Connecting silos with distributed and private computation

by

Praneeth Vepakomma

M.S., Mathematical and Applied Statistics
Rutgers University, New Brunswick, 2010

Submitted to the Program in Media Arts and Sciences
School of Architecture and Planning
in partial fulfillment of the requirements for the degree of

Doctor of Philosophy

at the

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

February 2024

Author . . . .
Program in Media Arts and Sciences
School of Architecture and Planning
December 20, 2023

Certified by . . . .
Ramesh Raskar, Thesis Supervisor
Associate Professor
Massachusetts Institute of Technology

Accepted by . . . .
Joseph Paradiso
Academic Head, Program in Media Arts and Sciences
Massachusetts Institute of Technology

Connecting silos with distributed and private computation

by

Praneeth Vepakomma

Submitted to the Program in Media Arts and Sciences
School of Architecture and Planning
on December 20, 2023, in partial fulfillment of the
requirements for the degree of
Doctor of Philosophy

Abstract

Data in today's world is increasingly siloed across a wide variety of entities with varying resource constraints. The quality of wisdom generated from a collaborative processing of such data is substantially better if the data from all these entities is shared across each other or centralized at a nodal entity. Such data sharing and centralization is often prohibited due to stringent privacy regulations, computational constraints, communication bottlenecks, trade secrets, trust issues and competition. This necessitates development of efficient methods for distributed computation while preserving privacy to generate wisdom whose quality is on par with the case of data centralization. This thesis covers methods introduced for the same in an inter-disciplinary manner to tackle several such problems using distributed and private computation.

Thesis Supervisor: Ramesh Raskar
Title: Associate Professor
Massachusetts Institute of Technology

Connecting Silos with Distributed and Private Computation

by

Praneeth Vepakomma

The following people served as readers for this thesis:

Thesis Reader____

Alex Pentland
Professor
Massachusetts Institute of Technology

Connecting Silos with Distributed and Private Computation

by

Praneeth Vepakomma

The following people served as readers for this thesis:

Thesis Reader____

Kun Zhang
Associate Professor
Carnegie Mellon University and
Mohamed bin Zayed University of Artificial Intelligence

Connecting Silos with Distributed and Private Computation

by

Praneeth Vepakomma

The following people served as readers for this thesis:

Thesis Reader____

Han Yu
Assistant Professor
Nanyang Technological University

Contents

Acknowledgments

I am extremely grateful to my advisor Ramesh Raskar for providing an exciting and creative lab environment, for his invaluable guidance, for providing a lot of intellectual freedom in pursuing research and for making some important implicit and thoughtful choices early on as an advisor as some of these panned out well in my favor with regards to the research journey that came ahead. This for instance, includes Ramesh's exceptional guidance on initial scoping and strategizing of some broad, exciting and impactful research directions that has been of tremendous value. I would like to collectively thank my dissertation committee members Ramesh Raskar, Alex Pentland, Kun Zhang and Han Yu for their time, tremendous support and in motivating me to map out potential societal applications of my results on top of my existing mapping of abstract/pure implications and connections. I would like to thank Alex 'Sandy' Pentland for providing invaluable advice on various occasions based on his vast experience in academia. I am thankful to all the members of the Camera Culture lab amazingly run by Ramesh for their camaraderie and countless discussions.

The doctoral study and research phase of my life has been a culmination of some implicit decisions, some fortuitous choices and some decisions made explicitly. Before arriving to MIT, I had spent several years in the industry from 2010-2018 (in large companies and startups), that shaped a good deal of my take on balancing professional life, and for several life skills that am thankful for. I would particularly like to thank William Kilmer in the industrial realm of my interactions.

Prior to that, I had an exciting and steep *learning curve* at the Department of Statistics (at the wonderful Hill Center for Mathematical Sciences), at Rutgers University over a couple years from 2009. This explicit choice of becoming a mathematically inclined individual has been set in motion by fortuitous luck when I got an opportunity for a summer internship in 2007 with Kyle Gallivan, a mathematics professor at Florida State University. Without this happening, I wouldn't have pursued my currently crafted path and I am extremely grateful to him for this internship. This helped cement such a strong taste in me for the rigorous, abstract and pure sciences. That style of work is a component of research that I have enjoyed and benefited from the most by far in my career so far. Looking back at this path of accidentally becoming a statistician (partly by a lucky accident of the internship followed by an explicit choice to be one), I cannot think of any other path that I would have enjoyed better.

In my time at the statistics department, I learnt the most over there from William Strawderman (on estimation theory, minimax estimation, Bayes estimation), late Larry Shepp (on stochastic processes), late Arthur Cohen (on introductory statistics), Cun-Hui Zhang (on statistical inference, some probability theory and the art of lengthy and detailed derivations during the class), Harold Sackrowitz (on design of experiments), late Kesar Singh (on bootstrap), David Tyler (on robust statistics, theory of statistics, M-estimation), Joseph Irwin Naus (on nonparametric statistics, scan statistics, summarizing findings), Javier Cabrera (on target estimation, empirical Bayes and applied statistical aspects of multiple testing, conditional testing and fascinating/inspiring tales about his close interactions with John Tukey) and Tong Zhang (on machine learning theory, basic probability/combinatorics and concentration bounds). It's hard to emphasize the profound impact, I have had from the collective brilliance of these individuals; both during my time over there and many more years long after that. I would particularly like to thank Ahmed Elgammal from the computer science department at Rutgers for providing a wonderful research experience at the intersection of applied machine learning and statistics in addition to invaluable early career guidance. In the same spirit I am thankful and grateful to external faculty Mehdi Bennis for his collaboration and career guidance along with Shekhar Bhansali for his invaluable support. There is a longer list of faculty from various places that I would like to thank for occasional but always useful discussions and research lessons. These include Edinah Gnang, Graham Cormode, Yulia Kempner, Nina Miolane, Eugene Soltes, Jayashree Kalpathy-Cramer, Seth Neel, Clément Canonne, Rajiv Gupta and Jihong Park. Students and collaborators that I would particularly thank, include Suat Evren, Abhishek Singh, Julia Balla, Rodmy Paredes Alfaro, Subha Nawer Pushpita, Kaustubh Ponkshe, Ege Kabasakalolu, Ken Chang, Chaoyang He, Mohammad Mohammadi Amiri, Saiteja Utpala, Sai Praneeth Karimireddi, Baihe Huang, Kevin Zhao, Niloofar Mireshghallah, Charles Lu, Gauri Gupta, Hangjie Ji, Vivek Sharma, Kushagra Tiwary, Sihun Baek, Otkrist Gupta, Zaid Tasneem, Ayush Chopra and Yusuke Koda for all the wonderful collaborations, discussions and countless attempts and research view-points. Other members of the lab that I have interacted with on a daily basis include Tzofi Klinghoffer, Siddharth Somasundaram, Guy Satat, Connor Henley, Subhash Sadhu, Neil Gaikwad, Lagnojita Sinha, Tomohiro Maeda, Tristan Swedish and Nikhil Behari. Thank you for being such wonderful lab mates. The academic environment around me ran smoothly due to Ashley Bell Clark, Hannah Corbin, Anel Urazova and Maggie Church. Kudos to your administrative superpowers; and thank you so much for all the hard work that you put in. I am thankful to the organizers of the SIAM Mathematical Problems in Industry workshops, Graduate Student Mathematical Modeling Camps and Boston Differential Privacy Summer School for their amazing problem solving sessions and research interactions and in particular Salil Vadhan for open DP related events at Harvard.

Regardless of this chronological view, the perennially most important, supportive, caring and inspiring people of my life have been my parents Dr. Bhujanga Rao Vepakomma and Lalitha Vepakomma, my brother Hemanth, Shilpa, Deepika, Sasidhar, Padma Sri, late Viswanadham, late grandparents Krishna Rao and Sita Devi, the new generation of Tanish, Reya, Samyu and my beloved and ever-supportive wife, Harika. I would like to dedicate this thesis to them.

Listing of figures

DEDICATED TO MY FAMILY, TEACHERS, MENTORS AND FRIENDS.

"I feel that one should employ methods that reflect the physics (specifics) of the problem at hand rather than the methods one happens to know."

Lawrence Shepp

1

Introduction

1.1 Summary of contributions

Data in today's world is increasingly siloed across a wide variety of entities with varying resource constraints. The quality of wisdom generated from the collaborative processing of such data is substantially better if the data from all these entities is shared across each other or centralized at a nodal entity. Data sharing and centralization are often prohibited due to stringent privacy regulations, computational constraints, communication bottlenecks, trade secrets, trust issues and competition. This necessitates the development of efficient methods for distributed computation while preserving privacy to generate wisdom whose quality is on par with the case of data centralization. This thesis covers methods introduced for the same in an interdisciplinary manner to tackle several such problems using distributed and private computation. The methods introduced in the thesis can be categorized into three parts, including Part I.) Distributed and Private Statistical Inference, Part II.) Distributed and Private Machine Learning and Part III.) Distributed and Private Scientific Computing. The work is motivated and guided by several downstream applications of the proposed methods, including distributed and private compliance for anti-corruption, private geo-location aggregation for location-based services, split learning-a popular variant of federated learning for distributed predictive analytics, distributed hypothesis testing for multi-center collaboration, collaboration on app/super-app ecosystems, digital advertising with a focus on revenue recovery due to a paradigm shift in enforcing rapidly evolving privacy regulations, distributed point-estimation of nonlinear correlations across two-parties, and intrinsic statistical data valuation for data markets that coordinates well with extrinsic game-theoretic approaches. The technical foundations for the introduced methods span multiple areas, including statistics, machine learning, geometry, partial differential equations, combinatorics and social choice.

1.2 Background

It has been recently shown that privacy-enhancing techniques based on previous approaches of K-Anonymity do not satisfy GDPR (European Union's privacy law) on the required clause of 'predicate singling out' while differentially private mechanisms (a mathematical notion of privacy) satisfy this clause[122]. This mathematical notion of privacy called differential privacy[160] was introduced in recent years and holds advantageous properties as it does not a.) require any specific modelling of attacks to prevent leakage of privacy, b.) provides accurate quantification of privacy loss, c.) holds forward-compatibility to post-process a differentially private release in any way without having to lose the privacy guarantees, d.) can perform multiple queries through private mechanisms while precisely budgeting (or accounting) the amount of privacy provided in the end. Differential privacy offers a method to formally release the *output* of a query applied on sensitive data to the public instead of cryptographic methods, which provide security of the *input* data and computational pipeline handled in an encrypted form. Therefore, these methods handle different parts of the data processing pipeline and are not at odds with each other. This is the technical distinction between "privacy" and "security".

1.2.1 What is differential privacy?

A widely accepted mathematical notion of privacy called differential privacy[160] was introduced in recent years and can be defined as follows. A randomized algorithm $\mathcal{M} : \mathcal{D} \to \mathcal{Y}$ is ϵ-differentially private if, for all neighbouring datasets (that defer in a record) $\mathbf{D}, \mathbf{D'} \in \mathcal{D}$ and for all $S \in \mathcal{Y}$,

$$\Pr[\mathcal{M}(\mathbf{D}) \in S] \leq e^{\epsilon} \Pr[\mathcal{M}(\mathbf{D'}) \in S]$$

An instance of this definition is illustrated as shown below, where the outputs of the probability densities of the output of the query applied on neighbouring datasets are bounded by e^{ϵ}. Note that when $\epsilon = 0$, one can see that the mechanism's output becomes independent of the input.

Definition 1.2.1. ***Post-Processing Invariance*** *A major advantage is that differential privacy-inducing mechanisms are immune to post-processing, meaning that an adversary without any additional knowledge about the dataset $\mathbf{X}$ cannot compute a function on the output $\mathcal{A}(\mathbf{X})$ to violate the stated privacy guarantees.*

1.2.2 What is approximate differential privacy?

In the case where slack is given in an event *(OR condition)* where the differential privacy does not hold, with probability δ, we have the following additive approximation of the probability distributions (due to the OR condition), in addition to the multiplicative approximation by e^ϵ.

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

This form of differential privacy is known as approximate differential privacy, where the approximation is via a multiplicative bound on the probabilities.

1.2.3 Accounting for two sources of randomness

In private statistical inference, two sources of randomness need to be accounted for

1. Sampling noise: modelling assumption about data generating process. Example: $\frac{\sigma}{\sqrt{n}}$ error needs to be paid for estimating mean even in the non-private setting.

2. Privacy noise: estimation error due to injection of privacy-preserving randomization/noise. (Note that additive noising is not the only form of randomization that leads to privacy).

This is a recurring theme of privacy-preserving statistics, including private statistical inference (confidence intervals, hypothesis tests, Bayesian inference, and beyond) where one must reason about both sources of randomness, unlike the non-private case. Upon accounting for two sources of randomness, it so happens that in some simpler queries, differential privacy can be obtained for free (as in at a better error rate due to privacy noise than the error rate for estimating under-sampling noise in the non-private setting). We explain this with the following example for privately estimating the mean of a Bernoulli distribution from its samples.

EXAMPLE: MEAN OF BERNOULLI

Consider that one is given samples $X_1, \dots, X_n \sim \text{Bernoulli}(p), 0 \leq p \leq 1$ and the goal is to estimate p (mean) privately. The following is a simple private estimator obtained by additive noise from a Laplacian distribution calibrated based on the required privacy level and the global sensitivity of the query.

$$\hat{p} = \frac{1}{n} \sum_{i=1}^{n} X_i + \text{Lap}\left(\frac{1}{\epsilon n}\right)$$

Note that in here the global sensitivity of this query $\frac{1}{n} \sum X_i$ is $\frac{1}{n}$). Note that the absolute privacy error here is of order $\frac{1}{\epsilon n}$.

$$\hat{p} = \frac{1}{n} \sum_{i=1}^{n} X_i + \text{Lap}\left(\frac{1}{\epsilon n}\right)$$

$$\text{Var}\left(\text{Lap}\left(\frac{1}{\epsilon n}\right)\right) = \frac{2}{\epsilon^2 n^2}.$$

Similarly, the overall absolute error is of order $\frac{1}{\sqrt{n}}$.

$$\text{Var}(\hat{p}) = \frac{p(1-p)}{n} + \frac{2}{\epsilon^2 n^2} = O\left(\frac{1}{n}\right)$$

Let us now compare this to a non-private estimator

$$\tilde{p} = \frac{1}{n} \sum_{i} X_i, \mathbb{E}[\tilde{\rho}] = p, \quad \text{Var}(\tilde{p}) = \frac{p(1-p)}{n}$$

Moreover, by the Cramer-Rao Lower Bound (CRLB), any unbiased estimator (private or not) has variance at least $\frac{p(1-p)}{n}$. We tabulate this in Table 1, and as can be seen, the error rate caused by privacy is, in fact, lower than that caused by estimation in the non-private setting.

	Variance / mean squared error	Absolute Error (whp)
Private	$\frac{p(1-p)}{n} + \frac{2}{\epsilon^2 n^2}$	$O\left(\frac{1}{\sqrt{n}} + \frac{1}{\epsilon n}\right)$
Non-Private	$\frac{p(1-p)}{n}$	$O\left(\frac{1}{\sqrt{n}}\right)$
Lower bound	$\frac{p(1-p)}{n}$	$\Omega\left(\frac{1}{\sqrt{n}}\right)$

Table 1.1: Privacy (absolute) rate for free phenomenon

1.3 Differential privacy: interpretation as a privacy loss random variable

The privacy loss random variable (PLRV for short) is the "actual" ε value for a specific output O. It is a random variable because typically, we consider the attacker's loss (Bayesian viewpoint in section 4) $\mathcal{L}_{D_1,D_2}(O)$ when O varies according to $A\left(D_1\right)$, which we assume is the real database. We show the privacy loss random variable's distribution (as an instantiation) and its interpretation in terms of ϵ, δ in Figure 1.1,1.2, 1.3 and its formula is shown below.

$$\mathcal{L}_{D_1,D_2}(O) = \ln\left(\frac{\mathbb{P}\left[M\left(D_1\right) = O\right]}{\mathbb{P}\left[M\left(D_2\right) = O\right]}\right)$$

The distribution (called privacy loss distribution/PLD) of the PLRV is bounded within the absolute value of the privacy level ϵ. Moreover, the PLD of the composition of queries is the convolution of individual query PLDs. This can easily be implemented as follows based on this property. Let μ and μ' be any distributions on real numbers. Their convolution, denoted by $\mu * \mu'$, is a distribution on real numbers where a sample $t \sim \mu * \mu'$ is drawn by first independently sampling $a \sim \mu, a' \sim \mu'$ and then letting $t = a + a'$. Therefore, given $\mu_{up}, \mu'_{up}, \mu_{lo}$ and μ'_{lo} as

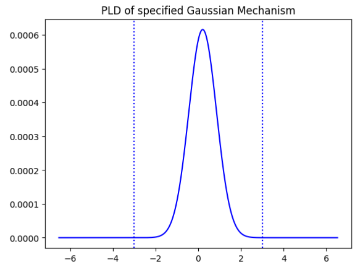


Figure 1.1: The distribution (called privacy loss distribution/PLD) of the PLRV[27]

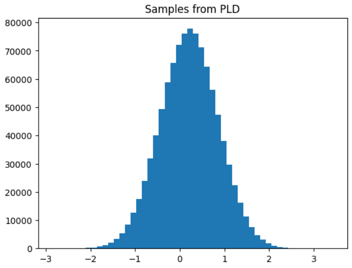


Figure 1.2: Samples from an instance of privacy loss distribution/PLD[27]

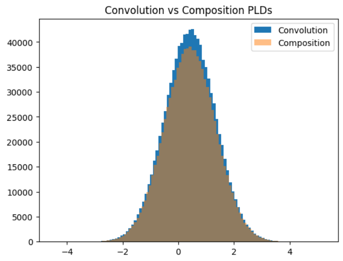


Figure 1.3: PLD of the composition of queries is the convolution of individual PLDs[27]

any distributions such that all of them are discrete or all of them are continuous. Then, we have

$$PLD_{\left(\mu_{up}\otimes\mu'_{up}\right)/\left(\mu_{lo}\otimes\mu'_{l_o}\right)} = PLD_{\mu_{up}/\mu_{lo}} * PLD_{\mu'_{up}/\mu'_{l_o}}$$

These notions help for tighter accounting of a privacy budget upon repeated queries and compositions of queries over a database and are in fact the inspiration behind modern variants of differential privacy.

1.4 Differential privacy: Interpretation as a divergence constraint

A mechanism $\mathcal{M}$ is (ε, δ)-differentially private (or $(\varepsilon, \delta) - DP$ for short) if and only if, for any neighbouring input datasets $\mathbf{x}, \mathbf{x}'$, it holds that $\mathfrak{D}_{e^{\varepsilon}}\left(\mathcal{M}(\mathbf{x}) \| \mathcal{M}\left(\mathbf{x}'\right)\right) \leq \delta$ using the hockey-stick divergence given by $\mathfrak{D}_{e^{\varepsilon}}\left(\mu \| \mu'\right) := \int \left[f_{\mu}(y) - e^{\varepsilon} \cdot f_{\mu'}(y)\right]_{+} dy$.

Moreover, for any two distributions μ_{up} and μ_{lo} where both are discrete or both are continuous, it holds that

$$\mathfrak{D}_{e^{\varepsilon}}\left(\mu_{up} \| \mu_{lo}\right) = \mathbb{E}_{y \sim PLD_{\mu_{up}/\mu_{l}}}\left[1 - e^{\varepsilon - y}\right]_{+}.$$

Due to this, a mechanism $\mathcal{M}$ is $(\varepsilon, \delta) - DP$ if and only if the following holds for all neighbouring input datasets $\mathbf{x}$ and $\mathbf{x}'$:

$$\delta \geq \mathbb{E}_{y \sim PLD_{\mathcal{M}(\mathrm{x})/\mathcal{M}(\mathrm{x}')}} \left[1 - e^{\varepsilon - y}\right]_{+}$$

1.5 Differential privacy: Interpretation as a hypothesis test

Given the output of a differentially private mechanism M, can one decide if Alice is in the dataset? This is the specific hypothesis test which has an equivalence to differential privacy.

1. Null Hypothesis H0: we observe $M(D) \Rightarrow$ Alice not in the dataset

2. Alternate Hypothesis H1: we observe $M\left(D'\right) \Rightarrow$ Alice in the dataset.

The adversary would want to minimize

1. Type I error: Alice is detected in the dataset while she's not

2. Type II error: Alice is not detected in the dataset while she is.

One would like to know a trade-off function of minimizing one type of error while keeping the other bounded below a threshold, as $T\left(\mathcal{M}(D), \mathcal{M}\left(D'\right)\right)(\alpha) = \inf\{$ type II error | type I error $\leq \alpha\}$. If a mechanism reveals some information, the adversary can discriminate $M(D)$ from $M\left(D'\right)$ with better odds (Type II error below the red line) and know something about Alice (whether she is in the dataset or not). A mechanism M being (ε, δ)-approximate-DP is equivalent to the following being true. For all the rejection rules,

$$\begin{cases} \Pr(\text{ Type I error }) + e^{\varepsilon} \Pr \ (\text{Type II error }) & \geq 1 - \delta \\ e^{\varepsilon} \Pr(\text{ Type I error }) + \Pr \ (\text{Type II error}) & \geq 1 - \delta \end{cases}$$

Which corresponds to the tradeoff function:

$$f_{\varepsilon,\delta}(\alpha) = \max\left\{0, 1-\delta-e^{\varepsilon}\alpha, e^{-\varepsilon}(1-\delta-\alpha)\right\}$$

Now, there is an equivalent notion of differential privacy called f-differential privacy specific to a choice of a trade-off function f. The following are the requirements for the trade-off function, its equivalence to differential privacy, and the Type-I and Type-II errors of any adversary.

Definition 1.5.1. *(Trade-off function[143]) For any two probability distributions Y and Y' on the same space, define the trade-off function $T\left(Y, Y'\right): [0,1] \rightarrow [0,1]$ as*

$$T\left(Y, Y'\right)(\alpha) = \inf\left\{\beta_\phi : \alpha_\phi \leq \alpha\right\}$$

where the infimum is taken over all (measurable) rejection rules ϕ.

The following gives the necessary and sufficient condition for f to be a trade-off function. Note that a function $f : [0,1] \rightarrow [0,1]$ is a trade-off function if and only if f is convex, continuous, non-increasing, and $f(x) \leq 1-x$ for $x \in [0,1]$ (as shown in the figure). f-DP allows the full trade-off between type I and type II errors in the simple hypothesis testing problem to be governed by a trade-off function f. A larger trade-off function implies stronger privacy guarantees.

Definition 1.5.2. *(f-differential privacy[143]). Let f be a trade-off function. A mechanism $\mathcal{M} : \mathcal{X} \rightarrow \mathcal{R}$ is f-differentially private if for every pair of neighboring datasets $x, x' \in \mathcal{X}$, we have*

$$T\left(\mathcal{M}(x), \mathcal{M}\left(x'\right)\right) \geq f$$

(ϵ, δ)*-DP is a special case of* f*-DP, taking* $f = f_{\epsilon,\delta}$*, where* $f_{\epsilon,\delta} = \max\{0, 1-\delta-\exp(\epsilon)\alpha, \exp(-\epsilon)(1-\delta-\alpha)\}$[558,143].

Trade-off function based interpretation of (ϵ, δ)

Based on the above concepts of trade-off functions, one can interpret ϵ as the slope formed by the approximation of a mechanism's trade-off function obtained by multiple choices of ϵ, δ. Similarly, δ is an additive slack that reduces the trade-off function (by reducing the errors) at the cost of loss of privacy with an (OR condition) probability of δ.

1.6 Variants of DP: higher accuracies from tighter budgeting (upon composition)

The composition of private mechanisms results in a loss of privacy budget. Initial estimates of this rate of composition were quite conservative, followed by the advanced composition theorem[268], which improved the rate to the following.

Definition 1.6.1. *(Advanced Composition). If* $M_1, \ldots, M_k$ *are* ϵ*-DP and* $M(x) = M_k\left(x, M_{k-1}\left(x, M_{k-2}(x, \ldots)\right)\right.$ *), then* $M(x)$ *is* $(\epsilon', \delta) - DP$ *for all* δ *and* $\epsilon' = \sqrt{2k \log 1/\delta} + k\left(e^{\epsilon} - 1\right)$.

However, composition bounds using (ϵ, δ)-DP are loose even for advanced composition. These bounds were improved *drastically* along with a more interpretable formula of composition via the introduction of 3 variants of differential privacy that are equivalent:

1. Rényi differential privacy[366]

2. zero-concentrated differential privacy[81]

3. f-differential privacy[143]

On a high level, both zCDP and RDP guarantee that the “distance” (technically, the Rényi divergence) between the distributions of $M(D)$ and $M(D')$ is below a certain threshold, for any two neighbors D and D' . Intuitively, since the two distributions are close, it is improbable for an attacker to deduce which of the neighbouring datasets was used by the algorithm. These two definitions do not allow for catastrophic failures and are stronger than approximate DP. Specifically, any zCDP or RDP guarantee can be converted to an approximate DP guarantee.

