Evaluating the quality of generated data itself is challenging. Popular metrics such as inception score and Fréchet inception distance are appropriate to use for evaluating color images. For the generated samples for tabular data and black and white images, we use the following three metrics: (a) Negative log-likelihood of generated samples given a ground truth model in Sec. 5.1; (b) α-way marginals of generated samples in Sec. 5.2 to judge whether the generated samples contain a similar correlation structure to the real data; (c) Test accuracy on the real data given classifiers trained with generated samples in Sec. 5.3 to judge the generalization performance from synthetic to real data.

As comparison methods, we tested PrivBayes (Zhang et al., 2017), DP-CGAN (Torkzadehmahani et al., 2019), DP-GAN (Xie et al., 2018) and DP-MERF (Harder et al., 2021). For image datasets we also trained GS-WGAN (Chen et al., 2020). Our experiments were implemented in PyTorch (Paszke et al., 2019) and run using Nvidia Kepler20 and Kepler80 GPUs. Our code is available at `https://github.com/ParkLabML/DP-HP`.

5.1. 2D Gaussian mixtures

We begin our experiments on Gaussian mixtures, as shown in Fig. 2 (left). We generate 4000 samples from each Gaussian, reserving 10% for the test set, which yields 90000 training samples from the following distribution: $p(\mathbf{x}, \mathbf{y}) = \prod_i^N \sum_{j \in C_{\mathbf{y}_i}} \frac{1}{C} \mathcal{N}(\mathbf{x}_i | \boldsymbol{\mu}_j, \sigma \mathbf{I}_2)$ where $N = 90000$, and $\sigma = 0.2$. $C = 25$ is the number of clusters and C_y denotes the set of indices for means $\boldsymbol{\mu}$ assigned to class y. Five Gaussians are assigned to each class, which leads to a uniform distribution over $\mathbf{y}$ and 18000 samples per class. We use the negative log likelihood (NLL) of the samples under the true distribution as a score[7] to measure the quality of the generated samples: $\text{NLL}(\mathbf{x}, \mathbf{y}) = -\log p(\mathbf{x}, \mathbf{y})$. The lower NLL the better.

We compare our method to DP-CGAN and DP-MERF at $(\epsilon, \delta) = (1, 10^{-5})$ in Fig. 2. Many of the generated samples by DP-CGAN fall out of the distribution and some modes are dropped (like the green one in the top right corner). DP-MERF preserves all modes. DP-HP performs better than DP-MERF by placing fewer samples in low density regions as indicated by the low NLL. This is due to the drastic difference in the size of the feature map. DP-MERF used $30,000$ random features (i.e., $30,000$-dimensional feature map). DP-HP used the 25-th order Hermite polynomials on both sum and product kernel approximation (i.e., $25^2 + 25 = 650$-dimensional feature map). in this example, as the input is 2-dimensional, it was not necessary to subsample the input dimensions to approximate the outer product kernel.

[7] Note that this is different from the other common measure of computing the negative log-likelihood of the true data given the learned model parameters.

Table 1. α-way marginals evaluated on generated samples with discretized Adult and Census datasets.

Adult	PrivBayes		DP-MERF		**DP-HP**		*Census*	PrivBayes		DP-MERF		DP-HP	
	ϵ=0.3	ϵ=0.1	ϵ=0.3	ϵ=0.1	ϵ=0.3	ϵ=0.1		ϵ=0.3	ϵ=0.1	ϵ=0.3	ϵ=0.1	ϵ=0.3	ϵ=0.1
α=3	0.446	0.577	0.405	0.480	**0.332**	**0.377**	α=2	0.180	0.291	0.190	0.222	**0.141**	**0.155**
α=4	0.547	0.673	0.508	0.590	**0.418**	**0.467**	α=3	0.323	0.429	0.302	0.337	**0.211**	**0.232**

5.2. α-way marginals with discretized tabular data

We compare our method to PrivBayes (Zhang et al., 2017) and DP-MERF. For PrivBayes, we used the published code from (McKenna et al., 2019), which builds on the original code with (Zhang et al., 2018) as a wrapper. We test the model on the discretized Adult and Census datasets. Although these datasets are typically used for classification, we use their inputs only for the task of learning the input distribution. Following (Zhang et al., 2017), we measure α-way marginals of generated samples at varying levels of ϵ-DP with $\delta = 10^{-5}$. We measure the accuracy of each marginal of the generated dataset by the total variation distance between itself and the real data marginal (i.e., half of the L1 distance between the two marginals, when both of them are treated as probability distributions). We use the average accuracy over all marginals as the final error metric for α-way marginals. In Table 1, our method outperforms other two at the stringent privacy regime. See Appendix Sec. G.1 for hyperparameter values we used, and Appendix Sec. G.2 for the impact of γ on the quality of the generated samples. We also show how the selection of D_{prod} affects the accuracy in Appendix Sec. G.5.

5.3. Generalization from synthetic to real data

Following (Chen et al., 2020; Torkzadehmahani et al., 2019; Yoon et al., 2019; Chen et al., 2020; Harder et al., 2021; Cao et al., 2021), we evaluate the quality of the (private and non-private) generated samples from these models using the common approach of measuring performance on downstream tasks. We train 12 different commonly used classifier models using generated samples and then evaluate the classifiers on a test set containing *real* data samples. Each setup is averaged over 5 random seeds. The test accuracy indicates how well the models generalize from the synthetic to the real data distribution and thus, the utility of using private data samples instead of the real ones. Details on the 12 models can be found in Table 10.

Tabular data. First, we explore the performance of DP-HP algorithm on eight different imbalanced tabular datasets with both numerical and categorical input features. The numerical features on those tabular datasets can be either discrete (e.g. age in years) or continuous (e.g. height) and the categorical ones may be binary (e.g. drug vs placebo group) or multi-class (e.g. nationality). The datasets are described in detail in Appendix Sec. G. As an evaluation metric, we use ROC (area under the receiver characteristics curve) and PRC (area under the precision recall curve) for datasets with binary labels, and F1 score for dataset with multi-class labels. Table 2 shows the average over the 12 classifiers trained on the generated samples (also averaged over 5 independent seeds), where overall DP-HP outperforms the other methods in both the private and non-private settings, followed by DP-MERF.[8] See Appendix Sec. G.3 for hyperparameter values we used. We also show the non-private MERF and HP results in Table 7 in Appendix.

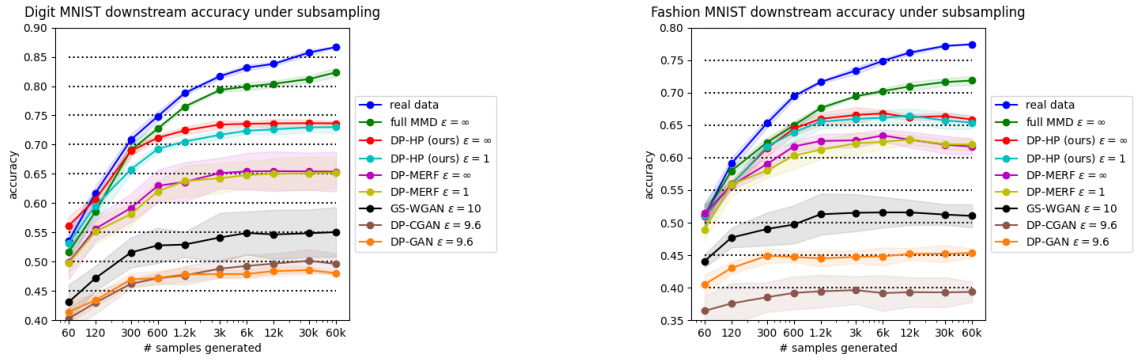
Image data. We follow previous work in testing our method on image datasets MNIST (LeCun et al., 2010) (license: CC BY-SA 3.0) and FashionMNIST (Xiao et al., 2017) (license: MIT). Both datasets contain 60000 images from 10 different balanced classes. We test both fully connected and convolutional generator networks and find that the former works better for MNIST, while the latter model achieves better scores on FashionMNIST. For the experimental setup of DP-HP on the image datasets see Table 9 in Appendix Sec. H.2. A qualitative sample of the generated images for DP-HP and comparison methods is shown in Fig. 4. While qualitatively GS-WGAN produces the cleanest samples, DP-HP outperforms GS-WGAN on downstream tasks. This can be explained by a lack of sample diversity in GS-WGAN shown in Fig. 3.

In Fig. 3, we compare the test accuracy on real image data based on private synthetic samples from DP-GAN, DP-CGAN, GS-WGAN, DP-MERF and DP-HP generators. As additional baselines we include performance of real data and of *full MMD*, a non-private generator, which is trained with the MMD estimator in eq. 2 in a mini-batch fashion. DP-HP gives the best accuracy over the other considered methods followed by DP-MERF but with a considerable difference especially on the MNIST dataset. For GAN-based methods, we use the same weak privacy constraints given in the original papers, because they do not produce meaningful samples at $\epsilon = 1$. Nonetheless, the accuracy these models achieve remains relatively low. Results for individual models for

[8] For the Cervical dataset, the non-privately generated samples by DP-MERF and DP-HP give better results than the baseline trained with real data. This may be due to the fact that the dataset is relatively small which can lead to overfitting. The generating samples by DP-MERF and DP-HP could bring a regularizing effect, which improves the performance as a result.

Table 2. Performance comparison on Tabular datasets. The average over five independent runs.

	Real		DP-CGAN $(1, 10^{-5})$-DP		DP-GAN $(1, 10^{-5})$-DP		DP-MERF $(1, 10^{-5})$-DP		**DP-HP** $(1, 10^{-5})$-DP	
adult	0.786	0.683	0.509	0.444	0.511	0.445	0.642	0.524	**0,688**	**0,632**
census	0.776	0.433	0.655	0.216	0.529	0.166	0.685	0.236	**0,699**	**0,328**
cervical	0.959	0.858	0.519	0.200	0.485	0.183	0.531	0.176	**0,616**	**0,312**
credit	0.924	0.864	0.664	0.356	0.435	0.150	0.751	0.622	**0,786**	**0,744**
epileptic	0.808	0.636	0.578	0.241	0.505	0.196	0.605	0.316	**0,609**	**0,554**
isolet	0.895	0.741	0.511	0.198	0.540	0.205	0.557	0.228	**0,572**	**0,498**
	F1		F1		F1		F1		F1	
covtype	0.820		0.285		0.492		0.467		**0.537**	
intrusion	0.971		0.302		0.251		**0.892**		0.890	



(a) MNIST (b) FashionMNIST

Figure 3. We compare the real data test accuracy as a function of training set size for models trained on synthetic data from DP-HP and comparison models. Confidence intervals show 1 standard deviation.

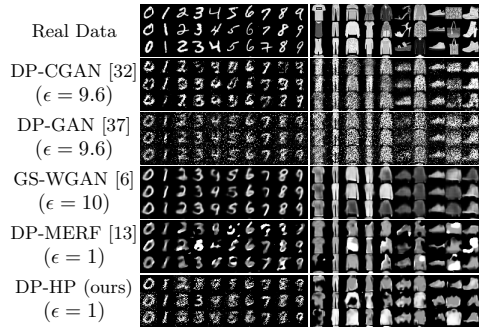


Figure 4. Generated MNIST and FashionMNIST samples from DP-HP and comparison models

both image datasets are given in Appendix Sec. H.

Finally, we show the downstream accuracy for smaller generated datasets down to 60 samples (or 0.1% of original dataset) in Fig. 3. The points, at which additional generated data does not lead to improved performance, gives us a sense of the redundancy present in the generated data. We observe that all generative models except *full MMD* see little increase in performance as we increase the number of synthetic data samples to train the classifiers. This indicates that the *effective dataset size* these methods produce lies only at about 5% (3k) to 10% (6k) of the original data. For DP-GAN and DP-CGAN this effect is even more pronounced, showing little to no gain in accuracy after the first 300 to 600 samples respectively on FashionMNIST.

6. Summary and Discussion

We propose a DP data generation framework that improves the privacy-accuracy trade-off using the Hermite polynomials features thanks to the orderedness of the polynomial features. We chose the combination of outer product and sum kernels computational tractability in handling high-dimensional data. The quality of generated data by our method is significantly higher than that by other state-of-the-art methods, in terms of three different evaluation metrics. In all experiments, we observed that assigning ϵ more to ϵ_1 than ϵ_2 and using the sum kernel's mean embedding as a main objective together with the outer product kernel's mean embedding as a constraint (weighted by γ) help improving the performance of DP-HP.

As the size of mean embedding grows exponentially with the input dimension under the outer product kernel, we chose to subsample the input dimensions. However, even with the

subsampling, we needed to be careful not to explode the size of the kernel's mean embedding, which limits the subsampling dimension to be less than 5, in practice. This gives us a question whether there are better ways to approximate the outer product kernel than random sampling across all input dimensions. We leave this for future work.

Acknowledgements

M.A. Charusaie and F. Harder thank the International Max Planck Research School for Intelligent Systems (IMPRS-IS) for its support. M. Vinaroz thanks the support by the Gibs Schule Foundation and the Institutional Strategy of the University of Tübingen (ZUK63). K. Adamczewski is grateful for the support of the Max Planck ETH Center for Learning Systems. M. Park thanks the CIFAR AI Chair fund (at AMII) for its support. M. Park also thanks Wittawat Jitkrittum and Danica J. Sutherland for their valuable time discussing the project. We thank our anonymous reviewers for their constructive feedback.

References

Nist 2018 differential privacy synthetic data challenge. https://www.nist.gov/ctl/pscr/open-innovation-prize-challenges/past-prize-challenges/2018-differential-privacy-synthetic.

Abadi, M., Chu, A., Goodfellow, I., McMahan, H. B., Mironov, I., Talwar, K., and Zhang, L. Deep learning with differential privacy. In *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security*, CCS '16, pp. 308–318, New York, NY, USA, 2016. Association for Computing Machinery. ISBN 9781450341394. doi: 10.1145/2976749.2978318.

Acs, G., Melis, L., Castelluccia, C., and De Cristofaro, E. Differentially private mixture of generative neural networks. *IEEE Transactions on Knowledge and Data Engineering*, 31(6):1109–1121, 2018.

Aronszajn, N. Theory of reproducing kernels. *Trans Am Math Soc*, 68(3):337–404, 1950. ISSN 00029947. URL www.jstor.org/stable/1990404.

Avron, H., Sindhwani, V., Yang, J., and Mahoney, M. W. Quasi-monte carlo feature maps for shift-invariant kernels. *Journal of Machine Learning Research*, 17(120):1–38, 2016. URL http://jmlr.org/papers/v17/14-538.html.

Balog, M., Tolstikhin, I., and Schölkopf, B. Differentially private database release via kernel mean embeddings. In *Proceedings of the 35th International Conference on Machine Learning (ICML)*, volume 80 of *Proceedings of Machine Learning Research*, pp. 423–431. PMLR, July 2018.

Cao, T., Bie, A., Vahdat, A., Fidler, S., and Kreis, K. Don't generate me: Training differentially private generative models with sinkhorn divergence. In *NeurIPS*, 2021.

Chen, D., Orekondy, T., and Fritz, M. Gs-wgan: A gradient-sanitized approach for learning differentially private generators. In *Advances in Neural Information Processing Systems 33*, 2020.

Chen, R., Xiao, Q., Zhang, Y., and Xu, J. Differentially private high-dimensional data publication via sampling-based inference. In *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 129–138, 2015.

Dua, D. and Graff, C. UCI machine learning repository, 2017. URL http://archive.ics.uci.edu/ml.

Dwork, C., Kenthapadi, K., McSherry, F., Mironov, I., and Naor, M. Our data, ourselves: Privacy via distributed noise generation. In *Advances in Cryptology - EUROCRYPT 2006, 25th Annual International Conference on the Theory and Applications of Cryptographic Techniques*, volume 4004 of *Lecture Notes in Computer Science*, pp. 486–503. Springer, 2006. doi: 10.1007/11761679_29. URL https://iacr.org/archive/eurocrypt2006/40040493/40040493.pdf.

Frigerio, L., de Oliveira, A. S., Gomez, L., and Duverger, P. Differentially private generative adversarial networks for time series, continuous, and discrete open data. In *ICT Systems Security and Privacy Protection - 34th IFIP TC 11 International Conference, SEC 2019, Lisbon, Portugal, June 25-27, 2019, Proceedings*, pp. 151–164, 2019. doi: 10.1007/978-3-030-22312-0_11.

Gretton, A., Borgwardt, K. M., Rasch, M. J., Schölkopf, B., and Smola, A. A kernel two-sample test. *Journal of Machine Learning Research*, 13(Mar):723–773, 2012.

Harder, F., Adamczewski, K., and Park, M. DP-MERF: Differentially private mean embeddings with randomfeatures for practical privacy-preserving data generation. In Banerjee, A. and Fukumizu, K. (eds.), *Proceedings of The 24th International Conference on Artificial Intelligence and Statistics*, volume 130 of *Proceedings of Machine Learning Research*, pp. 1819–1827. PMLR, 13–15 Apr 2021. URL http://proceedings.mlr.press/v130/harder21a.html.

Hardt, M., Ligett, K., and Mcsherry, F. A simple and practical algorithm for differentially private data release. In Pereira, F., Burges, C. J. C., Bottou, L., and Weinberger,

K. Q. (eds.), *Advances in Neural Information Processing Systems 25*, pp. 2339–2347. Curran Associates, Inc., 2012.

LeCun, Y., Cortes, C., and Burges, C. Mnist handwritten digit database. *ATT Labs [Online]. Available: http://yann.lecun.com/exdb/mnist*, 2, 2010.

McKenna, R., Sheldon, D., and Miklau, G. Graphical-model based estimation and inference for differential privacy. *arXiv preprint arXiv:1901.09136*, 2019.

Mehler, F. G. Ueber die entwicklung einer function von beliebig vielen variablen nach laplaceschen functionen höherer ordnung. 1866.

Mohammed, N., Chen, R., Fung, B. C., and Yu, P. S. Differentially private data release for data mining. In *Proceedings of the 17th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '11, pp. 493–501, New York, NY, USA, 2011. ACM. ISBN 978-1-4503-0813-7. doi: 10.1145/2020408.2020487.

Papernot, N., Abadi, M., Erlingsson, U., Goodfellow, I., and Talwar, K. Semi-supervised Knowledge Transfer for Deep Learning from Private Training Data. In *Proceedings of the International Conference on Learning Representations (ICLR)*, April 2017. URL http://arxiv.org/abs/1610.05755.

Paszke, A., Gross, S., Massa, F., Lerer, A., Bradbury, J., Chanan, G., Killeen, T., Lin, Z., Gimelshein, N., Antiga, L., Desmaison, A., Kopf, A., Yang, E., DeVito, Z., Raison, M., Tejani, A., Chilamkurthy, S., Steiner, B., Fang, L., Bai, J., and Chintala, S. Pytorch: An imperative style, high-performance deep learning library. In Wallach, H., Larochelle, H., Beygelzimer, A., d'Alché-Buc, F., Fox, E., and Garnett, R. (eds.), *Advances in Neural Information Processing Systems 32*, pp. 8024–8035. Curran Associates, Inc., 2019. URL http://papers.neurips.cc/paper/9015-pytorch-an-imperative-style-high-performance-deep-learning-library.pdf.

Qardaji, W., Yang, W., and Li, N. Priview: practical differentially private release of marginal contingency tables. In *Proceedings of the 2014 ACM SIGMOD international conference on Management of data*, pp. 1435–1446, 2014.

Rahimi, A. and Recht, B. Random features for large-scale kernel machines. In *Advances in neural information processing systems*, pp. 1177–1184, 2008.

Rasmussen, C. E. and Williams, C. K. I. *Gaussian Processes for Machine Learning (Adaptive Computation and Machine Learning)*. The MIT Press, 2005. ISBN 026218253X.

Rudin, W. *Fourier Analysis on Groups: Interscience Tracts in Pure and Applied Mathematics, No. 12*. Literary Licensing, LLC, 2013.

Sarpatwar, K., Shanmugam, K., Ganapavarapu, V. S., Jagmohan, A., and Vaculin, R. Differentially private distributed data summarization under covariate shift. In *Advances in Neural Information Processing Systems*, pp. 14432–14442, 2019.

Slepian, D. On the symmetrized kronecker power of a matrix and extensions of mehler's formula for hermite polynomials. *SIAM Journal on Mathematical Analysis*, 3 (4):606–616, 1972.

Smola, A., Gretton, A., Song, L., and Schölkopf, B. A Hilbert space embedding for distributions. In *ALT*, pp. 13–31, 2007.

Smola, A. J. and Schölkopf, B. *Learning with kernels*, volume 4. Citeseer, 1998.

Snoke, J. and Slavković, A. pmse mechanism: differentially private synthetic data with maximal distributional similarity. In *International Conference on Privacy in Statistical Databases*, pp. 138–159. Springer, 2018.

Sriperumbudur, B. K., Fukumizu, K., and Lanckriet, G. R. Universality, characteristic kernels and rkhs embedding of measures. *Journal of Machine Learning Research*, 12 (7), 2011.

Torkzadehmahani, R., Kairouz, P., and Paten, B. Dp-cgan: Differentially private synthetic data and label generation. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR) Workshops*, June 2019.

Unser, M. and Tafti, P. D. *An introduction to sparse stochastic processes*. Cambridge University Press, 2014.

Wang, Y.-X., Balle, B., and Kasiviswanathan, S. P. Subsampled rényi differential privacy and analytical moments accountant. PMLR, 2019.

Xiao, H., Rasul, K., and Vollgraf, R. Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms. *arXiv preprint arXiv:1708.07747*, 2017.

Xiao, Y., Xiong, L., and Yuan, C. Differentially private data release through multidimensional partitioning. In Jonker, W. and Petković, M. (eds.), *Secure Data Management*, pp. 150–168, Berlin, Heidelberg, 2010. Springer Berlin Heidelberg. ISBN 978-3-642-15546-8.

Xie, L., Lin, K., Wang, S., Wang, F., and Zhou, J. Differentially private generative adversarial network. *CoRR*, abs/1802.06739, 2018.

Yoon, J., Jordon, J., and van der Schaar, M. PATE-GAN: Generating synthetic data with differential privacy guarantees. In *International Conference on Learning Representations*, 2019.

Zhang, D., McKenna, R., Kotsogiannis, I., Hay, M., Machanavajjhala, A., and Miklau, G. Ektelo: A framework for defining differentially-private computations. SIGMOD, 2018.

Zhang, J., Cormode, G., Procopiuc, C. M., Srivastava, D., and Xiao, X. Privbayes: Private data release via bayesian networks. *ACM Transactions on Database Systems (TODS)*, 42(4):1–41, 2017.

Zhang, Z., Wang, T., Li, N., Honorio, J., Backes, M., He, S., Chen, J., and Zhang, Y. Privsyn: Differentially private data synthesis. In *30th {USENIX} Security Symposium ({USENIX} Security 21)*, 2021.

Zhu, H., Williams, C. K., Rohwer, R., and Morciniec, M. Gaussian regression and optimal finite dimensional linear models. 1997.

Zhu, T., Li, G., Zhou, W., and Yu, P. S. Differentially private data publishing and analysis: A survey. *IEEE Transactions on Knowledge and Data Engineering*, 29 (8):1619–1638, August 2017. ISSN 1041-4347. doi: 10.1109/TKDE.2017.2697856.

Appendix

A. Effect of length scale on the kernel approximation

Fig. 5 shows the effect of the kernel length scale on the kernel approximation for both HPs and RFs.

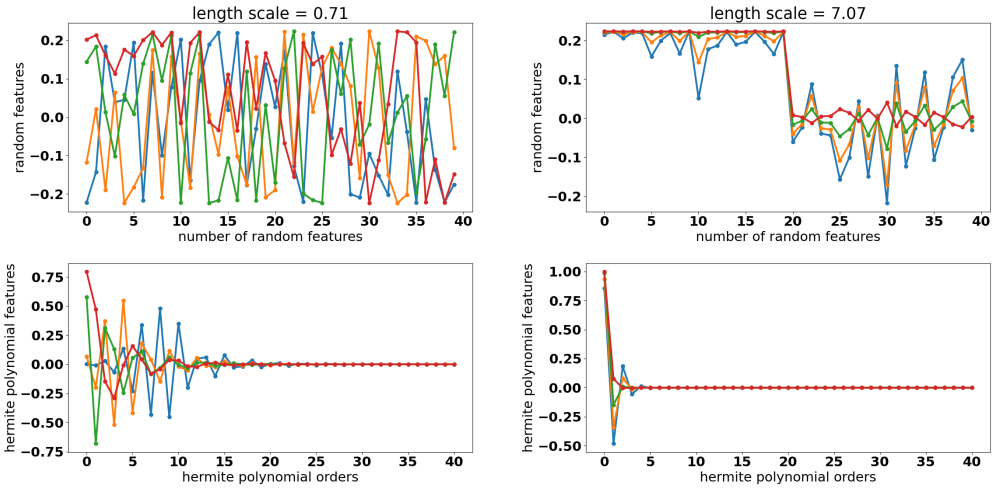


Figure 5. Comparison between HP and random features at a different length scale value. Different color indicates a different datapoint, where four datapoints are drawn from $\mathcal{N}(0, 1)$. **Left**: With length scale $l = 0.71$ (relatively small compared to 1), random features (top) at the four datapoints exhibit large variability while the Hermite polynomial features (bottom) at those datapoints decay at around order ≤ 20. **Right**: With $l = 7.07$ (large compared to 1), random features (top) exhibit less variability, while it is not clear how many features are necessary to consider. On the other hand, the Hermite polynomial features (bottom) decay fast at around order ≤ 5 and we can make a cut-off at that order without losing much information.