1.6.1 Renyi-DP

A randomized mechanism F satisfies $(\alpha, \bar{\epsilon})$-RDP if for all neighboring datasets D and D'

$$D_\alpha\left(F(D) \| F\left(D'\right)\right) \leq \bar{\epsilon}$$

In other words, RDP requires that the Rényi divergence of order α between $F(D)$ and $F\left(D'\right)$ to be bounded by $\bar{\epsilon}$. Note that one needs to use $\bar{\epsilon}$ to denote the ϵ parameter of RDP in order to distinguish it from the ϵ in pure ϵ-differential privacy and (ϵ, δ)-differential privacy. A key property of Rényi differential privacy is that a mechanism which satisfies RDP also satisfies (ϵ, δ) -differential privacy. Specifically, if F satisfies $(\alpha, \bar{\epsilon})$-RDP, then for $\delta > 0, F$ satisfies (ϵ, δ)-differential privacy for $\epsilon = \bar{\epsilon} + \frac{\log(1/\delta)}{\alpha-1}$. The analyst is free to pick any value of δ; a meaningful value (e.g. $\delta \leq \frac{1}{n^2}$) should be picked in practice. The major advantage of Rényi differential privacy is the tight composition for the Gaussian mechanism, and this advantage in composition comes without the need for a special advanced composition theorem. The sequential composition theorem of Rényi differential privacy states that if F_1 satisfies $(\alpha, \overline{\epsilon_1})$-RDP and F_2 satisfies $(\alpha, \overline{\epsilon_2})$-RDP - Then their composition satisfies $(\alpha, \overline{\epsilon_1} + \overline{\epsilon_2})$-RDP Based on this sequential composition theorem, running an $(\alpha, \bar{\epsilon})$-RDP mechanism k times results in $(\alpha, k\bar{\epsilon})$-RDP. For a given level of noise (i.e. a given value for σ^2), bounding the privacy cost of repeated applications of the Gaussian mechanism using RDP's sequential composition and then converting to (ϵ, δ)-differential privacy, will usually yield a much lower privacy cost than performing the composition directly in (ϵ, δ) world (even with advanced composition).

As a result, the ideas behind Rényi differential privacy have been used to improve significantly the privacy cost accounting in several recent iterative algorithms, including Google's differentially private version of TensorFlow.

INTERPRETATION OF RENYI DP USING PRLV

Rényi-DP can be interpreted in terms of Bayesian loss of attacker and privacy loss random variable. These two concepts were introduced in the previous sections above. Rényi-DP leads to an averaging of the PRLV with more extreme events being penalized based on the choices of α, the order of the used Rényi-divergence in controlling the divergence between the pdfs of the outputs of the privacy mechanism across neighbouring databases. This increases the Bayesian attacker's loss accordingly

$$\underset{O \sim A(D_1)}{\mathbb{E}} \left[\left(e^{\mathcal{L}_{D_1, D_2}(O)} \right)^{(\alpha-1)} \right] \leq \left(e^{\varepsilon} \right)^{(\alpha-1)}$$

This is Rényi differential privacy. If A satisfies the above inequality for all choices of D_1 and D_2, we say it's (α, ε)-Rényi differentially private. Some special values of α correspond to common averaging functions. - $\alpha \to 1$ bounds the arithmetic mean of $\mathcal{L}$ or, equivalently, the geometric mean of $e^{\mathcal{L}}$; - $\alpha = 2$ bounds the arithmetic mean of $e^{\mathcal{L}}$; - $\alpha = 3$ bounds the quadratic mean of $e^{\mathcal{L}}$; $\alpha = 4$ bounds the cubic mean of $e^{\mathcal{L}}$; - and it's also possible to pick $\alpha = \infty$, which bounds the maximum value of $e^{\mathcal{L}}$: it's then equivalent to ε-DP.

1.6.2 ZERO-CONCENTRATED DIFFERENTIAL PRIVACY (ZCDP)

This variant allows for covering all values of α (as used in Renyi DP) at once!

That's precisely what zero-concentrated differential privacy (zCDP) provides. Introduced by Mark Bun & Thomas Steinke, it can be interpreted in simple terms: given a single parameter ρ, the ε corresponding to each α must be at most $\rho\alpha$. In the formalism above, the mechanism is ρ zCDP if:

$$\mathop{\mathbb{E}}_{O \sim A(D_1)} \left[\left(e^{\mathcal{L}_{D_1,D_2}(O)} \right)^{(\alpha-1)} \right] \leq \left(e^{\rho\alpha} \right)^{(\alpha-1)}$$

It's easy to verify that it matches all the requirements above. 1. The single parameter ρ corresponds to the arithmetic average of the privacy loss. (Or, equivalently, to the geometric average of the $e^{\mathcal{L}}$.) 2. It guarantees that the relationship between α and ε is at most linear, which is very simple. 3. It describes the Gaussian mechanism beautifully. Suppose the statistics you're computing have a L^2 sensitivity of Δ. Then, add adding Gaussian noise of variance σ^2 to the result. Then the result satisfies $\rho-$zCDP, with $\rho = \frac{\Delta^2}{2\sigma^2}$. So much nicer than the formula giving the $(\varepsilon, \delta) -$ DP guarantee! 4. And composition is a breeze. If a mechanism is $\rho_1 -$ zCDP and another is $\rho_2 -$ zCDP, then publishing the result of both is $(\rho_1 + \rho_2) - zCDP$.

1.6.3 DP PREVENTS MEMORIZATION

One critical benefit of differential privacy is that it provably prevents memorization. The requirement that private algorithms perform similarly on neighbouring databases constrains the algorithm from overfitting to individual entries in the database. It thus ensures that no single entry has been memoized. This guarantee also provides strong generalization guarantees for differentially private algorithms, which have also been observed in other machine learning applications. Moreover, foundations of differential privacy and obfuscation are the driving forces behind recent algorithms for "Machine Unlearning" to implement the ad hoc "right to forget/delete sensitive information" in a trained model.

1.7 Edge-cases, boundaries and limitations

1.7.1 Universal optimality

We refer to a DP mechanism as universally optimal for a given kind of query if no other DP mechanism provides higher utility. It has been shown that all linear queries (including counting queries) have universally optimal DP mechanisms, while universal optimality for nonlinear queries is trickier. For example, in density estimation queries, the K-norm mechanism can achieve a minimax rate only in 1-D. This opens room for novel privacy mechanisms that improve utility towards being closer to universally optimal for nonlinear queries.

For binary data, the staircase mechanism is universally optimal only in low and high privacy regimes with respect to f-divergence metrics and all monotonic losses. For continuous linear queries, the Laplacian mechanism is universally optimal. The geometric mechanism is universally optimal for counting queries based on Euclidean metric and all monotonic losses. At the same time, there has been a proven impossibility of universal optimality for non-counting queries.

1.7.2 Choice of δ

Since δ controls the strength of the relaxation, it is important to ensure that a sufficiently small is used. The general recommendation in the literature is to choose $\delta \leq 1/n$, where n is the number of records in the dataset[160]. This recommendation stems from a worst-case analysis. Specifically, consider the following worst-case assumption on every record: if the record r is present in the dataset, the (ϵ, δ) mechanism will generate a certain output Er with probability δ, and furthermore, Er cannot happen otherwise. If an attacker observes Er, they can directly deduce that the record r is in the dataset. Thus, each record in the dataset has a probability δ of being successfully identified by the attacker in this worst-case scenario. The expected number of successful attacks is δn. Choosing $\delta \leq 1/n$, will ensure that the expected number of successful attacks is much smaller than 1.

1.7.3 Independence issues

We believe that under any reasonable formalization of evidence of participation, such proof can be encapsulated by exactly one tuple only when all tuples are independent (but not necessarily generated from the same distribution). We believe this independence assumption is a good rule of thumb when considering the applicability of differential privacy. Still, it leads to more than needed (more conservative) privacy budgeting when there is dependence across records. This makes privacy analysis and design of mechanisms quite challenging for spatial, temporal and spatio-temporal data. Quite a few mechanisms have been recently developed for this regime, but it is still a widely open area of research requiring a lot more work.

1.7.4 Explicit identifiability

For queries requiring explicit identifiability, such as Fraud Detection, DP is not a privacy solution as its goal is to prevent identifiability. DP is not a silver bullet for every situation, although it covers a wide range of scenarios.

1.7.5 Systems security issues

From a hardware and systems perspective, it is to be noted that DP mechanisms, when coded as part of hardware, lead to other issues that need to be taken care of for a proper deployment. These include

1. Timing Attacks (Systems Security) can lead to identifiability.

2. Latency time profiles of users could lead to identifiability.

3. Floating-point precisions of noise used reveal supplementary information that helps leakage.

4. Formal verification of mechanisms is needed to ensure it has been coded and integrated correctly.

For example, DP mechanisms can be designed to inject artificial latency such that the measured timings are indistinguishable to prevent identifiability[254]. On floating-point attacks (meta info leaked via precision used by a client), these have already been prevented in some DP-based queries through[85]. However, this needs to be generalized to other base mechanisms. For those interested in understanding why the float point implementation of the naive Laplace mechanism destroys differential privacy, the work in[365] explains this vulnerability.

1.8 Categorization of threats, other privacy and security methods for deep learning[361]

The success of Deep Neural Networks (DNNs) in various fields, including vision, medicine, recommendation systems, natural language processing, etc., has resulted in their deployment in numerous production systems[301,278,269,479]. In medicine, learning is used to find patterns in patient histories and recognize abnormalities in medical imaging, which help with disease diagnosis and prognosis. The use of machine learning in healthcare can compromise patient privacy by exposing the patient's genetic markers, as Fredrikson et al.[181]. Among many other applications, deep learning is also widely used in finance for predicting prices or creating portfolios. In these cases, usually, an entity trains its model, and the model parameters are considered confidential. Being able to find or infer them is regarded as a breach of privacy[65]. Ease of access to large datasets and high computational power (GPUs and TPUs) have paved the way for the aforementioned advances. These datasets are usually crowdsourced and might contain sensitive information. This poses serious privacy concerns, as neural networks are used in different aspects of our lives[508,181,387,182,452,459].

Figure 1.4 classifies possible threats to deep learning. One threat is the direct intentional or unintentional exposure of sensitive information through untrusted data curator, communication link, or cloud[110,28]. This information can be the training data, inference queries, model parameters, or hyperparameters. If we assume that information cannot be attained directly, there is still the threat of information exposure through indirect inference. Membership inference attacks[474] can infer whether a given data instance was part of a model's training process. Model inversion and attribute inference attacks can infer sensitive features about a data instance from observed predictions of a trained model and other non-sensitive features of that data instance[180,584]. Some attacks are targeted towards stealing information about a deployed model, such as its architecture[576], trained parameters[512] or a general property of the data it was trained on, for instance, if the images used for training were all taken outdoor[186].

There is a myriad of methods proposed to tackle these threats. Most of these methods focus on the data aggregation/dataset publishing and training stages of deep learning. We classify these methods into three classes. The first class of methods focuses on sanitizing the data and removing sensitive information from it while maintaining the statistical trends[494,157]. The second class focuses on making the DNN training phase private and protecting the data used for training[473,1,400,213,224,601]. The last class, of which there is only a handful of works, attempts to protect the privacy of the test-time inference phase by protecting the input data (request) that the user sends to a deployed DNN[397,146,359].

In this paper, we first briefly discuss existing attacks and privacy threats against deep learning. Then, we focus on the existing privacy-preserving methods for deep learning and demonstrate a gap in the literature regarding test-time inference privacy. There are a few other security vulnerabilities which can be exploited in a deep learning model, such as adversarial attacks[92], data poisoning[59]. This work focuses only on privacy-specific vulnerability, and other such attacks are out of the scope of this paper.

1.9 Existing Threats

In this section, we map the space of existing threats against privacy in deep learning and machine learning in general. While this survey focuses on privacy-preserving techniques, we provide a brief summary of attacks to situate the need for privacy protection better. Figure 1.4 shows the landscape of these threats, which we have divided into two main categories of direct and indirect information exposure hazards. Direct threats are those where the attacker can access the information. In indirect attacks, however, the attacker tries to infer or guess the information and does not have access to the actual information.

1.9.1 Direct Information Exposure

Direct intentional or unintentional data breaches can occur in many different settings and are not limited to machine learning. Dataset breaches through data curators or entities housing the data can be caused unintentionally by hackers, malware, viruses, or social engineering by tricking individuals into handing over sensitive data to adversaries[110]. A study by Intel Security[456] demonstrated that employees are responsible for 43% of data leakage, half of which is believed to be unintentional. A malicious party can exploit a system's backdoor to bypass a server's authentication mechanism and gain direct access to sensitive datasets or sensitive parameters and models[557,322,282]. For instance, the recent hacking of Equifax exploited a vulnerability in

Table 1.2: Properties of some notable attacks against machine learning privacy. MIA denotes Model Inversion Attack in the table.

Attack	Membership Inference	Model Inversion	Hyperparam Inference	Parameter Inference	Property Inference	Access to Model	Access to Output
Membership Inference[474]	●	○	○	○	○	Blackbox	Logits
Measuring Membership Privacy[331]	●	○	○	○	○	Blackbox	Logits
ML-Leaks[444]	●	○	○	○	○	Blackbox	Logits
The Natural Auditor[483]	●	○	○	○	○	Blackbox	Label
LOGAN[218]	●	○	○	○	○	Both	Logits
Data Provenance[484]	●	○	○	○	○	Blackbox	Logits
Privacy Risk in ML[584]	●	●	○	○	○	Whitebox	Logits+Auxilary
Fredrikson et al.[181]	○	●	○	○	○	Blackbox	Logits
MIA w/ Confidence Values[180]	○	●	○	○	○	Both	Logits
Adversarial NN Inversion[580]	○	●	○	○	○	Blackbox	Logits
Updates-Leak[443]	○	●	○	○	○	Blackbox	Logits
Collaborative Inference MIA[221]	○	●	○	○	○	Both	Logits
The Secret Sharer[86]	○	○	○	○	●	Blackbox	Logits
Property Inference on FCNNs[186]	○	○	○	○	●	Whitebox	Logits
Hacking Smart Machines w[37]	○	○	○	○	●	Whitebox	Logits
Cache Telepathy[576]	○	○	○	●	○	Blackbox	Logits
Stealing Hyperparameters[552]	○	○	○	●	○	Blackbox	Logits
Stealing ML Models[512]	○	○	●	●	○	Blackbox	Label

the Apache Struts software, which was used by Equifax[557].

Data sharing by transmitting confidential data without proper encryption is an example of data exposure through communication link[564]. Kaspersky Labs reported in 2018 that they found four million Android apps were sending unencrypted user profile data to advertisers' servers[521]. Private data can also be exposed through the cloud service that receives it to run a process, for instance, Machine Learning as a Service (MLaaS). Some of these services do not clarify what happens to the data once the process is finished, nor do they even mention that they are sending the user's data to the cloud and not processing it locally.

1.9.2 Indirect (Inferred) Information Exposure

As shown in figure 1.4, we categorize indirect attacks into 5 main groups: membership inference, model inversion, hyperparameter inference, parameter inference, and property inference attacks. Table 10.1a summarises different attacks and their properties. The "Access to Model" column determines whether the attack needs white-box or black-box access to the model to successfully mount. White-box access assumes access to the full target model. In contrast, the

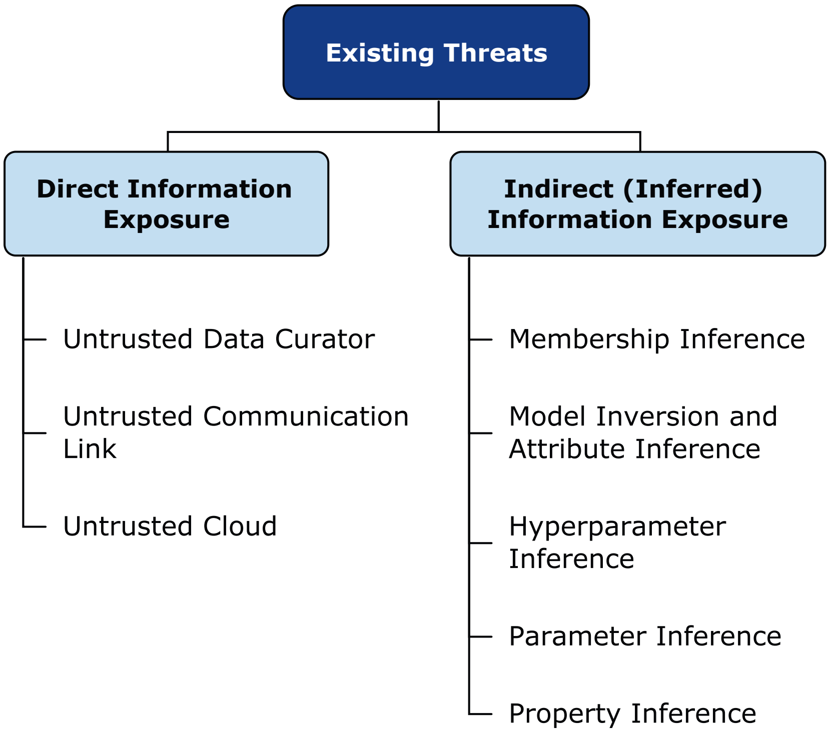

Existing Threats
Direct Information Exposure
Indirect (Inferred) Information Exposure
Untrusted Data Curator
Untrusted Communication Link
Untrusted Cloud
Membership Inference
Model Inversion and Attribute Inference
Hyperparameter Inference
Parameter Inference
Property Inference

Figure 1.4: Categorization of existing threats against deep learning

black box assumes only query access to the model without knowledge of the architecture or parameters of the target model. The last column shows whether the attacker needs access to the output confidence values of the model (the probabilities, logits) or whether only the predicted labels suffice.

Membership Inference

Given a data instance and (black-box or white-box) access to a pre-trained target model, a membership inference attack speculates whether or not the given data instance has contributed to the training step of the target model. Shokri et al.[474] propose the first membership inference attack on machine learning. They consider an attacker with black-box query access to the target model and can obtain the queried input's confidence scores (probability vector). The attacker uses this confidence score to deduce the participation of given data in training. They first train shadow models on a labelled dataset that can be generated using three methods: model inversion attack (we will see next), statistics-based synthesis (through assumptions about the underlying distribution of the training set), or noisy real data. Using these shadow models, the attacker trains an "attack model" that distinguishes the participation of a data instance in the training set of the shadow models. Lastly, for the main inference attack, the attacker queries the target deployed model to receive confidence scores for each given input data instance and infers whether or not the input was part of the target training data. This attack is built on the assumption that if a record were used in a model's training, it would yield a higher confidence score than a record not seen before by the model.

Some studies [437,514,584] attribute membership inference attacks to the generalization gap, the over-fitting of the model, and the data memorization capabilities of neural networks. Deep neural networks have been shown to memorize the training data[30,352,249] rather than learning the latent properties of the data, which means they often tend to over-fit the training data. Long et al. [331] propose an approach that more accurately tests a given instance's membership. They train the shadow models with and without this given instance, and then at inference time, the attacker tests to see if the example was used for training the target model, similar to Shokri et al.'s approach. More recently, Salem et al.[444] propose a more generic attack that could relax the main requirements in previous attacks (such as using multiple shadow models, knowledge of the target model structure, and having a dataset from the same distribution as the target model's training data), and show that such attacks are also applicable at a lower cost, without significantly degrading their effectiveness.

Membership inference attacks do not always need access to the confidence values (logits) of the target model, as shown by Song & Shmatikov in a recent attack[483], which can detect with very few queries to a model if a particular user's texts were used to train it.

Yeom et al. [584] suggest a membership inference attack for cases where the attacker can have white-box access to the target model and know the average training loss of the model. In this attack, for an input record, the attacker evaluates the loss of the model, and if the loss is smaller than a threshold (the average loss on the training set), the input record is deemed part of the training set. Membership inference attacks can also be applied to Generative Adversarial Networks (GANs), as shown by Hayes et al.[218].

Model Inversion and Attribute Inference

Model inversion and attribute inference attacks are against attribute privacy, where an adversary tries to infer sensitive attributes of given data instances from a released model and the instance's non-sensitive attributes[567]. The most prominent of these attacks is against a publicly-released linear regression model, where Fredrikson et al.[181] invert the model of a medicine (Warfarin) dosage prediction task. They recover genomic information about the patient based on the model output and several other non-sensitive attributes (e.g., height, age, weight). This attack can

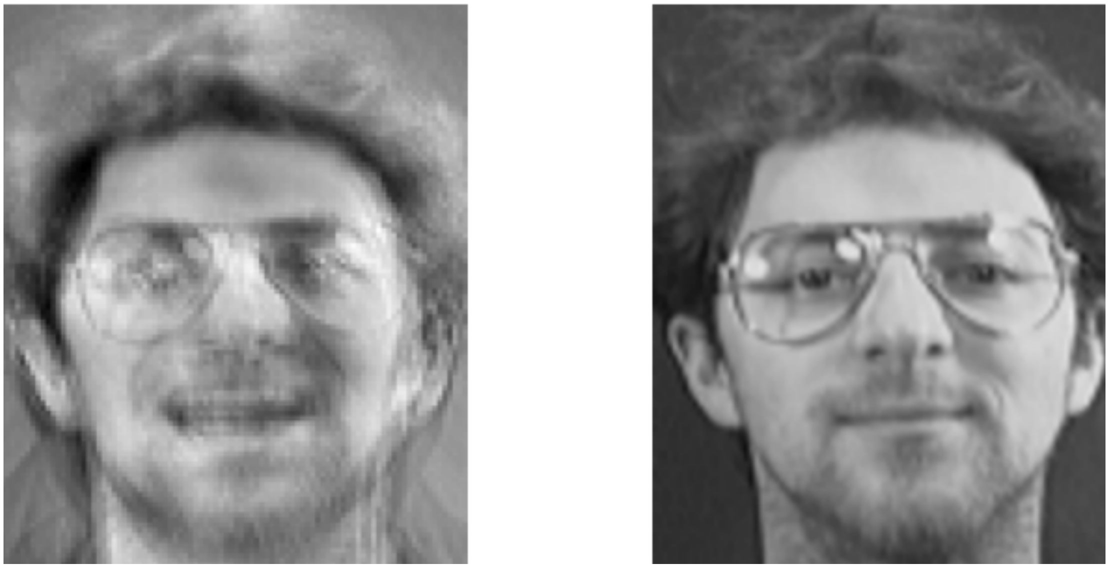


Figure 1.5: The image on the left was recovered using the model inversion attack of Fredrikson et al.[180]. The image on the right shows an image from the training set. The attacker is given only the person's name and access to a facial recognition system that returns a class confidence score[180].

only be applied to the target model with black-box API access. Fredrikson et al. formalize this attack as maximizing the posterior probability estimate of the sensitive attribute. In other words, the attacker assumes that features f_1 to f_{d-1} of the f_d features of each data instance are non-sensitive. The attacker then tries to maximize the posterior probability of feature f_d, given the nonsensitive features of f_1 to f_{d-1}, and the model output.

In another work, given white-box access to a neural network, Fredrikson et al.[180] show that they could extract training data instances from observed model predictions. Figure 1.5 shows a recovered face image that is similar to the input image and was reconstructed by utilizing the confidence score of the target model. Yeom et al.[584] also propose an attribute inference attack, built upon the same principle used for their membership inference attack, mentioned in Section 1.9.2. The attacker evaluates the model's loss on the input instance for different values of the sensitive attribute and infers the value that yields a loss value similar to that outputted by the original data as the sensitive value. Salem et al.[443] suggest a model inversion attack on online learning using a generative adversarial network based on the difference between a model before and after a gradient update. More recently, He et al.[221] propose a new set of attacks to compromise the privacy of test-time inference queries in collaborative deep learning systems where a DNN is split and distributed to different participants. This scheme is called split learning[209]. They demonstrate that with their attack, one malicious participant can recover an arbitrary input fed into this system, even without access to other participants' data or computations.

Model Stealing: Hyperparameter and Parameter Inference

Trained models are considered intellectual properties of their owners and can be regarded as confidential in many cases[65]; therefore, extracting the model can be viewed as a privacy breach. Apart from this, as discussed earlier, DNNs are shown to memorize information about their training data; therefore, exposing the model parameters could lead to exposure of training data. A model stealing attack is meant to recover the parameters via black-box access to the target model. Tramer et al.[512] devise an attack that finds the parameters of a model given the observation of its predictions (confidence values). Their attack tries to see model parameters through equation solving based on input-output pairs. This attack cannot be mounted in a setting where the confidence values are not provided.

Hyperparameter stealing attacks try to find the hyperparameters used during the model training, such as the regularization coefficient [552] or model architecture[576].

Property Inference

This class of attacks tries to infer specific patterns of information from the target model. An example of these attacks is the memorization attack that aims to find sensitive patterns in the training data of a target model[86]. These attacks have been mounted on Hidden Markov Models

(HMM) and Support Vector Machines (SVM) [37] and neural networks[186].

1.10 Privacy-Preserving Mechanisms

This section reviews the literature on privacy-preserving mechanisms for deep learning and machine learning in general. Figure 1.6 shows our classification of the landscape of this field. We divide the literature into three main groups. The first is private data aggregation methods, which aim at collecting data and forming datasets while preserving the privacy of the contributors[494,157]. The second group, comprised of a large body of work, focuses on devising mechanisms that make the training process of models private so that sensitive information about the participants of the training dataset is not exposed. Finally, the last group aims at the test-time inference phase of deep learning. It tries to protect the privacy of users of deployed models, who send their data to a trained model for having a given inference service carried out.

1.10.1 Data Aggregation

Here, we introduce the most prominent data privacy-preserving mechanisms. Not all these methods are applied to deep learning, but we briefly discuss them for the sake of comprehensiveness. These methods can be broadly divided into two groups context-free privacy and context-aware. Context-free privacy solutions, such as differential privacy, are unaware of the specific context or the purpose that the data will be used for. Whereas context-aware privacy solutions, such as information-theoretic privacy, are aware of the context where the data is going to be used, and can achieve an improved privacy-utility tradeoff[235].

Naive Data Anonymization

What we mean by naive anonymization in this survey is the removal of identifiers from data, such as the names, addresses, and full postcodes of the participants, to protect privacy. This method was used for protecting patients while processing medical data and has been shown to fail on many occasions[383,494,227]. Perhaps the most prominent failure is the Netflix prize case, where Narayanan & Shmatikov apply their de-anonymization technique to the Netflix Prize dataset. This dataset contains anonymous movie ratings of 500,000 subscribers on Netflix. They showed that an adversary with auxiliary knowledge (from the publicly available Internet Movie Database records) about individual subscribers can easily identify the user and uncover potentially sensitive information[383].

K-Anonymity

A dataset has a k-anonymity property if each participant's information cannot be distinguished from at least $k-1$ other participants whose information is in the dataset[494]. K-anonymity means that for any given combination of attributes that are available to the adversary (these attributes are called quasi-identifiers), there are at least k rows with the exact same set of attributes. K-anonymity has the objective of impeding re-identification. However, k-anonymization has been shown to perform poorly on the anonymization of high-dimensional datasets[10]. This has led to privacy notions such as l-diversity[337] and t-closeness[316], which are out of the scope of this survey.

Semantic Security and Encryption

Semantic security[200] (computationally secure) is a standard privacy requirement of encryption schemes which states that the advantage (a measure of how successfully an adversary can attack a cryptographic algorithm) of an adversary with background information should be cryptographically small. Semantic security is theoretically possible to break but it is infeasible to do so by any known practical means[389]. Secure Multiparty Computation (SMC), which we discuss in Section 1.10.2, is based on semantic security definition[325].

Information-Theoretic Privacy

Information-theoretic privacy is a context-aware privacy solution. Context-aware solutions explicitly model the dataset statistics, unlike context-free solutions that assume worst-case dataset statistics and adversaries. There is a body of work studying information-theoretic-based methods for both privacy and fairness, where privacy and fairness are provided through information degradation, through obfuscation or adversarial learning and demonstrated by mutual information reduction [138,414,235,264,533,233,311,362,435,569,574]. Huang et al. introduce a context-aware privacy framework called generative adversarial privacy (GAP), which leverages generative adversarial networks (GANs) to generate privatized datasets. Their scheme comprises a sanitizer that tries to remove private attributes and an adversary that tries to infer them[235]. They show that the privacy mechanisms learned from data (in a generative adversarial fashion) match the theoretically optimal ones.

1.10.2 Training Phase

The literature surrounding private training of deep learning and machine learning can be categorized based on the guarantee that these methods provide, which is most commonly either based on differential privacy or semantic security and encryption. Privacy using encryption is achieved by doing computation over encrypted data. The two most common methods for this are Homomorphic Encryption (HE) and Secure Multi-Party Computation (SMC).

Table 1.3: Categorization of some notable privacy-preserving mechanisms for training. In the table, the following abbreviations have been used: ERM for Empirical Risk Minimization, GM for Generative Model, AE for Auto Encoder, LIR for Linear Regression, LOR for Logistic Regression, LM for Linear Means, FLD for Fishers Linear Discriminant, NB for Naive Bayes and RF for Random Forest.

Method	**DP**	**SMC**	**HE**	**Dataset(s)**	**Task**
DPSGD[1]	●	○	○	MNIST, CIFAR-10	Image Classification w/ DNN
DP LSTM[350]	●	○	○	Reddit Posts	Language Model w/ LSTMs
DP LOR[103]	●	○	○	Artificial Data	Logistic Regression
DP ERM[104]	●	○	○	Adult, KDD-99	Classification w/ ERM
DP GAN[573]	●	○	○	MNIST, MIMIC-III	Data Generation w/ GAN
DP GM[8]	●	○	○	MNIST, CDR, TRANSIT	Data Generation w/ GM
DP AE[409]	●	○	○	Health Social Network Data	Behaviour Prediction w/ AE
DP Belief Network[410]	●	○	○	YesiWell, MNIST	Classification w/ DNN
Adaptive Laplace Mechanism[411]	●	○	○	MNIST, CIFAR-10	Image Classification w/ DNN
PATE[400]	●	○	○	MNIST, SVHN	Image Classification w/ DNN
Scalable Learning w/ PATE[401]	●	○	○	MNIST, SVHN, Adult, Glyph	Image Classification w/ DNN
DP Ensemble[213]	●	○	○	KDD-99, UCI-HAR, URLs	Classification w/ ERM
SecProbe[605]	●	○	○	US, MNIST, SVHN	Regress. & Class. w/ DNN
Distributed DP[44]	●	○	○	eICU, TCGA	Classification w/ DNN
DP model publishing[589]	●	○	○	MNIST, CIFAR	Image Classification w/ DNN
DP federated learning[194]	●	○	○	MNIST	Image Classification w/ DNN
ScalarDP, PrivUnit[57]	●	○	○	MNIST, CIFAR	Image Classification w/ DNN
DSSGD[473]	●	○	○	MNIST, SVHN	Image Classification w/ DNN
Private Collaborative NN[100]	●	●	○	MNIST	Image Classification w/ DNN
Secure Aggregation for ML[67]	○	●	○	-	Federated Learning
QUOTIENT[11]	○	●	○	MNIST, Thyroid, Credit	Classification w/ DNN
SecureNN[550]	○	●	○	MNIST	Image Classification w/ DNN
ABY3[371]	○	●	○	MNIST	LIR, LOR, NN
Trident[418]	○	●	○	MNIST, Boston Housing	LIR, LOR, NN
SecureML[372]	○	●	●	MNIST, Gisette, Arcene	LIR, LOR, NN
Deep Learning w/ AHE[412]	○	○	●	MNIST	Image Classification w/ DNN
ML Confidential[202]	○	○	●	Wisconsin Breast Cancer	LM, FLD
Encrypted Statistical ML[36]	○	○	●	20 datasets from UCI ML	LOR, NB, RF
CryptoDL[224]	○	○	●	MNIST, CIFAR-10	Image Classification w/ DNN
DPHE[586]	○	○	●	Caltech101/256, CelebA	Image Classification w/ SVM

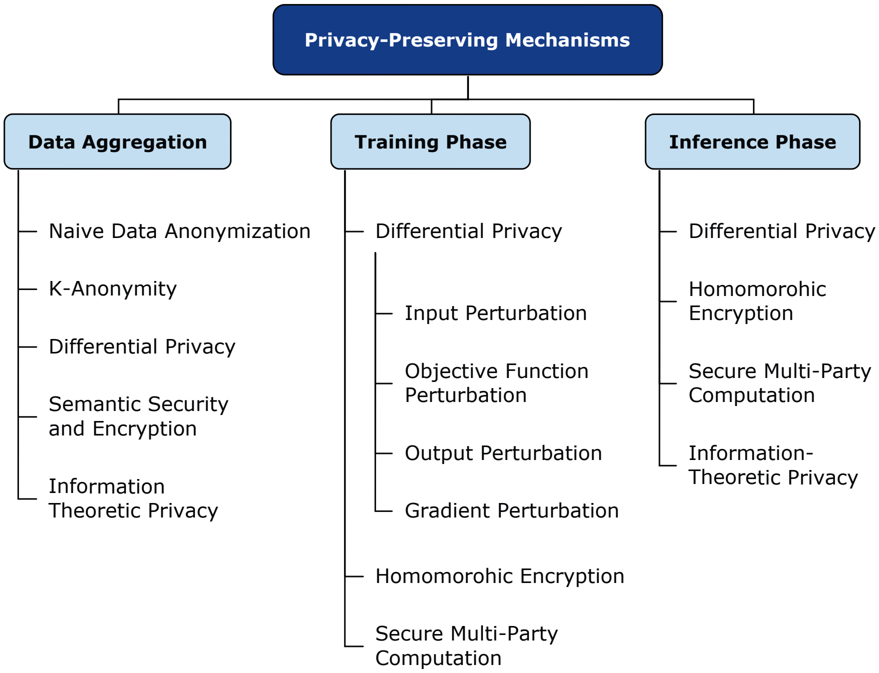


Figure 1.6: Categorization of privacy-preserving schemes for deep learning.

Homomorphic Encryption (HE). HE[192] allows computation over encrypted data. A client can send their data, in an encrypted format, to a server, and the server can compute over this data without decrypting it and then send a ciphertext (encrypted result) to the client for decryption. HE is extremely compute-intensive and is therefore not yet deployed in many production systems[427,338].

Secure Multi-Party Computation (SMC). SMC attempts to design a network of computing parties (not all of which the user necessarily has to trust) that carry out a given computation and make sure no data leaks. Each party in this network has access to only an encrypted part of the data. SMC ensures that as long as the owner of the data trusts at least one of the computing systems in the network, their input data remains secret. Simple functions can easily be computed using this scheme. Arbitrarily complex function computations can also be supported but with an often prohibitive computational cost[338].

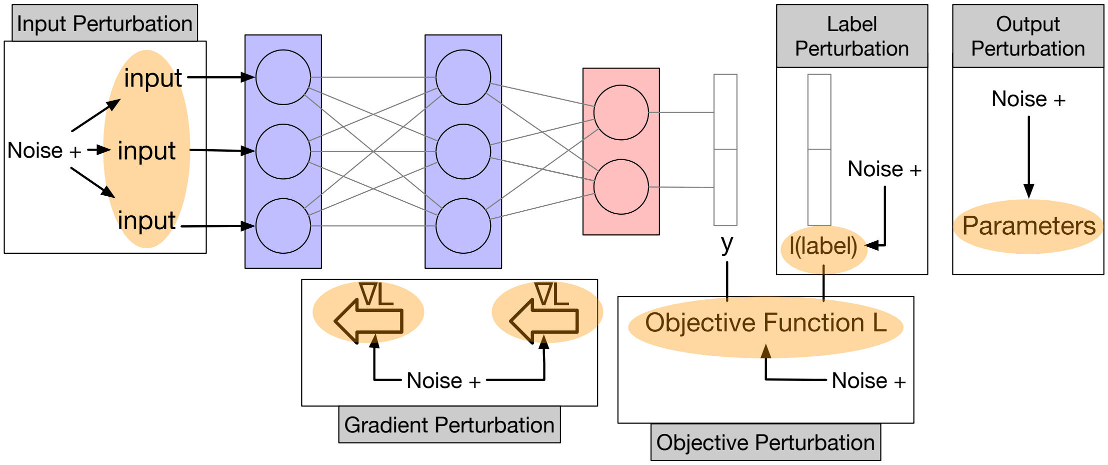


Figure 1.7: Overview of how a deep learning framework works and how differential privacy can be applied to different parts of the pipeline.

In this survey, we divided the literature on private training into three groups of methods that employ 1) Differential Privacy (DP), 2) Homomorphic Encryption (HE), and 3) Secure Multi-Party Computation (SMC). Table 1.3 shows this categorization for the literature we discuss in this section.

Differential Privacy

This section briefly discusses methods for modifying deep learning algorithms to satisfy differential privacy. Figure 1.7 shows an overview of a deep learning framework. As can be seen, the randomization required for differential privacy (or the privacy-preserving noise) can be inserted in five places: to the input, to the loss/objective function, to the gradient updates, to the output (the optimized parameters of the trained model) and the labels[400].

Input perturbations can be considered equivalent to using a sanitized dataset (discussed in Section 1.10.1) for training. **objective function perturbation** and **output perturbation** are explored for machine learning tasks with convex objective functions. For instance in the case of logistic regression, Chaudhuri et al. prove that objective perturbation requires sampling noise in the scale of $\frac{2}{n\epsilon}$, and output perturbation requires sampling noise in the scale of $\frac{2}{n\lambda\epsilon}$, where n is the number of samples and λ is the regularization coefficient[104]. More recently, Iyengar et al. [246] propose a more practical and general objective perturbation approach and benchmark it using high-dimensional real-world data. In deep learning tasks, due to the non-convexity of the objective function, calculating the sensitivity of the function (which is needed to determine the intensity of the added noise) becomes non-trivial. One solution is replacing the non-convex function with an approximate convex polynomial function [409,410,411] and then using objective function perturbation. This approximation limits the capabilities and the utility that a conventional DNN would have. Given the discussed limitations, **gradient perturbation** is the approach that is widely used for private training in deep learning. Applying perturbations on the gradients requires the gradient norms to be bounded since, in deep learning tasks, the gradient could be unbounded. Clipping is usually used to alleviate this issue.

Shokri et al. showed that deep neural networks can be trained in a distributed manner and with perturbed parameters to achieve privacy[473]. Still, their implementation requires ϵ proportional to the size of the target model, which can be in the order of a couple of millions. Abadi et al.[1] propose a mechanism dubbed the "moments accountant (MA)" for bounding the cumulative privacy budget of sequentially applied differentially private algorithms over deep neural networks. The moments accountant uses the moment-generating function of the privacy loss random variable to keep track of a bound on the privacy loss during composition. MA operates in three steps: first, it calculates the moment generating functions for the algorithms A_1, A_2,.., which are the randomizing algorithms. It then composes the moments together through a composition theorem and, finally, finds the best leakage parameter (δ) for a given privacy budget of ϵ. The moments accountant is widely used in different DP mechanisms for private deep learning. Papernot et al. use MA to aid in bounding the privacy budget for their teacher ensemble method that uses noisy voting and **label perturbation**[400,401]. MA is also employed by the works[44,589,194,573,8,57], all of which use perturbed gradients.

More recently, Bu et al. apply the Gaussian Differential Privacy (GDP) notion introduced by Dong et al.[142] to deep learning[80] to achieve a more refined analysis of neural network training, compared to that of Abadi et al.[1]. They analyze the privacy budget exhaustion of private DNN training using Adam optimizer without the need to develop sophisticated techniques such as the moments accountant. They demonstrate that GDP allows for a new privacy analysis that improves on the moments accountant analysis and provides better guarantees (i.e. lower ϵ values).

Inherently, applying differential privacy to deep learning yields a loss of utility due to the addition of noise and clipping. Bagdasaryan et al. have demonstrated that this loss in utility is disparate across different sub-groups of the population, with different sizes[40]. They experimentally show that sub-groups with fewer training samples (less representation) lose more accuracy compared to well-represented groups, i.e. the poor get poorer.

There is a body of work that tries to experimentally measure and audit the privacy brought by differentially private learning algorithms[250,251]. Jagielski et al.[250] investigate whether DP-SGD offers better privacy in practice than what is guaranteed by its analysis, using data poisoning attacks. Jayaraman et al. [251] apply membership and attribute inference attacks on multiple differentially private machine learning and deep learning algorithms and compare their performance.

Homomorphic Encryption

There are only a handful of works that exploit solely homomorphic encryption for private training of machine learning models[202,36,224]. Graepel et al. use a Somewhat HE (SHE) scheme to train Linear Means (LM) and Fishers Linear Discriminate (FLD) classifiers[202]. HE algorithms have some limitations in terms of the functions they can compute (for instance, they cannot implement non-linearities). For that reason, Graepel et al. propose division-free algorithms and focus on simple classifiers and not complex algorithms such as neural networks.

Hesamifard et al.[224] try to exploit HE for deep learning tasks. They introduce methods for approximating the most commonly used neural network activation functions (ReLU, Sigmoid, and Tanh) with low-degree polynomials. This is a crucial step for designing efficient homomorphic encryption schemes. They then train convolutional neural networks with those approximate polynomial functions and, finally, implement convolutional neural networks over encrypted data and measure the performance of the models.

Secure Multi-Party Computation (SMC)

A trend in research on private and secure computation consists of designing custom protocols for applications such as linear and logistic regression [372] and neural network training and inference[372,11,461]. These methods usually target settings where different datasets from different places are set to train a model together or where computation is off-loaded to a group of computing servers that do not collude with each other. SMC requires that all participants be online at all times, which requires a significant amount of communication[265]. Mohassel & Zhang proposed SecureML, which is a privacy-preserving stochastic gradient descent-based method to privately train machine learning algorithms such as linear regression, logistic regression, and neural networks in multiparty computation settings. SecureML uses secret sharing to achieve privacy during training. In a more recent work[371], Mohassel et al. design protocols for secure three-party training of DNNs with a majority of honest parties. Agrawal et al. propose QUOTIENT[11], where their goal is to design an optimization algorithm alongside a secure computation protocol customized for it instead of a conventional approach that uses encryption on top of existing optimization algorithms.

Table 1.4: Categorization of some notable privacy-preserving mechanisms for inference. In this table, NB is short for Naive Bayes, and DT is short for Decision Tree.

Method	DP	SMC	HE	IT	Dataset(s)	Task
ARDEN[553]	●	○	○	○	MNIST, CIFAR-10, SVHN	Image Classification w/ DNN
Cryptonets[147]	○	○	●	○	MNIST	Image Classification w/ DNN
Private Classification[90]	○	○	●	○	MNIST	Image Classification w/ DNN
TAPAS[449]	○	○	●	○	MNIST, Faces, Cancer, Diabetes	Image Classification w/ DNN
FHEDiNN[74]	○	○	●	○	MNIST	Image Classification w/ DNN
Face Match[62]	○	○	●	○	LFW, IJB-A, IJB-B, CASIA	Face recognition with CNNs
Cheetah[423]	○	○	●	○	MNIST, Imagenet	Image Classification w/ DNN
EPIC[338]	○	●	○	○	CIFAR-10, MIT, Caltech	Image Classification w/ DNN
DeepSecure[433]	○	●	○	○	MNIST, UCI-HAR	Classification w/ DNN
XONN[427]	○	●	○	○	MNIST, CIFAR-10	Image Classification w/ DNN
Chameleon[428]	○	●	○	○	MNIST, Credit Approval	Classification w/ DNN and SVM
CRYPTFLOW[295]	○	●	○	○	MNIST,CIFAR, ImageNet	Classification w/ DNN
Classification over Encrypted Data[72]	○	●	●	○	Wisconsin Breast Cancer	Classification w/ NB, DT
MiniONN[324]	○	●	●	○	MNIST, CIFAR-10	Image Classification w/ DNN
GAZELLE[260]	○	●	●	○	MNIST, CIFAR-10	Image Classification w/ DNN
DELPHI[368]	○	●	●	○	CIFAR-10, CIFAR-100	Image Classification w/ DNN
Shredder[359]	○	○	○	●	SVHN, VGG-Face, ImageNet	Classification w/ DNN
Sensor Data Obfuscation[339]	○	○	○	●	Iphone 6s Accelerometer Data	Activity Recognition w/ DNN
Olympus[422]	○	○	○	●	Driving images	Activity Recognition w/ DNN
DPFE[397]	○	○	○	●	CelebA	Image Classification w/ DNN
Cloak[358]	○	○	○	●	CIFAR-100, CelebA, UTKFace	Image Classification w/ DNN

1.10.3 Inference Phase

As shown in Table 1.4 there are fewer works in the field of inference privacy, compared to training. Inference privacy targets systems that are deployed to offer Inference-as-a-Service. In these cases, the deployed system is assumed to be trained and is not to learn anything new from the data provided by the user. It is only supposed to carry out its designated inference task. The categorization of literature for inference privacy is similar to training, except that there is one extra group here, named Information-Theoretic (IT) privacy. The works in this group usually offer information-theoretic mathematical or empirical evidence of how their methods operate and help privacy. These works are based on the context-aware privacy definition of Section 1.10.1, and they aim at decreasing the information content in the data sent to the service provider for inference so that there is only as much information in the input as needed for the service and not more.

One notable difference between training and inference privacy is the difference in the amount of literature on different categories. There seems to be a trend of using differential privacy for training and encryption methods (HE and SMC) for inference. One underlying reason could be computational complexity and implementation. Encryption methods, specifically homomorphic encryption, are shown to be at least two orders of magnitude slower than conventional execution[260]. That's why adopting them for training will increase training time significantly. Also, as mentioned in Section 1.10.2, due to approximating non-linear functions, the capabilities of neural networks in terms of performance become limited during training on encrypted data. For inference, however, adopting encryption is more trivial, since the model is already trained. Employing differential privacy, and noise addition, however, is less trivial for inference, since it could damage the accuracy of the trained model, if not done meticulously. Below we delve deeper into the literature of each category.

Differential Privacy

There are very few works using differential privacy for inference including the recent work on posthoc privacy[477] that is based on a variant of differential privacy called metric differential privacy. Cloud-based machine learning inference is an emerging paradigm where users query by sending their data through a service provider who runs an ML model on that data and returns the answer. Due to increased concerns over data privacy, recent works have proposed Collaborative Inference (CI) to learn a privacy-preserving encoding of sensitive user data before it is shared with an untrusted service provider. Existing works so far evaluate the privacy of these encodings through empirical reconstruction attacks. In this work, we develop a new framework 8 that provides formal privacy guarantees for an arbitrarily trained neural network by linking its local Lipschitz constant with its local sensitivity. To guarantee privacy 10 using local sensitivity, we extend the Propose-Test-Release (PTR) framework to make it tractable for neural network queries.

Wang et al.[553] propose Arden, a data nullification and differentially private noise injection mechanism for inference. Arden partitions the DNN across edge devices and the cloud. A simple data transformation is performed on the mobile device, while the computation-heavy and complex inference relies on the cloud data center. Arden uses data nullification, and noise injection to make different queries indistinguishable so that the privacy of the clients is preserved. The proposed scheme requires noisy retraining of the entire network, with noise injected at different layers. Since it is complicated to calculate the global sensitivity at each layer of the neural network, the input to the noise injection layer is clipped to the largest possible value created by a member of the training set, on the trained network.

HOMOMORPHIC ENCRYPTION

CryptoNets is one of the first works in HE inference[147]. Dowlin et al. present a method for converting a trained neural network into an encrypted one, named a CryptoNet. This allows the clients of an inference service to send their data in an encrypted format and receive the result, without their data being decrypted. CryptoNets allows the use of SIMD (Single Instruction Multiple Data) operations, which increase the throughput of the deployed system. However, for single queries, the latency of this scheme is still high.

Chabanne et al.[90] approximate the ReLu non-linear activation function using low-degree polynomials and provide a normalization layer before the activation function, which offers high accuracy. However, they do not show results on the latency of their method. More recently, Juvekar et al. propose GAZELLE[260], a system with lower latency (compared to prior work) for secure and private neural network inference. GAZELLE combines homomorphic encryption with traditional two-party computation techniques (such as garbled circuits). With the help of its homomorphic linear algebra kernels, which map neural network operations to optimized homomorphic matrix-vector multiplication and convolutions, GAZELLE is shown to be three orders of magnitude faster than CryptoNets. Sanyal et al. leverage binarized neural networks to speed up their HE inference method. They claim that unlike CryptoNets which only protects the data, their proposed scheme can protect the privacy of the model as well.

Secure Multi-Party Computation (SMC)

Liu et al. propose MiniONN[324], which uses additively homomorphic encryption (AHE) in a preprocessing step, unlike GAZELLE which uses AHE to speed up linear algebra directly. MiniONN demonstrates a significant performance improvement compared to CryptoNets, without loss of accuracy. However, it is only a two-party computation scheme and does not support computation over multiple parties. Riazi et al. introduces Chameleon, a two-party computation framework whose vector dot product of signed fixed-point numbers improves the efficiency of prediction in classification methods based upon heavy matrix multiplications. Chameleon achieves a 4.2× latency improvement over MiniONN. Most of the efforts in the field of SMC for deep learning are focused on speeding up the computation, as demonstrated above, and also by [427], [338], [433]. The accuracy loss of the aforementioned methods, compared to their pre-trained models is negligible (less than 1%).