B. Approximation error under HP and Random Fourier features

In the following proposition, we provide that provably our method converges with $O(\rho^{2C})$ where $\rho < 1$ is the constant in the Mehler's formula, while DP-MERF has the convergence $\Omega(1/C)$, where C is the number of features in each case.

Proposition B.1. *Let X and Y be standard normal random variables. There exists a C-dimensional Hermite feature map $\phi^{(C)}_{HP}(\cdot)$ with the expected predictive error bounded as*

$$\mathbb{E}_{X,Y}\left[\left|k(X,Y) - \langle\hat{\phi}^{(C)}_{HP}(X), \hat{\phi}^{(C)}_{HP}(Y)\rangle\right|\right] \leq \frac{1}{3\sqrt{2}}(\frac{1}{3})^C. \tag{17}$$

However, the expected predictive error of the random feature map $\hat{\phi}_{RF,\omega}(\cdot)$ with C number of features (i.e., ω is a vector of length C) and the same approximating kernel is equal to

$$\mathbb{E}_{\boldsymbol{\omega},X,Y}\left[\left|k(X,Y) - \langle\hat{\phi}_{RF,\omega}(X), \hat{\phi}_{RF,\omega}(Y)\rangle\right|\right] \geq \frac{1}{8C}. \tag{18}$$

Proof. We start by proving eq. 17. In this case, we write the squared error term as following:

$$A_{x,y} = \left|k(x,y) - \langle\hat{\phi}^{(C)}_{HP}(x), \hat{\phi}^{(C)}_{HP}(y)\rangle\right|^2 \overset{(a)}{=} \Big|\sum_{C+1}^{\infty} \frac{\lambda_l}{\sqrt{N_l}} H_l(x) e^{-\frac{\rho}{1+\rho}x^2} \frac{1}{\sqrt{N_l}} H_l(y) e^{-\frac{\rho}{1+\rho}y^2}\Big|^2 \tag{19}$$

$$= \sum_{l,l'=C+1}^{\infty} \frac{\lambda_l \lambda_{l'}}{N_l N_{l'}} H_l(x) H_{l'}(x) H_l(y) H_{l'}(y) e^{-\frac{2\rho}{1+\rho}x^2 - \frac{2\rho}{1+\rho}y^2}, \tag{20}$$

where (a) is followed by the definition of $\hat{\phi}_{HP}^{(C)}$ in eq. 6 and its approximation property (i.e., Mehler's formula eq. 5). Now, by setting $\rho = \frac{1}{3}$, we have

$$A_{x,y} = \sum_{l,l'=C+1}^{\infty} \frac{\lambda_l \lambda_{l'}}{N_l N_{l'}} H_l(x) H_{l'}(x) H_l(y) H_{l'}(y) e^{-\frac{1}{2}x^2 - \frac{1}{2}y^2}. \tag{21}$$

Next, we average out $A_{x,y}$ for xs and ys that are drawn from a standard normal distribution as

$$\mathbb{E}_{X,Y\sim N(0,1)}\big[A_{X,Y}\big] = \int_{x,y=-\infty}^{\infty} \sum_{l,l'=C+1}^{\infty} \frac{\lambda_l \lambda_{l'}}{N_l N_{l'}} H_l(x) H_{l'}(x) H_l(y) H_{l'}(y) e^{-\frac{1}{2}x^2 - \frac{1}{2}y^2} \frac{e^{-\frac{1}{2}x^2 - \frac{1}{2}y^2}}{2\pi} \mathrm{d}x\mathrm{d}y \tag{22}$$

$$= \sum_{l,l'=C+1}^{\infty} \frac{\lambda_l \lambda_{l'}}{N_l N_{l'}} \frac{\int H_l(x) H_{l'}(x) e^{-x^2} \mathrm{d}x \int H_l(y) H_{l'}(y) e^{-y^2} \mathrm{d}y}{2\pi} \tag{23}$$

$$\overset{(a)}{=} \sum_{l=C+1}^{\infty} \frac{\lambda_l^2}{N_l^2} \frac{1}{2\pi} \sqrt{\pi} 2^l l! \sqrt{\pi} 2^l l! \overset{(b)}{=} \sum_{l=C+1}^{\infty} \frac{(2/3)^2 (1/3)^{2l}}{\frac{1}{2} 2^{2l} (l!)^2} \frac{2^{2l} (l!)^2}{2} = \frac{4}{9} \sum_{l=C+1}^{\infty} (1/3)^{2l} \tag{24}$$

$$\overset{(c)}{=} \frac{1}{2} (1/3)^{2C+2}, \tag{25}$$

where (a) is followed by orthogonality of Hermite polynomials, (b) is followed by the definition of λ_l and N_l in Section 3.1, and (c) is due to the infinite Geometric series.

As a result of eq. 25, the definition of $A_{x,y}$, and Jensen's inequality we have

$$\mathbb{E}_{X,Y}\big[|k(X,Y) - \langle \hat{\phi}_{HP}^{(C)}(X), \hat{\phi}_{HP}^{(C)}(X)\rangle|\big] \leq \mathbb{E}_{X,Y}^{1/2}\big[A_{X,Y}\big] \leq \frac{1}{3\sqrt{2}} \big(\frac{1}{3}\big)^C. \tag{26}$$

For bounding the expected error of random features, we expand the squared error using the definition given in eq. 4:

$$B_{x,y,\boldsymbol{\omega}} = \big|k(x,y) - \langle \phi_{RF,\boldsymbol{\omega}}(x), \phi_{RF,\boldsymbol{\omega}}(y)\rangle\big|^2 = \big|e^{-\frac{\rho(x-y)^2}{1-\rho^2}} - \frac{2}{C}\sum_{i=1}^{C/2} \cos\omega_i x \cos\omega_i y - \frac{2}{C}\sum_{i=1}^{C/2} \sin\omega_i x \sin\omega_i y\big| \tag{27}$$

$$= \underbrace{\big|e^{-\frac{\rho(x-y)^2}{1-\rho^2}} - \frac{2}{C}\sum_{i=1}^{C/2} \cos\omega_i (x-y)\big|^2}_{B_{x,y,\boldsymbol{\omega}}}. \tag{28}$$

Next, by setting $\rho = \frac{1}{3}$, we have

$$B_{x,y,\boldsymbol{\omega}} = e^{-\frac{3}{4}(x-y)^2} - \frac{4}{C} e^{-\frac{3}{8}(x-y)^2} \underbrace{\sum_{i=1}^{C/2} \cos\omega_i (x-y)}_{E_{1,x,y,\boldsymbol{\omega}}} + \frac{4}{C^2} \underbrace{\big(\sum_{i=1}^{C/2} \cos\omega_i (x-y)\big)^2}_{E_{2,x,y,\boldsymbol{\omega}}}. \tag{29}$$

Next, we calculate the average of terms $E_{1,x,y,\boldsymbol{\omega}}$ and $E_{2,x,y,\boldsymbol{\omega}}$ over $\boldsymbol{\omega}$.

Due to the Bochner's theorem (see Theorem 3.7 of (Unser & Tafti, 2014)) that shows a shift-invariant positive kernel could be written in the form of Fourier transform of a density function, we have

$$\mathbb{E}_{\boldsymbol{\omega}}\big[E_{1,x,y,\boldsymbol{\omega}}\big] = \mathbb{E}_{\boldsymbol{\omega}}\big[\sum_{i=1}^{C/2} \cos\omega_i (x-y)\big] \tag{30}$$

$$= \sum_{i=1}^{C/2} \mathbb{E}_{\omega_i}\big[e^{j\omega_i (x-y)}\big] = \frac{C}{2} e^{-\frac{3}{8}(x-y)^2}, \tag{31}$$

Next, we obtain the average of $E_{2,x,y,\boldsymbol{\omega}}$ as following:

$$\mathbb{E}_{\boldsymbol{\omega}}\big[E_{2,x,y,\boldsymbol{\omega}}\big] = \mathbb{E}_{\boldsymbol{\omega}}\big[\sum_{i,k=1}^{C/2}\cos\omega_i(x-y)\cos\omega_k(x-y)\big] \tag{32}$$

$$= \mathbb{E}_{\boldsymbol{\omega}}\big[\sum_{i,k=1}^{C/2}\frac{e^{j(\omega_i+\omega_k)(x-y)}+e^{j(\omega_i-\omega_k)(x-y)}+e^{j(-\omega_i+\omega_k)(x-y)}+e^{j(-\omega_i-\omega_k)(x-y)}}{4}\big] \tag{33}$$

$$\overset{(a)}{=} \sum_{i,k=1,i\neq k}^{C/2}\mathbb{E}_{\omega_i}\big[e^{j\omega_i(x-y)}\big]\mathbb{E}_{\omega_k}\big[e^{j\omega_k(x-y)}\big] + \frac{1}{2}\sum_{i=1}^{C/2}\big(\mathbb{E}_{\omega_i}\big[e^{j\omega_i(2x-2y)}\big]+1\big) \tag{34}$$

$$\overset{(b)}{=} \big(\frac{C^2}{4}-\frac{C}{2}\big)e^{-\frac{3}{4}(x-y)^2} + \frac{C}{4}\big(e^{-\frac{3}{4}(x-y)^2}+1\big) \tag{35}$$

$$= \frac{C^2}{4}e^{-\frac{3}{4}(x-y)^2} + \frac{C}{4}\big(e^{-\frac{3}{2}(x-y)^2} - 2e^{-\frac{3}{4}(x-y)^2}+1\big), \tag{36}$$

where (a) is due to symmetry of the normal distribution of $\boldsymbol{\omega}$, and (b) is followed by independence of ω_i and ω_k and their distribution symmetry.

Substituting eq. 31 and eq. 36 in eq. 29, and using Jensen's inequality, we have

$$\mathbb{E}_{X,Y\sim N(0,1)}\mathbb{E}_{\boldsymbol{\omega}}\big[B_{x,y,\boldsymbol{\omega}}\big] = \frac{1}{C}\mathbb{E}_{X,Y\sim N(0,1)}\Big[\big(e^{-\frac{3}{4}(X-Y)^2}-1\big)^2\Big] \geq \frac{1}{C}\mathbb{E}^2_{X,Y}\Big[e^{-\frac{3}{4}(x-y)^2}-1\Big] \tag{37}$$

$$= \frac{1}{C}\big(\underbrace{\mathbb{E}_{X,Y\sim N(0,1)}\big[e^{-\frac{3}{4}(X-Y)^2}\big]}_{G}-1\big)^2. \tag{38}$$

To calculate G, we have

$$G = \mathbb{E}_{X,Y\sim N(0,1)}\big[e^{-\frac{3}{4}(X-Y)^2}\big] = \int_{x,y}\frac{e^{-\frac{3}{4}(x^2+y^2-2xy)}e^{-\frac{x^2}{2}-\frac{y^2}{2}}}{2\pi}\mathrm{d}x\mathrm{d}y \tag{39}$$

$$= \int_{x,y}\frac{e^{-\frac{5}{4}(x^2+y^2)+\frac{3}{2}xy}}{2\pi}\mathrm{d}x\mathrm{d}y \tag{40}$$

$$= \int_{x,y}\frac{e^{-\frac{5}{4}(x^2-\frac{6}{5}xy+\frac{9}{25}y^2)+\frac{9}{25}\frac{5}{4}y^2-\frac{5}{4}y^2}}{2\pi}\mathrm{d}x\mathrm{d}y \tag{41}$$

$$\overset{(a)}{=} \int_y\frac{e^{-\frac{4}{5}y^2}}{\sqrt{2\pi\frac{5}{2}}}\int_x\frac{e^{-\frac{5}{4}(x-\frac{3}{5}y)^2}}{\sqrt{2\pi\frac{2}{5}}}\mathrm{d}x\mathrm{d}y \tag{42}$$

$$= \int_y\frac{e^{-\frac{4}{5}y^2}}{\sqrt{2\pi\frac{5}{2}}}\mathrm{d}y = \frac{1}{2}\int_y\frac{e^{-\frac{4}{5}y^2}}{\sqrt{2\pi\frac{5}{8}}} \tag{43}$$

$$\overset{(b)}{=} \frac{1}{2}, \tag{44}$$

where (a) and (b) hold since for a normal distribution $f_{a,b}(x) = \frac{e^{-\frac{(x-b)^2}{2a}}}{\sqrt{2\pi a}}$, we have $\int_x f_{a,b}(x)\mathrm{d}x = 1$. As a result of eq. 38 and eq. 44 we have

$$\mathbb{E}_{X,Y,\boldsymbol{\omega}}\big[B_{X,Y,\boldsymbol{\omega}}\big] \geq \frac{1}{4C}. \tag{45}$$

Finally, since $0 \leq B_{x,y,\boldsymbol{\omega}} \leq 4$, we have

$$\frac{1}{16C} \leq \mathbb{E}_{X,Y,\boldsymbol{\omega}}\big[\frac{B_{X,Y,\boldsymbol{\omega}}}{4}\big] \leq \mathbb{E}_{X,Y,\boldsymbol{\omega}}\big[\frac{|B_{X,Y,\boldsymbol{\omega}}|^{1/2}}{2}\big] = \frac{1}{2}\mathbb{E}_{X,Y,\boldsymbol{\omega}}\big[\big|k(X,Y)-\langle\phi_{RF,\boldsymbol{\omega}}(X),\phi_{RF,\boldsymbol{\omega}}(Y)\rangle\big|\big], \tag{46}$$

which proves eq. 18. □

C. Mercer's theorem and the generalized Hermite polynomials

We first review Mercer's theorem, which is a fundamental theorem on how can we find the approximation of a kernel via finite-dimensional feature maps.

Theorem C.1 ((Smola & Schölkopf, 1998) Theorem 2.10 and Proposition 2.11)**.** *Suppose $k \in L_\infty(\mathcal{X}^2)$, is a symmetric real-valued function, for a non-empty set $\mathcal{X}$, such that the integral operator $T_k f(x) = \int_{\mathcal{X}} k(x, x')f(x')\partial\mu(x')$ is positive definite. Let $\psi_j \in L_2(\mathcal{X})$ be the normalized orthogonal eigenfunctions of T_k associated with the eigenvalues $\lambda_j > 0$, sorted in non-increasing order, then*

1. *$(\lambda_j)_j \in \ell_1$,*
2. *$k(x, x') = \sum_{j=1}^{N_\mathcal{H}} \lambda_j \psi_j(x)\psi_j(x')$ holds for almost all (x, x'). Either $N_\mathcal{H} \in \mathbb{N}$, or $N_\mathcal{H} = \infty$; in the latter case, the series converge absolutely and uniformly for almost all (x, x').*

Furthermore, for every $\epsilon > 0$, there exists n such that

$$|k(x, x') - \sum_{j=1}^{n} \lambda_j \psi_j(x)\psi_j(x')| < \epsilon, \tag{47}$$

for almost all $x, x' \in \mathcal{X}$.

This theorem states that one can define a feature map

$$\Phi_n(x) = \left[\sqrt{\lambda_1}\psi_1(x), \ldots, \sqrt{\lambda_n}\psi_n(x)\right]^T \tag{48}$$

such that the Euclidean inner product $\langle \Phi(x), \Phi(x') \rangle$ approximates $k(x, x')$ up to an arbitrarily small factor ϵ.

By means of uniform convergence in Mercer's theorem, we can prove the convergence of the approximated MMD using the following lemma.

Lemma C.1. *Let $\mathcal{H}$ be an RKHS that is generated by the kernel $k(\cdot, \cdot)$, and let $\widehat{\mathcal{H}}_n$ be an RKHS with a kernel $k_n(\mathbf{x}, \mathbf{y})$ that can uniformly approximate $k(\mathbf{x}, \mathbf{y})$. Then, for a positive real value ϵ, there exists n, such that for every pair of distributions P, Q, we have*

$$\left|\mathrm{MMD}^2_{\mathcal{H}}(P, Q) - \mathrm{MMD}^2_{\widehat{\mathcal{H}}_n}(P, Q)\right| < \epsilon. \tag{49}$$

Proof. Firstly, using Theorem C.1, we can find n such that $\left|k(x, y) - \langle \Phi_n(x), \Phi_n(y) \rangle\right| < \frac{\epsilon}{4}$. We define the RKHS $\widehat{H}_n$ as the space of functions spanned by $\Phi_n(\cdot)$. Next, we rewrite $\mathrm{MMD}^2_{\mathcal{H}}(P, Q) - \mathrm{MMD}^2_{\widehat{\mathcal{H}}_n}(P, Q)$, using the definition of MMD in Section 2.1, as

$$\begin{aligned}
&\mathrm{MMD}^2_{\mathcal{H}}(P, Q) - \mathrm{MMD}^2_{\widehat{\mathcal{H}}_n}(P, Q) \\
&= \mathbb{E}_{x,x' \sim P}\left[k(x, x')\right] + \mathbb{E}_{y,y' \sim Q}\left[k(y, y')\right] - 2\mathbb{E}_{x \sim P, y \sim Q}\left[k(x, y)\right] \\
&- \mathbb{E}_{x,x' \sim P}\left[\langle \Phi_n(x), \Phi_n(x') \rangle\right] + \mathbb{E}_{y,y' \sim Q}\left[\langle \Phi_n(y), \Phi_n(y') \rangle\right] - 2\mathbb{E}_{x \sim P, y \sim Q}\left[\langle \Phi_n(x), \Phi_n(y) \rangle\right]
\end{aligned} \tag{50}$$

Therefore, we can bound $\left|\mathrm{MMD}^2_{\mathcal{H}}(P, Q) - \mathrm{MMD}^2_{\widehat{\mathcal{H}}_n}(P, Q)\right|$ as

$$\left|\mathrm{MMD}^2_{\mathcal{H}}(P, Q) - \mathrm{MMD}^2_{\widehat{\mathcal{H}}_n}(P, Q)\right| \overset{(a)}{\leq} \left|\mathbb{E}_{x,x' \sim P}\left[k(x, x')\right] - \mathbb{E}_{x,x' \sim P}\left[\left\langle \Phi_n(x), \Phi_n(x') \right\rangle\right]\right|$$

$$+ \left|\mathbb{E}_{y,y' \sim Q}\left[k(y, y')\right] - \mathbb{E}_{y,y' \sim P}\left[\left\langle \Phi_n(y), \Phi_n(y') \right\rangle\right]\right| + 2\left|\mathbb{E}_{x,y \sim P,Q}\left[k(x, y)\right] - \mathbb{E}_{x,y \sim P,Q}\left[\left\langle \Phi_n(x), \Phi_n(y) \right\rangle\right]\right|$$

$$\overset{(b)}{\leq} \mathbb{E}_{x,x'\sim P}\left[\left|k(x,x') - \left\langle \Phi_n(x), \Phi_n(x')\right\rangle\right|\right] + \mathbb{E}_{y,y'\sim Q}\left[\left|k(y,y') - \left\langle \Phi_n(y), \Phi_n(y')\right\rangle\right|\right] + 2\mathbb{E}_{x,y\sim P,Q}\left[\left|k(x,y) - \left\langle \Phi_n(x), \Phi_n(y)\right\rangle\right|\right]$$

$$\overset{(c)}{\leq} \mathbb{E}_{x,x'\sim P}\left[\frac{\epsilon}{4}\right] + \mathbb{E}_{y,y'\sim Q}\left[\frac{\epsilon}{4}\right] + 2\mathbb{E}_{x,y\sim P,Q}\left[\frac{\epsilon}{4}\right] = \epsilon \tag{51}$$

where (a) holds because of triangle inequality, (b) is followed by Tonelli's theorem and Jensen's inequality for absolute value function, and (c) is correct because of the choice of n as mentioned earlier in the proof. □

As a result of the above theorems, we can approximate the MMD in RKHS $\mathcal{H}_k$ for a kernel $k(\cdot,\cdot)$ via MMD in RKHS $\widehat{\mathcal{H}}_n \subseteq \mathbb{R}^n$ that is spanned by the first n eigenfunctions weighted by square roots of eigenvalues of the kernel $k(\cdot,\cdot)$. Therefore, in the following section, we focus on finding the eigenfunctions/eigenvalues of a multivariate Gaussian kernel.

C.1. Generalized Mehler's approximation

As we have already seen in eq. 5, Mehler's theorem provides us with an approximation of a one-dimensional Gaussian kernel in terms of Hermite polynomials. To generalize Mehler's theorem to a uniform covergence regime (that enables us to approximate MMD via such feature maps as shown in Lemma C.1), and for a multivariate Gaussian kernel, we make use of the following theorem.

Theorem C.2 ((Slepian, 1972), Section 6)**.** *Let the joint Gaussian density kernel $k(\mathbf{x},\mathbf{y},C) : \mathbb{R}^n \times \mathbb{R}^n \to \mathbb{R}$ be*

$$k(\mathbf{x},\mathbf{y},C) = \frac{1}{(2\pi)^n |C|^{1/2}} \exp\Big(-\frac{1}{2}[\mathbf{x},-\mathbf{y}]C^{-1}[\mathbf{x},-\mathbf{y}]^T\Big) \tag{52}$$

where C is a positive-definite matrix as

$$C = \begin{bmatrix} C_{11} & C_{12} \\ C_{12}^T & C_{22} \end{bmatrix}, \tag{53}$$

in which $C_{ij} \in \mathbb{R}^{n\times n}$ for $i,j \in \{1,2\}$, and $C_{11} = C_{22}$. Further, let the integral operator be defined with respect to a measure with density

$$w(\mathbf{x}) = \frac{1}{\int k(\mathbf{x},\mathbf{y},C)\partial\mathbf{y}}. \tag{54}$$

Then, the orthonormal eigenfunctions and eigenvalues for such kernel are

$$\psi_{\mathbf{k}}(\mathbf{x}) = \sum_{\mathbf{l}:\|\mathbf{l}\|_1=\|\mathbf{k}\|_1} \big(\sigma_{\|\mathbf{k}\|_1}(P)^{-1}\big)_{\mathbf{kl}} \frac{\varphi_{\mathbf{l}}(\mathbf{x};C_{11})}{\sqrt{\prod_{i=1}^n l_i!}}, \tag{55}$$

and

$$\lambda_{\mathbf{k}} = \prod_{i=1}^n \mathbf{e}_i^{\mathbf{k}_i/2}. \tag{56}$$

Here, $\sigma_p(A)$ is symmetrized Kronecker power of a matrix A, defined as

$$\big(\sigma_{\|\mathbf{k}\|_1}(A)\big)_{\mathbf{kl}} = \sqrt{\prod_{i=1}^n k_i! l_i!} \sum_{M\in\mathbb{R}^{n\times n}: M\mathbb{1}_n=\mathbf{k}, \mathbb{1}_n^T M=\mathbf{l}} \frac{\prod_{ij} A_{ij}^{M_{ij}}}{\prod_{ij} M_{ij}!}, \tag{57}$$

for two n-dimensional vectors $\mathbf{k}$ and $\mathbf{l}$ with $\|\mathbf{k}\|_1 = \|\mathbf{l}\|_1$, the vector $\mathbf{e}$ (the matrix P) is formed by eigenvalues (eigenvectors) of $C_{11}^{-1}C_{12}$, and $\varphi_{\mathbf{l}}(\mathbf{x},A)$ is generalized Hermite functions defined as

$$\varphi_{\mathbf{l}}(\mathbf{x}, A) = \frac{1}{(2\pi)^{n/2}|A|^{1/2}} \frac{\partial^{\|\mathbf{l}\|_1}}{\partial {x_1}^{l_1} \dots \partial {x_n}^{l_n}} \exp\big(-\frac{1}{2}\mathbf{x}^T A^{-1}\mathbf{x}\big). \tag{58}$$

The above theorem provides us with eigenfunctions/eigenvalues of a joint Gaussian density function. We utilize this theorem to approximate Mahalanobis kernels (i.e., a generalization of Gaussian radial basis kernels where $A = cI_n$) via Hermite polynomials as follow.

Proposition C.3. *A Mahalanobis kernel $k(\mathbf{x}, \mathbf{y}, A) : \mathbb{R}^D \times \mathbb{R}^D \to \mathbb{R}$ defined as*

$$k(\mathbf{x}, \mathbf{y}, A) = \exp\big(-(\mathbf{x}-\mathbf{y})A(\mathbf{x}-\mathbf{y})^T\big)$$

can be uniformly approximated as

$$k(\mathbf{x}, \mathbf{y}, A) \simeq \Big\langle \Phi_N\Big(\sqrt{\frac{\alpha^2-1}{\alpha}}\sqrt{A}\mathbf{x}\Big), \Phi_N\Big(\sqrt{\frac{\alpha^2-1}{\alpha}}\sqrt{A}\mathbf{y}\Big)\Big\rangle, \tag{59}$$

where $\Phi(\mathbf{x}) \in N^D$ is defined as a tensor product

$$\Phi_N(\mathbf{x}) = \bigotimes_{i=1}^{n} [\phi_{k_i}(x_i)]_{k_i=1}^{N}, \tag{60}$$

where

$$\phi_{k_i}(x_i) = \left(\frac{(\alpha^2-1)\alpha^{-k_i}}{\alpha^2 k_i!}\right)^{1/4} \exp\left(\frac{-x_i^2}{\alpha+1}\right) H_{k_i}(x_i) \tag{61}$$

Remark 1. *Using Proposition C.3 and Lemma C.1, we can show that the MMD based on the tensor feature map in eq. 60 and between any two distributions approximates the real MMD based on Gaussian kernel with Mahalanobis norm.*

Proof of Proposition C.3. Let $C = \begin{bmatrix} \frac{1}{2}I_n & \frac{1}{2\alpha}I_n \\ \frac{1}{2\alpha}I_n & \frac{1}{2}I_n \end{bmatrix}$, or equivalently $C^{-1} = \begin{bmatrix} \frac{2\alpha^2}{\alpha^2-1}I_n & -\frac{2\alpha}{\alpha^2-1}I_n \\ -\frac{2\alpha}{\alpha^2-1}I_n & \frac{2\alpha^2}{\alpha^2-1}I_n \end{bmatrix}$, for $\alpha \in [1, \infty)$.

Since C is positive-definite, we can define a Gaussian density kernel as

$$k(\mathbf{x}, \mathbf{y}, C) = \frac{1}{(\frac{\pi\sqrt{\alpha^2-1}}{2\alpha})^n} \exp\big(-\frac{\alpha^2}{\alpha^2-1}\|\mathbf{x}\|^2 - \frac{\alpha^2}{\alpha^2-1}\|\mathbf{y}\|^2 + \frac{2\alpha}{\alpha^2-1}\mathbf{y}\cdot\mathbf{x}^T\big). \tag{62}$$

Moreover, we can calculate the integration over all values of $\mathbf{y}$ as

$$\int k(\mathbf{x}, \mathbf{y}, C)\partial\mathbf{y} = \int \frac{\exp\big(-\|\mathbf{x}\|^2\big)}{(\frac{\pi\sqrt{\alpha^2-1}}{2\alpha})^n} \exp\big(-\frac{\|\alpha\mathbf{y}-\mathbf{x}\|^2}{(\alpha^2-1)}\big)\partial\mathbf{y} = \frac{\exp\big(-\|\mathbf{x}\|^2\big)}{(\pi)^{n/2}}. \tag{63}$$

Next, by setting $w(\mathbf{x}) = \frac{1}{\int k(\mathbf{x},\mathbf{y},C)\partial\mathbf{y}}$ and using Theorem C.2, we have

$$\int \frac{1}{(\pi\frac{\alpha^2-1}{\alpha^2})^{n/2}} \psi_{\mathbf{k}}(\mathbf{x}) \exp\big(-\frac{\|\alpha\mathbf{y}-\mathbf{x}\|^2}{\alpha^2-1}\big)\partial\mathbf{x} = \lambda_{\mathbf{k}}\psi_{\mathbf{k}}(\mathbf{y}). \tag{64}$$

Now to find the eigenfunctions of the Gaussian kernel $k'(\mathbf{x}, \mathbf{y}) = \exp\big(-\frac{\alpha\|\mathbf{x}-\mathbf{y}\|^2}{(\alpha^2-1)}\big)$, we let $\psi'_{\mathbf{k}}(\mathbf{x}) = \psi_{\mathbf{k}}(\mathbf{x})\exp\big(\frac{\alpha}{\alpha+1}\|\mathbf{x}\|^2\big)$ and let the weight function be $w'(\mathbf{x}) = (\pi)^{n/2}\exp\big(-\frac{(\alpha-1)}{\alpha+1}\|\mathbf{x}\|^2\big)$. As a result of such as-

sumptions, we see that

$$\int \psi'_{\mathbf{k}}(\mathbf{x})k'(\mathbf{x},\mathbf{y})w'(\mathbf{x})\partial\mathbf{x}$$

$$= \int (\pi)^{n/2}\psi_{\mathbf{k}}(\mathbf{x})\exp\big(-\frac{1}{\alpha^2-1}\|\mathbf{x}\|^2 - \frac{\alpha}{\alpha^2-1}\|\mathbf{y}\|^2 + \frac{2\alpha}{\alpha^2-1}\mathbf{x}\cdot\mathbf{y}^T\big)\partial\mathbf{x} \tag{65}$$

$$= (\pi)^{n/2}\exp\big(\frac{\alpha}{\alpha+1}\|\mathbf{y}\|^2\big)\int \psi_{\mathbf{k}}(\mathbf{x})\exp\big(-\frac{\|\alpha\mathbf{y}-\mathbf{x}\|^2}{\alpha^2-1}\big)\partial\mathbf{x} \tag{66}$$

$$\overset{(a)}{=} (\pi)^{n/2}\exp\big(\frac{\alpha}{\alpha+1}\|\mathbf{y}\|^2\big)\sqrt{\lambda_{\mathbf{k}}}\psi_{\mathbf{k}}(\mathbf{y})\Big(\frac{\pi(\alpha^2-1)}{\alpha^2}\Big)^{n/2} \tag{67}$$

$$\overset{(b)}{=} (\pi)^{n}\Big(\frac{\alpha^2-1}{\alpha^2}\Big)^{n/2}\lambda_{\mathbf{k}}\psi'_{\mathbf{k}}(\mathbf{y}), \tag{68}$$

where (a) holds because of eq. 64, and (b) is followed by the definition of $\psi'_{\mathbf{k}}(\mathbf{y})$. As a result, $\psi'_{\mathbf{k}}(\mathbf{x})$ is an eigenfunction of the integral operator with kernel $k'(\mathbf{x},\mathbf{y})$ and with weight function $w'(\mathbf{x})$.

Equation eq. 68 shows that the eigenvalue of $k'(\mathbf{x},\mathbf{y})$ corresponding to $\psi_{\mathbf{k}}(\mathbf{x})$ is as

$$\lambda'_{\mathbf{k}} = (\pi)^{n}\Big(\frac{\alpha^2-1}{\alpha^2}\Big)^{n/2}\lambda_{\mathbf{k}} \tag{69}$$

Now we show that such eigenfunctions are orthonormal. Deploying the idea in eq. 68, for two eigenfunctions $\psi'_{\mathbf{k}}(\cdot)$ and $\psi'_{\mathbf{l}}(\cdot)$ for fixed vectors $\mathbf{k},\mathbf{l}\in\mathbb{N}^n$, we have

$$\int \psi'_{\mathbf{k}}(\mathbf{y})\psi'_{\mathbf{l}}(\mathbf{y})w'(\mathbf{y})\partial\mathbf{y} \overset{(a)}{=} \int \psi_{\mathbf{k}}(\mathbf{y})\psi_{\mathbf{l}}(\mathbf{y})\frac{(\pi)^{n/2}}{\exp\big(-\|\mathbf{x}\|^2\big)}\partial\mathbf{y} \overset{(b)}{=} \int \psi_{\mathbf{k}}(\mathbf{y})\psi_{\mathbf{l}}(\mathbf{y})w(\mathbf{y}) \overset{(c)}{=} \delta[\mathbf{l}-\mathbf{k}], \tag{70}$$

where (a) is followed by the definition of eigenfunctions $\psi'_{\mathbf{k}}(\cdot)$, $\psi'_{\mathbf{l}}(\cdot)$ and the definition of weight function $w'(\mathbf{x})$, (b) is due to the definition of $w(\mathbf{x})$ and eq. 63, and (c) holds because of orthonormality of $\psi_{\mathbf{k}}$s as a result of Theorem C.2.

Further, in this case we have $C_{11}^{-1}C_{12} = \frac{1}{\alpha}I_n$, or equivalently $P = I_n$ and $\mathbf{e} = \frac{1}{\alpha}\mathbb{1}_n$. Hence, firstly using eq. 56, one can see that

$$\lambda_{\mathbf{k}} = \alpha^{-\|\mathbf{k}\|/2}. \tag{71}$$

Secondly, in finding symmetrized Kronecker power $\sigma_{\|k\|_1}(P)$ in eq. 57, for non-diagonal matrices M, the term $\prod_{ij}P_{ij}^{M_{ij}} = 0$. Further, for a diagonal matrix M, we have $M\mathbb{1}_n = \mathbb{1}_n M$. This induces the fact that

$$\sigma_{\|k\|_1}(P) = \begin{cases} 0 & \mathbf{k}\neq\mathbf{l}, \\ 1 & \mathbf{k}=\mathbf{l} \end{cases}. \tag{72}$$

This shows that

$$\psi_{\mathbf{l}}(\mathbf{x}) = \frac{\varphi_{\mathbf{l}}(\mathbf{x})}{\sqrt{\prod_{i=1}^n l_i!}}. \tag{73}$$

To find the formulation of eigenfunction $\psi_k(\mathbf{x})$, we can rewrite the term $\varphi_{\mathbf{l}}(\mathbf{x},C_{11})$ in eq. 55 for $C_{11} = \frac{1}{2}I_n$ as

$$\varphi_{\mathbf{l}}(\mathbf{x},I) = \frac{1}{(\pi)^{n/2}}\frac{\partial^{\|\mathbf{l}\|_1}}{\partial {x_1}^{l_1}\dots\partial {x_n}^{l_n}}\exp\Big(-\sum_{i=1}^n x_i^2\Big). \tag{74}$$

We note that the exponential function can be written as the product of functions that are only dependent on one variable x_i for $i\in[n]$. Hence, we can rephrase eq. 74 as a product of the derivative of each function as

$$\varphi_{\mathbf{l}}(\mathbf{x},I) = \prod_{i=1}^n \frac{1}{\sqrt{\pi}}\frac{\partial^{l_i}}{\partial^{l_i}x_i}\exp\big(-x_i^2\big). \tag{75}$$

As a result of this equation and the definition of Hermite functions in one dimension, we have

$$\varphi_{\mathbf{1}}(\mathbf{x}, I) = \frac{\exp(-\|\mathbf{x}\|^2)}{(\pi)^{n/2}} \prod_{i=1}^{n} H_{l_i}(x_i) \tag{76}$$

Hence, we can calculate $\psi'_{\mathbf{k}}(\mathbf{x})$ as

$$\psi'_{\mathbf{k}}(\mathbf{x}) = \frac{1}{\sqrt{(\pi)^n \prod_{i=1}^{n} k_i!}} \exp(\frac{-\|\mathbf{x}\|^2}{\alpha+1}) \prod_{i=1}^{n} H_{k_i}(x_i). \tag{77}$$

Using above discussion, we see that $\mathbf{k}$-th element $[\Phi_N(\mathbf{x})]_{\mathbf{k}}$ of the tensor $\Phi_N(x)$, which is defined in the proposition statement, is equal to

$$[\Phi_N(\mathbf{x})]_{\mathbf{k}} = \sqrt{\lambda'_{\mathbf{k}}}\psi'_{\mathbf{k}}(\mathbf{x}). \tag{78}$$

This fact and Theorem C.1 concludes that we can uniformly approximate $k'(\mathbf{x}, \mathbf{y})$ as

$$k'(\mathbf{x}, \mathbf{y}) = \langle \Phi_N(\mathbf{x}), \Phi_N(\mathbf{y}) \rangle. \tag{79}$$

Further, for any positive-definite matrix A, since the singular values of $\sqrt{\frac{\alpha^2-1}{\alpha}}\sqrt{A}$ are bounded, one can uniformly approximate $k''(\mathbf{x}, \mathbf{y}) := \exp\big(-(\mathbf{x}-\mathbf{y})A(\mathbf{x}-\mathbf{y})^T\big) = k'\Big(\sqrt{\frac{\alpha^2-1}{\alpha}}\sqrt{A}\mathbf{x}, \sqrt{\frac{\alpha^2-1}{\alpha}}\sqrt{A}\mathbf{y}\Big)$ as

$$k''(\mathbf{x}, \mathbf{y}) \simeq \Big\langle \Phi_N\Big(\sqrt{\frac{\alpha^2-1}{\alpha}}\sqrt{A}\mathbf{x}\Big), \Phi_N\Big(\sqrt{\frac{\alpha^2-1}{\alpha}}\sqrt{A}\mathbf{y}\Big) \Big\rangle \tag{80}$$

□

D. Sum-kernel upper-bound

Instead of using Generalized Hermite mean embedding which takes a huge amount of memory, one could use an upper bound to the joint Gaussian kernel. We use the inequality of arithmetic and geometric means to prove that.

$$k(\mathbf{x}, \mathbf{y}) = \exp\Big(-\frac{1}{2l^2}(\mathbf{x}-\mathbf{y})^T(\mathbf{x}-\mathbf{y})\Big) = \exp(-\frac{1}{2l^2}\sum_{d=1}^{D}(x_d - y_d)^2\Big) \tag{81}$$

$$= \prod_{d=1}^{D} \exp\Big(-\frac{1}{2l^2}(x_d - y_d)^2\Big) \tag{82}$$

$$\overset{(a)}{\leq} \frac{1}{D}\sum_{d=1}^{D} \exp\Big(-\frac{D}{2l^2}(x_d - y_d)^2\Big) \tag{83}$$

$$= \frac{1}{D}\sum_{d=1}^{D} k_{X_d}(x_d, y_d), \tag{84}$$

where (a) holds due to inequality of arithmetic and geometric means (AM-GM), and $k_{X_d}(\cdot,\cdot)$ is defined as

$$k_{X_d}(x_d, y_d) := \exp\Big(-\frac{D}{2l^2}(x_d - y_d)^2\Big). \tag{85}$$

Next, we approximate such kernel via an inner-product of the feature maps

$$\boldsymbol{\phi}_C(\mathbf{x}) = \begin{bmatrix} \boldsymbol{\phi}^{(C)}_{HP,1}(x_1)/\sqrt{D} \\ \boldsymbol{\phi}^{(C)}_{HP,2}(x_2)/\sqrt{D} \\ \vdots \\ \boldsymbol{\phi}^{(C)}_{HP,D}(x_D)/\sqrt{D} \end{bmatrix} \in \mathbb{R}^{((C+1)\cdot D)\times 1}. \tag{86}$$

Although such feature maps are not designed to catch correlation among dimensions, they provide us with a guarantee on marginal distributions as follows.

Lemma D.1. *Define $k_{X_i}(\cdot,\cdot)$ as in eq. 85 and define $\phi_C(\mathbf{x})$ as in eq. 86. For $\epsilon \in \mathbb{R}^+$, there exists N such that for $C \geq N$ we have*

- $\left\|\mathbb{E}_{\mathbf{x}\sim P}\left[\phi_C(\mathbf{x})\right] - \mathbb{E}_{\mathbf{y}\sim Q}\left[\phi_C(\mathbf{y})\right]\right\|_2 \leq \epsilon \Rightarrow \mathrm{MMD}_{k_{X_i}}(P_i, Q_i) \leq \sqrt{D+1}\epsilon$ *for every* $i \in \{1,\dots,D\}$, *and*
- $\mathrm{MMD}_{k_{X_i}}(P_i, Q_i) \leq \epsilon$ *for every* $i \in \{1,\dots,D\} \Rightarrow \left\|\mathbb{E}_{\mathbf{x}\sim P}\left[\phi_C(\mathbf{x})\right] - \mathbb{E}_{\mathbf{y}\sim Q}\left[\phi_C(\mathbf{y})\right]\right\| \leq \sqrt{2}\epsilon$,

where P_i and Q_i are marginal probability distributions corresponding to P and Q, respectively.

Proof. Since $\boldsymbol{\phi}^{(C)}_{HPi}(x_i)$ has the certain form as in Theorem C.1, then Lemma C.1 shows that we can use such feature maps to uniformly approximate the MMD in an RKHS based on the kernel $k_i(x_i, y_i) = \exp\left(-\frac{1}{2l^2}(x_i - y_i)^2\right)$. As a result, there exists N such that for $C \geq N$, we have

$$\left|\left\|\mathbb{E}_{x_i\sim P_i}\left[\phi^{(C)}_{HP,i}(x_i)\right] - \mathbb{E}_{y_i\sim Q_i}\left[\phi^{(C)}_{HP,i}(y_i)\right]\right\|_2^2 - \mathrm{MMD}^2_{k_{X_i}}(P_i, Q_i)\right| \leq D\epsilon^2. \tag{87}$$

Now we prove the first part. Knowing

$$\left\|\mathbb{E}_{\mathbf{x}\sim P}\left[\phi_C(\mathbf{x})\right] - \mathbb{E}_{\mathbf{y}\sim Q}\left[\phi_C(\mathbf{y})\right]\right\|_2 \leq \epsilon, \tag{88}$$

and by the definition of $\phi_C(\cdot)$, we deduce that

$$\left\|\mathbb{E}_{x_i\sim P_i}\left[\phi^{(C)}_{HP,i}(x_i)\right] - \mathbb{E}_{y_i\sim Q_i}\left[\phi^{(C)}_{HP,i}(y_i)\right]\right\|_2^2 \leq \epsilon^2. \tag{89}$$

Using this and eq. 87 we can prove the first part.

Inversely, by setting $\mathrm{MMD}_{k_{X_i}}(P_i, Q_i) \leq \epsilon$ and eq. 87, one sees that

$$\left\|\mathbb{E}_{x_i\sim P_i}\left[\phi^{(C)}_{HP,i}(x_i)\right] - \mathbb{E}_{y_i\sim Q_i}\left[\phi^{(C)}_{HP,i}(y_i)\right]\right\|_2 \leq \sqrt{2}\epsilon. \tag{90}$$

This coupled with the definition of Φ_C completes the second part of lemma. □

E. ϕ Recursion

$$\begin{aligned}
\phi_{k+1}(x) &= ((1+\rho)(1-\rho))^{\frac{1}{4}} \frac{\rho^{\frac{k+1}{2}}}{\sqrt{2^{k+1}(k+1)!}} H_{k+1}(x) \exp\left(-\frac{\rho}{\rho+1}x^2\right), \quad \text{by definition} \\
&= ((1+\rho)(1-\rho))^{\frac{1}{4}} \frac{\rho^{\frac{k+1}{2}}}{\sqrt{2^{k+1}(k+1)!}} \left[2xH_k(x) - 2kH_{k-1}(x)\right] \exp\left(-\frac{\rho}{\rho+1}x^2\right), \\
&= \frac{\sqrt{\rho}}{\sqrt{2(k+1)}} 2x\phi_k(x) - \frac{\rho}{\sqrt{k(k+1)}} k\phi_{k-1}(x).
\end{aligned} \tag{91}$$

F. Sensitivity of mean embeddings (MEs)

F.1. Sensitivity of ME under the sum kernel

Here we derive the sensitivity of the mean embedding corresponding to the sum kernel.

$$S_{\widehat{\boldsymbol{\mu}}^s_P} = \max_{\mathcal{D},\mathcal{D}'} \|\widehat{\boldsymbol{\mu}}^s_P(\mathcal{D}) - \widehat{\boldsymbol{\mu}}^s_P(\mathcal{D}')\|_F = \max_{\mathcal{D},\mathcal{D}'} \|\tfrac{1}{m}\sum_{i=1}^m \mathbf{h}_s(\mathbf{x}_i)\mathbf{f}(\mathbf{y}_i)^T - \tfrac{1}{m}\sum_{i=1}^m \mathbf{h}_s(\mathbf{x}'_i)\mathbf{f}(\mathbf{y}'_i)^T\|_F$$

where $\|\cdot\|_F$ represents the Frobenius norm. Since $\mathcal{D}$ and $\mathcal{D}'$ are neighbouring, then $m-1$ of the summands on each side cancel and we are left with the only distinct datapoints, which we denote as $(\mathbf{x}, \mathbf{y})$ and $(\mathbf{x}', \mathbf{y}')$. We then apply the triangle

inequality and the definition of $\mathbf{f}$. As $\mathbf{y}$ is a one-hot vector, all but one column of $\mathbf{h}_s(\mathbf{x})\mathbf{f}(\mathbf{y})^\top$ are 0, so we omit them in the next step:

$$\begin{aligned} S_{\boldsymbol{\mu}_P^s} &= \max_{(\mathbf{x},\mathbf{y}),(\mathbf{x}',\mathbf{y}')} \|\tfrac{1}{m}\mathbf{h}_s(\mathbf{x})\mathbf{f}(\mathbf{y})^T - \tfrac{1}{m}\mathbf{h}_s(\mathbf{x}')\mathbf{f}(\mathbf{y}')^T\|_F \\ &\leq \max_{(\mathbf{x},\mathbf{y})} \tfrac{2}{m}\|\mathbf{h}_s(\mathbf{x})\mathbf{f}(\mathbf{y})^T\|_F = \max_{\mathbf{x}} \tfrac{2}{m}\|\mathbf{h}_s(\mathbf{x})\|_2. \end{aligned} \tag{92}$$

We recall the definition of the feature map given in eq. 10,

$$\|\mathbf{h}_s(\mathbf{x})\|_2 = \frac{1}{\sqrt{D}}\left(\sum_{d=1}^{D}\|\boldsymbol{\phi}_{HP,d}^{(C)}(x_d)\|_2^2\right)^{\frac{1}{2}}. \tag{93}$$

To bound $\|\mathbf{h}_s(\mathbf{x})\|_2$, we first prove that $\|\boldsymbol{\phi}_{HP,d}^{(C)}(x_d)\|_2^2 \leq 1$. Using Mehler's formula (see eq. 5), and by plugging in $y = x_d$, one can show that

$$1 = \exp\Big(-\frac{\rho}{1-\rho^2}(x_d - x_d)^2\Big) = \sum_{c=0}^{\infty}\lambda_c f_c(x_d)^2. \tag{94}$$

Using this, we rewrite the infinite sum in terms of the Cth-order approximation and the rest of summands to show that

$$1 = \sum_{c=0}^{\infty}\lambda_c f_c^2(x_d) \overset{(a)}{=} \|\boldsymbol{\phi}_{HP,d}^{(C)}(x_d)\|_2^2 + \sum_{c=C+1}^{\infty}\lambda_c f_c^2(x) \overset{(b)}{\geq} \|\boldsymbol{\phi}_{HP,d}^{(C)}(x_d)\|_2^2, \tag{95}$$

where (a) holds because of the definition of $\boldsymbol{\phi}_{HP,d}^{(C)}(x_d)$ in eq. 6: $\|\boldsymbol{\phi}_{HP,d}^{(C)}(x_d)\|_2^2 = \sum_{c=0}^{C}\lambda_c f_c^2(x_d)$, and (b) holds, because λ_c and $f_c^2(x)$ are non-negative scalars.

Finally, deploying eq. 92, eq. 93, and eq. 95, we bound the sensitivity as

$$S_{\boldsymbol{\mu}_P} \leq \max_{\mathbf{x}} \tfrac{2}{m}\|\mathbf{h}_s(\mathbf{x})\|_2 \leq \tfrac{2}{m\sqrt{D}}\sqrt{D} = \tfrac{2}{m}. \tag{96}$$

F.2. Sensitivity of ME under the product kernel

Similarly, we derive the sensitivity of the mean embedding corresponding to the product kernel.

$$S_{\widehat{\boldsymbol{\mu}}_P^p} = \max_{\mathcal{D},\mathcal{D}'}\|\widehat{\boldsymbol{\mu}}_P^p(\mathcal{D}) - \widehat{\boldsymbol{\mu}}_P^p(\mathcal{D}')\|_F \leq \max_{\mathbf{x}} \tfrac{2}{m}\|\mathbf{h}_p(\mathbf{x}^{D_{prod}})\|_2$$

Given the definition in eq. 8, the L2 norm is given by

$$\tfrac{2}{m}\|\mathbf{h}_p(\mathbf{x}^{D_{prod}})\|_2 = \tfrac{2}{m}\prod_{d=1}^{D_{prod}}\|\boldsymbol{\phi}_{HP}^{(C)}(x_d)\|_2, \tag{97}$$

$$\leq \tfrac{2}{m} \tag{98}$$

where the last line is due to eq. 95.

G. Descriptions on the tabular datasets

In this section we give more detailed information about the tabular datasets used in our experiments. Unless otherwise stated, the datasets were obtained from the UCI machine learning repository (Dua & Graff, 2017).

Adult

Adult dataset contains personal attributes like age, gender, education, marital status or race from the different dataset participants and their respective income as the label (binarized by a threshold set to 50K). The dataset is publicly available at the UCI machine learning repository at the following link: `https://archive.ics.uci.edu/ml/datasets/adult`.

Census

The Census dataset is also a public dataset that can be downloaded via the SDGym package [9]. This is a clear example of an imbalaned dataset since only 12382 of the samples are considered positive out of a total of 199523 samples.

Cervical

The Cervical cancer dataset comprises demographic information, habits, and historic medical records of 858 patients and was created with the goal to identify the cervical cancer risk factors. The original dataset can be found at the following link: https://archive.ics.uci.edu/ml/datasets/Cervical+cancer+%28Risk+Factors%29.

Covtype

This dataset contains cartographic variables from four wilderness areas located in the Roosevelt National Forest of northern Colorado and the goal is to predict forest cover type from the 7 possible classes. The data is publicly available at https://archive.ics.uci.edu/ml/datasets/covertype.

Credit

The Credit Card Fraud Detection dataset contains the categorized information of credit card transactions made by European cardholders during September 2013 and the corresponding label indicating if the transaction was fraudulent or not. The dataset can be found at: https://www.kaggle.com/mlg-ulb/creditcardfraud. The original dataset has a total number of 284807 samples where only 492 of them are frauds. In our experiments, we descarded the feature related to the time the transaction was done. The data is released under a Database Contents License (DbCL) v1.0.

Epileptic

The Epileptic Seizure Recognition dataset contains the brain activity measured in terms of the EEG across time. The dataset can be found at https://archive.ics.uci.edu/ml/datasets/Epileptic+Seizure+Recognition. The original dataset contains 5 different labels that we binarized into two: seizure (2300 samples) or not seizure (9200 samples).

Intrusion

The dataset was used for The Third International Knowledge Discovery and Data Mining Tools Competition held at the Conference on Knowledge Discovery and Data Mining, 1999, and can be found at http://kdd.ics.uci.edu/databases/kddcup99/kddcup99.html. We used the file named "kddcup.data10percent.gz" that contains the 10% of the orginal dataset. The goal is to distinguish between intrusion/attack and normal connections categorized in 5 different labels.