1.10.4 Trusted Execution Environments (TEEs)

Trusted execution environments, also referred to as secure enclaves, provide opportunities to move parts of decentralized learning or inference processes into a trusted environment in the cloud, whose code can be attested and verified. Recently, Mo et al. have suggested a framework that uses an edge devices Trusted Execution Environment (TEE) in conjunction with model partitioning to limit the attack surface against DNNs[369]. TEEs can provide integrity and confidentiality during execution. TEEs have been deployed in many forms, including Intels SGX-enabled CPUs[129,384], Arms TrustZone[inc.]. This execution model, however, requires the users to send their data to an enclave running on remote servers which allows the remote server to have access to the raw data and as the new breaches in hardware[282,322,561,83,504,503] show, the access can lead to comprised privacy.

Part I

Distributed and Private Statistical Inference

"If chance is the antithesis of law, then we need to discover the laws of chance."

Calyampudi Radhakrishna Rao

2 Private Estimation of Non-Linear Correlations

2.1 Introduction

Estimating correlations (linear and non-linear dependencies between random variables) and hypothesis tests of independence (from their samples) are two adjacent problems of fundamental importance in statistics. The impact of these problems in decision-making has gone beyond that of statistics into a wide range of adjacent and seemingly far-flung fields of science and engineering. This chapter is based on our work in[544]. The following describes two adjacent problems of consideration in this paper.

2.1.1 Problem statement

1. How can non-linear correlations between two random variables be estimated with formal privacy guarantees?
2. How can the sample *test-statistic* for a hypothesis test of independence between two random variables of arbitrary dimension be estimated *privately*?

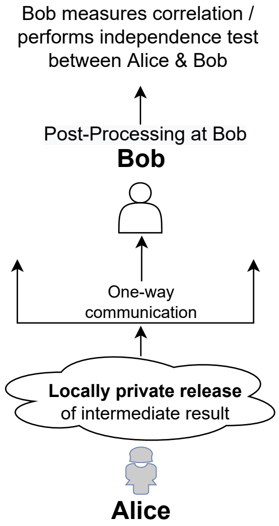

Bob measures correlation / performs independence test between Alice & Bob
Post-Processing at Bob
Bob
One-way communication
Locally private release of intermediate result
Alice

Figure 2.1: The two-party privacy model

Communication setup: In terms of communication, we consider a two-party setup, with Alice holding $\boldsymbol{X} \in \mathbb{R}^{n \times d}$ and Bob holding $\boldsymbol{Y} \in \mathbb{R}^{n \times m}$ that denote the data matrices of n samples in corresponding dimensions of d or m, respectively. The direction of allowed communication is one-way, from Alice to Bob: Alice sends an intermediate computation in a privatized form to Bob, who then finishes the rest of the computation on its premise. Bob never reveals the answer to Alice, so the on-device computation of Bob involves its own non-privatized data. Thereby, Bob obtains the results (on its side) while maintaining the privacy of Alice's (already privatized)

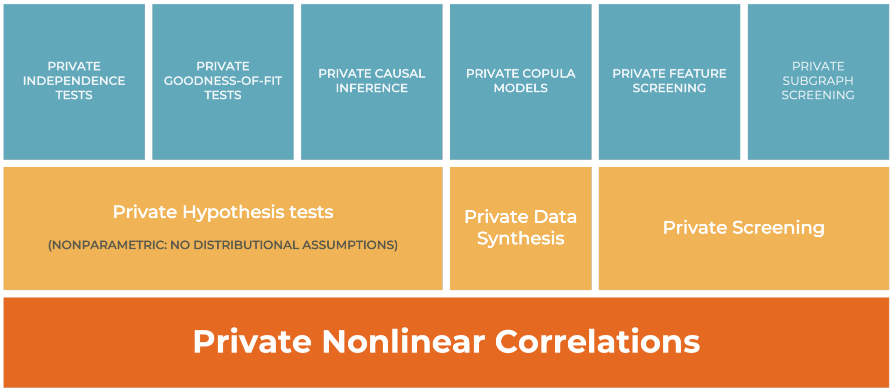

PRIVATE INDEPENDENCE TESTS
PRIVATE GOODNESS-OF-FIT TESTS
PRIVATE CAUSAL INFERENCE
PRIVATE COPULA MODELS
PRIVATE FEATURE SCREENING
PRIVATE SUBGRAPH SCREENING
Private Hypothesis tests
(NONPARAMETRIC: NO DISTRIBUTIONAL ASSUMPTIONS)
Private Data Synthesis
Private Screening
Private Nonlinear Correlations

Figure 2.2: A stack of technical applications that benefit from methods for private estimation of nonlinear correlations.

data. We refer to this setup as *one-way local differential privacy*, as the privatization mechanism used by Alice ensures a mathematical notion of privacy called differential privacy[160].

A more general way to think about it is as a data summary that, when published by Alice, allows any analyst to measure dependence and/or test independence with another data set they hold. We illustrate this setup in Figure 2.1.

2.2 Related work

2.2.1 Private nonlinear correlations

An ability to compute non-linear correlations between features hosted across multiple parties in a private manner opens new applications. The dependency measure we consider in this paper is called distance correlation[500], which is an instance of 'Energy Statistics' introduced in[499,431,497]. The listed solutions to the following problems predominantly depend on measuring distance correlation. Thereby, an ability to measure it privately in a setting where the features are siloed across two parties would lead to a wide variety of privatization schemes for these problems.

Private multi-party feature screening with distance correlation: The importance of distance correlation in optimally selecting features with a '*sure independence screening*' guarantee under a model agnostic setting was shown in the works of[317,609,172,335]. Thereby, the measurement of distance correlation in a privacy-preserving manner in multi-party settings would potentially allow for private feature selection.

Private multi-party independence testing with distance correlation: Hypothesis tests for testing independence of distributions using distance correlation were introduced in[500,496], where the test statistic is based on distance correlation. Thereby, performing private independence testing between samples that are distributed in multiple entities requires a private estimation of distance correlation. The work in[469] showed an equivalence between independence testing using distance correlation and k-sample testing.

Private multi-party causal direction estimation with distance correlation: The works in[296,297] privately estimate distance correlation in a single party setting, where samples from both the random variables are at the same entity. They then use this private estimate of distance correlation to infer the causal direction. We consider the more important setting where the samples from both the random variables are at two different corresponding entities as opposed to all of them being on-premise at one entity.

Private multi-party data synthesis by seeding copulas with distance correlations: Gaussian copulas have been used for private data synthesis, as shown in[35,314]. A key step in this process is to seed the Gaussian copula with a correlation matrix, and it has a significant impact on the quality of the synthesis. These current works have so far seeded it with linear measures of correlations such as Pearson's correlation or Kendall's Tau. A private multi-party measure of distance correlation allows for better seeding of the Gaussian copula for multi-party private data synthesis.

2.2.2 Private Independence Testing

In view of the large amount of literature addressing the problem of independence testing over the years in a variety of statistical settings, we focus here on the closest and most relevant to our work.This first line of work, which aims to develop differentially private independence tests, itself comes in two distinct flavors: one is the so-called *asymptotic regime* (limiting distribution and properties of the test statistic as sample size goes to infinity), and the second is the *finite-sample regime* (coarser utility guarantees, but with explicit finite bounds on the sample size required to achieve them).

Asymptotic tests: The work of[184] considers independence testing in the (central) model of differential privacy, from an *asymptotic* perspective: namely, they propose a differentially private analogue of the classical chi-squared tests of independence, and analyse the limiting distribution of the test statistic along with its resulting power (1-Type II error). Note that Type II error is the probability of failing to reject a null hypothesis when the null hypothesis is not true. To complement their asymptotic analysis, the authors further empirically assess the performance of those differentially private analogues of the standard χ^2 tests, focusing on a small sample size. Finally, the privacy model considered in this work (central model) differs from ours (one-way distributed), as the chi-squared tests privatized in[184] require access to the empirical contingency tables – something not available to either party in our setting.

Non-asymptotic tests: Later work by[467] focuses on the *finite sample* (non-asymptotic) version of the test under a more stringent model of local privacy. The question is formulated in a manner that is standard in theoretical computer science (specifically in the domain of distribution testing) and in minimax analysis in statistics as part of a composite hypothesis testing problem. They consider a set of product distributions as part of the null hypothesis and frame the alternative hypothesis to contain all the distributions that are “far” from product distributions, where "farness" is quantified by the total variation distance. They focus on the minimax sample complexity achievable using a specific locally private mechanism of Randomized Response[556]. Subsequent work by[6] improves on these results by showing a way to achieve significantly lower sample complexity (still in the locally private setting) by considering a different privacy mechanism and upon establishing matching lower bounds.

Non-private independence testing with dependency measures: There have been advances in the development of various statistical dependency measures, and an active route of modern independence testing is based on the derivation of test-statistics that depend on these measures in addition to other required terms. We now share some related works that fall into this category. Distance covariance was introduced in[500] and can be expressed as a weighted L_2 norm between the characteristic function of the joint distribution and the product of the marginal characteristic functions. This concept has also been studied in high dimensions[496,583], and for testing the independence of several random vectors[187]. In[458], tests based on distance covariance were shown to be equivalent to a reproducing kernel Hilbert space (RKHS) test for a specific choice of kernel. RKHS tests have been widely studied in the machine learning community, with a survey of the subject given by[215] and[205,223] in which the Hilbert-Schmidt independence criterion was proposed. These tests are based on embedding the joint distribution and product of the marginal distributions into a Hilbert space and considering the norm of their difference in this space. One drawback of the kernel paradigm here is the computational complexity, though[256,255] and[600] have recently attempted to address this issue. A conditional measure of dependence called conditional distance correlation was introduced by[555]. The works in[599,598,554,468] performed conditional independence testing and applied them to the problem of causal discovery.

To the best of our knowledge, no other work addresses the question of independence testing under differential privacy (be it local or central) other than the following works: the differentially private distribution estimation approach (under total variation distance)[137] or two-sample goodness-of-fit[7,18] that can be used to obtain sub-optimal sample complexity guarantees for this problem; and the (central) differentially private algorithm for independence testing of two random variables of Aliakbarpour et al. [17]. We note that this body of work differs from ours, both in the distributed model assumed (the way the data is partitioned across users) and in the guarantee provided (dependency measure used). They also restrict themselves to the discrete setting as opposed to ours. Our method can also be applied to test between samples lying in different dimensions. In particular, most of the works discussed above (except for Gaboardi et al. [184]) follow the norm in distribution testing and focus on the very stringent notion of minimax testing under total variation distance, which might be overly conservative in many settings.

2.2.3 Contributions

Our contributions are threefold and can be summarized as follows:

1. We introduce a differentially private method to measure nonlinear correlations between sensitive features hosted across two entities and provide some utility analysis of this estimator along with experimental results to compare the quality of our estimator on several benchmark datasets. As part of this utility analysis, we decompose the error into two terms, from which we obtain the following bound on one of the two error terms. We provide experimental results showing that the other error term is reasonably controlled for as well.

   $$\mathbb{P}\left\{ Error < \frac{4C_dC_mn^2\sqrt{2\left(\ln\left(\frac{1}{2\delta}\right)+\epsilon\right)}}{K\sqrt{k}(n-2)(n-3)\epsilon} \right\} \geq 1 + n\left(\frac{4}{e^{\frac{2n}{n-1}}}\right)^n - 2n\left(\frac{2}{e^{\frac{n}{n-1}}}\right)^n,$$

   where n is the sample size, ϵ and δ are the approximate differential privacy parameters, K is the number of random projections involved in the estimator, and k refers to the dimension into which the data is projected to after the random projection.

2. We then provide a mechanism to privatize a test-statistic required to perform a hypothesis test of independence between features hosted by Alice and corresponding features hosted by Bob under the communication setup discussed in Section 2.1.1. We show that one can express our test statistic as the ratio of the sum of *directional variance* queries (of non-private data) with respect to 'specific' covariance matrices of sensitive data (for the formal definition of *directional variance* queries, the reader is referred to Definition 2.3.4). This reduction enables us to privatize the test-statistic via the privatization of these covariances. This brings up the next question of deriving utility guarantees for the privatized test-statistic, leading to our next contribution.

3. We derive both lower and upper bounds on the utility of this privatized test-statistic in terms of additive and multiplicative errors. For a chosen $0 < \eta < 1$ with sample size n, in order to achieve (ϵ, δ)-differential privacy, our approach results in a multiplicative error factor of $1 \pm \eta$, and an additive error Δ bounded as

$$-\frac{(1-\eta)^2}{2(1+\eta)} \leq \Delta \leq \frac{2\tau}{(1-\eta)[(1-\eta)s - \tau]},$$

where

$$\tau := \left(\frac{2048 \ln(2/(m+n)\nu) \ln(2/\delta)}{\eta\epsilon^2}\right) \ln^2\left(\frac{128 \ln \frac{1}{(m+n)\nu}}{\eta^2\delta}\right),$$

for some $s > \frac{\tau}{1-\eta}$. The additive and multiplicative approximations hold together at the same time with probability at least $1 - (m+n)\nu$ for the user's choice of ν.

2.3 Background

Preliminaries for differential privacy

Definition 2.3.1. (ϵ, δ)-*Differential Privacy (2014)*

A randomized algorithm $\mathcal{A} : \mathcal{X} \rightarrow \mathcal{Y}$ is (ϵ, δ)-differentially private if, for all neighboring datasets $\mathbf{X}, \mathbf{X}' \in \mathcal{X}$ and for all $S \in \mathcal{Y}$

$$\Pr[\mathcal{A}(\mathbf{X}) \in S] \leq e^{\epsilon} \Pr\left[\mathcal{A}\left(\mathbf{X}'\right) \in S\right] + \delta$$

Definition 2.3.2. *Post-Processing Invariance*

Differential privacy is immune to post-processing, meaning that an adversary without any additional knowledge about the dataset $\mathbf{X}$ cannot compute a function on the output $\mathcal{A}(\mathbf{X})$ to violate the stated privacy guarantees.

Definition 2.3.3. *(l_2-Global Sensitivity)*

Let $f : \mathcal{X} \rightarrow \mathbb{R}^k$. The l_{2-} global sensitivity of f is

$$\Delta_2^{(f)} = \max_{\mathbf{x}, \mathbf{X}' \in \mathcal{X}} \left\| f(\mathbf{X}) - f\left(\mathbf{X}'\right) \right\|_2$$

where $\mathbf{X}, \mathbf{X}'$ are neighboring databases. In addition,[160] is a good extended resource on the topic of differential privacy.

Preliminaries for private nonlinear correlations

Notation: We use $\boldsymbol{X} \in \mathbb{R}^{n \times d}$ and $\boldsymbol{Y} \in \mathbb{R}^{n \times m}$ to refer to a data matrix of n samples in corresponding dimensions of d and m, respectively. We denote $\mathcal{X}, \mathcal{Y}$ to refer to population random variables from which we obtain the data samples. Note that, in the rest of the paper, we refer to rows k, l in $\boldsymbol{X}$ using lower-case representation of $\boldsymbol{x}_k, \boldsymbol{x}_l$. Similarly, we refer to columns k, l

in $\boldsymbol{X}$ using upper-case representation of $\boldsymbol{X}_k, \boldsymbol{X}_l$. We start by providing definitions of population distance covariance and population distance correlation, as these are central to the rest of the paper. We then share two existing non-private sample estimators for estimating the same. To avoid confusion, we now list the notations for denoting several estimators and population notions of distance covariance below as a reference and follow it up with definitions for each of these. We use $\Omega(\mathcal{X}, \mathcal{Y})$ to denote the *population distance covariance*, $\hat{\Omega}(\boldsymbol{X}, \boldsymbol{Y})$ to denote the *classical distance covariance* estimator (Eqn. 2.3), $\overline{\Omega}(\boldsymbol{X}, \boldsymbol{Y})$ to denote the *random projected distance covariance* estimator (Eqn. 2.4), and $\overline{\Omega}^{dp}(\boldsymbol{X}, \boldsymbol{Y})$ to denote the *private random projected distance covariance* estimator (Eqn. 2.5) in the rest of the paper.

Population distance covariance

For random variables $\mathcal{X} \in \mathbb{R}^d$ and $\mathcal{Y} \in \mathbb{R}^m$ with finite first moments, the population distance covariance[500] between them is a non-negative number given by

$$\Omega(\mathcal{X}, \mathcal{Y}) = \int_{\mathbb{R}^{d+m}} |f_{\mathcal{X},\mathcal{Y}}(t,s) - f_{\mathcal{X}}(t) f_{\mathcal{Y}}(s)|^2 w(t,s) dt ds, \tag{2.1}$$

where $f_{\mathcal{X}}, f_{\mathcal{Y}}$ are characteristic functions of $\mathcal{X}, \mathcal{Y}$, $f_{\mathcal{X},\mathcal{Y}}$ is the joint characteristic function, and $w(t,s)$ is a weight function defined as

$$w(t,s) = (C(d,\alpha) C(m,\alpha) |t|_d^{\alpha+d} |s|_m^{\alpha+m})^{-1}$$

with

$$C(d,\alpha) = \frac{2\pi^{d/2}\Gamma(1-\alpha/2)}{\alpha 2^{\alpha}\Gamma((\alpha+d)/2)},$$

for chosen values of α which impacts the choice of norm. More details about this weight function and the reasoning for its appropriateness for evaluating this integral are given in[500].

Γ refers to the popular complete Gamma function, which is defined to be an extension of the concept of a factorial to complex and real numbers as opposed to just the integers. Note that for random variables that admit a density, the characteristic function is the Fourier transform of the probability density function. Note that, for complex-valued functions such as the characteristic function used in 2.1, the norm is dependent on itself and its complex conjugate $\bar{f}$ as $|f|^2 = f\bar{f}$. It is worth noting that distance covariance between a variable and itself is referred to as distance variance.

POPULATION DISTANCE CORRELATION

Using this above definition of distance covariance, we have the following expression for distance correlation[500] $\rho(\mathcal{X}, \mathcal{Y})$ between random variables

$$\rho(\mathcal{X}, \mathcal{Y}) = \begin{array}{ll} \frac{\Omega(\mathcal{X},\mathcal{Y})}{\sqrt{\Omega(\mathcal{X},\mathcal{X})\Omega(\mathcal{Y},\mathcal{Y})}}, & \text{if } \Omega(\mathcal{X}, \mathcal{X})\Omega(\mathcal{Y}, \mathcal{Y}) > 0, \\ 0, & \text{if } \Omega(\mathcal{X}, \mathcal{X})\Omega(\mathcal{Y}, \mathcal{Y}) = 0. \end{array} \tag{2.2}$$

This always lies within the interval $[0, 1]$ with 0 indicating independence and 1 indicating dependence. For completeness, we provide a list of some popular measures of nonlinear statistical dependency as given below

1. *Energy statistics*: Distance correlation (DCOR)[500,496,497,499], Brownian distance covariance[495], Partial distance correlation[498], Partial martingale difference correlation[402]

2. *Kernel covariance operators*: Constrained covariance (COCO)[205], Hilbert-Schmidt independence criterion (HSIC)[204], Kernel Target Alignment (KTA)[131].

3. *Integral probability metrics*: Maximum mean discrepancey (MMD)[70], Wasserstein distance[530], Dudley metric[150] and Fortet-Mourier metric[486].

4. *Information theoretic measures*: Mutual information, f-divergence[148,490], Renyi divergence[532], Hellinger distance[487], Total variation distance[548,421], Maximal information coefficient[281].

In the setting when both of the datasets are hosted on the same entity, a privatized measure of a statistical dependency called Hilbert-Schmidt independence criterion was provided in[297]. That said, this paper specifically focuses on mechanisms for privatizing the measure of distance correlation in settings when the two sets of data features are hosted at *different* entities.

Sample estimators of distance correlation

The following are two different sample estimators for estimating population distance covariance in a *non-private* setting.

Unbiased classical sample distance covariance: From this section onwards, we use $|\cdot|$ to represent the Euclidean norm and should not be confused with the norm of a complex-valued function as used in defining population distance covariance. Using $\boldsymbol{X}_i, \boldsymbol{Y}_i$ to denote i'th rows in the corresponding data matrices, we first define $a_{ij} = |\boldsymbol{x}_i - \boldsymbol{x}_j|$, $b_{ij} = |\boldsymbol{y}_i - \boldsymbol{y}_j|$, $a_{i\cdot} = \sum_{l=1}^{n} a_{il}$, $b_{i\cdot} = \sum_{l=1}^{n} b_{il}$, $a_{\cdot\cdot} = \sum_{k,l=1}^{n} a_{kl}$ and $b_{\cdot\cdot} = \sum_{k,l=1}^{n} b_{kl}$. We now use these quantities to define an unbiased statistical estimator of distance covariance $\hat{\Omega}(\boldsymbol{X}, \boldsymbol{Y})$ as follows:

$$\hat{\Omega}(\boldsymbol{X}, \boldsymbol{Y}) = \frac{1}{n(n-3)} \sum_{i \neq j} a_{ij} b_{ij} - \frac{2}{n(n-2)(n-3)} \sum_{i=1}^{n} a_{i\cdot} b_{i\cdot} + \frac{a_{\cdot\cdot} b_{\cdot\cdot}}{n(n-1)(n-2)(n-3)}. \tag{2.3}$$

Random projected distance covariance: A faster unbiased estimator of distance covariance based on random projections denoted by $\overline{\Omega}(\boldsymbol{X}, \boldsymbol{Y})$ was introduced in[234]. In order to define this sample estimator of distance covariance, we first define a few constants based on π and the Gamma function as follows. These include $c_d = \frac{\pi^{(d+1)/2}}{\Gamma((d+1)/2)}$ and $c_m = \frac{\pi^{(m+1)/2}}{\Gamma((m+1)/2)}$, $C_d =$

$\frac{c_1 c_{d-1}}{c_d} = \frac{\sqrt{\pi}\Gamma((d+1)/2)}{\Gamma(d/2)}$ and $C_m = \frac{c_1 c_{m-1}}{c_m} = \frac{\sqrt{\pi}\Gamma((m+1)/2)}{\Gamma(m/2)}$. Let $\boldsymbol{u}$ and $\boldsymbol{v}$ be points on the hyperspheres: $\boldsymbol{u} \in \mathcal{S}^{d-1} = \left\{\boldsymbol{u} \in \mathbb{R}^d : |u| = 1\right\}$ and $\boldsymbol{v} \in \mathcal{S}^{m-1}$. For any vector $\boldsymbol{u}$ or $\boldsymbol{v}$, let $\boldsymbol{u}^T$ or $\boldsymbol{v}^T$ denote its transpose. We now share the non-private estimator below

$$\overline{\Omega}(\boldsymbol{X}, \boldsymbol{Y}) = \sum_{k=1}^{K} \frac{C_d C_m \Omega_n(\boldsymbol{u}_k^\top \boldsymbol{X}, \boldsymbol{v}_k^\top \boldsymbol{Y})}{K}, \tag{2.4}$$

where $\boldsymbol{u}_k^\top \boldsymbol{X} = (\boldsymbol{u}_k^\top \boldsymbol{x}_1, \ldots, \boldsymbol{u}_k^T \boldsymbol{x}_n)$ and $\boldsymbol{u}_k^\top \boldsymbol{Y} = (\boldsymbol{u}_k^\top \boldsymbol{y}_1, \ldots, \boldsymbol{u}_k^\top \boldsymbol{y}_n)$. Note that in this case $a_{ij} = |\boldsymbol{u}^\top(\boldsymbol{x}_i - \boldsymbol{x}_j)|$ and $b_{ij} = |\boldsymbol{v}^\top(\boldsymbol{y}_i - \boldsymbol{y}_j)|$. Apart from being unbiased, a concentration bound on the deviation around the true value exists[234] as $\boldsymbol{P}\left(\left|\overline{\Omega}(\boldsymbol{X}, \boldsymbol{Y}) - \hat{\Omega}(\boldsymbol{X}, \boldsymbol{Y})\right| > \epsilon\right) \leq 2\exp\left\{-\frac{CK\epsilon^2}{\mathrm{Tr}[\boldsymbol{\Sigma}_{\boldsymbol{X}}]\,\mathrm{Tr}[\boldsymbol{\Sigma}_{\boldsymbol{Y}}]}\right\}$, where K is the number of random projections used in obtaining $\overline{\Omega}(\boldsymbol{X}, \boldsymbol{Y})$, $\boldsymbol{\Sigma}_{\boldsymbol{X}}$ and $\boldsymbol{\Sigma}_{\boldsymbol{Y}}$ denote the covariance matrices of $\boldsymbol{X}$ and $\boldsymbol{Y}$, respectively, Tr returns to their matrix traces, and $C = \frac{2}{25C_d^2 C_m^2}$ is a constant depending on C_d, C_m which were introduced previously. Note that, in terms of uniformly sampling points on the sphere, we may use the property of centered and normalized normal random vectors are uniformly distributed on the unit sphere. That is, if we let $\mathcal{S}_r^m \equiv \{\boldsymbol{x} \in \mathbb{R}^m | \sum \boldsymbol{x}_i^2 = r^2\}$ denote the m-dimensional sphere with radius r, then we have $\frac{\boldsymbol{X}}{||\boldsymbol{X}||} \sim \mathrm{U}(\mathcal{S}_1^m)$.