Isolet

The Isolet dataset contains the features sounds as spectral coefficients, contour features, sonorant features, pre-sonorant features, and post-sonorant features of the different letters on the alphabet as inputs and the corresponding letter as the label. The original dataset can be found at https://archive.ics.uci.edu/ml/datasets/isolet. However, in our experiments we considered this dataset as a binary classification task as we only considered the labels as constants or vowels.

Table 3 summarizes the number of samples, labeled classes and type of different inputs (numerical, ordinal or categorical) for each tabular dataset used in our experiments. The content of the table reflects the results after pre-processing or binarizing the corresponding datasets.

G.1. Hyperparameters for discrete tabular datasets

Here we include the hyperparameters used in obtaining the results obtained in Table 1. In the main text we describe the choices of the Hermitian hyperparameters. In the separate section G.2 we present a broader view over the gamma hyperparameter.

[9] SDGym package website: https://pypi.org/project/sdgym/

Table 3. Tabular datasets. Size, number of classes and feature types descriptions.

dataset	# samps	# classes	# features
isolet	4366	2	617 num
covtype	406698	7	10 num, 44 cat
epileptic	11500	2	178 num
credit	284807	2	29 num
cervical	753	2	11 num, 24 cat
census	199523	2	7 num, 33 cat
adult	48842	2	6 num, 8 cat
intrusion	394021	5	8 cat, 6 ord, 26 num

Table 4. Hyperparameters for discrete tabular datasets

	privacy	batch rate	order hermite prod	prod dimension	gamma	order hermite
Adult	$\varepsilon = 0.3$	0.1	10	5	1	100
	$\varepsilon = 0.1$	0.1	5	7	1	100
Census	$\varepsilon = 0.3$	0.01	5	7	0.1	100
	$\varepsilon = 0.1$	0.01	5	7	0.1	100

G.2. Gamma hyperparameter ablation study

Here we study the impact of gamma γ hyperparameter on the quality of the generated samples. Gamma describes the weight that is given to the product kernel in relation to the sum kernel. We elaborate on the results from the Table 1 which describe α-way marginals evaluated on generated samples with discretized Census dataset. We fix all the hyperparameters and vary gamma. The Table 5 shows the impact of gamma. The $k-$way results remain indifferent for $\gamma \leq 1$ but deterioriate for $\gamma > 1$. In this experiment, we set $\epsilon_1 = \epsilon_2 = \epsilon/2$. Here, "order hermite prod " means the HP order for the outer product kernel, "prod dimension" means the number of subsampled input dimensions, and "order hermite" means the HP order for the sum kernel.

Table 5. The impact of gamma hyperparamer.

epsilon	batch rate	order hermite prod	prod dimension	gamma	epochs	3-way	4-way
0.3	0.1	10	5	0.001	8	0.474	0.570
0.3	0.1	10	5	0.01	8	0.473	0.570
0.3	0.1	10	5	0.1	8	0.499	0.597
0.3	0.1	10	5	1	8	0.474	0.570
0.3	0.1	10	5	10	8	0.585	0.671
0.3	0.1	10	5	100	8	0.674	0.757
0.3	0.1	10	5	1000	8	0.676	0.761

G.3. Training DP-HP generator

Here we provide the details of the DP-HP training procedure we used on the tabular data experiments. Table 6 shows the Hermite polynomial order, the fraction of dataset used in a batch, the number of epochs and the undersampling rate we used during training for each tabular dataset.

G.4. Non-private results

We also show the non-private MERF and HP results in Table 7.

Table 6. Tabular datasets. Hyperparameter settings for private constraints $\epsilon = 1$ and $\delta = 10^{-5}$.

data name	batch rate	order hermite prod	prod dimension	order hermite	gamma
adult	0.1	5	5	100	0.1
census	0.5	5	5	100	0.1
cervical	0.5	13	5	20	1
credit	0.5	7	5	20	1
epileptic	0.1	5	7	20	0.1
isolet	0.5	13	5	150	1
covtype	0.01	7	2	10	1
intrusion	0.01	5	5	7	1

Table 7. Performance comparison on Tabular datasets. The average over five independent runs.

	Real		DP-MERF (non-priv)		DP-HP (non-priv)		DP-CGAN $(1, 10^{-5})$-DP		DP-GAN $(1, 10^{-5})$-DP		DP-MERF $(1, 10^{-5})$-DP		**DP-HP** $(1, 10^{-5})$-DP	
adult	0.786	0.683	0.642	0.525	**0,673**	**0,621**	0.509	0.444	0.511	0.445	0.642	0.524	**0,688**	**0,632**
census	0.776	0.433	0.696	0.244	**0,707**	**0,32**	0.655	0.216	0.529	0.166	0.685	0.236	**0,699**	**0,328**
cervical	0.959	0.858	**0.863**	**0.607**	0,823	0,574	0.519	0.200	0.485	0.183	0.531	0.176	**0,616**	**0,312**
credit	0.924	0.864	**0.902**	0.828	0.89	**0,863**	0.664	0.356	0.435	0.150	0.751	0.622	**0,786**	**0,744**
epileptic	0.808	0.636	0.564	0.236	**0,602**	**0,546**	0.578	0.241	0.505	0.196	0.605	0.316	**0,609**	**0,554**
isolet	0.895	0.741	0.755	0.461	**0,789**	**0,668**	0.511	0.198	0.540	0.205	0.557	0.228	**0,572**	**0,498**
	F1		F1		F1		F1		F1		F1		F1	
covtype	0.820		**0.601**		0.580		0.285		0.492		0.467		**0.537**	
intrusion	0.971		0.884		**0.888**		0.302		0.251		**0.892**		0.890	

G.5. The effect of subsampled input dimensions for the product kernel on Adult dataset

Table 8 shows the 3-way (Left) and 4-way (Right) marginals evaluated at different number of dimensions for the product kernel (D_{prod}) where the rest of hyperparameters are fixed. The results show that increasing the number of dimensions in the product kernel improved the result.

Table 8. Trade-off for subsampling dimensions in the product kernel for Adult dataset.

	D_{prod}			D_{prod}		
ϵ	2	5	7	2	5	7
1	0.367	0.34	**0.332**	0.466	0.434	**0.422**

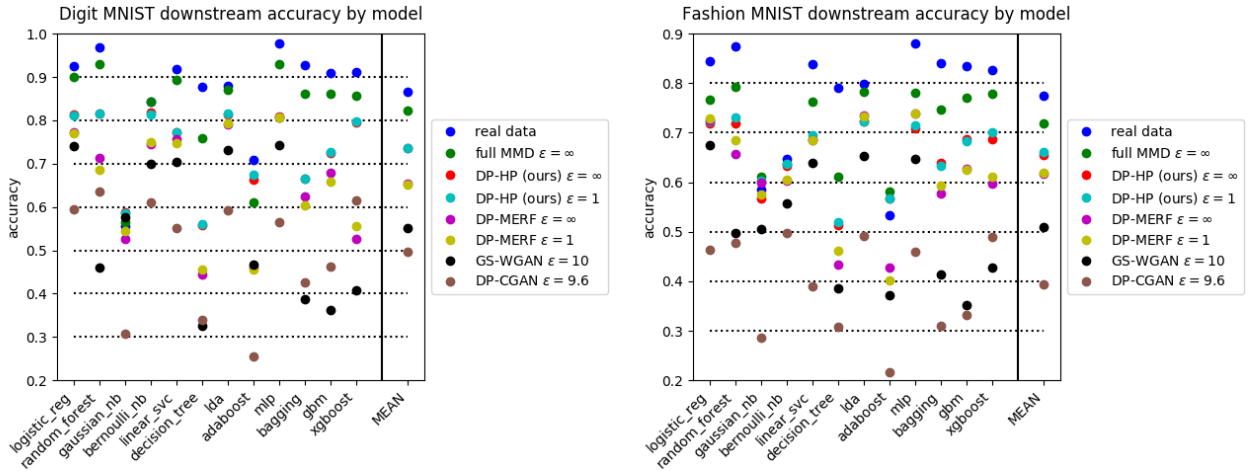
H. Image data

H.1. Results by model

In the following we provide a more detailed description of the downstreams models accuracy over the different methods considered in the image datasets.

H.2. MNIST and fashionMNIST hyper-parameter settings

Here we give a detailed hyper-parameter setup and the architectures used for generating synthetic samples via DP-HP for MNIST and FashionMNIST datasets in Table 9. The non-private version of our method does not exhibit a significant accuracy difference between 2, 3 and 4 subsampled dimensions for the product kernel, so we considered product dimension to be 2 for memory savings. Table 10 summarizes the 12 predictive models hyper-parameters setup for the image datasets trained on the generated samples via DP-HP. In this experiment, we optimize this loss $\min_{\boldsymbol{\theta}} \; ||\widehat{\boldsymbol{\mu}}_P^p - \widehat{\boldsymbol{\mu}}_{Q_{\boldsymbol{\theta}}}^p||_2^2 + \gamma||\widehat{\boldsymbol{\mu}}_P^s - \widehat{\boldsymbol{\mu}}_{Q_{\boldsymbol{\theta}}}^s||_2^2$, where γ is multiplied by the sum kernel's loss.



(a) MNIST (b) FashionMNIST

Figure 6. We compare the real data test accuracy of models trained on synthetic data for various models: DP-HP, DP-MERF, GS-WGAN and DP-CGAN. As baselines we also include results for real training data and a generator, which is non-privately trained with MMD, listed as "full MMD". We show accuracy sorted by downstream classifier and the mean accuracy across classifiers on the right. Each score is the average of 5 independent runs.

Table 9. Hyperparameter settings for image data experiments. All parameters not listed here are used with their default values.

	MNIST		FashionMNIST	
	(non-priv)	$(1, 10^{-5})$-DP	(non-priv)	$(1, 10^{-5})$-DP
Hermite order (sum kernel)	100	100	100	100
Hermite order (product kernel)	20	20	20	20
kernel length (sum kernel)	0.005	0.005	0.15	0.15
kernel length (product kernel)	0.005	0.005	0.15	0.15
product dimension	2	2	2	2
subsample product dimension	beginning of each epoch	beginning of each epoch	beginning of each epoch	beginning of each epoch
gamma	5	20	20	10
mini-batch size	200	200	200	200
epochs	10	10	10	10
learning rate	0.01	0.01	0.01	0.01
architecture	fully connected	fully connected	CNN + bilinear upsampling	CNN + bilinear upsampling

Table 10. Hyperparameter settings for downstream models used in image data experiments. Models are taken from the scikit-learn 0.24.2 and xgboost 0.90 python packages and hyperparameters have been set to achieve reasonable accuracies while limiting runtimes. Paramters not listed are kept at their default values.

Model	Parameters
Logistic Regression	solver: lbfgs, max_iter: 5000, multi_class: auto
Gaussian Naive Bayes	-
Bernoulli Naive Bayes	binarize: 0.5
LinearSVC	max_iter: 10000, tol: 1e-8, loss: hinge
Decision Tree	class_weight: balanced
LDA	solver: eigen, n_components: 9, tol: 1e-8, shrinkage: 0.5
Adaboost	n_estimators: 1000, learning_rate: 0.7, algorithm: SAMME.R
Bagging	max_samples: 0.1, n_estimators: 20
Random Forest	n_estimators: 100, class_weight: balanced
Gradient Boosting	subsample: 0.1, n_estimators: 50
MLP	-
XGB	colsample_bytree: 0.1, objective: multi:softprob, n_estimators: 50

D

PRE-TRAINED PERCEPTUAL FEATURES IMPROVE DIFFERENTIALLY PRIVATE IMAGE GENERATION

1 https://creativecommons.org/licenses/by/4.0/

Pre-trained Perceptual Features Improve Differentially Private Image Generation

Frederik Harder *fharder@tue.mpg.de*
Max Planck Institute for Intelligent Systems & University of Tübingen

Milad Jalali Asadabadi *miladj7@cs.ubc.ca*
University of British Columbia

Danica J. Sutherland *dsuth@cs.ubc.ca*
University of British Columbia & Alberta Machine Intelligence Institute

Mijung Park *mijungp@cs.ubc.ca*
University of British Columbia & Alberta Machine Intelligence Institute

Reviewed on OpenReview: *https://openreview.net/forum?id=R6W7zkMz0P*

Abstract

Training even moderately-sized generative models with differentially-private stochastic gradient descent (DP-SGD) is difficult: the required level of noise for reasonable levels of privacy is simply too large. We advocate instead building off a good, relevant representation on an informative public dataset, then learning to model the private data with that representation. In particular, we minimize the maximum mean discrepancy (MMD) between private target data and a generator's distribution, using a kernel based on perceptual features learned from a public dataset. With the MMD, we can simply privatize the data-dependent term once and for all, rather than introducing noise at each step of optimization as in DP-SGD. Our algorithm allows us to generate CIFAR10-level images with $\epsilon \approx 2$ which capture distinctive features in the distribution, far surpassing the current state of the art, which mostly focuses on datasets such as MNIST and FashionMNIST at a large $\epsilon \approx 10$. Our work introduces simple yet powerful foundations for reducing the gap between private and non-private deep generative models. Our code is available at `https://github.com/ParkLabML/DP-MEPF`.[1]

1 INTRODUCTION

The gold standard privacy notion, *differential privacy* (DP), is now ubiquitous in a diverse range of academic research, industry products (Apple, 2017), and even government databases (National Conference of State Legislatures, 2021). DP provides a mathematically provable privacy guarantee, which is its main strength and reason for its popularity. DP even offers means of tracking the effect of multiple accesses to the same data on it's overall privacy level, but with each access, the privacy of the data gradually degrades. To guarantee a high level of privacy, one thus needs to limit access to data, a challenge in applying DP with the usual iterative optimization algorithms used in machine learning.

Differentially private data generation solves this problem by creating a synthetic dataset that is *similar* to the private dataset, in terms of some chosen similarity metric. While producing such a synthetic dataset incurs a privacy loss, the resulting dataset can be used repeatedly without further loss of privacy. Classical approaches, however, typically assume a certain class of pre-specified purposes on how the synthetic data can be used (Mohammed et al., 2011; Xiao et al., 2010; Hardt et al., 2012; Zhu et al., 2017). If data analysts use the data for other tasks outside these pre-specified purposes, the theoretical guarantees on its utility are lost.

[1]This is a revision of the first published version which contained erroneous FID scores. Please refer to this paper's OpenReview page for a clarification of our errors and the older version.

To produce synthetic data usable for potentially *any* purpose, many papers on DP data generation have utilized the recent advances in deep generative modelling. The majority of these approaches are based on the generative adversarial network (GAN; Goodfellow et al., 2014) framework, where a discriminator and a generator play an adversarial game to optimize a given distance metric between the true and synthetic data distributions. Most approaches under this framework have used DP-SGD (Abadi et al., 2016), where the gradients of the discriminator (which compares generated samples to private data) are privatized in each training step, resulting in a high overall privacy loss (Park et al., 2017; Torkzadehmahani et al., 2019; Yoon et al., 2019; Xie et al., 2018; Frigerio et al., 2019). Another challenge is that, as the gradients must have bounded norm to derive the DP guarantee, the amount of noise for privatization in DP-SGD increases proportionally to the dimension of the discriminator. Hence, these methods are typically bound to relatively small discriminators, limiting the ability to learn data distributions beyond, say, MNIST (LeCun & Cortes, 2010) or FashionMNIST (Xiao et al., 2017).

Given these challenges, the heavy machinery such as GANs and large-scale auto-encoder-based methods – capable of generating complex datasets in a non-private setting – fails to model datasets such as CIFAR-10 (Krizhevsky, 2009) or CelebA (Liu et al., 2015) with a meaningful privacy guarantee (e.g., $\epsilon \approx 2$). Typical deep generative modeling papers have moved well beyond these datasets, but to the best of our knowledge, currently there is no DP data generation method that can produce reliable samples at a reasonable privacy level.

How can we reduce this huge gap between the performance of non-private deep generative models and that of private counterparts? We argue that we can narrow this gap by using the abundant resource of *public* data, in line with the core message of Tramèr & Boneh (2021): *We simply need better features for differentially private learning*. While Tramèr & Boneh demonstrated this in the context of DP classification, we aim to show the applicability of this reasoning for the more challenging problem of DP data generation, with a focus on high-dimensional image generation.

We propose to exploit public data to learn *perceptual features* (PFs) from public data, which we will use to compare synthetic and real data distributions. Following dos Santos et al. (2019), we use "perceptual features" to mean the vector of all activations of a pretrained deep network for a given data point, e.g. the hundreds of thousands of hidden activations from applying a trained deep classifier to an image. Building on dos Santos et al. (2019), who use PFs for transfer learning in natural image generation, our goal is to improve the quality of natural images generated with differential privacy constraints.

We construct a kernel on images using these powerful PFs, then train a generator by minimizing the Maximum Mean Discrepancy (MMD) (Gretton et al., 2012) between distributions (as in Harder et al., 2021; Li et al., 2015; Dziugaite et al., 2015; dos Santos et al., 2019). This scheme is non-adversarial, leading to simpler and more stable optimization; moreover, it allows us to privatize the mean embedding of the private dataset *once*, using it at each step of generator training without incurring cumulative privacy losses.

We observe in our experiments that as long as the public data contains more complex patterns than private data, e.g., transferring the knowledge learned from ImageNet as public data to generate CIFAR-10 images as private data, the learned features from public data are useful enough to generate good synthetic data. We successfully generate reasonable samples for CIFAR-10, CelebA, MNIST, and FashionMNIST in high-privacy regimes. We also theoretically analyze the effect of privatizing our loss function, helping understand the privacy-accuracy trade-offs in our method.

The main point of our paper is that features from public data are a key tool for improved DP data generation, a point we think our experiments make resoundingly; this may be "obvious", but has not been explored for image generation. Our proposed method, in particular, is a simple (which, we think, is a good thing) initial technique exploiting this idea, which outperforms simple pretraining of DP-GAN and DP-Sinkhorn (see Section 6). We hope this work will inspire future work on other ways to use public features for improving image generation with differential privacy.

2 BACKGROUND

We provide background information on maximum mean discrepancy and differential privacy.

Maximum Mean Discrepancy The MMD is a distance between distributions based on a kernel $k_\phi(x, y) = \langle \phi(x), \phi(y) \rangle_{\mathcal{H}}$, where ϕ maps data in $\mathcal{X}$ to a Hilbert space $\mathcal{H}$ (Gretton et al., 2012). One definition is

$$\mathrm{MMD}_{k_\phi}(P, Q) = \left\| \mathbb{E}_{x \sim P}[\phi(x)] - \mathbb{E}_{y \sim Q}[\phi(y)] \right\|_{\mathcal{H}},$$

where $\boldsymbol{\mu}_\phi(P) = \mathbb{E}_{x\sim P}[\phi(x)] \in \mathcal{H}$ is known as the (kernel) *mean embedding* of P, and is guaranteed to exist if $\mathbb{E}_{x\sim P}\sqrt{k(x,x)} < \infty$ (Smola et al., 2007). If k_ϕ is *characteristic* (Sriperumbudur et al., 2011), then $P \mapsto \boldsymbol{\mu}_\phi(P)$ is injective, so $\mathrm{MMD}_{k_\phi}(P,Q) = 0$ if and only if $P = Q$.

For a sample set $\mathcal{D} = \{\mathbf{x}_i\}_{i=1}^m \sim P^m$, the empirical mean embedding $\boldsymbol{\mu}_\phi(\mathcal{D}) = \frac{1}{m}\sum_{i=1}^m \phi(\mathbf{x}_i)$ is the "plug-in" estimator of $\boldsymbol{\mu}_\phi(P)$ using the empirical distribution of $\mathcal{D}$. Given $\tilde{\mathcal{D}} = \{\tilde{\mathbf{x}}_i\}_{i=1}^n \sim Q^n$, we can estimate $\mathrm{MMD}_{k_\phi}(P,Q)$ as the distance between empirical mean embeddings,

$$\mathrm{MMD}_{k_\phi}(\mathcal{D}, \tilde{\mathcal{D}}) = \left\| \frac{1}{m}\sum_{i=1}^m \phi(\mathbf{x}_i) - \frac{1}{n}\sum_{i=1}^n \phi(\tilde{\mathbf{x}}_i) \right\|_{\mathcal{H}}. \tag{1}$$

We would like to minimize the distance between a target data distribution P (based on samples $\mathcal{D}$) and the output distribution $Q_{g_{\boldsymbol{\theta}}}$ of a generator network $g_{\boldsymbol{\theta}}$. If the feature map is finite-dimensional and norm-bounded, following Harder et al. (2021); Vinaroz et al. (2022), we can privatize the mean embedding of the data distribution $\boldsymbol{\mu}_\phi(\mathcal{D})$ with a known DP mechanism such as the Gaussian or Laplace mechanisms, to be discussed shortly. As the summary of the real data does not change over the course of a generator training, we only need to privatize $\boldsymbol{\mu}_\phi(\mathcal{D})$ once.

Differential privacy A mechanism $\mathcal{M}$ is (ϵ, δ)-DP for a given $\epsilon \geq 0$ and $\delta \geq 0$ if and only if

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

for all possible sets of the mechanism's outputs S and all neighbouring datasets $\mathcal{D}$, $\mathcal{D}'$ that differ by a single entry. One of the most well-known and widely used DP mechanisms is the *Gaussian mechanism*. The Gaussian mechanism adds a calibrated level of noise to a function $\mu : \mathcal{D} \mapsto \mathbb{R}^p$ to ensure that the output of the mechanism is (ϵ, δ)-DP: $\widetilde{\mu}(\mathcal{D}) = \mu(\mathcal{D}) + n$, where $n \sim \mathcal{N}(0, \sigma^2\Delta_\mu^2\mathbf{I}_p)$. Here, σ is often called a privacy parameter, which is a function[2] of ϵ and δ. Δ_μ is often called the *global sensitivity* (Dwork et al., 2006), which is the maximum difference in L_2-norm given two neighbouring $\mathcal{D}$ and $\mathcal{D}'$, $||\mu(\mathcal{D}) - \mu(\mathcal{D}')||_2$. In this paper, we will use the Gaussian mechanism to ensure the mean embedding of the data distribution is DP.

3 METHOD

In this paper, to transfer knowledge from public to private data distributions, we construct a particular kernel k_Φ to use in Equation 1 based on *perceptual features* (PFs).

3.1 MMD with perceptual features as a feature map

We call our proposed method *Differentially Private Mean Embeddings with Perceptual Features (DP-MEPF)*, analogous to the related method DP-MER**R**F (Harder et al., 2021). We use high-dimensional, over-complete perceptual features from a feature extractor network pre-trained on a public dataset, as illustrated in **Step 1** of Figure 1. Given a vector input $\mathbf{x}$, the pre-trained feature extractor network outputs the perceptual features from each layer, where the jth layer's PF is denoted by $\mathbf{e}_j(\mathbf{x})$. Each of the J layers' perceptual features is of a different length, $\mathbf{e}_j(\mathbf{x}) \in \mathbb{R}^{d_j}$; the total dimension of the perceptual feature vector is $D = \sum_{j=1}^J d_j$.

As illustrated in **Step 2** in Figure 1, we use those PFs to form our feature map $\Phi(\mathbf{x}) := [\boldsymbol{\phi}_1(\mathbf{x}), \boldsymbol{\phi}_2(\mathbf{x})]$, where the first part comes from a concatenation of PFs from all the layers: $\boldsymbol{\phi}_1(\mathbf{x}) = [\mathbf{e}_1(\mathbf{x}), \cdots, \mathbf{e}_J(\mathbf{x})]$, while the second part comes from their squared values: $\boldsymbol{\phi}_2(\mathbf{x}) = [\mathbf{e}_1^2(\mathbf{x}), \cdots, \mathbf{e}_J^2(\mathbf{x})]$, where $\mathbf{e}_j^2(\mathbf{x})$ means each entry of $\mathbf{e}_j(\mathbf{x})$ is squared. Using this feature map, we then construct the mean embedding of a data distribution given the data samples $\mathcal{D} = \{\mathbf{x}_i\}_{i=1}^m$:

$$\boldsymbol{\mu}_P(\mathcal{D}) = \begin{bmatrix} \boldsymbol{\mu}_P^{\boldsymbol{\phi}_1}(\mathcal{D}) \\ \boldsymbol{\mu}_P^{\boldsymbol{\phi}_2}(\mathcal{D}) \end{bmatrix} = \begin{bmatrix} \frac{1}{m}\sum_{i=1}^m \boldsymbol{\phi}_1(\mathbf{x}_i) \\ \frac{1}{m}\sum_{i=1}^m \boldsymbol{\phi}_2(\mathbf{x}_i) \end{bmatrix}. \tag{2}$$

Lastly (**Step 3** in Figure 1), we will train a generator $g_{\boldsymbol{\theta}}$ that maps latent vectors $\mathbf{z}_i \sim \mathcal{N}(0, I)$ to a synthetic data sample $\tilde{\mathbf{x}}_i = g_{\boldsymbol{\theta}}(\mathbf{z}_i)$; we need to find good parameters $\boldsymbol{\theta}$ for the generator. In non-private settings, we estimate the generator's

[2]The relationship can be numerically computed by packages like `auto-dp` (Wang et al., 2019), among other methods.

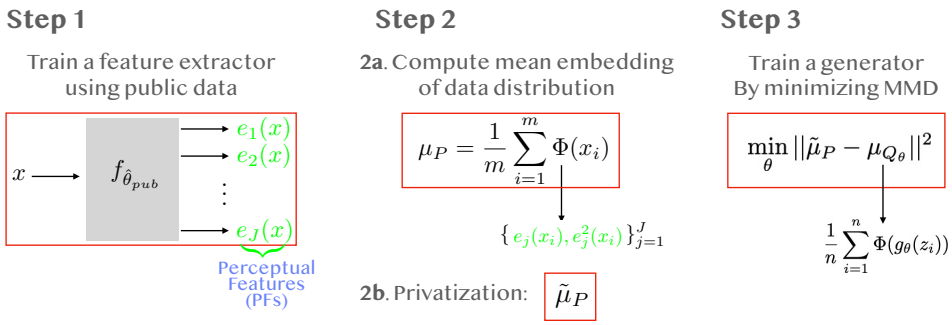


Figure 1: Three steps in *differentially private mean embedding with perceptual features (DP-MEPF)*. **Step 1:** We train a feature extractor neural network, $f_{\hat{\theta}_{pub}}$, using public data. This is a function of public data, with no privacy cost to train. A trained $f_{\hat{\theta}_{pub}}$ maps an input $\mathbf{x}$ to perceptual features (in green), the outputs of each layer. **Step 2:** We compute the mean embedding of the data distributions using a feature map consisting of the first and second moments (in green) of the perceptual features, and privatize it based on the Gaussian mechanism (see text). **Step 3:** We train a generator g_θ, which produces synthetic data from latent codes $z_i \sim \mathcal{N}(0, I)$, by minimizing the privatized MMD.

parameters by minimizing an estimate of $\mathrm{MMD}^2_{k_\Phi}(P, Q_{g_{\boldsymbol{\theta}}})$, using $\tilde{\mathcal{D}} = \{\tilde{\mathbf{x}}_i\}$ in Equation 1, similar to Dziugaite et al. (2015); Li et al. (2015); dos Santos et al. (2019). In private settings, we privatize $\mathcal{D}$'s mean embedding to $\tilde{\boldsymbol{\mu}}_\phi(\mathcal{D})$ with the Gaussian mechanism (details below), and minimize

$$\widetilde{\mathrm{MMD}}^2_{k_\Phi}(\mathcal{D}, \tilde{\mathcal{D}}) = \left\|\tilde{\boldsymbol{\mu}}_\phi(\mathcal{D}) - \boldsymbol{\mu}_\phi(\tilde{\mathcal{D}})\right\|^2. \tag{3}$$

A natural question that arises is whether the MMD using the PFs is a metric: if $\mathrm{MMD}_{k_\Phi}(P, Q) = 0$ only if $P = Q$. As PFs have a finite-dimensional embedding, we in fact know this cannot be the case (Sriperumbudur et al., 2011). Thus, there exists *some* pair of distributions which our MMD cannot distinguish. However, given that linear functions in perceptual feature spaces can obtain excellent performance on nearly any natural image task (as observed in transfer learning), it seems that PFs are "nearly" universal for natural distributions of images (dos Santos et al., 2019). Thus we expect the MMD with this kernel to do a good job of distinguishing "natural" distributions from one another, though the possibility of "adversarial attacks" perhaps remains.

A more important question in our context is whether this MMD serves as a good loss for training a generator, and whether the resulting synthetic data samples are reasonably faithful to the original data samples. Our experiments in Section 6, as well as earlier work by dos Santos et al. (2019) in non-private settings, imply that it is.

Privatization of mean embedding We privatize the mean embedding of the data distribution only once, and reuse it repeatedly during the training of the generator $g_{\boldsymbol{\theta}}$. We use the Gaussian mechanism to separately privatize the first and second parts of the feature map. We normalize each type of perceptual features such that $\|\boldsymbol{\phi}_1(\mathbf{x}_i)\|_2 = 1$ and $\|\boldsymbol{\phi}_2(\mathbf{x}_i)\|_2 = 1$ for each sample $\mathbf{x}_i$. After this change, the sensitivity of each part of the mean embedding is

$$\max_{\mathcal{D},\mathcal{D}' \text{ s.t. } |\mathcal{D}-\mathcal{D}'|=1} \|\boldsymbol{\mu}_{\boldsymbol{\phi}_t}(\mathcal{D}) - \boldsymbol{\mu}_{\boldsymbol{\phi}_t}(\mathcal{D}')\|_2 \leq \tfrac{2}{m}, \tag{4}$$

where $\boldsymbol{\mu}_{\boldsymbol{\phi}_t}(\mathcal{D})$ denotes the two parts of the mean embedding for $t = 1, 2$. Using these sensitivities, we add Gaussian noise to each part of the mean embedding, obtaining

$$\tilde{\boldsymbol{\mu}}_\Phi(\mathcal{D}) = \begin{bmatrix}\tilde{\boldsymbol{\mu}}_{\boldsymbol{\phi}_1}(\mathcal{D}) \\ \tilde{\boldsymbol{\mu}}_{\boldsymbol{\phi}_2}(\mathcal{D})\end{bmatrix} = \begin{bmatrix}\frac{1}{m}\sum_{i=1}^m \boldsymbol{\phi}_1(\mathbf{x}_i) + \mathbf{n}_1 \\ \frac{1}{m}\sum_{i=1}^m \boldsymbol{\phi}_2(\mathbf{x}_i) + \mathbf{n}_2\end{bmatrix}, \tag{5}$$

where $\mathbf{n}_t \sim \mathcal{N}(0, \frac{4\sigma^2}{m^2} I)$ for $t = 1, 2$.

Since we are using the Gaussian mechanism twice, we simply compose the privacy losses from each mechanism. More precisely, given a desired privacy level ϵ, δ, we use the package of Wang et al. (2019) to find the corresponding σ for the two Gaussian mechanisms.

Labeled data generation Extending our framework to generate both labels and input images is straightforward. As done by Harder et al. (2021), we construct a separate mean embedding for each class-conditional input distribution and then concatenate them into a single embedding

$$\tilde{\boldsymbol{\mu}}_{\boldsymbol{\phi}_t}(\mathcal{D}) = \left[\frac{1}{m}\sum_{i\in C_1}\boldsymbol{\phi}_t(\mathbf{x}_i) + \mathbf{n}_{t,1} \quad \cdots \quad \frac{1}{m}\sum_{i\in C_K}\boldsymbol{\phi}_t(\mathbf{x}_i) + \mathbf{n}_{t,K}\right]^\top, \tag{6}$$

where K is the number of classes and $C_k = \{i \in [m] | y_i = k\}$ is the set of indices belonging to class k. As a result, the size of the final mean embedding is $D \times K$ (number of perceptual features by the number of classes) if we use only the first moment, or $2 \times D \times K$ if we use the first two moments. This is exactly the conditional mean embedding with a discrete kernel on the class label (Song et al., 2013). In the case of imbalanced data, an estimate of the label distribution can be obtained at low privacy cost with a DP release of the class counts, as done in Harder et al. (2021). Since all datasets considered in this paper are balanced, this step is not necessary in our experiments.

3.2 Differentially private early stopping

On some datasets (CelebA and Cifar10) we observe that the generated sample quality deteriorates if the model is trained for too many iterations in high-privacy settings. This is indicated by a steady increase in FID score (Heusel et al., 2017), and likely due to overfitting to the static noisy embedding. Since the FID score is based on the training data, simply choosing the iteration with the best FID score after training has completed would violate privacy.

Privatizing the FID score requires privatizing the covariance of the output of the final pooling layer in the Inception network, which is quite sensitive. Instead, we privatize the first and second moment of data embeddings as in Equation 2, but using only the output of the final pooling layer in the Inception network. We then use this quantity as a private proxy for FID, and select the iteration with the lowest score. To minimize the privacy cost, we choose a larger noise parameter than for the main objective: $\sigma_{stopping} = 10\sigma$, where σ is the noise scale for privatizing each part of the data mean embeddings, works well. Again, we compose these σs with the analysis of Wang et al. (2019).

4 THEORETICAL ANALYSIS

We now bound the effect of adding noise to our loss function, showing that asymptotically our noise does not hurt the rate at which our model converges to the optimal model.

Appendix A proves full finite-sample versions of all of the following bounds, which are stated here using $\mathcal{O}_p$ notation for simplicity. The statment $X = \mathcal{O}_p(A_n)$ essentially means that X is $\mathcal{O}(A_n)$ with probability at least $1 - \rho$ for any constant choice of failure probability $\rho > 0$.

The full version in the supplementary material is also ambivalent to the choice of covariance for the noise variable $\mathbf{n}$, allowing in particular analysis of DP-MEPF based either on one or two moments of PFs. (The full version gives a slightly more refined treatment of the two-moment case, but the difference is typically not asymptotically relevant.)

To begin, we use standard results on Gaussians to establish that the privatized MMD is close to the non-private MMD:

Proposition 4.1. *Given datasets $\mathcal{D}$ and $\tilde{\mathcal{D}}$, the absolute difference between the privatized and non-private squared MMDs, a random function of only $\mathbf{n}$, satisfies*

$$\left|\widetilde{\mathrm{MMD}}^2_{k_\Phi}(\mathcal{D}, \tilde{\mathcal{D}}) - \mathrm{MMD}^2_{k_\Phi}(\mathcal{D}, \tilde{\mathcal{D}})\right| = \mathcal{O}_p\left(\tfrac{\sigma^2}{m^2}D + \tfrac{\sigma}{m}\,\mathrm{MMD}_{k_\Phi}(\mathcal{D}, \tilde{\mathcal{D}})\right).$$

One key quantity in the bound is σ/m, the ratio of the noise scale σ (inversely proportional to ε) to the number of observed (private) data points m. Note that σ depends only on the given privacy level, not on m, so the error becomes zero as long as $m \to \infty$. In the second term, σ/m is multiplied by the (non-private, non-squared) MMD, which is bounded for our features, but for good generators (where our optimization hopefully spends most of its time) this term will also be nearly zero. The other term accounts for adding independent noise to each of the D feature dimensions; although D is typically large, so is m^2. Having $m = 50\text{K}$ private samples, e.g. for CIFAR-10, allows for a strong error bound as long as $D\sigma^2 \ll 625\text{M}$.

The above result is for a fixed pair of datasets. Because we only add noise $\mathbf{n}$ once, across all possible comparisons, we can use this to obtain a bound uniform over all possible generator distributions, in particular implying that the minimizer of the privatized MMD approximately minimizes the original, non-private MMD:

Proposition 4.2. *Fix a target dataset $\mathcal{D}$. For each $\boldsymbol{\theta}$ in some set Θ, fix a corresponding $\tilde{\mathcal{D}}_{\boldsymbol{\theta}}$; in particular, $\Theta = \mathbb{R}^p$ could be the set of all generator parameters, and $\tilde{\mathcal{D}}_{\boldsymbol{\theta}}$ either the outcome of running a generator $g_{\boldsymbol{\theta}}$ on a fixed set of "seeds," $\tilde{\mathcal{D}}_{\boldsymbol{\theta}} = \{g_{\boldsymbol{\theta}}(\mathbf{z}_i)\}_{i=1}^n$, or the full output distribution of the generator $Q_{g_{\boldsymbol{\theta}}}$. Let $\widetilde{\boldsymbol{\theta}} \in \arg\min_{\theta\in\Theta} \widetilde{\mathrm{MMD}}^2_{k_\Phi}(\mathcal{D}, \tilde{\mathcal{D}}_\theta)$ be the private minimizer, and $\widehat{\boldsymbol{\theta}} \in \arg\min_{\theta\in\Theta} \mathrm{MMD}^2_{k_\Phi}(\mathcal{D}, \tilde{\mathcal{D}}_\theta)$ the non-private minimizer. Then $\mathrm{MMD}^2_{k_\Phi}(\mathcal{D}, \tilde{\mathcal{D}}_{\widetilde{\boldsymbol{\theta}}}) - \mathrm{MMD}^2_{k_\Phi}(\mathcal{D}, \tilde{\mathcal{D}}_{\widehat{\boldsymbol{\theta}}}) = \mathcal{O}_p\left(\frac{\sigma^2 D}{m^2} + \frac{\sigma\sqrt{D}}{m}\right)$.*

The second term of this bound will generally dominate; it arises from uniformly bounding the $\frac{\sigma}{m}\mathrm{MMD}_{k_\Phi}(\mathcal{D}, \tilde{\mathcal{D}}_{\boldsymbol{\theta}})$ term of Proposition 4.1 over all possible $\tilde{\mathcal{D}}_{\boldsymbol{\theta}}$. This approach, although the default way to prove this type of bound, misses that $\mathrm{MMD}_{k_\Phi}(\mathcal{D}, \tilde{\mathcal{D}}_{\boldsymbol{\theta}})$ is hopefully small for $\widetilde{\boldsymbol{\theta}}$ and $\widehat{\boldsymbol{\theta}}$. We can in fact take advantage of this to provide an "optimistic" rate (Srebro et al., 2010; Zhou et al., 2021) that achieves faster convergence if the generator is capable of matching the target features (an "interpolating" regime):

Proposition 4.3. *In the setting of Proposition 4.2,*

$$\mathrm{MMD}^2_{k_\Phi}(\mathcal{D}, \tilde{\mathcal{D}}_{\widetilde{\boldsymbol{\theta}}}) - \mathrm{MMD}^2_{k_\Phi}(\mathcal{D}, \tilde{\mathcal{D}}_{\widehat{\boldsymbol{\theta}}}) = \mathcal{O}_p\left(\frac{\sigma^2 D}{m^2} + \frac{\sigma\sqrt{D}}{m}\mathrm{MMD}_{k_\Phi}(\mathcal{D}, \tilde{\mathcal{D}}_{\widehat{\boldsymbol{\theta}}})\right).$$

Note that this bound implies the previous one, since $\mathrm{MMD}_{k_\Phi}(\mathcal{D}, \tilde{\mathcal{D}})$ is bounded. But in the case where the generator is capable of exactly matching the features of the target distribution, the second term becomes zero, and the rate with respect to m is greatly improved.

In either regime, our approximate minimization of the empirical MMD is far faster than the rate at which minimizing the empirical $\mathrm{MMD}(\mathcal{D}, Q_{g_{\boldsymbol{\theta}}})$ converges to minimizing the true, distribution-level $\mathrm{MMD}(P, Q_{g_{\boldsymbol{\theta}}})$: the known results there (e.g. Dziugaite et al., 2015, Theorem 1) give a $1/\sqrt{m}$ rate, compared to our $1/m$ or even $1/m^2$.

We show that minimizing DP-MEPF's loss actually pays *no* asymptotic penalty for privacy (especially when a perfect generator exists), with the privacy loss dwarfed by the statistical error for large datasets; this essentially agrees with experiments (see Section 6). This is not the case for all DP methods, and other DP generation papers didn't prove any such guarantees: DP-Sinkhorn only proved privacy, and DP-MERF showed only a much weaker guarantee (its gradient is asymptotically unbiased).

5 RELATED WORK

Initial work on differentially private data generation assumed strong constraints on the type of data and the intended use of the released data (Snoke & Slavković, 2018; Mohammed et al., 2011; Xiao et al., 2010; Hardt et al., 2012; Zhu et al., 2017). While these studies provide theoretical guarantees on the utility of the synthetic data, they typically do not scale to our goal of large-scale image data generation.

Recently, several papers focused on discrete data generation with limited domain size (Zhang et al., 2017; Qardaji et al., 2014; Chen et al., 2015; Zhang et al., 2021). These methods learn the correlation structure of small subsets of features and privatize them in order to produce differentially private synthetic data samples. These methods often require discretization of the data and have limited scalability, so are also unsuitable for high-dimensional image data generation.

More recently, however, a new line of work has emerged that adopt the core ideas from the recent advances in deep generative models for a broad applicability of synthetic data with differential privacy constraints. The majority of this work (Xie et al., 2018; Torkzadehmahani et al., 2019; Frigerio et al., 2019; Yoon et al., 2019; Chen et al., 2020) uses generative adversarial networks (GANs; Goodfellow et al., 2014) along with some form of DP-SGD (Abadi et al., 2016). Other works in this line include PATE-GAN based on the private aggregation of teacher ensembles (Papernot et al., 2017) and variational autoencoders (Acs et al., 2018).

The closest prior work to the proposed method is DP-MERF (Harder et al., 2021), where the kernel mean embeddings are constructed using random Fourier features (Rahimi & Recht, 2008). A recent variant of DP-MERF uses Hermite

polynomial-based mean embeddings (Vinaroz et al., 2022). Unlike these methods, we use the perceptual features from a pre-trained network to construct kernel mean embeddings. Neither previous method applies to the perceptual kernels used here, so their empirical results are far worse (as we'll see shortly). Our theoretical analysis is also much more extensive: they only proved a bound on the expected error between the private and non-private empirical MMD for a fixed pair of datasets.

More recently, a similar work to DP-MERF utilizes the Sinkhorn divergence for private data generation (Cao et al., 2021), which performs similarly to DP-MERF when the cost function is the L2 distance with a large regularizer. Another related work proposes to use the characteristic function and an adversarial re-weighting objective (Liew et al., 2022) in order to improve the generalization capability of DP-MERF.

A majority of these related methods were evaluated only on relatively simple datasets such as MNIST and FashionMNIST. Even so, the DP-GAN-based methods mostly require a large privacy budget of $\epsilon \approx 10$ to generate synthetic data samples that are reasonably close to the real data samples. Our method goes far beyond this quality with much more stringent privacy constraints, as we will now see.

6 EXPERIMENTS

We will now compare our method to state-of-the-art methods for DP data generation.

Table 1: Downstream accuracies by Logistic regression and MLP, evaluated on the generated data samples using MNIST and FashionMNIST as private data and SVHN and CIFAR-10 as public data, respectively. In all cases, we set $\epsilon = 10, \delta = 10^{-5}$. In our method, we used both features ϕ_1, ϕ_2.

		DP-MEPF	DP-Sinkhorn (Cao et al., 2021)	GS-WGAN (Chen et al., 2020)	DP-MERF (Harder et al., 2021)	DP-HP (Vinaroz et al., 2022)
MNIST	LogReg	**83**	83	79	79	81
	MLP	**90**	83	79	78	82
F-MNIST	LogReg	**76**	75	68	76	73
	MLP	**76**	75	65	75	71

Datasets. We considered four image datasets[3] of varying complexity. We started with the commonly used datasets MNIST (LeCun & Cortes, 2010) and FashionMNIST (Xiao et al., 2017), where each consist of 60,000 28×28 pixel grayscale images depicting hand-written digits and items of clothing, respectively, sorted into 10 classes. We also looked at the more complex CelebA (Liu et al., 2015) dataset, containing 202,599 color images of faces which we scale to sizes of 32×32 or 64×64 pixels and treat as unlabeled. We also study CIFAR-10 (Krizhevsky, 2009), a 50,000-sample dataset containing 32×32 color images of 10 classes of objects, including vehicles like ships and trucks, and animals such as horses and birds.

Implementation. We implemented our code for all the experiments in PyTorch (Paszke et al., 2019), using the `auto-dp` package[4] (Wang et al., 2019) for the privacy analysis. Following Harder et al. (2021), we used the generator that consists of two fully connected layers followed by two convolutional layers with bilinear upsampling, for generating both MNIST and FashionMNIST datasets. For MNIST, we used the SVHN dataset as public data to pre-train ResNet18 (He et al., 2016), from which we took the perceptual features. For FashionMNIST, we used perceptual features from a ResNet18 trained on CIFAR-10. For CelebA and CIFAR-10, we followed dos Santos et al. (2019) in using perceptual features from a pre-trained VGG (Simonyan & Zisserman, 2014) on ImageNet, and a ResNet18-based generator. Further implementation details are given in the supplementary material, which also studies how different public datasets and feature extractors impact the performance.

Evaluation metric. Evaluating the quality of generated data is a challenging problem of its own. We use two conventional measures. The first is the *Frechet Inception Distance (FID)* score (Heusel et al., 2017), which directly measures the quality of the generated samples. The FID score correlates with human evaluations of visual similarity to the real

[3]Dataset licenses: MNIST: CC BY-SA 3.0; FashionMNIST:MIT; CelebA: see `https://mmlab.ie.cuhk.edu.hk/projects/CelebA.html`; Cifar10: MIT

[4]`https://github.com/yuxiangw/autodp`

Table 2: Downstream accuracies of our method for MNIST and FashionMNIST at varying values of ϵ.

		MNIST				FashionMNIST			
		$\epsilon = 5$	$\epsilon = 2$	$\epsilon = 1$	$\epsilon = 0.2$	$\epsilon = 5$	$\epsilon = 2$	$\epsilon = 1$	$\epsilon = 0.2$
MLP	DP-MEPF (ϕ_1, ϕ_2)	90	89	89	80	76	75	75	70
	DP-MEPF (ϕ_1)	88	88	87	77	75	76	75	69
LogReg	DP-MEPF (ϕ_1, ϕ_2)	83	83	82	76	75	76	75	73
	DP-MEPF (ϕ_1)	81	80	79	72	75	76	76	72

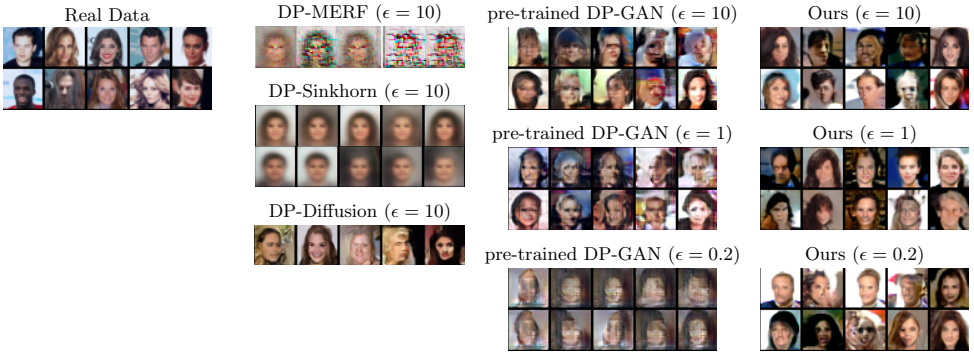


Figure 2: Synthetic 32×32 CelebA samples generated at different levels of privacy. Samples for DP-MERF and DP-Sinkhorn are taken from Cao et al. (2021) and DP-Diffusion samples are taken from Dockhorn et al. (2022). The pre-trained GAN is our baseline utilizing public data. Even at $\epsilon = 0.2$, DP-MEPF (ϕ_1, ϕ_2) yields samples of higher visual quality than the comparison methods.

data, and is commonly used in deep generative modelling. We computed FID scores with the `pytorch_fid` package (Seitzer, 2020), based on 5 000 generated samples, matching dos Santos et al. (2019). As discussed in Section 3.2, we use a private proxy for FID for early stopping, while the FID scores we report in this section are non-DP measures of our final model for fair comparison to other existing methods. The second metric we use is the accuracy of downstream classifiers, trained on generated datasets and then test on the real data test sets (used by Chen et al., 2020; Torkzadehmahani et al., 2019; Yoon et al., 2019; Chen et al., 2020; Harder et al., 2021; Cao et al., 2021). This test accuracy indicates how well the downstream classifiers generalize from the synthetic to the real data distribution and thus, the utility of using synthetic data samples instead of the real ones. We computed the downstream accuracy on MNIST and FashionMNIST using the logistic regression and MLP classifiers from scikit-learn (Pedregosa et al., 2011). For CIFAR-10, we used ResNet9 taken from FFCV[5] (Leclerc et al., 2022).

In all experiments, we tested non-private training and settings with various levels of privacy, ranging from $\epsilon = 10$ (no meaningful guarantee) to $\epsilon = 0.2$ (strong privacy guarantee). We set $\delta = 10^{-5}$ for MNIST, FashionMNIST, and Cifar10 and $\delta = 10^{-6}$ for CelebA. In DP-MEPF, we also tested cases based on embeddings with only the first moment, written (ϕ_1), and using the first two moments, written (ϕ_1, ϕ_2). Each value in all tables is an average of 3 or more runs; standard deviations are in the supplementary material.

Since we are unaware of any prior work on DP data generation for image data using auxiliary datasets, we instead mostly compare to recent methods which do not access auxiliary data. As expected, due to the advantage of non-private data our approach outperforms these methods by a significant margin on the more complex datasets. As a simple baseline based on public data, we also pretrain a GAN on a downscaled version of ImageNet, at 32×32, and fine-tune this model with DP-SGD on CelebA and Cifar10. We use architectures based on ResNet9 with group normalization (Wu & He, 2018) for both generator and discriminator. As suggested by Bie et al. (2023), we update the generator at a lower frequency than the discriminator and use increased minibatch sizes. Further details can be found in the supplementary material.

MNIST and FashionMNIST. We compare DP-MEPF to existing methods on the most common settings used in the literature, MNIST and FashionMNIST at $\epsilon = 10$, in Table 1. For an MLP on MNIST, DP-MEPF's samples far

[5] `https://github.com/libffcv/ffcv/blob/main/examples/cifar/train_cifar.py`

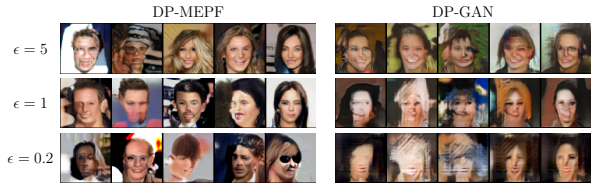


Figure 3: Synthetic 64×64 CelebA samples generated at different levels of privacy with DP-MEPF (ϕ_1, ϕ_2).

Table 3: CelebA FID scores (lower is better) for images of resolution 32×32 and 64×64. Results for DP Diffusion (DPDM) and DP Sinkhorn taken from Dockhorn et al. (2022) and Cao et al. (2021).