Computational advantage: This estimator has a $\mathcal{O}(nKlogn)$ computational complexity and a memory requirement of $\mathcal{O}(max\{n, K\})$ as opposed to the classical estimator, which is dependent on computing all pairs of distances between the samples in $\boldsymbol{X}$ and $\boldsymbol{Y}$.

Preliminaries for independence testing

To formalise the problem of hypothesis testing for independence, consider random variables $\mathcal{X}$ and $\mathcal{Y}$ that have densities $f_{\mathcal{X}}$ on $\mathbb{R}^d$ and $f_{\mathcal{Y}}$ on $\mathbb{R}^m$, respectively, and let $\mathcal{Z} = (\mathcal{X}, \mathcal{Y})$ have density f on $\mathbb{R}^q$, where $q := d + m$. Given independent and identically distributed copies $Z_1, \ldots, Z_n$ of $\mathcal{Z}$, we wish to test the null hypothesis that $\mathcal{X}$ and $\mathcal{Y}$ are independent, denoted by $H_0\colon \mathcal{X} \perp\!\!\!\perp \mathcal{Y}$, against the alternative hypothesis that $\mathcal{X}$ and $\mathcal{Y}$ are not independent, denoted by $H_1\colon \mathcal{X} \not\perp\!\!\!\perp \mathcal{Y}$. A distribution D is said to be *independent* if it is equal to the product of its marginals.

Test-statistic: A predominant methodology for hypothesis testing (in the non-private setting) involves the computation of a sample *test statistic* from the data. The key idea is that the asymptotic distribution of a test statistic (that can be estimated from data samples) is derived (and hence, known) upon conditioning on the null hypothesis being true. Therefore, the observed value of the sample test-statistic (a scalar) is compared with this asymptotic distribution in conjunction with a chosen level of confidence to decide if the observed value falls within the acceptance region or the rejection region. The rejection region refers to the observed value of the sample test statistic being highly unlikely, given the asymptotic distribution, if the null hypothesis was supposed to be true.

Definition 2.3.4 (Directional variance)**.** *For an $n \times d$ matrix $\boldsymbol{M}$, a* directional variance query *is specified by a unit-length direction $\boldsymbol{x}$, and is given by $\Phi_{\boldsymbol{M}}(\boldsymbol{x}) = \boldsymbol{x}^\top \boldsymbol{M}^\top \boldsymbol{M}\boldsymbol{x}$.*

2.4 Method for private nonlinear correlations

Privately estimating sample distance correlation between $\boldsymbol{X} \in \mathbb{R}^{n\times d}$ and $\boldsymbol{Y} \in \mathbb{R}^{n\times m}$ requires a private estimation of a distance covariance term in the numerator and one of the distance variance terms in the denominator of $\frac{\overline{\Omega}(\boldsymbol{X},\boldsymbol{Y})}{\sqrt{\overline{\Omega}(\boldsymbol{X},\boldsymbol{X})\overline{\Omega}(\boldsymbol{Y},\boldsymbol{Y})}}$ that depends on $\boldsymbol{X}$.

Privatizing $\overline{\Omega}(\boldsymbol{X},\boldsymbol{Y})$: For the distance covariance in the numerator, we provide a differentially private estimator of $\overline{\Omega}(\boldsymbol{X},\boldsymbol{Y})$ that we denote by $\overline{\Omega}^{dp}(\boldsymbol{X},\boldsymbol{Y})$. Note that $\overline{\Omega}(\boldsymbol{X},\boldsymbol{Y})$ is the non-private 'random projected distance covariance estimator' given in equation 2.4 to estimate population distance covariance $\Omega(\boldsymbol{X},\boldsymbol{Y})$. We provide details on our differentially private estimator $\overline{\Omega}^{dp}(\boldsymbol{X},\boldsymbol{Y})$ in section 3.2.1 and provide our utility proofs for it in Theorems 1 & 2.

Privatizing $\overline{\Omega}(\boldsymbol{Y}, \boldsymbol{Y})$: The term of $\overline{\Omega}(\boldsymbol{Y}, \boldsymbol{Y})$ does not require private estimation as it is computed on-premise by the entity that holds $\boldsymbol{Y}$. The entity that holds $\boldsymbol{X}$ (say, Alice) makes a one-way communication of our proposed private estimates for $\overline{\Omega}^{dp}(\boldsymbol{X}, \boldsymbol{Y})$, $\overline{\Omega}^{dp}(\boldsymbol{X}, \boldsymbol{X})$ to the entity that holds $\boldsymbol{Y}$ (say, Bob). Bob computes a non-private and $\overline{\Omega}(\boldsymbol{Y}, \boldsymbol{Y})$ on-premise.

PRIVATE CORRELATION

These three estimates can be put together by Bob to privately estimate the distance correlation as

$$\begin{cases} \frac{\overline{\Omega}^{dp}(\boldsymbol{X},\boldsymbol{Y})}{\sqrt{\overline{\Omega}^{dp}(\boldsymbol{X},\boldsymbol{X})\overline{\Omega}(\boldsymbol{Y},\boldsymbol{Y})}}, & \text{if } \overline{\Omega}^{dp}(\boldsymbol{X},\boldsymbol{X})\overline{\Omega}(\boldsymbol{Y},\boldsymbol{Y}) > 0, \\ 0, & \text{if } \overline{\Omega}^{dp}(\boldsymbol{X},\boldsymbol{X})\overline{\Omega}(\boldsymbol{Y},\boldsymbol{Y}) = 0. \end{cases}$$

Note that Bob does not reveal the estimated private distance correlation to Alice. In a scenario where it needs to reveal it to Alice (or anyone in the public), it could do that by using the private estimator in the subsection on privatizing $\overline{\Omega}(\boldsymbol{X}, \boldsymbol{X})$ with respect to $\boldsymbol{X}$. This approach can be used by Bob to privatize $\overline{\Omega}(\boldsymbol{Y}, \boldsymbol{Y})$ with respect to $\boldsymbol{Y}$ instead. We now describe our two proposed private estimators for $\overline{\Omega}^{dp}(\boldsymbol{X}, \boldsymbol{Y})$ which is in the two-party setting and $\overline{\Omega}^{dp}(\boldsymbol{X}, \boldsymbol{X})$ which is in the single party setting.

2.4.1 PROPOSED PRIVATE ESTIMATOR FOR DISTANCE COVARIANCE

Our estimation starts with Alice sending K differentially private random projections of sensitive data $\boldsymbol{X}$ to Bob that hosts the sensitive data $\boldsymbol{Y}$. An average of K distance covariances between the random projections of $\boldsymbol{X}$ and the raw data $\boldsymbol{Y}$ is computed at Bob's premises to get $\overline{\Omega}^{dp}(\boldsymbol{X}, \boldsymbol{Y})$. We then perform K differentially private random projections of the data $\boldsymbol{X}$ by adding the necessary noise N^x required for privacy. The next sub-section explains the process of choosing N^x in order to guarantee differential privacy. Therefore, this estimator is given by

$$\overline{\Omega}^{dp}(\boldsymbol{X}, \boldsymbol{Y}) = \sum_{k=1}^{K} \frac{C_d C_m \overline{\Omega}_n(u_k^T \boldsymbol{X} + N^x, v_k^T \boldsymbol{Y})}{K}, \tag{2.5}$$

where $N^x = (N_1^x, \ldots, N_n^x)$. Note that in this case $\tilde{a}_{ij} = |\boldsymbol{u}^\top(\boldsymbol{x}_i - \boldsymbol{x}_j) + (N_i^x - N_j^x)|$ and $\tilde{b}_{ij} = |\boldsymbol{v}^\top(\boldsymbol{y}_i - \boldsymbol{y}_j)|$.

Avoiding sequential composition: Applying K differentially private random projections u_k^X in Equation 2.4 would lead to a differentially private estimate of nonlinear correlation. Studying the utility of this estimator would be of interest. That said, if all the K random projections are applied on the entire dataset $\mathbf{X}$, it will lead to an overall privacy guarantee of $K\epsilon$ due to the sequential composition property of differential privacy. This loss of privacy budget can be avoided if the samples in the dataset (i.e. the rows of $\mathbf{X}, \mathbf{Y}$) are partitioned into K disjoint subsets prior to a random projection-based distance covariance measured disjointly on each subset prior to averaging them out. This would lead to an improved accounting with regards to the privacy budget, as this falls under the parallel composition property of differential privacy, thereby leading to an overall ϵ-DP of the estimator if each of the individual projections was also performed with ϵ-DP. This estimator is summarized below.

Private estimator on disjoint partitions $\Omega_{Disjt}^{dp}(X, Y)$: Our estimator requires us to first partition the n records of $\mathbf{X}, \mathbf{Y}$ into K blocks of $\{[X_1, Y_1]\ [X_2, Y_2] \ldots [X_K, Y_K]\}$ in order to avoid sequential composition. Therefore, this estimator is given by $\Omega_{\text{Disjt}}^{dp}(\mathbf{X}, \mathbf{Y}) = \sum_{k=1}^{K} \frac{C_d C_m \overline{\Omega}_n(u_k^t X_k + N^x, v_k^t Y_k)}{K}$.

2.5 Differentially private random projections

As our method is based on the connections between the Johnson-Lindenstrauss transform and differential privacy to release statistics of distances, graph Laplacians, and directional variances, we now provide some relevant background and references on this topic. We state one of the

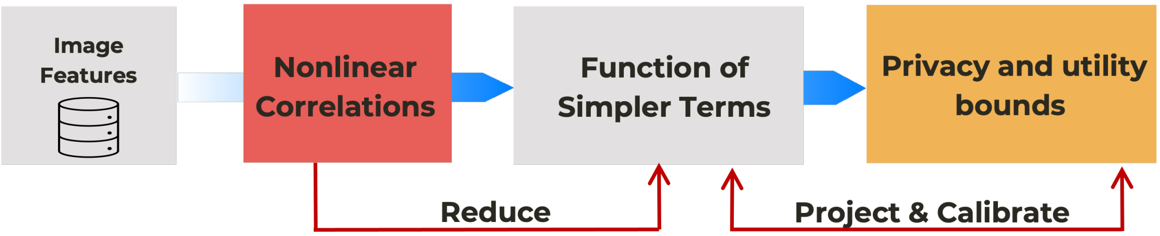


Figure 2.3: The reduce, project, and calibrate approach to private estimation.

classic mechanisms from[277] that is relevant to some aspects of our proposed method. We first share a prerequisite definition that is required prior to re-stating this mechanism. We denote the random projection matrix by $\boldsymbol{P}$ and note that one of the popular choices for building it is to have each entry of the matrix drawn independently from a Normal distribution with mean 0 and $\sigma^2 = 1/k$ where k refers to the dimension into which the data is projected to after the random projection. We now define the ℓ_ρ-Sensitivity of P.

Definition 2.5.1. (ℓ_ρ*-Sensitivity of* $\boldsymbol{P}$). *Define the* l_ρ*-sensitivity of a* $d \times k$ *projection matrix* $\boldsymbol{P} = \{\boldsymbol{P}_{ij}\}_{d\times k}$ *denoted by* $w_\rho(\boldsymbol{P})$*, as the maximum* ℓ_ρ*-norm of any row in* $\boldsymbol{P}$*, i.e.,* $w_\rho(\boldsymbol{P}) = \max_{1\leq i\leq d}\left(\sum_{j=1}^{k}|\boldsymbol{P}_{ij}|^\rho\right)^{\frac{1}{\rho}}$. *Equivalently,* $w_\rho(\boldsymbol{P})$ *can be defined as* $\max_{\boldsymbol{e}_i}\|\boldsymbol{e}_i\boldsymbol{P}\|_\rho$*, where* $\{\boldsymbol{e}_i\}_{i=1}^{d}$ *are standard basis unit vectors.*

Theorem 1. Let $w_2(\boldsymbol{P})$ be the ℓ_2-sensitivity of the projection matrix P (see Definition 2). Assuming $\delta < \frac{1}{2}$, let the entries of the noise matrix be drawn from $N\left(0, \sigma^2\right)$ with $\sigma \geq w_2(\boldsymbol{P})\frac{\sqrt{2\left(\ln\left(\frac{1}{2\delta}\right)+\epsilon\right)}}{\epsilon}$, then releasing $\boldsymbol{Z} = \boldsymbol{X}\boldsymbol{P} + \boldsymbol{\Delta}$ satisfies (ϵ, δ)-differential privacy.

Proof. Refer to proofs of Theorem 1 and Lemma 1 (for a more generalized version) in[277]. □

In addition, unlike the above work that requires explicit additive noise to privately release distances upon the random projection, the work in[61] instead uses random projections to privately release graph Laplacians and directional variances. These two papers are of good relevance to our work within this context.

Utility results: We now study the utility of the private estimator $\overline{\Omega}^{dp}(\boldsymbol{X},\boldsymbol{Y})$ in estimating the non-private $\overline{\Omega}(\boldsymbol{X},\boldsymbol{Y})$ given the effect of noise in the private estimator. The utility can be expressed by $\overline{\Omega}^{dp}(\boldsymbol{X},\boldsymbol{Y}) - \overline{\Omega}(\boldsymbol{X},\boldsymbol{Y})$.

Theorem 2. **(Error bound)** The error term

$$Error = \frac{4C_dC_m}{Kn(n-2)(n-3)} \sum_{i=1}^{n} \left(\sum_{l=1}^{n} |N_i^x - N_l^x| \sum_{l=1}^{n} V_k^T(\boldsymbol{y}_i - \boldsymbol{y}_l)| \right)$$

is bounded by

$$\mathbb{P}\left\{ Error < \frac{4C_dC_mn^2}{K\sqrt{k}(n-2)(n-3)} \frac{\sqrt{2\left(\ln\left(\frac{1}{2\delta}\right)+\epsilon\right)}}{\epsilon} \right\}$$
$$\geq 1 + n\left(\frac{4}{e^{\frac{2n}{n-1}}}\right)^n - 2n\left(\frac{2}{e^{\frac{n}{n-1}}}\right)^n.$$

Sketch of proof: The lower case k used in $\sqrt{k}$ refers to the dimension into which the data is projected after any random projection, as a generalization to not just having to project to 1 dimension. The proof is based on characterizing the terms in this error via folded normal distributions and their corresponding moment-generating functions in order to obtain a concentration bound. An empirical characterization of this concentrated upper bound on the error is provided in the experiments section.

We will first introduce the following sub-lemma that helps with the rest of this proof.

Lemma 2.5.1. *For a, b and $c \in \mathbb{R}^+$, given that $p\left(a_i < c\right) > b$ for $i = 1, 2, \ldots, n$, we have*

$$P\left(\sum a_i < nc\right) \geqslant 1 - n(1-b).$$

Proof. $p\left(\sum a_i < nc\right) \geqslant P\left(a_1 < c \cap a_2 < c \ldots . \cap a_n < c\right) = 1 - P\left(a_1 \geqslant c \cup a_2 \geqslant c \cup a_3 \geqslant C\right.$

$\ldots \quad \cup\, a_n \geqslant c) \geqslant 1 - p\,(a_1 \geqslant c) - p\,(a_2 \geqslant c) \ldots - p\,(a_n > c) \geq 1 - n(1-b).$ □

The following puts together the complete proof of Theorem 2.

Proof. As the N_i^X's are drawn from a zero mean Gaussian distribution where $\sigma^2 \geq w_2(P)\frac{\sqrt{2ln(1/2\delta)+\epsilon}}{\epsilon}$. Assuming N_i^X and N_l^X are sampled independently for any $l \neq i$, if $s_l = N_i^X - N_l^X$, the distribution of s_l terms will be therefore $N(0, 2\sigma^2)$. Now let $\tau_i = \sum_{l=1}^n |N_i^X - N_l^X| = \sum_{l=1, l\neq i}^n |N_i^X - N_l^X| = \sum_{l=1,l\neq i}^n |s_l|$. It is critical to note that the s_l's are not independent anymore. Therefore, we consider τ_i's instead as for the expectation $\mathbb{E}(\tau_i)$, the correlation between s_l's would not matter anymore. Now, $\mathbb{E}(\tau_i) = (n-1)\mathbb{E}(s_1)$. This brings us to the regime of half-normal or folded normal distributions. If $\boldsymbol{X} \sim N(0, \sigma^2), \boldsymbol{Y} = |\boldsymbol{X}|$, then we know that $\mathbb{E}[\boldsymbol{Y}] = \sigma\sqrt{\frac{2}{\pi}}$ and $Var(\boldsymbol{Y}) = \sigma^2(1 - \frac{2}{\pi})$. In our case, $\mathbb{E}[s_1] = \sigma\sqrt{2} \times \sqrt{\frac{2}{\pi}} = \frac{2\sigma}{\sqrt{\pi}}$ and $Var(s_1) = 2\sigma^2(1-\frac{2}{\pi})$. We now need a concentration bound for $\sum_{l=1,l\neq i}^n |N_i^X - N_l^X|$. We have

$$\sum_{l=1,l\neq i}^{n} |N_i^X - N_l^X| \leq (n-1)|N_i^X| + \sum_{l=1,l\neq i}^{n} |N_l^X|.$$

The moment generating function of a standard Gaussian random variable $\boldsymbol{X}$ is $\mathbb{E}[e^{t\boldsymbol{X}}] = e^{\frac{\sigma^2 t^2}{2}}$ for all $t \in \mathbb{R}$. We would want to find a concentration bound on $|(n-1)N_i^X| + \sum_{l=1,l\neq i}^{l=n} |N_i^X|$. We denote $(n-1)N_i^X$ as N_i'. Note that $(n-1)N_i^X$ and N_l^X where $l \neq i$ are all i.i.d. The required concentration bound can be written in terms of the moment generating function as

$$
\begin{aligned}
P\left\{\frac{1}{n}\left(|(n-1)N_i^X| + \sum_{l=1;l\neq i}^{l=n} |N_i^X|\right) \geq t\right\} &= P\left\{\exp\left(\frac{\lambda}{n}(|(n-1)N_i^X| + \sum_{l\neq i}^{l=n} |N_i^X|)\right) \geq e^{\lambda t}\right\} \\
&\leq e^{-\lambda t}\mathbb{E}\left[\exp\left(\frac{\lambda}{n}(|(n-1)N_i^X| + \sum_{l=1,l\neq i}^{l=n} |N_i^X|)\right)\right] \\
&= e^{-\lambda t}\mathbb{E}\left[\left(\prod_{k=1;k\neq i}^{n} e^{\frac{\lambda}{n}|N_k^X|}\right) e^{\frac{\lambda}{n}|(n-1)N_i^X|}\right] \\
&= e^{-\lambda t}\prod_{k=1;k\neq i}^{n} \mathbb{E}\left[e^{\frac{\lambda}{n}|N_k^X|}\right]\mathbb{E}\left[e^{\frac{\lambda}{n}|(n-1)N_i^X|}\right] \text{(as they are independent)}
\end{aligned}
$$

Now, based on the properties of moment-generating functions, we have

$$
\begin{aligned}
\mathbb{E}\left[e^{t|\boldsymbol{X}_k|}\right] &\leq \mathbb{E}\left[e^{t|\boldsymbol{X}_k|} + e^{-t|\boldsymbol{X}_k|}\right] \qquad (2.6) \\
&= \mathbb{E}[e^{t\boldsymbol{X}_k} + e^{-t\boldsymbol{X}_k}] \\
&= \mathbb{E}[e^{t\boldsymbol{X}_k}] + \mathbb{E}[e^{t\boldsymbol{X}_k}] \\
&= 2\mathbb{E}[e^{t\boldsymbol{X}_k}] \\
&= 2e^{\frac{\sigma^2 t^2}{2}} \text{as m.g.f for Gaussians is } m_X(t) = e^{\mu t + \frac{\sigma^2 t^2}{2}}.
\end{aligned}
$$

We have that $\mathbb{E}[e^{\frac{\lambda}{n}|N_k^X|}] \leq 2e^{\frac{\lambda^2\sigma^2}{2n^2}}$ and $\mathbb{E}[e^{\frac{\lambda}{n}|(n-1)N_i^X|}] \leq 2e^{\frac{\lambda^2}{2n^2}(n-1)\sigma^2}$.

From eqn. 2.6 the r.h.s of the inequality is $e^{-\lambda t}2^n e^{\frac{\lambda^2\sigma^2(n-1)}{n^2}}$. Therefore, $\mathbb{P}\{expression \geq nt\} \leq e^{-\lambda t}2^n e^{\frac{\lambda^2\sigma^2(n-1)}{n^2}}$. As we want to minimize the upper bound, $\therefore \frac{\partial}{\partial l}(-\lambda t + \frac{\lambda^2\sigma^2(n-1)}{n^2} = 0)$ or, $-t + 2\lambda\frac{(n-1)\sigma^2}{n^2} = 0$. So we have, $\lambda = \frac{n^2 t}{2\sigma^2(n-1)}$. Therefore, $\mathbb{P}\{expression \geq nt\} \leq e^{-\frac{n^2t^2}{4\sigma^2(n-1)}}2^n$. Now for $\sigma_1^2 = \frac{1}{k}$ (assume $\boldsymbol{V}_k^\top \sim N\left(0, \frac{1}{k}\right)$, we would have

$$
\mathbb{P}\left(\sum_{l=1}^{n}\left|\boldsymbol{V}_k^T(\boldsymbol{y}_i - \boldsymbol{y}_l)\right| \geqslant nt_1\right) \leq e^{\frac{-n^2t_1^2}{4(n-1)\sigma_1^2}} \times 2^n
$$

$$\Rightarrow \mathbb{P}\left(\sum_{k=1}^{n}\left|V_k^T\left(\boldsymbol{y}_i-\boldsymbol{y}_l\right)\right| \geqslant 2n\sigma_1\right) \leqslant e^{\frac{-n^2t_1^2}{4(n-1)\sigma_1^2}\times 2^n}$$

Now, considering $t \geq 2\sigma$, we have $e^{\frac{-n^2 4\sigma_1^2}{4(n-1)\sigma_1^2}\times 2^n} = e^{-\frac{n^2}{n-1}\times 2^n} = \left(\frac{2}{e^{\frac{n}{n-1}}}\right)^n$. Therefore, this implies that

$$\mathbb{P}\left(\sum_{i=1}^{n}\left|V_k^T\left(\boldsymbol{y}_i-\boldsymbol{y}_1\right)\right| \geqslant 2n\times\frac{1}{\sqrt{k}}\right) \leqslant \left(\frac{2}{e^{\frac{n}{n-1}}}\right)^n.$$

The more n increases, the tighter the bound gets. Similar to the above result we have

$$P\left(\sum_{1=1}^{n}|N_i^x-N_i^x| \geqslant nt_2\right) \leqslant e^{\frac{-n^2t_2^2}{4(n-1)\sigma_2^2}}\times 2^n.$$

We know that $\sigma_2 > w_2(p)\frac{\sqrt{2\left(\ln\left(\frac{1}{2\delta}\right)+\epsilon\right)}}{\epsilon}$, where $w_2(p)=1$; in this case, so $\sigma_2 \geqslant \frac{\sqrt{2\left(\ln\left(\frac{1}{2\delta}\right)+\epsilon\right)}}{\epsilon}$. Considering $t_2 = 2\sigma_2$, we have

$$\mathbb{P}\left(\sum_{i=1}^{n}|N_i^x-N_i^x| \geqslant 2n\sigma_2\right) \leqslant e^{-\frac{n^2}{n-1}\times 2^n} = e^{-\frac{n^2}{n-1}} = \left(\frac{2}{e^{\frac{n}{n-1}}}\right)^n,$$

$$\mathbb{P}\left(\sum_{i=1}^{n}|N_i^x-N_i^x| \geqslant 2n\frac{\sqrt{2\Bigl(\ln\Bigl(\frac{1}{2_\delta}\Bigr)+t}}{\epsilon}\right) \leq \left(\frac{2}{e^{\frac{n}{n-1}}}\right)^n.$$

Note that

$$P(\sum_{l=1}^{n}|\boldsymbol{V}_k^T(\boldsymbol{y}_i-\boldsymbol{y}_l)|\sum_{l=1}^{n}|N_i^X-N_l^X| < n^2t_1t_2) \geq P(\sum_{l=1}^{n}|\boldsymbol{V}_k^T(\boldsymbol{y}_i-\boldsymbol{y}_l)| < nt_1)P(\sum_{l=1}^{n}|N_i^X-N_l^X| < nt_2),$$

since these two events are clearly independent and for independent a, b, $P(ab < c^2) > P(a < c)P(b < c)$ holds. Therefore, $P(\sum_{l=1}^{n}|\boldsymbol{V}_k^T(\boldsymbol{y}_i-\boldsymbol{y}_l)|\sum_{l=1}^{n}|N_i^X-N_l^X| < n^2t_1t_2) \geq (1-$

$\alpha_1)(1-\alpha_2)$, where $\alpha_1 = \left(\frac{2}{e^{\frac{n}{n-1}}}\right)^n, \alpha_2 = \left(\frac{2}{e^{\frac{n}{n-1}}}\right)^n$. Now, using Lemma 2.5.1, we get

$$P(\sum_{i=1}^{n}(\sum_{l=1}^{n}|\boldsymbol{V}_k^T(\boldsymbol{y}_i-\boldsymbol{y}_l)|\sum_{l=1}^{n}|N_i^X - N_l^X| < n^3 t_1 t_2)) \geq 1 - n(\alpha_1+\alpha_2-\alpha_1\alpha_2).$$

So, for any constant C, we can write

$$P(C\sum_{i=1}^{n}(\sum_{l=1}^{n}|\boldsymbol{V}_k^T(\boldsymbol{y}_i-\boldsymbol{y}_l)|\sum_{l=1}^{n}|N_i^X - N_l^X| < Cn^3 t_1 t_2)) \geq 1 - n(\alpha_1+\alpha_2-\alpha_1\alpha_2),$$

and substituting the values for α_1, α_2, we get the desired value. Therefore, for each n,

$$\frac{4C_dC_m}{Kn(n-2)(n-3)}\sum_{i=1}^{n}\left(\sum_{j=1}^{n}|N_i^X - N_j^X|\sum_{j=1}^{n}\boldsymbol{V}_K^T(\boldsymbol{y}_i-\boldsymbol{y}_j)|\right)$$

is upper bounded with a very high probability and roughly for $n \geq 50$ this value is almost 1, and the probability value is an increasing function in n; so for large n, this value can be treated as approximately 1. The error term is a summation for all n from 1 to K, and hence, the error term is bounded with a high probability. □

Theorem 3. **(Decomposition theorem)** The difference between estimators of $\overline{\Omega}^{dp}(\boldsymbol{X},\boldsymbol{Y})$ and $\overline{\Omega}(\boldsymbol{X},\boldsymbol{Y})$ can be expressed as

$$\begin{aligned}&\overline{\Omega}^{dp}(\boldsymbol{X},\boldsymbol{Y}) - \overline{\Omega}(\boldsymbol{X},\boldsymbol{Y}) = \overline{\Omega}(N^x,\boldsymbol{Y})\\ &+\sum_{n=1}^{K}\frac{4C_dC_m}{Kn(n-2)(n-3)}\sum_{i=1}^{n}\left(\sum_{l=1}^{n}|N_i^x - N_l^x|\sum_{l=1}^{n}\boldsymbol{V}_k^\top(\boldsymbol{y}_i-\boldsymbol{y}_l)|\right).\end{aligned} \tag{2.7}$$

Proof.

$$\overline{\Omega}^{dp}(\boldsymbol{X}+N_X,\boldsymbol{Y})=\sum_{n=1}^{K}\frac{C_pC_q\Omega_n(u_K^\top\boldsymbol{X}+N_X,\boldsymbol{v}_K\boldsymbol{Y})}{K}$$

$$=\sum_{n=1}^{K}\frac{C_pC_q}{K}\Big\{\frac{1}{n(n-3)}\sum_{i\neq j}\tilde{a}_{ij}\tilde{b}_{ij}-\frac{2}{n(n-2)(n-3)}\sum\tilde{a_{i}}.\tilde{b_{i}}.+\frac{a_{..}b_{..}}{n(n-1)(n-2)(n-3)}\Big\}$$

$$=\sum_{n=1}^{K}\frac{C_pC_q}{K}\Bigg\{\frac{1}{n(n-3)}\sum_{i\neq j}|\boldsymbol{u}_K^\top(\boldsymbol{x}_i-\boldsymbol{x}_j)+N_i^X-N_j^X||\boldsymbol{V}_K^\top(\boldsymbol{y}_i-\boldsymbol{y}_j)|$$

$$-\frac{2}{n(n-2)(n-3)}\sum_{i=1}^{n}(\sum_{j=1}^{n}|\boldsymbol{u}_K^\top(\boldsymbol{x}_i-\boldsymbol{x}_j)+N_i^X-N_j^X|\sum_{j=1}^{n}|\boldsymbol{V}_K^\top(\boldsymbol{y}_i-\boldsymbol{y}_j)|)$$

$$+\frac{(\sum_{i,j}^{n}|\boldsymbol{u}_K^\top(\boldsymbol{x}_i-\boldsymbol{x}_j)+N_i^X-N_j^X|\sum_{k,l}^{n}|\boldsymbol{V}_K^T(\boldsymbol{y}_k-\boldsymbol{y}_l)|)}{n(n-1)(n-2)(n-3)}\Bigg\}$$

$$\leq\sum_{n=1}^{K}\frac{C_pC_q}{K}\Big\{\frac{1}{n(n-3)}\sum_{i\neq j}|\boldsymbol{u}_K^\top(\boldsymbol{X}_i-\boldsymbol{X}_j)||\boldsymbol{V}_K^\top(\boldsymbol{y}_i-\boldsymbol{y}_j)|+\frac{1}{n(n-3)}\sum_{i\neq j}\boldsymbol{y}_i-\boldsymbol{y}_j)|$$

$$-\frac{2}{n(n-2)(n-3)}\sum_{i=1}^{n}\Bigg(\sum_{j=1}^{n}|\boldsymbol{u}_K^\top(\boldsymbol{x}_i-\boldsymbol{x}_j)|\sum_{j=1}^{n}V_K^T(\boldsymbol{y}_i-\boldsymbol{y}_j)|\Bigg)$$

$$+\frac{2}{n(n-2)(n-3)}\sum_{i=1}^{n}\Bigg(\sum_{j=1}^{n}|N_i^X-N_j^X|\sum_{j=1}^{n}\boldsymbol{V}_K^T(\boldsymbol{y}_i-\boldsymbol{y}_j)|\Bigg)$$

$$+\frac{1}{n(n-1)(n-2)(n-3)}\Bigg(\sum_{i,j}^{n}|\boldsymbol{u}_K^\top(\boldsymbol{x}_i-\boldsymbol{x}_j)|\sum_{k,l}^{n}|\boldsymbol{V}_K^\top(\boldsymbol{y}_k-\boldsymbol{y}_l)|+\sum_{i,j}^{n}|N_i^X-N_j^X|\sum_{k,l}^{n}|\boldsymbol{V}_K^T(\boldsymbol{y}_k-\boldsymbol{y}_l)|\Bigg)\Big\}$$

$$=\sum_{n=1}^{K}\frac{C_pC_q}{K}\Big\{\frac{1}{n(n-3)}\sum_{i\neq j}|\boldsymbol{u}_K^\top(\boldsymbol{x}_i-\boldsymbol{x}_j)||\boldsymbol{V}_K^\top(\boldsymbol{y}_i-\boldsymbol{y}_j)| \tag{2.8}$$

$$-\frac{2}{n(n-2)(n-3)}\sum_{i=1}^{n}\Bigg(\sum_{j=1}^{n}|\boldsymbol{u}_K^T(\boldsymbol{x}_i-\boldsymbol{x}_j)|\sum_{j=1}^{n}\boldsymbol{V}_K^\top(\boldsymbol{y}_i-\boldsymbol{y}_j)|\Bigg)$$

$$+\frac{1}{n(n-1)(n-2)(n-3)}\Bigg(\sum_{i,j}^{n}|\boldsymbol{u}_K^\top(\boldsymbol{x}_i-\boldsymbol{x}_j)|\sum_{k,l}^{n}|\boldsymbol{V}_K^\top(\boldsymbol{y}_k-\boldsymbol{y}_l)|\Bigg)$$

$$-\frac{2}{n(n-2)(n-3)}\sum_{i=1}^{n}\Bigg(\sum_{j=1}^{n}|N_i^X-N_j^X|\sum_{j=1}^{n}V_K^T(\boldsymbol{y}_i-\boldsymbol{y}_j)|\Bigg)+\frac{1}{n(n-3)}\sum_{i\neq j}|(N_i^X-N_j^X)||\boldsymbol{V}_K^T(\boldsymbol{y}_i-\boldsymbol{y}_j)|$$

$$+\frac{1}{n(n-1)(n-2)(n-3)}\Bigg(\sum_{i,j}^{n}|N_i^X-N_j^X|\sum_{k,l}^{n}|\boldsymbol{V}_K^\top(\boldsymbol{y}_k-\boldsymbol{y}_l)|\Bigg)\Big\}$$

$$+\sum_{n=1}^{K}\frac{4C_pC_q}{Kn(n-2)(n-3)}\sum_{i=1}^{n}\Bigg(\sum_{j=1}^{n}|N_i^X-N_j^X|\sum_{j=1}^{n}\boldsymbol{V}_K^\top(\boldsymbol{y}_i-\boldsymbol{y}_j)|\Bigg)$$

$$=\overline{\Omega}(\boldsymbol{X},\boldsymbol{Y})+\overline{\Omega}(N^X,\boldsymbol{Y})+\sum_{n=1}^{K}\frac{4C_pC_q}{Kn(n-2)(n-3)}\sum_{i=1}^{n}\Bigg(\sum_{j=1}^{n}|N_i^X-N_j^X|\sum_{j=1}^{n}\boldsymbol{V}_K^\top(\boldsymbol{y}_i-\boldsymbol{y}_j)|\Bigg). \tag{2.9}$$

PNC-Estimator

1. **Alice's input:** Data matrix $\boldsymbol{X}$
2. **Bob's input:** Data matrix $\boldsymbol{Y}$
3. **Alice's side:** The client takes the following actions:
   (a) Computes $\overline{\Omega}^{dp}(\boldsymbol{X}, \boldsymbol{Y})$ using one of the proposed equations in 2.5 or 2.4.1.
   (b) Computes $\overline{\Omega}^{dp}(\boldsymbol{X}, \boldsymbol{X})$ by adding noise calibrated to global sensitivity of $\frac{12n-11}{(n-1)^2}$ to $\overline{\Omega}(\boldsymbol{X}, \boldsymbol{X})$
   (c) The client sends the obtained $\overline{\Omega}^{dp}(\boldsymbol{X}, \boldsymbol{X})$ and either of $\overline{\Omega}^{dp}(\boldsymbol{X}, \boldsymbol{Y})$ or $\Omega^{dp}_{\text{Disjt}}(\boldsymbol{X}, \boldsymbol{Y})$ to Bob.
4. **Bob's side:**
   (a) Bob computes non-private $\overline{\Omega}(\boldsymbol{Y}, \boldsymbol{Y})$.
   (b) Bob puts together all these estimates to compute the distance correlation between $\boldsymbol{X}$ and $\boldsymbol{Y}$ according to 2.4.

Figure 2.4: Protocol for private estimation of non-linear correlations between data of Alice and Bob.

□

Therefore, the error in estimation within this context depends on two error terms of $\bar{\Omega}(N^x, \boldsymbol{Y})$ and $\frac{4C_dC_m}{Kn(n-2)(n-3)} \sum_{i=1}^{n} \left(\sum_{l=1}^{n} |N_i^x - N_l^x| \sum_{l=1}^{n} \boldsymbol{V}_k^T(\boldsymbol{y}_i - \boldsymbol{y}_l)|\right)$. We now upper bound the second error term in the following theorem. *Choosing N^x for differential privacy*: Several mechanisms have been proposed for releasing random projections of data with differential privacy, including[277,61,522] and more recently[575,201]. We now restate one such mechanism as shown below.

PRIVATIZING $\overline{\Omega}(\boldsymbol{X}, \boldsymbol{X})$

The problem of privatizing $\overline{\Omega}(\boldsymbol{X}, \boldsymbol{X})$ is simpler than that of privatizing $\overline{\Omega}(\boldsymbol{X}, \boldsymbol{Y})$, in the sense that in the former case, the entire data $\boldsymbol{X}$ is hosted on one entity as opposed to $\boldsymbol{X}$ and $\boldsymbol{Y}$ being hosted on two different entities. For the private estimation of distance variance terms $\overline{\Omega}^{dp}(\boldsymbol{X}, \boldsymbol{X})$, we first use an equivalence between distance covariance and another popular dependency measure called Hilbert-Schmidt Independence Criterion (HSIC)[204]. We then use the global sensitivity of this equivalently obtained HSIC to privatize the distance variance. We begin by defining the empirical estimate of the HSIC.

Given unique positive definite kernels k, l in the context of kernel methods and reproducing kernel Hilbert space (RKHS) theory in machine learning, we have the following definition for the sample estimator of HSIC.

Definition 2.5.2. *(**HSIC**[204]) Let $Z := \{(\boldsymbol{x}_1, \boldsymbol{y}_1), \ldots, (\boldsymbol{x}_m, \boldsymbol{y}_m)\} \subseteq \mathcal{X} \times \mathcal{Y}$ be a series of m independent observations drawn from $p_{\boldsymbol{x},\boldsymbol{y}}$. An estimator of HSIC, written by $\operatorname{HSIC}(Z, \mathcal{F}, \mathcal{G})$, is given by*

$$\operatorname{HSIC}(Z, \mathcal{F}, \mathcal{G}) := (m-1)^{-2} \operatorname{Tr}(\boldsymbol{KHLH}),$$

where $\boldsymbol{H}, \boldsymbol{K}, \boldsymbol{L} \in \mathbb{R}^{m \times m}, \boldsymbol{K}_{ij} := k(\boldsymbol{x}_i, \boldsymbol{x}_j), \boldsymbol{L}_{ij} := l(\boldsymbol{y}_i, \boldsymbol{y}_j)$ and $\boldsymbol{H}$ is a double-centering matrix.

We assume k, l are bounded above by 1 (e.g., using the squared exponential kernel or the Matern kernel[565]). A classic way to calibrate the amount of noise required to achieve $\epsilon-$differential privacy[160,152,154] is to add noise with a variance of $\frac{\Delta}{\epsilon}$ where Δ is the global sensitivity of the query.

Global sensitivity of HSIC: The global sensitivity of HSIC was derived in[296,297] to be at most

$\frac{12n-11}{(n-1)^2}$. Specifically,

$$\left|\widehat{HSIC}_{k,l}\left(\boldsymbol{x},\boldsymbol{y}\right)-\widehat{HSIC}_{k,l}\left(\boldsymbol{x}',\boldsymbol{y}'\right)\right|\leq\frac{12n-11}{(n-1)^2},$$

for all neighboring[160] datasets.

The following equivalence was shown between distance covariance and HSIC. As we have the global sensitivity for HSIC and since we have the following equivalence, we can use this global sensitivity to privatize $\overline{\Omega}(\boldsymbol{X},\boldsymbol{X})$, which belongs to the one-party setting of privately releasing distance covariance.

Equivalence map between Distance Covariance and HSIC

Definition 2.5.3. *Bijective induced kernel*

Given sample data $\{x_i, i=1,\ldots,n\}$, for any metric $d(\cdot,\cdot)$, we define its bijective induced kernel[469,470] as

$$\hat{k}_d\left(\boldsymbol{x}_i,\boldsymbol{x}_j\right)=\max_{s,t\in[n]}\left(d\left(\boldsymbol{x}_s,\boldsymbol{x}_t\right)\right)-d\left(\boldsymbol{x}_i,\boldsymbol{x}_j\right).$$

For any kernel $k(\cdot,\cdot)$, we define the induced metric as

$$\hat{d}_k\left(\boldsymbol{x}_i,\boldsymbol{x}_j\right)=\max_{s,t\in[n]}\left(k\left(\boldsymbol{x}_s,\boldsymbol{x}_t\right)\right)-k\left(\boldsymbol{x}_i,\boldsymbol{x}_j\right).$$

The subscripts $s,t\in[n]$ is a shorthand for $s=1,\ldots,n$ and $t=1,\ldots,n$.

Other alternative definitions for this bijection are also given in[470]. These are widely used depending on the problem at hand.

$$\hat{d}_k\left(\boldsymbol{x}_i, \boldsymbol{x}_j\right) = 1 - k\left(\boldsymbol{x}_i, \boldsymbol{x}_j\right) / \max_{s,t\in[n]}\left(k\left(\boldsymbol{x}_s, \boldsymbol{x}_t\right)\right),$$

$$\hat{k}_d\left(\boldsymbol{x}_i, \boldsymbol{x}_j\right) = 1 - d\left(\boldsymbol{x}_i, \boldsymbol{x}_j\right) / \max_{s,t\in[n]}\left(d\left(\boldsymbol{x}_s, \boldsymbol{x}_t\right)\right).$$

This is an equivalent definition up to scaling by the maximum elements and can be succinctly expressed in a matrix form:

$$\hat{\boldsymbol{D}}_{\boldsymbol{K}} = \boldsymbol{J} - \boldsymbol{K}/\max(\boldsymbol{K}),$$

$$\hat{\boldsymbol{K}}_{\boldsymbol{D}} = \boldsymbol{J} - \boldsymbol{D}/\max(\boldsymbol{D}).$$

Theorem 4. [470] Suppose distance covariance (DCOV) uses a given metric $d(\cdot,\cdot)$, and the Hilbert Schmidt independence criterion HSIC uses the bijective induced kernel $\hat{k}_d(\cdot,\cdot)$. Given any sample data $(\boldsymbol{X}, \boldsymbol{Y})$, it holds that

$$\mathrm{DCOV}_n(\boldsymbol{X}, \boldsymbol{Y}) = \mathrm{HSIC}_n(\boldsymbol{X}, \boldsymbol{Y}),$$

where the remainder term $O\left(\frac{1}{n^2}\right)$ is invariant to permutation.

2.6 Computational complexity

Fast estimators of distance correlation requires $\mathcal{O}(nlogn)$ [102,238] computational complexity for univariate and $\mathcal{O}(nKlogn)$ complexity[234] for multivariate settings with $\mathcal{O}(\max(n, K))$ memory, where K is the number of random projections required as part of the estimation. In addition to being differentiable and easily computable with a closed-form, it requires no other tuning of parameters and is self-contained, unlike HSIC or other dependency measures such as Maximum Mean Discrepancy (MMD) or Kernel Target Alignment (KTA) that depend on a choice of separate kernels for features as well as labels along with their respective tuning parameters.

"Doing a meta-analysis is easy... Doing one well is hard."

Ingram Olkin

3

Private Independence Testing

3.1 Method for private independence testing

The preliminaries for independence testing were provided in Section 2.3. This chapter is based on our work in[535]. As a starting point for the problem of private independence testing, we first provide the test-statistic for non-private independence testing in section 3.1. Privatizing this test-statistic requires privatization of **(a).** Its numerator, given by the *distance covariance* and denoted by $\hat{\Omega}^2(\boldsymbol{X}, \boldsymbol{Y})$ as well as **(b).** Its denominator, given by a product of individually computed means of pairwise distances of $\boldsymbol{X}$ and $\boldsymbol{Y}$, that is denoted by $\hat{S}(\boldsymbol{X}, \boldsymbol{Y})$. These terms are formally defined in Section 3.1. The privatization strategy that we employ for both of these terms relies on a reduction to *directional variance queries*, which we define below.

Alternate privatization strategy for $\hat{\Omega}^2(\boldsymbol{X}, \boldsymbol{Y})$: We first express $\hat{\Omega}^2(\boldsymbol{X}, \boldsymbol{Y})$ as a function of i) a covariance matrix formed by incidence matrices (or a matrix square root) of a specific weighted graph formed over samples in $\boldsymbol{X}$ as its vertices (that we refer to as incidence-covariance) and ii) the covariance matrix of $\boldsymbol{Y}$. We then show that $\hat{\Omega}^2(\boldsymbol{X}, \boldsymbol{Y})$ is exactly equal to a sum of *directional variances* of columns of $\boldsymbol{Y}$ with respect to the incidence-covariance matrix of $\boldsymbol{X}$.

Alternate privatization strategy for $\hat{S}(\boldsymbol{X}, \boldsymbol{Y})$: We are able to express $\hat{S}(\boldsymbol{X}, \boldsymbol{Y})$ in a similar form as $\hat{\Omega}^2(\boldsymbol{X}, \boldsymbol{Y})$, but with some key differences. Namely, we can write it as a function of the covariance of $\boldsymbol{X}$ (given by $\boldsymbol{X}\boldsymbol{X}^\top$) and a data-independent graph Laplacian matrix $\boldsymbol{L}^{\boldsymbol{S}} = n\boldsymbol{I} - \boldsymbol{e}\boldsymbol{e}^\top$. We show that this is exactly equal to the sum of directional variances of the columns of the incidence matrix (or a matrix square root such that $\boldsymbol{L}^{\boldsymbol{S}} = \boldsymbol{B}\boldsymbol{B}^\top$) with respect to covariance matrix $\boldsymbol{X}\boldsymbol{X}^\top$. A key difference with respect to the case of $\hat{\Omega}^2(\boldsymbol{X}, \boldsymbol{Y})$ is that, here, the directional variance is with respect to the covariance matrix of $\boldsymbol{X}$ while in $\hat{\Omega}^2(\boldsymbol{X}, \boldsymbol{Y})$ the directional variance is with respect to the incidence-covariance matrix defined over $\boldsymbol{X}$.

Therefore, in the notation of directional variances, we show that $\hat{\Omega}^2(\boldsymbol{X}, \boldsymbol{Y}) = \sum_i \Phi_{\boldsymbol{B}(\boldsymbol{X})}(\boldsymbol{y}_i)$, where $\boldsymbol{y}_i$ refers to the i'th column of $\boldsymbol{Y}$, and $\hat{S}(\boldsymbol{X}, \boldsymbol{Y}) = \sum_i \Phi_{\boldsymbol{X}}(\boldsymbol{g}_i)$, where $\boldsymbol{g}_i$ refers to the i'th column of $\boldsymbol{G}$. A utility bound on computing directional variance queries upon privatization of covariance matrices can be expressed in terms of additive and multiplicative errors. We now define the structure of such approximations as below. In the rest of the paper, for presentation simplicity, we refer to $\hat{\Omega}^2(\boldsymbol{X}, \boldsymbol{Y})$ and $\hat{S}(\boldsymbol{X}, \boldsymbol{Y})$ by $\hat{\Omega}^2$ and $\hat{S}$, respectively.

Definition 3.1.1 (Additive and multiplicative approximation)**.** *We say a privacy mechanism that* *privatizes an estimate* $\hat{A}(x)$ *using* $\bar{A}(x)$*, provides an* (η, τ, ν)-approximation*, if the following holds:*

$$\Pr\left[(1-\eta)\hat{A}(x) - \tau \leq \bar{A}(x) \leq (1+\eta)\hat{A}(x) + \tau\right] \geq 1 - \nu.$$

We use $\bar{\Omega}^2, \bar{S}$ to denote the privatized versions of $\hat{\Omega}^2$ and $\hat{S}$, respectively, that can be obtained upon privatization of the covariance matrix $\boldsymbol{X}\boldsymbol{X}^\top$ and incidence-covariance $\boldsymbol{B}(\boldsymbol{X})\boldsymbol{B}^\top(\boldsymbol{X})$. The goal is to bound the ratio $\frac{\bar{\Omega}^2}{\bar{S}}$ as follows:

$$\alpha_\ell \frac{\hat{\Omega}^2}{\hat{S}} + \beta_\ell \leq \frac{\bar{\Omega}^2}{\bar{S}} \leq \alpha_u \frac{\hat{\Omega}^2}{\hat{S}} + \beta_u. \tag{3.1}$$

In section 3.2, we first show that privatization of covariances and incidence-covariances results in separate additive and multiplicative approximations for $\hat{\Omega}^2$ and $\hat{S}^2$, respectively. A first-pass attempt towards our main goal of an approximation of the form in equation 3.1 just involves a simple rearrangement of the terms using the individual bounds on $\hat{\Omega}^2$ and $\hat{S}$, leading us to the following result with probability $\geq 1-2\nu$ where we have, $\frac{(1-\eta)\hat{\Omega}^2-\tau}{(1+\eta)\hat{S}+\tau} \leq \frac{\bar{\Omega}^2}{\bar{S}} \leq \frac{(1+\eta)\hat{\Omega}^2+\tau}{(1-\eta)\hat{S}-\tau}$. Unfortunately, this first attempt falls short of our goal, as the additive and multiplicative errors in equation 3.1 do not decompose well to provide a more convenient expression of the form in equation 3.1. Obtaining a final form of this kind requires extra work, leading to our main result.