		$\epsilon = 10$	$\epsilon = 5$	$\epsilon = 2$	$\epsilon = 1$	$\epsilon = 0.5$	$\epsilon = 0.2$
32	DP-MEPF (ϕ_1, ϕ_2)	17.4	17.5	18.1	19.0	21.4	25.8
	DP-MEPF (ϕ_1)	16.3	16.9	16.5	17.2	21.8	25.5
	DP-GAN (pre-trained)	58.1	66.9	67.1	81.3	109.1	192.0
	DPDM (no public data)	21.2	-	-	71.8	-	-
	DP Sinkhorn (no public data)	189.5	-	-	-	-	-
64	DP-MEPF (ϕ_1, ϕ_2)	18.5	19.1	18.4	19.0	21.4	26.8
	DP-MEPF (ϕ_1)	17.4	16.5	16.9	18.4	20.4	27.7
	DP-GAN (pre-trained)	57.1	62.3	65.2	72.5	91.9	133.3

outperform other methods for logistic regression and both classifiers on FashionMNIST, scores match or slightly exceed those of existing models. This might be because the domain shift between public dataset (CIFAR-10, color images of scenes) and private dataset (FashionMNIST, grayscale images of fashion items) is too large, or because the task is simple enough that random features as found in DP-MERF or DP-HP are already good enough. This will change as we proceed to more complex datasets. Table 2 shows that downstream test accuracy only starts to drop in high privacy regimes, $\epsilon < 1$, due to the low sensitivity of $\boldsymbol{\mu}_\phi$. Samples for visual comparison between methods are included in the supplementary material.

CelebA Figure 2 shows that previous attempts to generate CelebA samples without auxiliary data using DP-MERF or DP-Sinkhorn have only managed to capture very basic features of the data. Each sample depicts a face, but offers no details or variety. DP-MEPF produces more accurate samples at the same 32×32 resolution, which is also reflected in improved FID scores of around 17, while DP-Sinkhorn, as reported in Cao et al. (2021), achieves an FID of 189.5. Table 3 gives FID scores for both resolutions at varying ϵ. DP-MEPF consistently outperforms our pre-trained DP-GAN baseline and the scores reported for DP diffusion Dockhorn et al. (2022), As the dataset has over 200 000 samples, the feature embeddings have low sensitivity, and offer similar quality between $\epsilon = 10$ and $\epsilon = 1$, although quality begins to decline at $\epsilon < 1$. Samples for 64×64 images are shown in Figure 3, with similar quality, and a quicker loss of quality in high privacy settings due to its larger embedding. In all cases, the ϕ_1 embedding yields better results than ϕ_1, ϕ_2, suggesting that the second moment does not contribute useful information, perhaps because on the limited variance of the dataset.

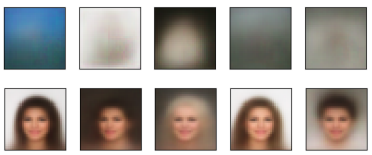


Figure 4: Samples from non-DP Sinkhorn. Top: ImageNet32. Bottom: CelebA after pretraining.

Because DP-Sinkhorn is the best-performing method without public data, we perform experiments on DP-Sinkhorn, pretraining it non-DP on ImageNet32 and fine-tuning with DP on CelebA ($\epsilon = 10$). After seeing no improvement, we tested non-DP fine-tuning and still saw no improvements beyond what is shown in Figure 4; we tried both BigGan- and ResNet18-based generators with hyperparameter grid searches. DP-Sinkhorn only compares features at image-level, without domain-specific priors, and it appears that even non-DP the method is not powerful enough to model image data beyond MNIST. (A DP-MEPF analogue that extracts features learned from public data might help, but this would be a novel method beyond scope for comparison.) DP-MERF is similarly limited by its random features, not DP noise, as shown by non-DP versions matching $\epsilon = 10$ performance.

Differentially private early stopping. For CelebA and Cifar10, we use DP early stopping as explained in Section 3.2 with a privacy parameter ten times larger than the σ used for the training objective. Keeping (ϵ, δ) fixed, this additional

Table 4: Two examples of beneficial early stopping: For CelebA at 64×64 resolution and labeled Cifar10, DP-MEPF (ϕ_1) sample quality (measured in FID) degrades with long training in high privacy settings (here $\epsilon \leq 1$). This makes the final model at the end of training a poor choice. Our DP selection of the best iteration via proxy stays close to the optimal choice.

		$\epsilon = 1$	$\epsilon = 0.5$	$\epsilon = 0.2$
CelebA 64×64	Best FID (not DP)	17.7	20.1	27.0
	DP proxy for FID	18.4	20.4	27.7
	At the end of training	18.4	22.1	45.2
Cifar10 (labeled)	Best FID (not DP)	54.8	92.0	268.3
	DP proxy for FID	56.5	92.0	268.3
	At the end of training	198.6	267.7	357.1

Figure 5: Labeled samples from DP-MEPF (ϕ_1, ϕ_2) and DP-MERF (Harder et al., 2021).

release results only in a small increase in σ, and gives us a simple way for choosing the best iteration. In Table 4, we compare the true best FID, the FID picked by our private proxy, and the FID at the end of training to illustrate the advantage in high DP settings. FID scores were computed every $5\,000$ iterations, while the model trained for $200\,000$ iterations in total.

CIFAR-10 Finally, we investigate a dataset which has not been covered in DP data generation. While CelebA depicts a centered face in every image, CIFAR-10 includes 10 visually distinct object classes, which raises the required minimum quality of samples to somewhat resemble the dataset. At only $5\,000$ samples per class, the dataset is also significantly smaller, which poses a challenge in the private setting.

Figure 5 shows that DP-MEPF is capable of producing labelled private data (generating both labels and input images together) resembling the real data, but the quality does suffer in high privacy settings. This is also reflected in the FID scores (Table 5): at $\epsilon \leq 1$ labeled DP-MEPF scores deteriorate at a much quicker rate than the unlabeled counterpart. As the unlabeled embedding dimension is smaller by a factor of 10 (the number of classes), it is easier to release privately and retains some semblance of the data even in the highest privacy settings, as shown in Figure 6. The FID scores of our pre-trained DP-GAN baseline consistently exceed our results, usually by over 10 points. These scores are better than the DP-GAN results for CelebA, likely because 32×32 ImageNet is very similar to Cifar10. Nonetheless, the high privacy cost of DP-SGD makes DP-GAN a poor fit for a dataset of this complexity and limited size.

Table 5: FID scores for synthetic CIFAR-10 data; labeled generates both labels and images.

		$\epsilon = 10$	$\epsilon = 5$	$\epsilon = 2$	$\epsilon = 1$	$\epsilon = 0.5$	$\epsilon = 0.2$
unlabeled	DP-MEPF (ϕ_1, ϕ_2)	38.8	37.0	38.7	43.0	49.4	67.3
	DP-MEPF (ϕ_1)	38.5	38.6	40.1	45.1	49.8	72.3
	DP-GAN	54.6	54.7	62.4	74.9	62.7	73.4
labeled	DP-MEPF (ϕ_1, ϕ_2)	29.1	30.0	39.5	54.0	76.4	226.0
	DP-MEPF (ϕ_1)	30.3	35.6	42.0	56.5	92.0	268.3

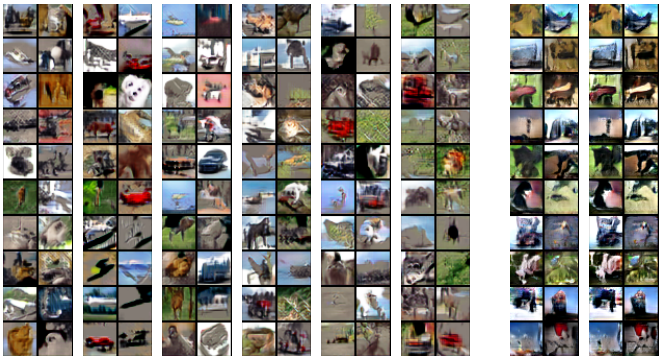


Figure 6: Unlabeled CIFAR-10 samples from DP-MEPF (ϕ_1, ϕ_2) and DP-GAN.

In Table 6 we show the test accuracy of models trained synthetic datasets applied to real data. While there is still a large gap between the 88.3% accuracy on the real data and our results, DP-MEPF achieves nontrivial results around 50% for $\epsilon = 10$, which degrade as privacy is increased. While the drop in sample quality due to high privacy is quite substantial, it is less of a problem in the unlabelled case, since our embedding dimension is smaller by a factor of 10 (the number of classes) and thus easier to release privately.

Table 6: Test accuracies (higher is better) of ResNet9 trained on CIFAR-10 synthetic data with varying privacy guarantees. When trained on real data, test accuracy is 88.3%

	$\epsilon = 10$	$\epsilon = 5$	$\epsilon = 2$	$\epsilon = 1$	$\epsilon = 0.5$	$\epsilon = 0.2$
DP-MEPF (ϕ_1, ϕ_2)	53.0	43.9	40.0	28.5	18.0	16.2
DP-MEPF (ϕ_1)	40.7	32.3	42.6	33.2	18.8	15.3
DP-MERF	13.2	13.4	13.5	13.8	13.1	10.4

7 DISCUSSION

We have demonstrated the advantage of using auxiliary public data in DP data generation. Our method DP-MEPF takes advantage of features from pre-trained classifiers that are readily available, and allows us to tackle datasets like CelebA and CIFAR-10, which have been unreachable for private data generation up to this point.

There are several avenues to extend our method in future work, in particular finding better options for the encoder features: the choice of VGG19 by dos Santos et al. (2019) works well in private settings, but a lower-dimensional embedding that still works well for training generative models – perhaps based on some kind of pruning scheme – might help reduce the sensitivity of $\boldsymbol{\mu}_\phi$ and improve quality.

Training other generative models such as GANs or VAEs with pretrained components is also exploring further than our initial attempt here. It may also be possible to take a "middle ground" and introduce some adaptation for features in DP-MEPF, to allow for more powerful, GAN-like models, without suffering too much privacy loss. In the non-private generative modelling community, this has proved important, but the challenge will be to do so while limiting the number of DP releases to allow modelling with, e.g., $\epsilon \leq 2$.

8 ACKNOWLEDGEMENTS

We thank the anonymous reviewers for their valuable time helping us improve our manuscript.

F. Harder is supported by the Max Planck Society and the Gibs Schüle Foundation and the Institutional Strategy of the University of Tübingen (ZUK63) and the German Federal Ministry of Education and Research (BMBF): Tübingen AI

Center, FKZ: 01IS18039B. F. Harder is also grateful for the support of the International Max Planck Research School for Intelligent Systems (IMPRS-IS).

M. Jalali Asadabadi, M. Park, and D. J. Sutherland are supported in part by the Natural Sciences and Engineering Research Council of Canada (NSERC) and the Canada CIFAR AI Chairs program.

9 BROADER IMPACT STATEMENT

Our work is motivated by the need for strong and scalable data privacy, which we expect will have mainly beneficial societal impact. However, our work touches on two topics, which are known to contain a risk of harmful impact on individuals and thus need to be treated with caution.

9.1 Differential privacy and fairness

Firstly, recent research has shown that DP is at odds with notions of fairness when if comes to under-represented groups in the data. For instance Chang & Shokri (2021) show that minorities are more susceptible to membership inference attacks in fair non-DP models (i.e. fairness reduces privacy) and Bagdasaryan et al. (2019) show the reverse effect: when training an unfair model with strong DP guarantees, the fairness is reduced further. The dilemma is intuitive: Fairness requires amplifying the impact of samples from minorities in the data, so they will not be ignored, while DP needs to limit the impact each individual sample can have in order to keep sensitivity low. Since its discovery, this trade-off has received attention both in works seeking a more detailed understanding (Cummings et al., 2019; Mangold et al., 2022; Esipova et al., 2022; Zhong et al., 2022; Sanyal et al., 2022) and works proposing custom approaches to DP fair machine learning (Ding et al., 2020; Xu et al., 2019; Jagielski et al., 2019; Tran et al., 2021a;b; Esipova et al., 2022). Given that the impact of DP on fairness is an active area of research and independent of our particular approach, we do not see the need to perform our own experiments on this matter.

We will, however, provide an intuition on how the problem manifests in DP-MEPF by looking at labelled data generation with significant class imbalance. Assuming an imbalanced dataset with two classes and $|C_1| = 100$ and $|C_2| = 10$, we obtain the following mean embedding:

$$\tilde{\boldsymbol{\mu}}_{\boldsymbol{\phi}_t}(\mathcal{D}) = \begin{bmatrix} \frac{1}{m} \sum_{i \in C_1} \boldsymbol{\phi}_t(\mathbf{x}_i) + \mathbf{n}_{t,1} \\ \frac{1}{m} \sum_{i \in C_2} \boldsymbol{\phi}_t(\mathbf{x}_i) + \mathbf{n}_{t,2} \end{bmatrix}. \tag{7}$$

With $\|\boldsymbol{\phi}_t(\mathbf{x}_i)\|_2 = 1$, we know that the norm of the unperturbed mean embedding for class 1, given by $\|\frac{1}{m} \sum_{i \in C_1} \boldsymbol{\phi}_t(\mathbf{x}_i)\|_2 \leq 100/110$, may be ten times as large as the maximum possible norm for the class 2 embedding $\|\frac{1}{m} \sum_{i \in C_2} \boldsymbol{\phi}_t(\mathbf{x}_i)\|_2 \leq 10/110$. Nonetheless, in order to preserve DP, both embeddings are perturbed with noise of the same magnitude, leading to a significantly worse signal-to-noise ratio for the class 2 embedding. As a result, the generative model trained on this embedding will produce more accurate samples for class 1 than for class 2.

9.2 Differential privacy with public data

The second issue regards the use of public data in DP. In a recent position paper, Tramèr et al. (2022) raise several concerns about the increasing trend of using auxiliary datasets in DP research. Their critique has two main arguments, the first being that publicly available data may still be sensitive and using such data may cause unintended privacy violations. Given that many large datasets are scraped from the internet with limited human oversight, this data may contain personal data that was released involuntarily or shared exclusively for a specific context. The authors suggest that responsible use of public data requires improved curation practices, including e.g. collection of explicit consent for data use, auditing for and removal of sensitive content, and providing channels for reporting privacy concerns.

The other main criticism raised by Tramèr et al. (2022) is that the datasets used to demonstrate the benefits of public data in DP, such as Cifar10 or ImageNet, are poorly chosen, because they are often from nearly the same distribution as the private data. In contrast, they argue, using public data in realistic application scenarios such as medical imaging would likely require considerable domain shift, since no public data close to the target domain is available. This disparity leads to overly optimistic claims, as the experiments don't actually demonstrate good performance under significant domain shift. They further point out that the quality of a DP method becomes difficult to measure if it builds on e.g. a

non-privately pre-trained model, as overall improvements may stem both from either the private and the non-private part of the method. The authors propose dedicated benchmarks for DP machine learning should be developed, in order to obtain results which are comparable and predictive of model performance in real-world applications. They also acknowledge that such benchmarks don't currently exist and their design requires careful consideration.

We agree with the authors in their analysis of the challenges facing DP machine learning research and value their proposals for future directions and experiment design. In the light of all these problems introduced by public data, one might ask whether this is at all a research direction worth pursuing. Here, we emphasize a fact that is acknowledged in the final paragraph of Tramèr et al. (2022): *"many recent works employing public data have played an important role in showing that differential privacy can be preserved for certain complex machine learning problems, without suffering devastating impacts on utility."* DP currently sees little to no practical application in machine learning, in large part because the loss of utility it causes is often unacceptable. Auxiliary public data is the best candidate for achieving sufficient utility for practical use and so, in our eyes, the potential of these approaches outweighs the complications they introduce. It is thus vital that research in DP ML with public data is pursued further.

References

Martin Abadi, Andy Chu, Ian J. Goodfellow, H. Brendan McMahan, Ilya Mironov, Kunal Talwar, and Li Zhang. Deep learning with differential privacy. In *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security*, CCS '16, pp. 308–318, New York, NY, USA, 2016. Association for Computing Machinery. ISBN 9781450341394. doi: 10.1145/2976749.2978318.

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):1109–1121, 2018.

Differential Privacy Team, Apple. Learning with privacy at scale, 2017. URL `https://machinelearning.apple.com/research/learning-with-privacy-at-scale`.

Eugene Bagdasaryan, Omid Poursaeed, and Vitaly Shmatikov. Differential privacy has disparate impact on model accuracy. *Advances in neural information processing systems*, 32, 2019.

Alex Bie, Gautam Kamath, and Guojun Zhang. Private GANs, revisited. *arXiv preprint arXiv:2302.02936*, 2023.

S. Boucheron, G. Lugosi, and P. Massart. *Concentration inequalities: A nonasymptotic theory of independence*. Oxford University Press, 2013.

Tianshi Cao, Alex Bie, Arash Vahdat, Sanja Fidler, and Karsten Kreis. Don't generate me: Training differentially private generative models with sinkhorn divergence. In *Neural Information Processing Systems (NeurIPS)*, 2021.

Hongyan Chang and Reza Shokri. On the privacy risks of algorithmic fairness. In *2021 IEEE European Symposium on Security and Privacy (EuroS&P)*, pp. 292–303. IEEE, 2021.

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 33*, 2020.

Rui Chen, Qian Xiao, Yu Zhang, and Jianliang Xu. Differentially private high-dimensional data publication via sampling-based inference. In *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 129–138, 2015.

Rachel Cummings, Varun Gupta, Dhamma Kimpara, and Jamie Morgenstern. On the compatibility of privacy and fairness. In *Adjunct Publication of the 27th Conference on User Modeling, Adaptation and Personalization*, pp. 309–315, 2019.

Jiahao Ding, Xinyue Zhang, Xiaohuan Li, Junyi Wang, Rong Yu, and Miao Pan. Differentially private and fair classification via calibrated functional mechanism. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pp. 622–629, 2020.

Tim Dockhorn, Tianshi Cao, Arash Vahdat, and Karsten Kreis. Differentially private diffusion models. *arXiv preprint arXiv:2210.09929*, 2022.

Cícero Nogueira dos Santos, Youssef Mroueh, Inkit Padhi, and Pierre L. Dognin. Learning implicit generative models by matching perceptual features. In *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 4460–4469. IEEE, 2019. doi: 10.1109/ICCV.2019.00456.

Cynthia Dwork, Krishnaram Kenthapadi, Frank McSherry, Ilya Mironov, and Moni Naor. Our data, ourselves: Privacy via distributed noise generation. In *Advances in Cryptology - EUROCRYPT 2006, 25th Annual International Conference on the Theory and Applications of Cryptographic Techniques*, volume 4004 of *Lecture Notes in Computer Science*, pp. 486–503. Springer, 2006. doi: 10.1007/11761679_29.

Gintare Karolina Dziugaite, Daniel M. Roy, and Zoubin Ghahramani. Training generative neural networks via Maximum Mean Discrepancy optimization. In *UAI*, 2015.

Maria S Esipova, Atiyeh Ashari Ghomi, Yaqiao Luo, and Jesse C Cresswell. Disparate impact in differential privacy from gradient misalignment. *arXiv preprint arXiv:2206.07737*, 2022.

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 *ICT Systems Security and Privacy Protection - 34th IFIP TC 11 International Conference, SEC 2019, Lisbon, Portugal, June 25-27, 2019, Proceedings*, pp. 151–164, 2019. doi: 10.1007/978-3-030-22312-0_11.

I. J. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio. Generative adversarial networks. In *Advances in Neural Information Processing Systems*, 2014.

Arthur Gretton, Karsten M Borgwardt, Malte J Rasch, Bernhard Schölkopf, and Alexander Smola. A kernel two-sample test. *Journal of Machine Learning Research*, 13(Mar):723–773, 2012.

Frederik Harder, Kamil Adamczewski, and Mijung Park. DP-MERF: Differentially private mean embeddings with random features for practical privacy-preserving data generation. In *AISTATS*, volume 130 of *Proceedings of Machine Learning Research*, pp. 1819–1827. PMLR, 2021.

Moritz Hardt, Katrina Ligett, and Frank Mcsherry. A simple and practical algorithm for differentially private data release. In *Advances in Neural Information Processing Systems 25*, pp. 2339–2347. Curran Associates, Inc., 2012.

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*, pp. 770–778, 2016.

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. *Advances in Neural Information Processing Systems*, 30, 2017.

Matthew Jagielski, Michael Kearns, Jieming Mao, Alina Oprea, Aaron Roth, Saeed Sharifi-Malvajerdi, and Jonathan Ullman. Differentially private fair learning. In *International Conference on Machine Learning*, pp. 3000–3008. PMLR, 2019.

Alex Krizhevsky. Learning multiple layers of features from tiny images. 2009.

Beatrice Laurent and Pascal Massart. Adaptive estimation of a quadratic functional by model selection. *Annals of Statistics*, pp. 1302–1338, 2000.

Guillaume Leclerc, Andrew Ilyas, Logan Engstrom, Sung Min Park, Hadi Salman, and Aleksander Madry. ffcv. `https://github.com/libffcv/ffcv/`, 2022.

Yann LeCun and Corinna Cortes. MNIST handwritten digit database. 2010. URL `http://yann.lecun.com/exdb/mnist/`.

Yujia Li, Kevin Swersky, and Richard S. Zemel. Generative moment matching networks. In *ICML*, 2015.

Seng Pei Liew, Tsubasa Takahashi, and Michihiko Ueno. PEARL: Data synthesis via private embeddings and adversarial reconstruction learning. In *International Conference on Learning Representations*, 2022.

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)*, December 2015.

Paul Mangold, Michaël Perrot, Aurélien Bellet, and Marc Tommasi. Differential privacy has bounded impact on fairness in classification. 2022.

Noman Mohammed, Rui Chen, Benjamin C.M. Fung, and Philip S. Yu. Differentially private data release for data mining. In *Proceedings of the 17th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '11, pp. 493–501, New York, NY, USA, 2011. ACM. ISBN 978-1-4503-0813-7. doi: 10.1145/2020408.2020487.

National Conference of State Legislatures. Differential privacy for census data, 2021. URL `https://www.ncsl.org/research/redistricting/differential-privacy-for-census-data-explained.aspx`.

Nicolas Papernot, Martín Abadi, Úlfar Erlingsson, Ian J. Goodfellow, and Kunal Talwar. Semi-supervised knowledge transfer for deep learning from private training data. In *Proceedings of the International Conference on Learning Representations (ICLR)*, 2017.

Mijung Park, James Foulds, Kamalika Choudhary, and Max Welling. DP-EM: Differentially Private Expectation Maximization. In *AISTATS*, volume 54 of *Proceedings of Machine Learning Research*, pp. 896–904, Fort Lauderdale, FL, USA, April 2017. PMLR.

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. Pytorch: An imperative style, high-performance deep learning library. In *Advances in Neural Information Processing Systems 32*, pp. 8024–8035. Curran Associates, Inc., 2019.

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*, 12:2825–2830, 2011.

Wahbeh Qardaji, Weining Yang, and Ninghui Li. Priview: practical differentially private release of marginal contingency tables. In *Proceedings of the 2014 ACM SIGMOD international conference on Management of data*, pp. 1435–1446, 2014.

Ali Rahimi and Benjamin Recht. Random features for large-scale kernel machines. In *Advances in Neural Information Processing Systems*, pp. 1177–1184, 2008.

Amartya Sanyal, Yaxi Hu, and Fanny Yang. How unfair is private learning? In *Uncertainty in Artificial Intelligence*, pp. 1738–1748. PMLR, 2022.

Maximilian Seitzer. pytorch-fid: FID Score for PyTorch. `https://github.com/mseitzer/pytorch-fid`, August 2020. Version 0.2.1.

Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. 2014.

A. Smola, A. Gretton, L. Song, and B. Schölkopf. A Hilbert space embedding for distributions. In *ALT*, pp. 13–31, 2007.

Joshua Snoke and Aleksandra Slavković. pmse mechanism: differentially private synthetic data with maximal distributional similarity. In *International Conference on Privacy in Statistical Databases*, pp. 138–159. Springer, 2018.

Le Song, Kenji Fukumizu, and Arthur Gretton. Kernel embeddings of conditional distributions: A unified kernel framework for nonparametric inference in graphical models. *IEEE Signal Processing Magazine*, 30(4):98–111, 2013.

Nathan Srebro, Karthik Sridharan, and Ambuj Tewari. Optimistic rates for learning with a smooth loss, 2010.

Bharath K Sriperumbudur, Kenji Fukumizu, and Gert RG Lanckriet. Universality, characteristic kernels and rkhs embedding of measures. *Journal of Machine Learning Research*, 12(7), 2011.

Reihaneh Torkzadehmahani, Peter Kairouz, and Benedict Paten. Dp-cgan: Differentially private synthetic data and label generation. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR) Workshops*, June 2019.

Florian Tramèr and Dan Boneh. Differentially private learning needs better features (or much more data). In *International Conference on Learning Representations*, 2021.

Florian Tramèr, Gautam Kamath, and Nicholas Carlini. Considerations for differentially private learning with large-scale public pretraining. *arXiv preprint arXiv:2212.06470*, 2022.

Cuong Tran, My Dinh, and Ferdinando Fioretto. Differentially private empirical risk minimization under the fairness lens. *Advances in Neural Information Processing Systems*, 34:27555–27565, 2021a.

Cuong Tran, Ferdinando Fioretto, and Pascal Van Hentenryck. Differentially private and fair deep learning: A lagrangian dual approach. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pp. 9932–9939, 2021b.

Margarita Vinaroz, Mohammad-Amin Charusaie, Frederik Harder, Kamil Adamczewski, and Mi Jung Park. Hermite polynomial features for private data generation. In *ICML*, volume 162 of *Proceedings of Machine Learning Research*, pp. 22300–22324. PMLR, 2022.

Yu-Xiang Wang, Borja Balle, and Shiva Prasad Kasiviswanathan. Subsampled rényi differential privacy and analytical moments accountant. In *AISTATS*, 2019.

Yuxin Wu and Kaiming He. Group normalization. In *Proceedings of the European conference on computer vision (ECCV)*, pp. 3–19, 2018.

Han Xiao, Kashif Rasul, and Roland Vollgraf. Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms. 2017.

Yonghui Xiao, Li Xiong, and Chun Yuan. Differentially private data release through multidimensional partitioning. In *Secure Data Management*, pp. 150–168, Berlin, Heidelberg, 2010. Springer Berlin Heidelberg. ISBN 978-3-642-15546-8.

Liyang Xie, Kaixiang Lin, Shu Wang, Fei Wang, and Jiayu Zhou. Differentially private generative adversarial network. 2018.

Depeng Xu, Shuhan Yuan, and Xintao Wu. Achieving differential privacy and fairness in logistic regression. In *Companion proceedings of The 2019 world wide web conference*, pp. 594–599, 2019.

Jinsung Yoon, James Jordon, and Mihaela van der Schaar. PATE-GAN: Generating synthetic data with differential privacy guarantees. In *International Conference on Learning Representations*, 2019.

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):1–41, 2017.

Zhikun Zhang, Tianhao Wang, Ninghui Li, Jean Honorio, Michael Backes, Shibo He, Jiming Chen, and Yang Zhang. Privsyn: Differentially private data synthesis. In *30th USENIX Security Symposium (USENIX Security 21)*, 2021.

Da Zhong, Haipei Sun, Jun Xu, Neil Gong, and Wendy Hui Wang. Understanding disparate effects of membership inference attacks and their countermeasures. In *Proceedings of the 2022 ACM on Asia Conference on Computer and Communications Security*, pp. 959–974, 2022.

Lijia Zhou, Frederic Koehler, Danica J. Sutherland, and Nathan Srebro. Optimistic rates: A unifying theory for interpolation learning and regularization in linear regression, 2021.

T. Zhu, G. Li, W. Zhou, and P. S. Yu. Differentially private data publishing and analysis: A survey. *IEEE Transactions on Knowledge and Data Engineering*, 29(8):1619–1638, August 2017. ISSN 1041-4347. doi: 10.1109/TKDE.2017.2697856.

Supplementary Material

A Proofs

We will conduct our analysis in terms of general noise covariance Σ for the added noise, $\mathbf{n} \sim \mathcal{N}(0, \Sigma)$. The results will depend on various norms of Σ, as well as $\|\Sigma^{1/2}\mathbf{a}\|$, where $\mathbf{a} = \boldsymbol{\mu}_\phi(\mathcal{D}) - \boldsymbol{\mu}_\phi(\tilde{\mathcal{D}})$ is the difference between empirical mean embeddings $\boldsymbol{\mu}_\phi(\mathcal{D}) = \frac{1}{|\mathcal{D}|}\sum_{\mathbf{x}\in\mathcal{D}}\phi(\mathbf{x})$. (Recall that $\mathrm{MMD}(\mathcal{D}, \tilde{\mathcal{D}}) = \|\mathbf{a}\|$.)

When we use only normalized first-moment features, the quantities appearing in the bounds are

$$\begin{gathered}\Sigma = \frac{4\sigma^2}{m^2} I_D \\ \|\Sigma\|_{op} = \frac{4\sigma^2}{m^2} \qquad \|\Sigma\|_F = \frac{4\sigma^2}{m^2}\sqrt{D} \qquad \mathrm{Tr}(\Sigma) = \frac{4\sigma^2}{m^2} D \\ \|\Sigma^{1/2}\mathbf{a}\|_2 = \sqrt{\mathbf{a}^\top \Sigma \mathbf{a}} = \frac{2\sigma}{m}\,\mathrm{MMD}_{k_{\boldsymbol{\phi}}}(\mathcal{D}, \tilde{\mathcal{D}}).\end{gathered} \tag{8}$$

When we use first- and second-moment features with respective scales C_1 and C_2 (both 1 in our experiments here), we have

$$\begin{gathered}\Sigma = \begin{bmatrix} \sigma^2 \left(\frac{2C_1}{m}\right)^2 I_D & 0 \\ 0 & \sigma^2 \left(\frac{2C_2}{m}\right)^2 I_D \end{bmatrix} = \frac{4\sigma^2}{m^2}\begin{bmatrix} C_1^2 I_D & 0 \\ 0 & C_2^2 I_D \end{bmatrix} \\ \|\Sigma\|_{op} = \frac{4\sigma^2}{m^2}\max(C_1^2, C_2^2) \quad \|\Sigma\|_F = \frac{4\sigma^2}{m^2}(C_1^2 + C_2^2)\sqrt{D} \quad \mathrm{Tr}(\Sigma) = \frac{4\sigma^2}{m^2}(C_1^2 + C_2^2)\, D \\ \|\Sigma^{1/2}\mathbf{a}\|_2 = \sqrt{\mathbf{a}^\top \Sigma \mathbf{a}} = \frac{2\sigma}{m}\sqrt{C_1^2\,\mathrm{MMD}\,k_{\boldsymbol{\phi}_1}(\mathcal{D}, \tilde{\mathcal{D}})^2 + C_2^2\,\mathrm{MMD}\,k_{\boldsymbol{\phi}_2}(\mathcal{D}, \tilde{\mathcal{D}})^2}.\end{gathered} \tag{9}$$

Note that if $C_1 = C_2 = C$, then

$$\sqrt{C_1^2\,\mathrm{MMD}\,k_{\boldsymbol{\phi}_1}(\mathcal{D}, \tilde{\mathcal{D}})^2 + C_2^2\,\mathrm{MMD}\,k_{\boldsymbol{\phi}_2}(\mathcal{D}, \tilde{\mathcal{D}})^2} = C\,\mathrm{MMD}_{k_\Phi}(\mathcal{D}, \tilde{\mathcal{D}}).$$

A.1 Mean absolute error of loss function

Proposition A.1. *Given datasets $\mathcal{D} = \{\mathbf{x}_i\}_{i=1}^m$ and $\tilde{\mathcal{D}} = \{\tilde{\mathbf{x}}_j\}_{j=1}^n$ and a kernel k_ϕ with a D-dimensional embedding ϕ, let $\mathbf{a} = \mu_\phi(\mathcal{D}) - \mu_\phi(\tilde{\mathcal{D}})$. Define $\widetilde{\mathrm{MMD}}^2_{k_\Phi}(\mathcal{D}, \tilde{\mathcal{D}}) = \|\mathbf{a} + \mathbf{n}\|^2$ for a noise vector $\mathbf{n} \sim \mathcal{N}(0, \Sigma)$. Introducing the noise $\mathbf{n}$ affects the expected absolute error as*

$$\mathbb{E}_{\mathbf{n}}\left[\left|\widetilde{\mathrm{MMD}}^2_{k_\Phi}(\mathcal{D}, \tilde{\mathcal{D}}) - \mathrm{MMD}^2_{k_\Phi}(\mathcal{D}, \tilde{\mathcal{D}})\right|\right] \leq \mathrm{Tr}(\Sigma) + 2\sqrt{\frac{2}{\pi}}\|\Sigma^{1/2}\mathbf{a}\|. \tag{10}$$

Proof. We have that

$$\begin{aligned}&\mathbb{E}_{\mathbf{n}}\left[\left|\widetilde{\mathrm{MMD}}^2_{k_\Phi}(\mathcal{D}, \tilde{\mathcal{D}}) - \mathrm{MMD}^2_{k_\Phi}(\mathcal{D}, \tilde{\mathcal{D}})\right|\right] \\ &\qquad = \mathbb{E}_{\mathbf{n}}\left[\left|\,\|\mathbf{a}+\mathbf{n}\|^2 - \|\mathbf{a}\|^2\,\right|\right] = \mathbb{E}_{\mathbf{n}}\left[\left|\mathbf{n}^\top\mathbf{n} + 2\mathbf{n}^\top\mathbf{a}\right|\right] \leq \mathbb{E}_{\mathbf{n}}\left[\mathbf{n}^\top\mathbf{n}\right] + 2\,\mathbb{E}_{\mathbf{n}}\left[\left|\mathbf{n}^\top\mathbf{a}\right|\right].\end{aligned} \tag{11}$$

The first term is standard:

$$\mathbb{E}\,\mathbf{n}^\top\mathbf{n} = \mathbb{E}\,\mathrm{Tr}(\mathbf{n}^\top\mathbf{n}) = \mathbb{E}\,\mathrm{Tr}(\mathbf{n}\mathbf{n}^\top) = \mathrm{Tr}(\mathbb{E}\,\mathbf{n}\mathbf{n}^\top) = \mathrm{Tr}(\Sigma).$$

For the second, note that

$$\mathbf{a}^\top\mathbf{n} \sim \mathcal{N}(0, \mathbf{a}^\top\Sigma\mathbf{a}),$$

and so its absolute value is $\sqrt{\mathbf{a}^\top\Sigma\mathbf{a}}$ times a $\chi(1)$ random variable. Since the mean of a $\chi(1)$ distribution is $\frac{\sqrt{2}\,\Gamma(1)}{\Gamma(1/2)} = \sqrt{\frac{2}{\pi}}$, we obtain the desired bound. □

A.2 High-probability bound on the error

Proposition A.2. *Given datasets $\mathcal{D} = \{\mathbf{x}_i\}_{i=1}^m$ and $\tilde{\mathcal{D}} = \{\tilde{\mathbf{x}}_j\}_{j=1}^n$, let $\mathbf{a} = \mu_\phi(\mathcal{D}) - \mu_\phi(\tilde{\mathcal{D}})$, and define $\widetilde{\mathrm{MMD}}^2_{k_\Phi}(\mathcal{D}, \tilde{\mathcal{D}}) = \|\mathbf{a} + \mathbf{n}\|^2$ for a noise vector $\mathbf{n} \sim \mathcal{N}(0, \Sigma)$. Then for any $\rho \in (0, 1)$, it holds with probability at least $1 - \rho$ over the choice of $\mathbf{n}$ that*

$$\left| \widetilde{\mathrm{MMD}}^2_{k_\Phi}(\mathcal{D}, \tilde{\mathcal{D}}) - \mathrm{MMD}^2_{k_\Phi}(\mathcal{D}, \tilde{\mathcal{D}}) \right| \leq \mathrm{Tr}(\Sigma) + \sqrt{\tfrac{2}{\pi}} \|\Sigma^{\frac{1}{2}}\mathbf{a}\|_2 + 2\left(\|\Sigma\|_F + \sqrt{2}\|\Sigma^{\frac{1}{2}}\mathbf{a}\|_2\right)\sqrt{\log(\tfrac{2}{\rho})} + 2\|\Sigma\|_{op}\log(\tfrac{2}{\rho}). \quad (12)$$

This implies that

$$\left| \widetilde{\mathrm{MMD}}^2_{k_\Phi}(\mathcal{D}, \tilde{\mathcal{D}}) - \mathrm{MMD}^2_{k_\Phi}(\mathcal{D}, \tilde{\mathcal{D}}) \right| = \mathcal{O}_p\left(\mathrm{Tr}(\Sigma) + \|\Sigma^{1/2}\mathbf{a}\|_2\right).$$

Proof. Introduce $\mathbf{z} \sim \mathcal{N}(0, I)$ such that $\mathbf{n} = \Sigma^{\frac{1}{2}}\mathbf{z}$ into Equation 11:

$$\left| \widetilde{\mathrm{MMD}}^2_{k_\Phi}(\mathcal{D}, \tilde{\mathcal{D}}) - \mathrm{MMD}^2_{k_\Phi}(\mathcal{D}, \tilde{\mathcal{D}}) \right| \leq \mathbf{n}^\top\mathbf{n} + 2\left|\mathbf{n}^\top\mathbf{a}\right| = \mathbf{z}^\top\Sigma\mathbf{z} + 2\left|\mathbf{a}^\top\Sigma^{1/2}\mathbf{z}\right|. \quad (13)$$

For the first term, denoting the eigendecomposition of Σ as $\mathbf{Q}\mathbf{\Lambda}\mathbf{Q}^\top$, we can write

$$\mathbf{z}^\top\Sigma\mathbf{z} = (\mathbf{Q}^\top\mathbf{z})^\top\mathbf{\Lambda}(\mathbf{Q}^\top\mathbf{z}),$$

in which $\mathbf{Q}^\top\mathbf{z} \sim \mathcal{N}(0, I)$ and $\mathbf{\Lambda}$ is diagonal. Thus, applying Lemma 1 of Laurent & Massart (2000), we obtain that with probability at least $1 - \frac{\rho}{2}$,

$$\mathbf{z}^\top\Sigma\mathbf{z} \leq \mathrm{Tr}(\Sigma) + 2\|\Sigma\|_F\sqrt{\log(\tfrac{2}{\rho})} + 2\|\Sigma\|_{op}\log(\tfrac{2}{\rho}). \quad (14)$$

In the second term, $\left|\mathbf{a}^\top\Sigma^{\frac{1}{2}}\mathbf{z}\right|$, can be viewed as a function of a standard normal variable $\mathbf{z}$ with Lipschitz constant at most $\|\Sigma^{\frac{1}{2}}\mathbf{a}\|_2$. Thus, applying the standard Gaussian Lipschitz concentration inequality (Boucheron et al., 2013, Theorem 5.6), we obtain that with probability at least $1 - \frac{\rho}{2}$,

$$\left|\mathbf{z}^\top\Sigma^{\frac{1}{2}}\mathbf{a}\right| \leq \mathbb{E}\left|\mathbf{z}^\top\Sigma^{\frac{1}{2}}\mathbf{a}\right| + \|\Sigma^{\frac{1}{2}}\mathbf{a}\|_2\sqrt{2\log(\tfrac{2}{\rho})} = \|\Sigma^{\frac{1}{2}}\mathbf{a}\|_2\left(\sqrt{\tfrac{2}{\pi}} + \sqrt{2\log(\tfrac{2}{\rho})}\right).$$

The first statement in the theorem follows by a union bound. The $\mathcal{O}_p$ form follows by Lemma A.1 and the fact that $\mathrm{Tr}(A) \geq \|A\|_F \geq \|A\|_{op}$ for positive semi-definite matrices A. □

The following lemma shows how to convert high-probability bounds with both sub-exponential and sub-Gaussian tails into a $\mathcal{O}_p$ statement.

Lemma A.1. *If a sequence of random variables X_n satisfies*

$$X_n \leq A_n + B_n\sqrt{\log\frac{b_n}{\rho}} + C_n\log\frac{c_n}{\rho} \qquad \textit{with probability at least } 1 - \rho,$$

then the sequence of variables X_n is

$$\mathcal{O}_p\left(\max\left(A_n, B_n\max(\sqrt{\log b_n}, 1), C_n\max(\log c_n, 1)\right)\right).$$

Proof. The definition of a sequence of random variables X_n being $\mathcal{O}_p(Q_n)$, where Q_n is a sequence of scalars, means that the sequence $\frac{X_n}{Q_n}$ is stochastically bounded: for each ρ, there is some constant R_ρ such that $\Pr(X_n/Q_n \geq R_\rho) \leq \rho$.

Here, we have for all n with probability at least $1-\rho$ that

$$\begin{aligned}
\frac{X_n}{\max\left(A_n, B_n \max(\sqrt{\log b_n}, 1), C_n \max(\log c_n, 1)\right)} &\leq \frac{A_n + B_n\sqrt{\log \frac{b_n}{\rho}} + C_n \log \frac{c_n}{\rho}}{\max\left(A_n, B_n \max(\sqrt{\log b_n}, 1), C_n \max(\log c_n, 1)\right)} \\
&= \frac{A_n + B_n\sqrt{\log b_n + \log \frac{1}{\rho}} + C_n \left[\log c_n + \log \frac{1}{\rho}\right]}{\max\left(A_n, B_n \max(\sqrt{\log b_n}, 1), C_n \max(\log c_n, 1)\right)} \\
&\leq \frac{A_n + B_n\sqrt{\log b_n} + B_n\sqrt{\log \frac{1}{\rho}} + C_n \log c_n + C_n \log \frac{1}{\rho}}{\max\left(A_n, B_n \max(\sqrt{\log b_n}, 1), C_n \max(\log c_n, 1)\right)} \\
&\leq 1 + 1 + \sqrt{\log \frac{1}{\rho}} + 1 + \log \frac{1}{\rho}.
\end{aligned}$$

Thus the desired bound holds with $R_\rho = 3 + \sqrt{\log \frac{1}{\rho}} + \log \frac{1}{\rho}$. □

A.3 Quality of the private minimizer: worst-case analysis

We first show uniform convergence of the privatized MMD to the non-private MMD.

Proposition A.3. *Suppose that $\Phi : \mathcal{X} \to \mathbb{R}^D$ is such that $\sup_x \|\Phi(x)\| \leq B$, and let $\widetilde{\mathrm{MMD}}_{k_\Phi}(\mathcal{D}, \tilde{\mathcal{D}}) = \|\mu_\Phi(\mathcal{D}) - \mu_\Phi(\tilde{\mathcal{D}}) + \mathbf{n}\|$ for $\mathbf{n} \sim \mathcal{N}(0, \Sigma)$. Then, with probability at least $1-\rho$ over the choice of $\mathbf{n}$,*

$$\begin{aligned}
&\sup_{\mathcal{D}, \tilde{\mathcal{D}}} \left| \widetilde{\mathrm{MMD}}^2_{k_\Phi}(\mathcal{D}, \tilde{\mathcal{D}}) - \mathrm{MMD}^2_{k_\Phi}(\mathcal{D}, \tilde{\mathcal{D}}) \right| \\
&\quad \leq \mathrm{Tr}(\Sigma) + 4B\sqrt{\mathrm{Tr}(\Sigma)} + 2\left(\|\Sigma\|_F + 2B\|\Sigma\|_{op}^{\frac{1}{2}}\right)\sqrt{\log(\tfrac{2}{\rho})} + 2\|\Sigma\|_{op}\log(\tfrac{2}{\rho}) = \mathcal{O}_p\left(\mathrm{Tr}(\Sigma) + B\sqrt{\mathrm{Tr}(\Sigma)}\right),
\end{aligned}$$

where the supremum is taken over all distributions, including the empirical distribution of datasets $\mathcal{D}, \tilde{\mathcal{D}}$ of any size.

Proof. Introducing $\mathbf{z} \sim \mathcal{N}(0, I_D)$ such that $\mathbf{n} = \Sigma^{1/2}\mathbf{z}$, we have that

$$\begin{aligned}
\sup_{\mathcal{D}, \tilde{\mathcal{D}}} \left| \widetilde{\mathrm{MMD}}^2_{k_\Phi}(\mathcal{D}, \tilde{\mathcal{D}}) - \mathrm{MMD}^2_{k_\Phi}(\mathcal{D}, \tilde{\mathcal{D}}) \right| &\leq \sup_{\mathcal{D}, \tilde{\mathcal{D}}} \mathbf{z}^\top \Sigma \mathbf{z} + 2\left|\mathbf{a}^\top \Sigma^{1/2}\mathbf{z}\right| \\
&\leq \mathbf{z}^\top \Sigma \mathbf{z} + 2 \sup_{\mathbf{a}: \|\mathbf{a}\| \leq 2B} \left|\mathbf{a}^\top \Sigma^{1/2}\mathbf{z}\right| \\
&\leq \mathbf{z}^\top \Sigma \mathbf{z} + 2 \sup_{\mathbf{a}: \|\mathbf{a}\| \leq 2B} \|\mathbf{a}\| \|\Sigma^{1/2}\mathbf{z}\| \\
&= \mathbf{z}^\top \Sigma \mathbf{z} + 4B\|\Sigma^{1/2}\mathbf{z}\|.
\end{aligned}$$

To apply Gaussian Lipschitz concentration, we also need to know that

$$\mathbb{E}\|\Sigma^{1/2}\mathbf{z}\| \leq \sqrt{\mathbb{E}\|\Sigma^{1/2}\mathbf{z}\|^2} = \sqrt{\mathrm{Tr}(\Sigma)};$$

the exact expectation of a χ variable with more than one degree of freedom is inconvenient, but the gap is generally not asymptotically significant. Then we get that, with probability at least $1 - \frac{\rho}{2}$,

$$\|\Sigma^{1/2}\mathbf{z}\| \leq \sqrt{\mathrm{Tr}(\Sigma)} + \|\Sigma\|_{op}^{1/2}\sqrt{2\log \tfrac{2}{\rho}}.$$

Again combining with the bound of Equation 14, we get the stated bound. □

This bound is looser than in Proposition A.2, since the term depending on $\mathbf{a}$ is now "looking at" $\mathbf{z}$ in many directions rather than just one: we end up with a $\chi(\dim(\Sigma))$ random variable instead of $\chi(1)$.

We can use this uniform convergence bound to show that the minimizer of the private loss approximately minimizes the non-private loss:

Proposition A.4. *Fix a target dataset $\mathcal{D}$. For each $\boldsymbol{\theta}$ in some set Θ, fix a corresponding $\tilde{\mathcal{D}}_{\boldsymbol{\theta}}$; in particular, $\Theta = \mathbb{R}^p$ could be the set of all generator parameters, and $\tilde{\mathcal{D}}_{\boldsymbol{\theta}}$ either the outcome of running a generator $g_{\boldsymbol{\theta}}$ on a fixed set of "seeds," $\tilde{\mathcal{D}}_{\boldsymbol{\theta}} = \{g_{\boldsymbol{\theta}}(\mathbf{z}_i)\}_{i=1}^n$, or the full output distribution of the generator $Q_{g_{\boldsymbol{\theta}}}$. Suppose that $\Phi : \mathcal{X} \to \mathbb{R}^D$ is such that $\sup_x \|\Phi(x)\| \le B$, and let $\widetilde{\mathrm{MMD}}_{k_\Phi}(\mathcal{D}, \tilde{\mathcal{D}}) = \|\mu_\Phi(\mathcal{D}) - \mu_\Phi(\tilde{\mathcal{D}}) + \mathbf{n}\|$ for $\mathbf{n} \sim \mathcal{N}(0, \Sigma)$. Let $\widetilde{\boldsymbol{\theta}} \in \arg\min_{\theta\in\Theta} \widetilde{\mathrm{MMD}}^2_{k_\Phi}(\mathcal{D}, \tilde{\mathcal{D}}_\theta)$ be the private minimizer, and $\widehat{\boldsymbol{\theta}} \in \arg\min_{\theta\in\Theta} \widehat{\mathrm{MMD}}^2_{k_\Phi}(\mathcal{D}, \tilde{\mathcal{D}}_\theta)$ the non-private minimizer. For any $\rho \in (0,1)$, with probability at least $1-\rho$ over the choice of $\mathbf{n}$,*

$$\begin{aligned}
&\mathrm{MMD}^2_{k_\Phi}(\mathcal{D}, \tilde{\mathcal{D}}_{\widetilde{\boldsymbol{\theta}}}) - \mathrm{MMD}^2_{k_\Phi}(\mathcal{D}, \tilde{\mathcal{D}}_{\widehat{\boldsymbol{\theta}}}) \\
&\le 2\mathrm{Tr}(\Sigma) + 8B\sqrt{\mathrm{Tr}(\Sigma)} + 4\left(\|\Sigma\|_F + 2B\|\Sigma\|_{op}^{\frac{1}{2}}\right)\sqrt{\log(\tfrac{2}{\rho})} + 4\|\Sigma\|_{op}\log(\tfrac{2}{\rho}) = \mathcal{O}_p\left(\mathrm{Tr}(\Sigma) + B\sqrt{\mathrm{Tr}(\Sigma)}\right).
\end{aligned}$$

Proof. Let α represent the uniform error bound of Proposition A.2. Applying Proposition A.2, the definition of $\widetilde{\boldsymbol{\theta}}$, then Proposition A.2 again:

$$\mathrm{MMD}^2_{k_\Phi}(\mathcal{D}, \tilde{\mathcal{D}}_{\widetilde{\boldsymbol{\theta}}}) \le \widetilde{\mathrm{MMD}}^2_{k_\Phi}(\mathcal{D}, \tilde{\mathcal{D}}_{\widetilde{\boldsymbol{\theta}}}) + \alpha \le \widetilde{\mathrm{MMD}}^2_{k_\Phi}(\mathcal{D}, \tilde{\mathcal{D}}_{\widehat{\boldsymbol{\theta}}}) + \alpha \le \mathrm{MMD}^2_{k_\Phi}(\mathcal{D}, \tilde{\mathcal{D}}_{\widehat{\boldsymbol{\theta}}}) + 2\alpha. \qquad \square$$

A.4 Quality of the private minimizer: "optimistic" analysis

The preceding analysis is quite "worst-case," since we upper-bounded the MMD by the maximum possible value everywhere. Noticing that the approximation in Proposition A.2 is tighter when $\|\Sigma^{1/2}\mathbf{a}\|$ is smaller, we can instead show an "optimistic" rate which takes advantage of this fact to show tighter approximation for the minimizer of the noised loss. In the "interpolating" case where the generator can achieve zero empirical MMD, the convergence rate substantially improves (generally improving the squared MMD from $\mathcal{O}_p(1/m)$ to $\mathcal{O}_p(1/m^2)$).