Test statistic for independence testing

The test-statistic we consider is a ratio of sample distance covariance (an un-normalized measure of statistical dependency) denoted by $\hat{\Omega}^2(\boldsymbol{X}, \boldsymbol{Y})$ and a product of average of distances denoted by $\hat{S}(\boldsymbol{X}, \boldsymbol{Y})$ defined as $\hat{S}(\boldsymbol{X}, \boldsymbol{Y}) = \frac{1}{n^2}\sum_{k,l=1}^n \|\boldsymbol{x}_k - \boldsymbol{x}_l\|^2 \frac{1}{n^2}\sum_{k,l=1}^n \|\boldsymbol{y}_k - \boldsymbol{y}_l\|^2$. In order to define the test statistic, we need to define distance covariance denoted as $\hat{\Omega}^2(\boldsymbol{X}, \boldsymbol{Y}) = \hat{R} + \hat{S} -$

$2\hat{T}$, where

$$\begin{aligned}
\hat{R} &= \frac{1}{n^2}\sum_{k,l=1}^{n} \left\|\boldsymbol{x}_k - \boldsymbol{x}_l\right\|^2 \left\|\boldsymbol{y}_k - \boldsymbol{y}_l\right\|^2 , \\
\hat{S} &= \frac{1}{n^2}\sum_{k,l=1}^{n} \left\|\boldsymbol{x}_k - \boldsymbol{x}_l\right\|^2 \frac{1}{n^2}\sum_{k,l=1}^{n} \left\|\boldsymbol{y}_k - \boldsymbol{y}_k\right\|^2 , \\
\hat{T} &= \frac{1}{n^3}\sum_{k=1}^{n}\sum_{l,m=1}^{n} \left\|\boldsymbol{x}_k - \boldsymbol{x}_l\right\|^2 \left\|\boldsymbol{y}_k - \boldsymbol{y}_l\right\|^2 .
\end{aligned}$$

We use $\Gamma(X, Y, \alpha, n)$ to denote the test statistic[500] that rejects independence when

$$\Gamma(\mathbf{X}, \mathbf{Y}, \alpha, n) = \frac{n\hat{\Omega}^2(\mathbf{X}, \mathbf{Y})}{\hat{S}(\mathbf{X}, \mathbf{Y})} > \left(\phi^{-1}(1-\alpha/2)\right)^2 ,$$

where $\phi(\cdot)$ denotes the standard normal cumulative distribution function, and α denotes the achieved significance level. A test rejecting independence of X and Y when $\sqrt{n\hat{\Omega}^2(\boldsymbol{X}, \boldsymbol{Y})/\hat{S}(\boldsymbol{X}, \boldsymbol{Y})} \geq \phi^{-1}(1- \alpha/2)$ is said to have an asymptotic significance level of at most α.

3.2 Privatization of test statistic

We first express the numerator and denominator of the test-statistic as different functions of directional variance. This is useful because the directional variance in our case is solely a function of a specific covariance (specific to the choice of numerator or denominator) of our sensitive data and the non-sensitive $\boldsymbol{Y}$. Therefore, the directional variances corresponding to the numerator and denominator of test-statistic are private upon applying the post-processing property of differential privacy after the privatization of that specific covariance. The following two results express the numerator $\hat{\Omega}^2(\boldsymbol{X}, \boldsymbol{Y})$ in terms of directional variance.

Lemma 3.2.1. *Distance covariance $\hat{\Omega}^2(\boldsymbol{X}, \boldsymbol{Y})$ can be estimated using the Euclidean distance matrix $\boldsymbol{E_X}$ formed over the rows in $\boldsymbol{X}$, the double-centering matrix $\boldsymbol{J} = \boldsymbol{I} - n^{-1}\boldsymbol{e}\boldsymbol{e}^{\boldsymbol{T}}$, to form* *an adjacency matrix given by* $\boldsymbol{W}(\boldsymbol{X}) = \boldsymbol{J}\boldsymbol{E}_{\boldsymbol{X}}\boldsymbol{J}$ *and a corresponding graph Laplacian*

$$\boldsymbol{L}^{\boldsymbol{W}}(\boldsymbol{X}) = \boldsymbol{D}(\boldsymbol{W}(\boldsymbol{X})) - \boldsymbol{W}(\boldsymbol{X})$$

where $\boldsymbol{D}(\boldsymbol{W}(\boldsymbol{X}))$ *is the degree matrix of* $\boldsymbol{W}(\boldsymbol{X})$ *to get*

$$\hat{\Omega}^2(\boldsymbol{X}, \boldsymbol{Y}) = \frac{1}{n}\sqrt{2\operatorname{Tr}\left(\boldsymbol{Y}^\top \boldsymbol{L}^{\boldsymbol{W}}(\boldsymbol{X})\boldsymbol{Y}\right)} = \frac{1}{n}\sqrt{2\operatorname{Tr}\left(\boldsymbol{X}^\top \boldsymbol{L}^{\boldsymbol{W}}(\boldsymbol{Y})\boldsymbol{X}\right)}$$

Proof. The proof is provided in Appendix 3.5.4. □

Now that distance covariance has been expressed in terms of a specific graph Laplacian, the following corollary follows to reformulate it as a sum of specific kinds of directional variances that depend on this graph Laplacian.

Corollary 3.2.1. *Distance covariance can be expressed as a sum of directional variances as* $\hat{\Omega}^2(\boldsymbol{X}, \boldsymbol{Y}) = \sum_i \Phi_{\boldsymbol{B}(\boldsymbol{X})}(\boldsymbol{y}_i)$ *where directional variance was defined in Definition 2.3.4.*

Proof. We expand the distance covariance estimator as follows to get this result.

$$\begin{aligned}\hat{\Omega}^2(\boldsymbol{X}, \boldsymbol{Y}) &= \operatorname{Tr}\left(\boldsymbol{X}^\top \boldsymbol{L}^{\boldsymbol{W}}(\boldsymbol{Y})\boldsymbol{X}\right) = \operatorname{Tr}\left(\boldsymbol{X}\boldsymbol{X}^\top \boldsymbol{L}^{\boldsymbol{W}}(\boldsymbol{Y})\right) = \sum_i^d \left(\boldsymbol{X}_{\boldsymbol{i}}^\top \boldsymbol{L}^{\boldsymbol{W}}(\boldsymbol{Y})\boldsymbol{X}_{\boldsymbol{i}}\right) \quad (3.2)\\ &= \sum_{i=1}^d \operatorname{Tr}\left(\boldsymbol{X}_{\boldsymbol{i}}\boldsymbol{X}_{\boldsymbol{i}}{}^\top \boldsymbol{L}^{\boldsymbol{W}}(\boldsymbol{Y})\right) = \sum_{i=1}^m \operatorname{Tr}\left(\boldsymbol{Y}_{\boldsymbol{i}}\boldsymbol{Y}_{\boldsymbol{i}}{}^\top \boldsymbol{L}^{\boldsymbol{W}}(\boldsymbol{X})\right) = \sum_{i=1}^m \operatorname{Tr}\left(\boldsymbol{Y}_{\boldsymbol{i}}\boldsymbol{Y}_{\boldsymbol{i}}{}^\top \boldsymbol{B}(\boldsymbol{X})\boldsymbol{B}^\top(\boldsymbol{X})\right)\\ &= \sum_i^m \left(\boldsymbol{Y}_{\boldsymbol{i}}{}^\top \boldsymbol{B}(\boldsymbol{X})\boldsymbol{B}^\top(\boldsymbol{X})\boldsymbol{Y}_{\boldsymbol{i}}\right) = \sum_{i=1}^m \hat{\Omega}_i(\boldsymbol{X}, \boldsymbol{Y}) = \sum_i \Phi_{\boldsymbol{B}(\boldsymbol{X})}(\boldsymbol{Y}_i). \quad (3.3)\end{aligned}$$

□

Lemma 3.2.2. *We now express the denominator term of the test-statistic as a specific sum of* *directional variances as follows. The denominator in the test-statistic,*

$$\hat{S}(\boldsymbol{X},\boldsymbol{Y}) = \frac{1}{n^2}\sum_{k,l=1}^{n} \|\boldsymbol{x}_k - \boldsymbol{x}_l\|^2 \, \frac{1}{n^2}\sum_{k,l=1}^{n} \|\boldsymbol{y}_k - \boldsymbol{y}_l\|^2 \, ,$$

can be expressed as a sum of directional variances as $\hat{S}(\boldsymbol{X},\boldsymbol{Y}) = \sum_i \phi_{\boldsymbol{X}}(\boldsymbol{g_i})$ *where* $\boldsymbol{g}_i$ *is the* i*'th column of a matrix* $\boldsymbol{G}$ *such that* $\boldsymbol{L^S} = \boldsymbol{G}\boldsymbol{G}^T$ *for a graph Laplacian matrix* $\mathbf{L^S}$ *given by* $\mathbf{L^S} = n\boldsymbol{I} - \boldsymbol{e}\boldsymbol{e^T}$*. We use the superscript* $\boldsymbol{S}$ *to distinguish from the Laplacian* $\boldsymbol{L^W}$ *used in the expression for* $\hat{\Omega}^2$*.*

Proof. The proof is provided in the Appendix 3.5.5. □

The privatization strategy therefore is to privatize $\boldsymbol{B(X)B^T(X)}$ in order to privatize $\hat{\Omega}^2(\boldsymbol{X},\boldsymbol{Y}) = \sum_i \phi_{\boldsymbol{B(X)}}(\boldsymbol{y}_i)$. This privatizes the numerator of the test-statistic. With regards to the denominator, the strategy is to privatize $\boldsymbol{X}\boldsymbol{X^T}$ in order to privatize $\hat{S}(\boldsymbol{X},\boldsymbol{Y}) = \sum_i \phi_{\boldsymbol{X}}(\boldsymbol{g_i})$. Note that in both cases, the privatization required is with respect to covariance matrices. Privatization of covariances was studied in[61,22,60]. We summarize our proposed mechanism called π-test in Figure 7.5. With covariance matrices being central to many downstream queries, the question of understanding the effect of their privatization on the utility of downstream queries often arises.

3.2.1 Utility results

In that spirit, the work in[61] provides a utility analysis of directional variances (expressed as functions of covariances) upon privatization of these covariances. This utility bound is formulated in terms of additive and multiplicative errors as part of a probability bound. The notion of additive and multiplicative approximation was defined in preliminaries under definition 3.1.1. This result only provides us individual bounds (with additive and multiplicative factors) separately

$\pi-$test mechanism

1. **Alice's input:** Data matrix $\boldsymbol{X}_{n\times d}$, parameters for privacy (ϵ, δ), confidence ν & error η.
2. **Bob's input:** Data matrix $\boldsymbol{Y}$.
3. **Alice's side:** Compute adjacency matrix $\boldsymbol{W}(\boldsymbol{X}) = \boldsymbol{J}\boldsymbol{E}_{\boldsymbol{X}}\boldsymbol{J}$
4. **Alice's side:** Compute graph Laplacian $\boldsymbol{L}^{\boldsymbol{W}}(\boldsymbol{X})$ for adjacency of $\boldsymbol{W}(\boldsymbol{X})$.
5. **Alice's side:** Express $\boldsymbol{L}^{\boldsymbol{W}}(\boldsymbol{X})$ as $\boldsymbol{B}(\boldsymbol{X})\boldsymbol{B}(\boldsymbol{X})^{\boldsymbol{T}}$ via a matrix square-root.
6. **Alice's side:** Privatize covariances $\boldsymbol{B}(\boldsymbol{X})\boldsymbol{B}(\boldsymbol{X})^{\boldsymbol{T}}$ using either of[61,60,22] and send them to Bob.
7. **Alice's side:** Compute $\boldsymbol{X}\boldsymbol{X}^{\boldsymbol{T}}$ and privatize these covariances as in the previous step and send them to Bob.
8. **Bob's side:** Compute $\bar{\Omega}^2(\boldsymbol{X},\boldsymbol{Y}) = \frac{1}{n}\sqrt{2\,\mathrm{Tr}\left(\boldsymbol{Y}^\top \boldsymbol{L}^{\boldsymbol{W}}(\boldsymbol{X})\boldsymbol{Y}\right)}$ using Alice's private estimate of $\mathbf{L}^{\mathbf{W}}(\mathbf{X})$.
9. **Bob's side:** Compute $\bar{S}(\boldsymbol{X},\boldsymbol{Y}) = \frac{4}{n^4}\,\mathrm{Tr}\left(\boldsymbol{G}^{\boldsymbol{T}}\boldsymbol{X}\boldsymbol{X}^\top\boldsymbol{G}\right).\,\mathrm{Tr}\left(\boldsymbol{Y}^\top\boldsymbol{L}^{\boldsymbol{S}}\boldsymbol{Y}\right)$ using Alice's private $\boldsymbol{X}\boldsymbol{X}^\top$.
10. **Bob's side:** Perform the test-using a rejection region of $\Gamma(\mathbf{X},\mathbf{Y},\alpha,n) = \frac{n\bar{\Omega}_n^2(\mathbf{X},\mathbf{Y})}{\bar{S}(\mathbf{X},\mathbf{Y})} > \left(\Phi^{-1}(1-\alpha/2)\right)^2$.

Figure 3.1: Protocol for the proposed π-test mechanism for two-party independence testing

for each summand (in summation of directional variances) in the numerator and denominator of our test-statistic. But we need a single bound (with additive and multiplicative factors) for the test-statistic in its entirety, which is a ratio of sums of directional variances. The following results that we work out in the rest of the paper let us obtain that bound. We denote each summand in the sum of directional variances corresponding to the numerator and denominator by $\hat{\Omega}_i^2$ for $i \in [1, 2 \ldots m]$ and $\hat{S}_i$ for $i \in [1, 2 \ldots, k \leq n]$, respectively, where k denotes the rank of $\boldsymbol{G}$. That is, $\hat{\Omega}^2 = \sum_i \hat{\Omega}_i^2 = \sum_i \phi_{\boldsymbol{B}(\boldsymbol{X})}(\boldsymbol{y}_i)$ and $\hat{S} = \sum_i \hat{S}_i = \sum_i \phi_{\boldsymbol{X}}(\boldsymbol{g_i})$.

We use $\bar{\Omega}_i^2, \bar{S}_i$ to denote the privatized versions of $\hat{\Omega}_i^2$ and $\hat{S}_i$, respectively, that can be obtained upon privatization of the covariance matrix $\boldsymbol{X}\boldsymbol{X}^\top$ and incidence-covariance $\boldsymbol{B}(\boldsymbol{X})\boldsymbol{B}^\top(\boldsymbol{X})$. As a starting point,[61] provided these additive and multiplicative error bounds using $r = \frac{8\ln(2/\nu)}{\eta^2}$, $w = \frac{16\sqrt{r\ln(2/\delta)}}{\epsilon}\ln(16r/\delta)$ for the utility of a single directional variance query expressed using private covariance, which in our setting refers to $\bar{\Omega}_i^2$ and $\bar{S}_i$ for any $i \in [1, 2 \ldots, k \leq n]$.

$$\mathbb{P}\left((1-\eta)\hat{\Omega}_i^2 - \tau \leq \bar{\Omega}_i^2 \leq (1+\eta)\hat{\Omega}_i^2 + \tau\right) \geq 1 - \nu, \tag{3.4}$$

and similarly for S as

$$\mathbb{P}\left((1-\eta)\hat{S}_i - \tau \leq \bar{S}_i \leq (1+\eta)\hat{S}_i + \tau\right) \geq 1 - \nu. \tag{3.5}$$

But we need a bound on the ratio of summation of specific directional variances. From the inequality on the intersection of e events $E_1, ..., E_e$ given as $\mathbb{P}(\cap_{i=1}^e E_i) \geq \sum_{i=1}^e \mathbb{P}(E_i) - (e-1)$, for $\nu \geq 1/(m+n)$, it follows that

$$\mathbb{P}\left((1-\eta)\hat{\Omega}^2 - m\tau \leq \bar{\Omega}^2 \leq (1+\eta)\hat{\Omega}^2 + m\tau\right) \geq 1 - m\nu, \tag{3.6}$$

and

$$\mathbb{P}\left((1-\eta)\hat{S} - n\tau \leq \bar{S} \leq (1+\eta)\hat{S} + n\tau\right) \geq 1 - n\nu. \tag{3.7}$$

We now provide results for our lower and upper bounds on additive and multiplicative errors for our private estimator of the test-statistic.

Lower bound

Theorem 5. For $\hat{S} > \frac{n\tau}{1-n}$ and $n \gg m$, we have the following lower bound on $\frac{\bar{\Omega}^2}{\bar{S}}$.

$$\mathbb{P}\left(\frac{\bar{\Omega}^2}{\bar{S}} \geq \frac{1-\eta}{1+\eta}\left(\frac{\hat{\Omega}}{\hat{S}}\right) - \frac{(1-\eta)^2}{2(1+\eta)}\right) \geq 1 - (m+n)\nu.$$

Proof. From the naive re-arrangement of the individual bounds, we have the following with probability $\geq 1 - (m+n)\nu$

$$\frac{(1-\eta)\hat{\Omega}^2 - m\tau}{(1+\eta)\hat{S} + n\tau} \leq \frac{\bar{\Omega}^2}{\bar{S}} \leq \frac{(1+\eta)\hat{\Omega}^2 + m\tau}{(1-\eta)\hat{S} - n\tau}. \tag{3.8}$$

Rearranging the lower bound on $\frac{\bar{\Omega}^2}{\bar{S}}$ yields

$$\begin{aligned}\frac{\bar{\Omega}^2}{\bar{S}} \geqslant \frac{(1-\eta)\hat{\Omega}^2 - m\tau}{(1+\eta)\hat{S} + n\tau} = \frac{(1-\eta)\hat{\Omega}^2 - m\tau}{(1+\eta)\hat{S}\left(1 + \frac{n\tau}{(1+\eta)\hat{S}}\right)} = \frac{(1-\eta)\hat{\Omega}^2\left(1 + \frac{n\tau}{(1+\eta)\hat{S}}\right) - m\tau - \frac{(1-\eta)\hat{\Omega}^2 n\tau}{(1+\eta)\hat{S}}}{(1+\eta)\hat{S}\left(1 + \frac{n\tau}{(1+\eta)\hat{S}}\right)} \\ = \left(\frac{1-\eta}{1+\eta}\right)\frac{\hat{\Omega}^2}{\hat{S}} - \frac{m\tau(1+\eta)\hat{S} + n\tau(1-\eta)\hat{\Omega}^2}{(1+\eta)\hat{S}\left((1+\eta)\hat{S} + n\tau\right)}.\end{aligned} \tag{3.9}$$

For $\hat{S} > \frac{n\tau}{1-\eta}$, we have

$$\frac{m\tau(1+\eta)\hat{S}+n\tau(1-\eta)\hat{\Omega}^2}{(1+\eta)\hat{S}\left((1+\eta)\hat{S}+n\tau\right)} \overset{\text{(a)}}{\leq} \frac{m\tau(1+\eta)+n\tau(1-\eta)}{(1+\eta)\left((1+\eta)\hat{S}+n\tau\right)} \overset{\text{(b)}}{\leq} \frac{m(1+\eta)+n(1-\eta)}{2n\left(\frac{1+\eta}{1-\eta}\right)}, \quad (3.10)$$

where (a) follows from Lemma 3.5.1, i.e., $\hat{\Omega}^2 \leq \hat{S}$, and (b) follows by replacing $\hat{S}$ with $\frac{n\tau}{1-\eta}$ since $\hat{S} > \frac{n\tau}{1-\eta}$. For $n \gg m$, we can approximate equation 3.10 as follows

$$\frac{m\tau(1+\eta)\hat{S}+n\tau(1-\eta)\hat{\Omega}^2}{(1+\eta)\hat{S}\left((1+\eta)\hat{S}+n\tau\right)} \leq \frac{m(1+\eta)+n(1-\eta)}{2n\left(\frac{1+\eta}{1-\eta}\right)} \approx \frac{(1-\eta)^2}{2(1+\eta)}. \quad (3.11)$$

Plugging equation 3.11 into equation 3.9 yields

$$\frac{\bar{\Omega}^2}{\bar{S}} \geq \left(\frac{1-\eta}{1+\eta}\right)\frac{\hat{\Omega}^2}{\hat{S}} - \frac{(1-\eta)^2}{2(1+\eta)} \quad (3.12)$$

□

3.2.2 Upper bound

Theorem 6. For some $s > \frac{\tau}{1-\eta}$ and $n \gg m$, we have the following upper bound on $\frac{\bar{\Omega}^2}{\bar{S}}$.

$$\mathbb{P}\left(\frac{\bar{\Omega}^2}{\bar{S}} \leqslant \frac{1+\eta}{1-\eta}\left(\frac{\hat{\Omega}^2}{\hat{S}}\right) + \frac{\tau}{(1-\eta)s-\tau}\right) \geq 1-(m+n)\nu.$$

Proof. Rearranging the upper bound on $\frac{\bar{\Omega}^2}{\bar{S}}$, given in equation 3.8, yields

$$\begin{aligned}\frac{\bar{\Omega}^2}{\bar{S}} &\leq \frac{(1+\eta)\hat{\Omega}^2 + m\tau}{(1-\eta)\hat{S}\left(1 - \frac{n\tau}{(1-\eta)\hat{S}}\right)} = \frac{(1+\eta)\hat{\Omega}^2\left(1 - \frac{n\tau}{(1-\eta)\hat{S}}\right) + m\tau + \frac{(1+\eta)\hat{\Omega}^2 n\tau}{(1-\eta)\hat{S}}}{(1-\eta)\hat{S}\left(1 - \frac{n\tau}{(1-\eta)\hat{S}}\right)} \\ &= \left(\frac{1+\eta}{1-\eta}\right)\frac{\hat{\Omega}^2}{\hat{S}} + \frac{m\tau(1-\eta)\hat{S} + n\tau(1+\eta)\hat{\Omega}^2}{(1-\eta)\hat{S}\left((1-\eta)\hat{S} - n\tau\right)}. \end{aligned} \tag{3.13}$$

We have

$$\frac{m\tau(1-\eta)\hat{S} + n\tau(1+\eta)\hat{\Omega}^2}{(1-\eta)\hat{S}\left((1-\eta)\hat{S} - n\tau\right)} \overset{\text{(a)}}{\leq} \frac{m\tau(1-\eta) + n\tau(1+\eta)}{(1-\eta)\left((1-\eta)\hat{S} - n\tau\right)} \overset{\text{(b)}}{\leq} \frac{m\tau(1-\eta) + n\tau(1+\eta)}{(1-\eta)\left((1-\eta)ns - n\tau\right)}, \tag{3.14}$$

where (a) follows from Lemma 3.5.1, i.e., $\hat{\Omega}^2 \leq \hat{S}$, and (b) follows since $S \geq ns$, for some $s > \frac{\tau}{1-\eta}$. For $n \gg m$, equation 3.14 is approximated as

$$\frac{m\tau(1-\eta)\hat{S} + n\tau(1+\eta)\hat{\Omega}^2}{(1-\eta)\hat{S}\left((1-\eta)\hat{S} - n\tau\right)} \leq \frac{m\tau(1-\eta) + n\tau(1+\eta)}{(1-\eta)\left((1-\eta)ns - n\tau\right)} \approx \frac{\tau}{(1-\eta)s - \tau}. \tag{3.15}$$

Plugging equation 3.15 into equation 3.13 yields

$$\frac{\hat{\Omega}^2}{\hat{S}} \leq \left(\frac{1+\eta}{1-\eta}\right)\frac{\Omega^2}{S} + \frac{\tau}{(1-\eta)s - \tau}. \tag{3.16}$$

□

3.3 CONCLUSION

Current works on private independence testing focus on discrete data (contingency tables) and are based on stringent minimax testing under total variation distance, which might be overly conservative in many settings. Our work differs in this sense in terms of the distributed model assumed (the way the data is partitioned across users), its compatibility with continuous data, and its applicability to samples lying in different dimensions, unlike these prior methods. While information-theoretic formulations are fairly mature, inference with energy statistics, in general, brings a newer viewpoint to private testing problems. In addition, distance correlation happens to be a special case of a broader concept of energy statistics[497,431,499]. Thereby, solutions for its private estimation open up a door for investigating multi-party private solutions for downstream problems that depend on distance correlation, such as multi-party private independence testing, multi-party private feature screening, and multi-party private causal inference.

3.4 Limitations and future work

That said, universality results on power analysis of distance covariance-based tests are being formulated by various groups in recent preprints such as[214]. A further analysis of our test from this viewpoint is needed as part of future work. Although we used the privacy mechanism for covariance given in[61] as part of our π-test protocol, other options such as[60,22] could be used, and theoretical effects of their utility in privatizing the test-statistic is of open interest. We used[61] given its suitability to our theoretical study. Similarly, the question of coming up with mechanisms for private conditional independence testing using conditional distance covariance is still of open interest.