Proposition A.5. *In the setup of Proposition A.4, we have with probability at least $1-\rho$ over $\mathbf{n}$ that*

$$\begin{aligned}
&\mathrm{MMD}^2_{k_\Phi}(\mathcal{D}, \tilde{\mathcal{D}}_{\widetilde{\boldsymbol{\theta}}}) - \mathrm{MMD}^2_{k_\Phi}(\mathcal{D}, \tilde{\mathcal{D}}_{\widehat{\boldsymbol{\theta}}}) \\
&\quad\le 9\mathrm{Tr}(\Sigma) + 4\sqrt{\mathrm{Tr}(\Sigma)}\,\mathrm{MMD}_{k_\Phi}(\mathcal{D}, \tilde{\mathcal{D}}_{\widehat{\boldsymbol{\theta}}}) \\
&\qquad + 2\left(9\|\Sigma\|_F + 2\sqrt{2\|\Sigma\|_{op}}\,\mathrm{MMD}_{k_\Phi}(\mathcal{D}, \tilde{\mathcal{D}}_{\widehat{\boldsymbol{\theta}}})\right)\sqrt{\log\frac{2}{\rho}} + 18\|\Sigma\|_{op}\log\frac{2}{\rho} \\
&\quad= \mathcal{O}_p\left(\mathrm{Tr}(\Sigma) + \sqrt{\mathrm{Tr}(\Sigma)}\,\mathrm{MMD}_{k_\Phi}(\mathcal{D}, \tilde{\mathcal{D}}_{\widehat{\boldsymbol{\theta}}})\right).
\end{aligned}$$

Proof. Let's use $\widehat{\mathrm{MMD}}(\boldsymbol{\theta})$ to denote $\mathrm{MMD}_{k_\Phi}(\mathcal{D}, \tilde{\mathcal{D}}_{\boldsymbol{\theta}})$, and $\widetilde{\mathrm{MMD}}(\boldsymbol{\theta})$ for $\widetilde{\mathrm{MMD}}_{k_\Phi}(\mathcal{D}, \tilde{\mathcal{D}}_{\boldsymbol{\theta}})$.

For all $\boldsymbol{\theta}$, we have that

$$\begin{aligned}
\left|\widetilde{\mathrm{MMD}}^2(\boldsymbol{\theta}) - \widehat{\mathrm{MMD}}^2(\boldsymbol{\theta})\right| &\le \mathbf{z}^\top\Sigma\mathbf{z} + 2\left|(\mu^\Phi(\mathcal{D}) - \mu^\Phi(\tilde{\mathcal{D}}))^\top\Sigma^{1/2}\mathbf{z}\right| \\
&\le \mathbf{z}^\top\Sigma\mathbf{z} + 2\,\widehat{\mathrm{MMD}}(\boldsymbol{\theta})\|\Sigma^{1/2}\mathbf{z}\|.
\end{aligned}$$

Thus, applying this inequality in both the first and third lines,

$$\begin{aligned}
\widehat{\mathrm{MMD}}^2(\widetilde{\boldsymbol{\theta}}) &\le \widetilde{\mathrm{MMD}}^2(\widetilde{\boldsymbol{\theta}}) + \mathbf{z}^\top\Sigma\mathbf{z} + 2\,\widehat{\mathrm{MMD}}(\widetilde{\boldsymbol{\theta}})\|\Sigma^{1/2}\mathbf{z}\| \\
&\le \widetilde{\mathrm{MMD}}^2(\widehat{\boldsymbol{\theta}}) + \mathbf{z}^\top\Sigma\mathbf{z} + 2\,\widehat{\mathrm{MMD}}(\widetilde{\boldsymbol{\theta}})\|\Sigma^{1/2}\mathbf{z}\| \\
&\le \widehat{\mathrm{MMD}}^2(\widehat{\boldsymbol{\theta}}) + 2\mathbf{z}^\top\Sigma\mathbf{z} + 2\left(\widehat{\mathrm{MMD}}(\widetilde{\boldsymbol{\theta}}) + \widehat{\mathrm{MMD}}(\widehat{\boldsymbol{\theta}})\right)\|\Sigma^{1/2}\mathbf{z}\|;
\end{aligned}$$

in the second line we used that $\widetilde{\mathrm{MMD}}(\widetilde{\boldsymbol{\theta}}) \le \widetilde{\mathrm{MMD}}(\widehat{\boldsymbol{\theta}})$. Rearranging, we get that

$$\widehat{\mathrm{MMD}}^2(\widetilde{\boldsymbol{\theta}}) - \beta\,\widehat{\mathrm{MMD}}(\widetilde{\boldsymbol{\theta}}) - \gamma \le 0, \tag{15}$$

where

$$\beta = 2\|\Sigma^{1/2}\mathbf{z}\| \geq 0$$
$$\gamma = \widehat{\mathrm{MMD}}^2(\widehat{\boldsymbol{\theta}}) + 2\mathbf{z}^\top\Sigma\mathbf{z} + 2\,\widehat{\mathrm{MMD}}(\widehat{\boldsymbol{\theta}})\|\Sigma^{1/2}\mathbf{z}\| \geq 0.$$

The left-hand side of Equation 15 is a quadratic in $\widehat{\mathrm{MMD}}(\tilde{\boldsymbol{\theta}})$ with positive curvature; it has two roots, at

$$\frac{\beta}{2} \pm \sqrt{\left(\frac{\beta}{2}\right)^2 + \gamma}.$$

Thus the inequality Equation 15 can only hold in between the roots; the root with a minus sign is negative, and so does not concern us since we know that $\widehat{\mathrm{MMD}}(\boldsymbol{\theta}) \geq 0$. Thus, for Equation 15 to hold, we must have

$$\begin{aligned}
\widehat{\mathrm{MMD}}(\widetilde{\boldsymbol{\theta}}) &\leq \tfrac{\beta}{2} + \sqrt{\left(\tfrac{\beta}{2}\right)^2 + \gamma} \\
\widehat{\mathrm{MMD}}^2(\widetilde{\boldsymbol{\theta}}) &\leq \tfrac{\beta^2}{4} + \left(\tfrac{\beta}{2}\right)^2 + \gamma + \beta\sqrt{\left(\tfrac{\beta}{2}\right)^2 + \gamma} \\
&\leq \gamma + \beta^2 + \beta\sqrt{\gamma}.
\end{aligned}$$

Also note that

$$\gamma = \widehat{\mathrm{MMD}}^2(\widehat{\boldsymbol{\theta}}) + 2\mathbf{z}^\top\Sigma\mathbf{z} + 2\,\widehat{\mathrm{MMD}}(\widehat{\boldsymbol{\theta}})\|\Sigma^{1/2}\mathbf{z}\| \leq \left(\widehat{\mathrm{MMD}}(\widehat{\boldsymbol{\theta}}) + \sqrt{2}\|\Sigma^{1/2}\mathbf{z}\|\right)^2.$$

Thus, substituting in for β and γ then simplifying, we have that

$$\widehat{\mathrm{MMD}}^2(\widetilde{\boldsymbol{\theta}}) \leq \widehat{\mathrm{MMD}}^2(\widehat{\boldsymbol{\theta}}) + (6 + 2\sqrt{2})\mathbf{z}^\top\Sigma\mathbf{z} + 4\|\Sigma^{1/2}\mathbf{z}\|\,\widehat{\mathrm{MMD}}(\widehat{\boldsymbol{\theta}}).$$

Using the same bounds on $\mathbf{z}^\top\Sigma\mathbf{z}$ and $\|\Sigma^{1/2}\mathbf{z}\|$ as in Proposition A.3, and $6\sqrt{2} < 9$, gives the claimed bound. □

B Extended Implementation details

Repository. Our code is available at `https://github.com/ParkLabML/DP-MEPF`; the readme files contain further instructions on how to run the code.

B.1 Hyperparameter settings

For each dataset, we tune the generator learning rate (LR_{gen}) and moving average learning rate (LR_{mavg}) from choices 10^{-k} and $3 \cdot 10^{-k}$ with $k \in \{3, 4, 5\}$ once for the non-private setting and once at $\epsilon = 2$. The latter is used in all private experiments for that dataset, as shown in 7. After some initial unstructured experimentation, hyperparameters are chosen with identical values across dataset shown in 8

For the Cifar10 DP-MERF baseline we tested random tuned random features dimension $d \in \{10000, 50000\}$, random features sampling distribution $\sigma \in \{100, 300, 1000\}$, learning rate decay by 10% every $e \in \{1000, 10000\}$ iterations and learning rate 10^{-k} with $k \in \{2, 3, 4, 5, 6\}$. Results presented use $d = 500000, \sigma = 1000, e = 10000, k = 3$.

The DP-GAN baseline for Cifar10 and CelebA uses the same generator as DP-MEPF with 3 residual blocks and a total of 8 convolutional layers and is paired with a ResNet9 discriminator which uses Groupnorm instead of Batchnorm to allow for per-sample gradient computation. We pre-train the model non-privately to convergence on downsampled imagenet in order to maintain the same resolution of 32×32 and then fine-tune the model for a smaller number of epochs. In case of the CelebA 64×64 data we add another residual block to discriminator and generator to account for the doubling in resolution. The base multiplier for number of feature maps is reduced from 64 to 50 to lessen the increase in number of weights. Results are the best scores of a grid-search over the following parameters at $\epsilon = 2$, which is then used in all settings: number of epochs $\{1, 10, 30, 50\}$ generator and discriminator learning rate separately for 10^{-k} and $3 \cdot 10^{-k}$ with $k \in \{3, 4, 5\}$, clip-norm $\{10^{-3}, 10^{-4}, 10^{-5}, 10^{-6}\}$, batch size $\{128, 256, 512\}$ and, as advised in Bie et al. (2023), number of discriminator updates per generator $\{1, 10, 30, 50\}$. The chosen values are given in table 9.

Table 7: Learning rate hyperparameters across datasets

Dataset	ε	(ϕ_1, ϕ_2)		(ϕ_1)	
		LR_{gen}	LR_{mavg}	LR_{gen}	LR_{mavg}
MNIST	$\varepsilon = \infty$	10^{-5}	10^{-3}	10^{-5}	10^{-3}
	$\varepsilon < \infty$	10^{-5}	10^{-4}	10^{-5}	10^{-4}
FashionMNIST	$\varepsilon = \infty$	10^{-5}	10^{-3}	10^{-5}	10^{-3}
	$\varepsilon < \infty$	10^{-4}	10^{-3}	10^{-4}	10^{-3}
CelebA32	$\{\infty, 10, 5\}$	$3 \cdot 10^{-4}$	10^{-4}	$3 \cdot 10^{-4}$	10^{-4}
	$\{2, 1\}$	$3 \cdot 10^{-4}$	$3 \cdot 10^{-4}$	$3 \cdot 10^{-4}$	$3 \cdot 10^{-4}$
	$\{0.5, 0.2\}$	10^{-3}	$3 \cdot 10^{-4}$	$\cdot 10^{-3}$	$3 \cdot 10^{-4}$
CelebA64	$\{\infty, 10, 5\}$	$3 \cdot 10^{-4}$	10^{-4}	$3 \cdot 10^{-4}$	$3 \cdot 10^{-4}$
	$\{2, 1\}$	$3 \cdot 10^{-4}$	10^{-3}	$3 \cdot 10^{-4}$	$3 \cdot 10^{-4}$
	$\{0.5, 0.2\}$	10^{-3}	10^{-3}	10^{-3}	10^{-3}
Cifar10 labeled	$\{\infty, 10, 5\}$	10^{-3}	$3 \cdot 10^{-4}$	10^{-3}	10^{-4}
	$\{2, 1\}$	10^{-3}	10^{-2}	10^{-3}	10^{-2}
	$\{0.5, 0.2\}$	10^{-3}	10^{-2}	10^{-3}	10^{-2}
Cifar10 unlabeled	$\{\infty, 10, 5\}$	10^{-3}	10^{-3}	10^{-3}	10^{-3}
	$\{2, 1\}$	10^{-3}	10^{-3}	10^{-3}	10^{-3}
	$\{0.5, 0.2\}$	10^{-3}	10^{-3}	10^{-3}	10^{-3}

Table 8: Hyperparameters fixed across datasets

Parameter	Value
(ϕ_1)-bound	1
(ϕ_2)-bound	1
iterations (MNIST & FashionMNIST)	100,000
batch size (MNIST and FashionMNIST)	100
iterations (Cifar10 & CelebA)	200,000
batch size (Cifar10 and CelebA)	128
seeds	1,2,3,4,5

C Detailed Tables

Below we present the results from the main paper with added $a \pm b$ notation, where a is the mean and b is the standard deviation of the score distribution across three independent runs for MNIST and FashionMNIST and 5 independent runs for Cifar10 and CelebA.

Table 10: Downstream accuracies of our method for MNIST at varying values of ϵ

		$\epsilon = \infty$	$\epsilon = 10$	$\epsilon = 5$	$\epsilon = 2$	$\epsilon = 1$	$\epsilon = 0.2$
MLP	DP-MEPF (ϕ_1, ϕ_2)	91.4 ± 0.3	89.8 ± 0.5	89.9 ± 0.2	89.3 ± 0.3	89.3 ± 0.6	79.9 ± 1.3
	DP-MEPF (ϕ_1)	88.2 ± 0.6	88.8 ± 0.1	88.4 ± 0.5	88.0 ± 0.2	87.5 ± 0.6	77.1 ± 0.4
LogReg	DP-MEPF (ϕ_1, ϕ_2)	84.6 ± 0.5	83.4 ± 0.6	83.3 ± 0.7	82.9 ± 0.7	82.5 ± 0.5	75.8 ± 1.1
	DP-MEPF (ϕ_1)	81.4 ± 0.4	80.8 ± 0.9	80.8 ± 0.8	80.5 ± 0.6	79.0 ± 0.6	72.1 ± 1.4

Table 9: Hyperparameters of DP-GAN for Cifar10 and CelebA

	Cifar10	CelebA 32×32	CelebA 64×64			
			$\epsilon \in \{0.2, 0.5\}$	$\epsilon = 1$	$\epsilon = 2$	$\epsilon \in \{5, 10\}$
LR_{gen}	10^{-4}	$3 \cdot 10^{-4}$	$3 \cdot 10^{-4}$	$3 \cdot 10^{-4}$	$3 \cdot 10^{-4}$	$3 \cdot 10^{-4}$
LR_{dis}	10^{-3}	$3 \cdot 10^{-4}$	10^{-3}	$3 \cdot 10^{-4}$	10^{-3}	10^{-3}
batch size	512	512	512	512	512	512
epochs	10	10	10	10	10	10
discriminator frequency	10	10	30	30	10	10
clip norm	10^{-5}	10^{-4}	10^{-5}	10^{-5}	10^{-4}	10^{-5}

Table 11: Downstream accuracies of our method for FashionMNIST at varying values of ϵ

		$\epsilon = \infty$	$\epsilon = 10$	$\epsilon = 5$	$\epsilon = 2$	$\epsilon = 1$	$\epsilon = 0.2$
MLP	DP-MEPF (ϕ_1, ϕ_2)	74.4 ± 0.3	76.0 ± 0.4	75.8 ± 0.6	75.1 ± 0.3	74.7 ± 1.1	70.4 ± 1.9
	DP-MEPF (ϕ_1)	73.8 ± 0.5	75.5 ± 0.6	75.1 ± 0.8	75.8 ± 0.7	75.0 ± 1.8	69.0 ± 1.5
LogReg	DP-MEPF (ϕ_1, ϕ_2)	74.3 ± 0.1	75.7 ± 1.0	75.2 ± 0.4	75.8 ± 0.4	75.4 ± 1.1	72.5 ± 1.2
	DP-MEPF (ϕ_1)	72.8 ± 0.5	75.5 ± 0.1	75.5 ± 0.8	76.4 ± 0.8	76.2 ± 0.8	71.7 ± 0.4

Table 12: CelebA FID scores 32×32 (lower is better)

	$\epsilon = \infty$	$\epsilon = 10$	$\epsilon = 5$	$\epsilon = 2$	$\epsilon = 1$	$\epsilon = 0.5$	$\epsilon = 0.2$
DP-MEPF (ϕ_1, ϕ_2)	18.5 ± 0.5	17.4 ± 0.7	17.5 ± 0.6	18.1 ± 0.8	19.0 ± 0.5	21.4 ± 1.3	25.8 ± 2.1
DP-MEPF (ϕ_1)	16.6 ± 0.7	16.3 ± 0.9	16.9 ± 0.5	16.5 ± 0.8	17.2 ± 0.9	21.8 ± 1.0	25.5 ± 1.1

Table 13: CelebA FID scores 64×64 (lower is better)

	$\epsilon = \infty$	$\epsilon = 10$	$\epsilon = 5$	$\epsilon = 2$	$\epsilon = 1$	$\epsilon = 0.5$	$\epsilon = 0.2$
DP-MEPF (ϕ_1, ϕ_2)	18.6 ± 1.0	18.5 ± 1.2	19.1 ± 0.9	18.4 ± 1.0	19.0 ± 1.2	21.4 ± 1.3	26.8 ± 1.5
DP-MEPF (ϕ_1)	16.3 ± 0.4	17.4 ± 1.4	16.5 ± 0.8	16.9 ± 1.1	18.4 ± 0.9	20.4 ± 0.8	27.7 ± 2.1

Table 14: FID scores for synthetic *labelled* CIFAR-10 data (generating both labels and input images)

	$\epsilon = \infty$	$\epsilon = 10$	$\epsilon = 5$	$\epsilon = 2$	$\epsilon = 1$	$\epsilon = 0.5$	$\epsilon = 0.2$
DP-MEPF (ϕ_1, ϕ_2)	27.7 ± 3.1	29.1 ± 1.3	30.0 ± 0.8	39.5 ± 1.9	54.0 ± 1.3	76.4 ± 3.9	226.0 ± 5.4
DP-MEPF (ϕ_1)	28.4 ± 2.8	30.3 ± 2.1	35.6 ± 5.8	42.0 ± 3.0	56.5 ± 3.4	92.0 ± 3.5	268.3 ± 8.5

Table 15: Test accuracies (higher better) of ResNet9 trained on CIFAR-10 synthetic data with varying privacy guarantees. When trained on real data, test accuracy is 88.3%

	$\epsilon = \infty$	$\epsilon = 10$	$\epsilon = 5$	$\epsilon = 2$	$\epsilon = 1$	$\epsilon = 0.5$	$\epsilon = 0.2$
DP-MEPF (ϕ_1, ϕ_2)	57.5 ± 3.3	53.0 ± 2.8	43.9 ± 1.2	40.0 ± 1.9	28.5 ± 4.5	18.0 ± 1.0	16.2 ± 1.8
DP-MEPF (ϕ_1)	43.8 ± 3.5	40.7 ± 4.2	32.3 ± 6.2	42.6 ± 1.6	33.2 ± 2.6	18.8 ± 4.0	15.3 ± 2.5

D Encoder architecture comparison

We are testing a large collection of classifiers of different sizes from the torchvision library including VGG, ResNet, ConvNext and EfficientNet. For each we look at unlabelled Cifar10 generation quality in the non-DP setting and at $\epsilon = 0.2$. In each architecture, we use all activations from convolutional layers with a kernel size greater than 1x1. We list the number of extracted features along with the achieved FID score in table 17, where each result is the best result obtained by tuning learning rates. As already observed in dos Santos et al. (2019), we find that VGG architectures appear to learn particularly useful features for feature matching. We hypothesized that in the private setting other architectures with fewer features might outperform the VGG model, but have found this to not be the case.

E Public dataset comparison

We pretrained a ResNet18 using ImageNet, CIFAR10, and SVHN as our public data, respectively. We then used the perceptual features to train a generator using CelebA dataset as our private data at a privacy budget of $\epsilon = 0.2$ and obtained the scores shown in 18. These numbers reflect our intuition that as long as the public data is sufficiently similar and contains more complex patterns than private data, e.g., transferring the knowledge learned from ImageNet as public data to generate CelebA images as private data, the learned features from public data are useful enough to generate good synthetic data. In addition, as the public data become more simplistic (from CIFAR10 to SVHN), the usefulness of such features reduces in producing good CelebA synthetic samples.

Table 16: FID scores for synthetic *unlabelled* CIFAR-10 data

	$\epsilon = \infty$	$\epsilon = 10$	$\epsilon = 5$	$\epsilon = 2$	$\epsilon = 1$	$\epsilon = 0.5$	$\epsilon = 0.2$
DP-MEPF (ϕ_1, ϕ_2)	38.5 ± 1.5	38.8 ± 2.0	37.0 ± 1.1	38.7 ± 2.2	43.0 ± 1.1	49.4 ± 1.0	67.3 ± 2.6
DP-MEPF (ϕ_1)	38.5 ± 0.6	38.5 ± 0.4	38.6 ± 1.3	40.1 ± 1.1	45.1 ± 2.4	49.8 ± 2.5	72.3 ± 4.0

Table 17: Unlabeled Cifar10 FID scores achieved with different feature extractors. VGG models yield the best results in both non-DP and high DP settings.

Encoder model	#features	$\epsilon = \infty$		$\epsilon = 0.2$	
		(ϕ_1, ϕ_2)	(ϕ_1)	(ϕ_1, ϕ_2)	(ϕ_1)
VGG19	303104	35.0	37.0	56.2	85.8
VGG16	276480	37.4	39.8	71.4	72.2
VGG13	249856	38.2	36.7	78.1	71.2
VGG11	151552	40.5	41.6	65.4	68.6
ResNet152	429568	71.8	70.1	88.6	87.9
ResNet101	300544	77.5	73.7	76.0	82.4
ResNet50	196096	71.5	76.3	90.0	105.1
ResNet34	72704	74.8	103.3	89.1	93.1
ResNet18	47104	84.9	85.0	104.5	95.2
ConvNext large	161280	141.9	232.0	138.2	221.6
ConvNext base	107520	142.4	248.0	157.0	200.1
ConvNext small	80640	171.7	212.3	169.9	202.9
ConvNext tiny	52992	145.6	218.2	138.8	205.8
EfficientNet L	119168	200.9	229.0	243.7	226.6
EfficientNet M	68704	185.7	177.1	218.7	227.1
EfficientNet S	47488	157.5	160.6	171.5	186.7

Table 18: FID scores achieved for CelebA 32×32 using a ResNet encoder with different public training sets

	ImageNet	Cifar10	SVHN
FID	47.6	51.2	65.2

F Training DP-MEPF without auxiliary data

While DP-MEPF is explicitly designed to take advantage of available public data, one might wonder how the method performs if no such data is available. The following experiment on CIFAR10 explores this scenario. We assume that a privacy budget of $\epsilon = 10$ is given. We use some part of the budget for feature extractor (i.e. the classifier) training and the rest of the budget for the generator training.

For a feature extractor, we have trained ResNet-20 classifiers with DP-SGD at three different levels of $\epsilon \in \{2, 5, 8\}$ for classifying the CIFAR10 dataset. We set the clipping norm to 0.01 and trained the classifiers for $7, 49$ and 98 epochs, respectively. Their test accuracies are $38.4\%, 49.5\%$ and 54.0% respectively. We also include scores for DP-MEPF applied to the untrained Classifier, denoted as $\epsilon = 0$.

Then, we train the generator using these four sets of features to generate CIFAR10 images, where each generator training uses the rest of the budget, i.e., $\epsilon \in \{8, 5, 2\}$ and $\epsilon = 10$ for the untrained classifier. We tune the learning rate in each of the four settings and keep other hyperparameters at default values.

Table 19: DP-MEPF results in CIFAR10 when using a DP feature extractor ($\epsilon = 0$ is an untrained extractor)

ϵ for feature extractor training	for generator training	FID
0	10	111.1
2	8	127.0
5	5	90.8
8	2	119.0

As expected, in Table 19 we see a considerable increase in the FID score, compared to DP-MEPF with public data. A balanced allocation of privacy budget with $\epsilon = 5$ each for classifier and generator training yields the best result at an FID score of 90.8 and performs significantly better than just using a randomly initialized feature extractor, which only achieves a score of 111.1. For comparison: with public data DP-MEPF achieves an FID score of 37.0 at $\varepsilon = 5$, highlighting the importance of such data to our method.

G Additional Plots

Below we show samples from our generated MNIST and FashionMNIST data in Figure 7 and Figure 8 respectively.

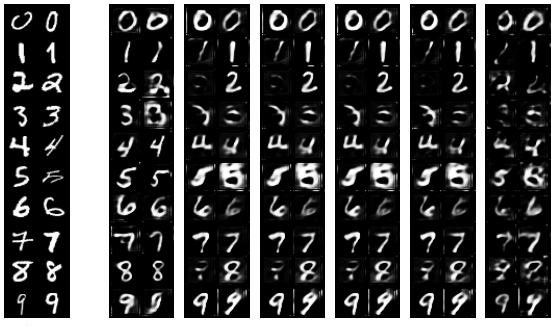


Figure 7: MNIST samples produced with DP-MEPF (ϕ_1, ϕ_2) at various levels of privacy

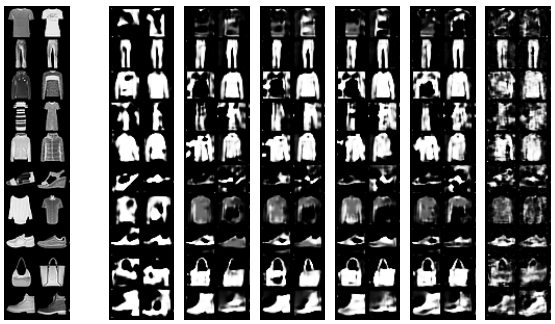


Figure 8: Fashion-MNIST samples produced with DP-MEPF (ϕ_1, ϕ_2) at various levels of privacy