3.5 Appendix for this chapter

3.5.1 Error bound on ratio of estimators

Lemma 3.5.1. *Assuming that* $d_{\max}/d_{\min}^2 \leq \frac{(n-1)}{2}$, *where* $d_{\max} = \max_{i,j} \|\boldsymbol{x}_i - \boldsymbol{x}_j\|^2$ *and* $d_{\min} = \min_{i,j,i\neq j} \|\boldsymbol{x}_i - \boldsymbol{x}_j\|^2$, *we have* $\Omega^2 \leq S$.

Proof. We have

$$\begin{aligned}\Omega^2 &= \frac{1}{n}\sqrt{2\,\mathrm{Tr}\left(\boldsymbol{Y}^\top \boldsymbol{L}_{\boldsymbol{X}}\boldsymbol{Y}\right)} = \frac{1}{n}\sqrt{\sum_{i,j} W^X_{i,j}\left\|\boldsymbol{y}_i - \boldsymbol{y}_j\right\|^2} \\ &= \frac{1}{n}\sqrt{\sum_{i,j}[\boldsymbol{J}\boldsymbol{E}_{\boldsymbol{X}}\boldsymbol{J}]_{i,j}\left\|\boldsymbol{y}_i - \boldsymbol{y}_j\right\|^2}, \qquad (3.17a)\\ S &= \frac{1}{n^4}\sum_{i,j}\|\boldsymbol{x}_i - \boldsymbol{x}_j\|^2 \sum_{i,j}\left\|\boldsymbol{y}_i - \boldsymbol{y}_j\right\|^2. \qquad (3.17b)\end{aligned}$$

Assuming that there are a total of c classes in the dataset, we denote by $\mathcal{C}_l$ the set of data samples indices that belong to the label l, $l = 0, ..., c-1$. For $i \in \mathcal{C}_l$, we assume that $\boldsymbol{y}_i$ is a vector with only one non-zero entry at the l-th coordinate, where for simplicity, we set the value in the l-th coordinate to 1. Accordingly, for $l = 0, ..., c-1$, we have

$$\left\|\boldsymbol{y}_i - \boldsymbol{y}_j\right\|^2 = \begin{cases} 0, & i,j \in \mathcal{C}_l \\ 2, & \text{otherwise,} \end{cases} \qquad (3.18)$$

which results in

$$\begin{aligned}\Omega^2 &= \frac{1}{n}\sqrt{2\sum_{i,j,(i,j)\notin\mathcal{C}_l^2}[\boldsymbol{J}\boldsymbol{E}_{\boldsymbol{X}}\boldsymbol{J}]_{i,j}} \\ &\leq \frac{1}{n}\sqrt{2\sum_{i,j}[\boldsymbol{J}\boldsymbol{E}_{\boldsymbol{X}}\boldsymbol{J}]_{i,j}} \\ &\overset{(a)}{=} \frac{1}{n}\sqrt{\frac{2}{n}\sum_{i,j}\|\boldsymbol{x}_i - \boldsymbol{x}_j\|^2}, \qquad (3.19)\end{aligned}$$

where (a) follows from Lemma 3.5.2. For simplicity of the analysis, we assume that the number of data samples with each class is uniform, i.e., $|\mathcal{C}_l| = n/c$, in which case $\sum_{i,j}\left\|\boldsymbol{y}_i - \boldsymbol{y}_j\right\|^2 =$

$2n^2(1-1/c)$. Accordingly, we have

$$S = \frac{2(1-1/c)}{n^2} \sum_{i,j} \|\boldsymbol{x}_i - \boldsymbol{x}_j\|^2 \,, \tag{3.20}$$

which is minimized for $c = 2$ (assuming that $c \geq 2$), i.e.,

$$S \geq \frac{1}{n^2} \sum_{i,j} \|\boldsymbol{x}_i - \boldsymbol{x}_j\|^2 \,. \tag{3.21}$$

According to equation 3.19 and equation 3.21, in order to prove $\Omega^2 \leq S$, it suffices to show that

$$\sum_{i,j} \|\boldsymbol{x}_i - \boldsymbol{x}_j\|^2 \leq \frac{1}{2n}\Big(\sum_{i,j} \|\boldsymbol{x}_i - \boldsymbol{x}_j\|^2\Big)^2. \tag{3.22}$$

We have

$$\begin{aligned} \sum_{i,j} \|\boldsymbol{x}_i - \boldsymbol{x}_j\|^2 &\leq n(n-1)d_{\max}, \\ \frac{1}{2n}\Big(\sum_{i,j} \|\boldsymbol{x}_i - \boldsymbol{x}_j\|^2\Big)^2 &\geq \frac{n^2(n-1)^2}{2n} d_{\min}^2. \end{aligned} \tag{3.23}$$

Accordingly, having $d_{\max} \leq \frac{(n-1)}{2} d_{\min}^2$ guarantees equation 3.22.

□

3.5.2 Double-centering lemmas

Lemma 3.5.2. *For the centering matrix given by $\boldsymbol{J} = \boldsymbol{I} - \frac{1}{n}\boldsymbol{e}\boldsymbol{e}^\top$, we have the following property when applied to Euclidean distance matrices of data sample X denoted by E_X,*

$$\sum_{i,j} [\boldsymbol{J}\boldsymbol{E}_{\boldsymbol{X}}\boldsymbol{J}]_{i,j} = \frac{1}{n} \sum_{i,j} d_{i,j}^2(X)$$

Proof. We have

$$\boldsymbol{J}\boldsymbol{E}_{\boldsymbol{X}}\boldsymbol{J} = \left(\boldsymbol{I} - \frac{1}{n}\boldsymbol{e}\boldsymbol{e}^{\top}\right)\boldsymbol{E}_{\boldsymbol{X}}\left(\boldsymbol{I} - \frac{1}{n}\boldsymbol{e}\boldsymbol{e}^{\top}\right) \tag{3.24}$$

$$= \boldsymbol{E}_{\boldsymbol{X}} - \frac{1}{n}\boldsymbol{e}\boldsymbol{e}^{\top}\boldsymbol{E}_{\boldsymbol{X}} - \frac{1}{n}\boldsymbol{E}_{\boldsymbol{X}}\boldsymbol{e}\boldsymbol{e}^{\top} + \frac{1}{n^2}\boldsymbol{e}\boldsymbol{e}^{\top}\boldsymbol{E}_{\boldsymbol{X}}\boldsymbol{e}\boldsymbol{e}^{\top}, \tag{3.25}$$

according to which

$$(\boldsymbol{J}\boldsymbol{E}_{\boldsymbol{X}}\boldsymbol{J})_{ij} = d_{ij}^2(\boldsymbol{X}) - \frac{1}{n}\sum_{i'=1}^{n} d_{i'j}^2(\boldsymbol{X}) - \frac{1}{n}\sum_{j'=1}^{n} d_{ij'}^2(\boldsymbol{X}) + \frac{1}{n^2}\sum_{i'j'} d_{i'j'}^2(\boldsymbol{X}). \tag{3.26}$$

Summing equation 3.26 over all i, j yields

$$\sum_{i,j}[\boldsymbol{J}\boldsymbol{E}_{\boldsymbol{X}}\boldsymbol{J}]_{ij} = \sum_{i,j} d_{ij}^2(\boldsymbol{X}) - \frac{1}{n}\sum_{i,j}\sum_{i'=1}^{n} d_{i'j}^2(\boldsymbol{X}) \tag{3.27}$$

$$- \frac{1}{n}\sum_{i,j}\sum_{j'=1}^{n} d_{i'j'}^2(\boldsymbol{X}) + \frac{n(n-1)}{n^2}\sum_{i',j'} d_{i'j'}^2(\boldsymbol{X}) \tag{3.28}$$

$$= \sum_{ij} d_{ij}^2(\boldsymbol{X}) - \frac{n-1}{n}\sum_{ij} d_{ij}^2(\boldsymbol{X}) - \frac{n-1}{n}\sum_{i,j} d_{ij}^2(\boldsymbol{X}) + \frac{n-1}{n}\sum_{i,j} d_{ij}^2(\boldsymbol{X}) \tag{3.29}$$

$$= \frac{1}{n}\sum_{i,j} d_{ij}^2(\boldsymbol{X}). \tag{3.30}$$

□

Lemma 3.5.3. *The data matrix $\boldsymbol{X}$ and its Euclidean distance matrix $\boldsymbol{E}_{\boldsymbol{X}}$ can be connected using the centering matrix given by $\boldsymbol{J} = \boldsymbol{I} - \frac{1}{n}\boldsymbol{e}\boldsymbol{e}^{\top}$ as,*

$$\boldsymbol{J}\boldsymbol{X}^{\boldsymbol{T}}\boldsymbol{X}\boldsymbol{J} = -\frac{1}{2}\boldsymbol{J}\boldsymbol{E}_{\boldsymbol{X}}\boldsymbol{J}$$

Proof.

$$\left(\|\mathbf{x}_i - \mathbf{x}_j\|^2 = \mathbf{x}_i^\top \mathbf{x}_i + \mathbf{x}_j^\top \mathbf{x}_j - 2\mathbf{x}_i^\top \mathbf{x}_j\right) \tag{3.31}$$

$$\mathbf{x}_i^T \mathbf{x}_j = -\frac{1}{2}\left(\|\mathbf{x}_i - \mathbf{x}_j\|^2 - \|\mathbf{x}_i\|^2 - \|\mathbf{x}_j\|^2\right) \tag{3.32}$$

$$\mathbf{X}^T\mathbf{X} = \frac{-1}{2}\mathbf{E}_{\mathbf{X}} + \frac{1}{2}\delta.\mathbf{1}^\top + \frac{1}{2}\mathbf{1}\delta^\top \text{ where } \delta \text{ is a vector with } \delta_i = \|\mathbf{x}_i\|. \tag{3.33}$$

$$\mathbf{J}\mathbf{X}^\top\mathbf{X}\mathbf{J} = -\frac{1}{2}\mathbf{J}\mathbf{E}_{\mathbf{X}}\mathbf{J} + \frac{1}{2}J\delta.\mathbf{1}^\top\mathbf{J} + \frac{1}{2}\mathbf{J}\mathbf{1}\delta^\top\mathbf{J}$$

But since, $\mathbf{1}^\top\mathbf{J} = 0$ and $\mathbf{J}\mathbf{1} = \mathbf{0}$, we have

$$\mathbf{J}\mathbf{X}^\top\mathbf{X}\mathbf{J} = -\frac{1}{2}\mathbf{J}\mathbf{E}_{\mathbf{X}}\mathbf{J}$$

□

3.5.3 Proof of Lemma 3.5.1

We have

$$\begin{aligned}\Omega^2(\mathbf{X},\mathbf{Y}) &= \frac{1}{n}\sqrt{2\,\mathrm{Tr}\left(\mathbf{Y}^\top\mathbf{L}_{\mathbf{X}}\mathbf{Y}\right)} = \frac{2}{n^2}\sum_{ij}\mathbf{W}^{\mathbf{X}}_{ij}d^2_{ij}(\mathbf{Y}) \\ &= \frac{1}{n}\sqrt{\sum_{ij}[\mathbf{J}\mathbf{E}_{\mathbf{X}}\mathbf{J}]_{ij}d^2_{ij}(\mathbf{Y})}.\end{aligned}$$

3.5.4 Laplacian formulation of $\Omega^2(\mathbf{X},\mathbf{Y})$[547]

Lemma 3.5.4. *Distance covariance $\hat{\Omega}^2(\boldsymbol{X},\boldsymbol{Y})$ can be estimated using the Euclidean distance matrix $\boldsymbol{E_X}$ formed over the rows in $\boldsymbol{X}$, the double-centering matrix $\boldsymbol{J} = \boldsymbol{I} - n^{-1}\boldsymbol{e}\boldsymbol{e}^{\boldsymbol{T}}$, to form an adjacency matrix given by $\boldsymbol{W}(\boldsymbol{X}) = \boldsymbol{J}\boldsymbol{E_X}\boldsymbol{J}$ and a corresponding graph Laplacian*

$\boldsymbol{L}^{\boldsymbol{W}}(\boldsymbol{X}) = \boldsymbol{D}(\boldsymbol{W}(\boldsymbol{X})) - \boldsymbol{W}(\boldsymbol{X})$ *where* $\boldsymbol{D}(\boldsymbol{W}(\boldsymbol{X}))$ *is the degree matrix of* $\boldsymbol{W}(\boldsymbol{X})$ *to get*

$$\hat{\Omega}^2(\boldsymbol{X}, \boldsymbol{Y}) = \frac{1}{n}\sqrt{2\operatorname{Tr}\left(\boldsymbol{Y}^\top \boldsymbol{L}^{\boldsymbol{W}}(\boldsymbol{X})\boldsymbol{Y}\right)} = \frac{1}{n}\sqrt{2\operatorname{Tr}\left(\boldsymbol{X}^\top \boldsymbol{L}^{\boldsymbol{W}}(\boldsymbol{Y})\boldsymbol{X}\right)}$$

Proof. This result and the corresponding proof are from[547]. We replicate the same over here for quick reference. Given matrices $\widehat{\boldsymbol{E}}_{\boldsymbol{X}}, \widehat{\boldsymbol{E}}_{\boldsymbol{Y}}$, and column centered matrices $\widetilde{\boldsymbol{X}}, \tilde{\boldsymbol{Y}}$, from result of[511] we have that $\widehat{\boldsymbol{E}}_{\boldsymbol{X}} = -2\widetilde{\boldsymbol{X}}\widetilde{\boldsymbol{X}}^T$ and $\widehat{\boldsymbol{E}}_{\boldsymbol{Y}} = -2\tilde{\boldsymbol{Y}}\tilde{\boldsymbol{Y}}^T$. In the problem of multidimensional scaling (MDS) (Borg and Groenen, 2005), we know for a given adjacency matrix say $\boldsymbol{W}$ and a Laplacian matrix $\boldsymbol{L}$,

$$\operatorname{Tr}\left(\boldsymbol{X}^T \boldsymbol{L}\boldsymbol{X}\right) = \frac{1}{2}\sum_{i,j}[\boldsymbol{W}]_{ij}[\boldsymbol{E}_{\boldsymbol{X}}]_{i,j}$$

Now for the Laplacian $\boldsymbol{L} = \boldsymbol{L}_{\boldsymbol{X}}$ and adjacency matrix $\boldsymbol{W} = \widehat{\boldsymbol{E}}_{\boldsymbol{Y}}$ we can represent $\operatorname{Tr}\left(\boldsymbol{X}^T \boldsymbol{L}_{\boldsymbol{Y}}\boldsymbol{X}\right)$ in terms of $\widehat{\boldsymbol{E}}_{\boldsymbol{Y}}$ as follows,

$$\operatorname{Tr}\left(\boldsymbol{X}^T \boldsymbol{L}_{\boldsymbol{Y}}\boldsymbol{X}\right) = \frac{1}{2}\sum_{i,j=1}^{n}\left[\widehat{\boldsymbol{E}}_{\boldsymbol{Y}}\right]_{i,j}[\boldsymbol{E}_{\boldsymbol{X}}]_{i,j}$$

From the fact $[\boldsymbol{E}_{\boldsymbol{X}}]_{i,j} = (\langle\widetilde{\boldsymbol{x}}_i, \widetilde{\boldsymbol{x}}_i\rangle + \langle\widetilde{\boldsymbol{x}}_j, \widetilde{\boldsymbol{x}}_j\rangle - 2\langle\widetilde{\boldsymbol{x}}_i, \widetilde{\boldsymbol{x}}_j\rangle)$, and also $\widehat{\boldsymbol{E}}_{\boldsymbol{X}} = -2\widetilde{\boldsymbol{X}}\widetilde{\boldsymbol{X}}^T$ we get

$$\begin{aligned}
\operatorname{Tr}\left(\boldsymbol{X}^T \boldsymbol{L}_{\boldsymbol{Y}}\boldsymbol{X}\right) = &-\frac{1}{4}\sum_{i,j=1}^{n}\left[\widehat{\boldsymbol{E}}_{\boldsymbol{Y}}\right]_{i,j}\left(\left[\widehat{\boldsymbol{E}}_{\boldsymbol{X}}\right]_{i,i} + \left[\widehat{\boldsymbol{E}}_{\boldsymbol{X}}\right]_{j,j} - 2\left[\widehat{\boldsymbol{E}}_{\boldsymbol{X}}\right]_{i,j}\right)\\
=&\frac{1}{2}\sum_{i,j}\left[\widehat{\boldsymbol{E}}_{\boldsymbol{X}}\right]_{i,j}\left[\widehat{\boldsymbol{E}}_{\boldsymbol{Y}}\right]_{i,j} - \frac{1}{4}\sum_{j}^{n}\left[\widehat{\boldsymbol{E}}_{\boldsymbol{X}}\right]_{j,j}\sum_{i}^{n}\left[\widehat{\boldsymbol{E}}_{\boldsymbol{Y}}\right]_{i,j}\\
&-\frac{1}{4}\sum_{i}^{n}\left[\widehat{\boldsymbol{E}}_{X}\right]_{i,i}\sum_{j}^{n}\left[\widehat{\boldsymbol{E}}_{\boldsymbol{Y}}\right]_{i,j}
\end{aligned}$$

Since $\widehat{\boldsymbol{E}}_{\boldsymbol{X}}$ and $\widehat{\boldsymbol{E}}_{\boldsymbol{Y}}$ are double centered matrices $\sum_{i=1}^{n}\left[\widehat{\boldsymbol{E}}_{\boldsymbol{Y}}\right]_{i,j} = \sum_{j=1}^{n}\left[\widehat{\boldsymbol{E}}_{\boldsymbol{Y}}\right]_{i,j} = 0$ it follows that

$$\operatorname{Tr}\left(\boldsymbol{X}^T \boldsymbol{L_Y} \boldsymbol{X}\right) = \frac{1}{2} \sum_{i,j} \left[\widehat{\boldsymbol{E}}_{\boldsymbol{X}}\right]_{i,j} \left[\widehat{\boldsymbol{E}}_{\boldsymbol{Y}}\right]_{i,j}$$

It also follows that

$$\hat{\nu^2}(\boldsymbol{X}, \boldsymbol{Y}) = \frac{1}{n^2} \sum_{i,j=1}^{n} \left[\widehat{\boldsymbol{E}}_{\boldsymbol{Y}}\right]_{i,j} [\boldsymbol{E}_{\boldsymbol{X}}]_{i,j} = \frac{2}{n^2} \operatorname{Tr}\left(\boldsymbol{X}^T \boldsymbol{L_Y} \boldsymbol{X}\right)$$

Similarly, we can express the sample distance covariance using Laplacians $\boldsymbol{L_X}$ and $\boldsymbol{L_Y}$ as

$$\hat{\nu^2}(\boldsymbol{X}, \boldsymbol{Y}) = \left(\frac{2}{n^2}\right) \operatorname{Tr}\left(\boldsymbol{X}^T \boldsymbol{L_Y} \boldsymbol{X}\right) = \left(\frac{2}{n^2}\right) \operatorname{Tr}\left(\boldsymbol{Y}^T \boldsymbol{L_X} \boldsymbol{Y}\right)$$

The sample distance variances can be expressed as $\hat{\nu}^2(\boldsymbol{X}, \boldsymbol{X}) = \left(\frac{2}{n^2}\right) \operatorname{Tr}\left(\boldsymbol{X}^T \boldsymbol{L_X} \boldsymbol{X}\right)$ and $\hat{\nu}^2(\boldsymbol{Y}, \boldsymbol{Y}) = \left(\frac{2}{n^2}\right) \operatorname{Tr}\left(\boldsymbol{Y}^T \boldsymbol{L_Y} \boldsymbol{Y}\right)$ substituting back into expression of sample □

3.5.5 $S(\mathbf{X}, \mathbf{Y})$ AS A SUM OF DIRECTIONAL VARIANCES

Lemma 3.5.5. *We now express the denominator term of the test-statistic as a specific sum of directional variances as follows. The denominator in the test-statistic,*

$$\hat{S}(\boldsymbol{X}, \boldsymbol{Y}) = \frac{1}{n^2} \sum_{k,l=1}^{n} \|\boldsymbol{x}_k - \boldsymbol{x}_l\|^2 \frac{1}{n^2} \sum_{k,l=1}^{n} \|\boldsymbol{y}_k - \boldsymbol{y}_l\|^2 ,$$

can be expressed as a sum of directional variances as $\hat{S}(\boldsymbol{X}, \boldsymbol{Y}) = \sum_i \phi_{\boldsymbol{X}}(\boldsymbol{g_i})$ where $\boldsymbol{g}_i$ is the i'th column of a matrix $\boldsymbol{G}$ such that $\boldsymbol{L^S} = \boldsymbol{G}\boldsymbol{G}^T$ for a graph Laplacian matrix $\mathbf{L^S}$ given by $\mathbf{L^S} = n\boldsymbol{I} - \boldsymbol{e}\boldsymbol{e^T}$. We use the superscript $\boldsymbol{S}$ to distinguish from the Laplacian $\boldsymbol{L^W}$ used in the expression for $\hat{\Omega}^2$.

Proof. We first start simply by expressing the Euclidean distance between any two pairs of points using a matrix trace formulation as follows.

$$d_{ij}^2(\boldsymbol{X}) = \sum_{a=1}^{m} \boldsymbol{x}_a^\top (\boldsymbol{e}_i - \boldsymbol{e}_j)(\boldsymbol{e}_i - \boldsymbol{e}_j)^\top \boldsymbol{x}_a = \sum_{a=1}^{m} \boldsymbol{x}_a^\top \boldsymbol{A}_{ij} \boldsymbol{x}_a = \operatorname{tr} \boldsymbol{X}^\top \boldsymbol{A}_{ij} \boldsymbol{X}$$

where $\boldsymbol{A}_{ij} = (e_i - e_j)(e_i - e_j)^\top$. Similarly, $\sum_{i<j} d_{ij}(\boldsymbol{X}) = \operatorname{tr} \boldsymbol{X}^\top \left(\sum_{i<j} d_{ij}^{-1}(\boldsymbol{X})\boldsymbol{A}_{ij}\right) \boldsymbol{X}$. We now extend this to express the sum of all pairs of Euclidean distance matrices as follows

$$\eta(\boldsymbol{X}) = \sum_{i<j} w_{ij} d_{ij}^2(\boldsymbol{X}) = \operatorname{tr} \boldsymbol{X}^\top \left(\sum_{i<j} w_{ij} \boldsymbol{A}_{ij}\right) \boldsymbol{X} = \operatorname{tr} \boldsymbol{X}^\top \boldsymbol{L^S} \boldsymbol{X}$$

where $\mathbf{L^S} = \left(\sum_{i<j} w_{ij}\boldsymbol{A}_{ij}\right)$. As we would like to express the sum of all pairs of distances, we consider the case where $\boldsymbol{W}_{ij} = 1, \forall i, j \in [n]$. Note that this matrix has the exact structure of a graph Laplacian. This results in a graph Laplacian $\boldsymbol{L^S}$ where

$$L^S = \begin{bmatrix} n-1 & -1 & -1 \\ -1 & \ddots & \vdots \\ -1 & \cdots & n-1 \end{bmatrix} = n\boldsymbol{I} - \boldsymbol{e}\boldsymbol{e}^{\boldsymbol{T}}$$

Therefore,

$$\boldsymbol{S^\alpha(X, Y)} = \eta(\boldsymbol{X})\eta(\boldsymbol{Y}) = \frac{4}{n^4} Tr\left(\boldsymbol{X}^\top \boldsymbol{L^S X}\right) \cdot \operatorname{Tr}\left(\boldsymbol{Y}^T \boldsymbol{L^S Y}\right)$$

Note that we could also express $Tr\left(\boldsymbol{X}^\top \boldsymbol{L^S X}\right)$ as

$$Tr\left(\boldsymbol{X}^\top \boldsymbol{L^S X}\right) = Tr\left(\boldsymbol{X}^\top \boldsymbol{G}\boldsymbol{G^T}\boldsymbol{X}\right) = Tr\left(\boldsymbol{G^T}\boldsymbol{X}\boldsymbol{X}^\top\boldsymbol{G}\right) = \sum_{i}^{k \le n} \left(\boldsymbol{g_i}^{\boldsymbol{T}} \boldsymbol{X}\boldsymbol{X}^{\boldsymbol{T}} \boldsymbol{g_i}\right) = \sum_i \phi_{\boldsymbol{X}}(\boldsymbol{g_i}),$$

where k is the rank of the matrix $\boldsymbol{L^S}$. □

"Only yesterday the practical things of today were decried as impractical, and the theories which will be practical tomorrow will always be branded as valueless games by the practical man of today."

William Feller

4

Moving to Manifolds: Private Estimation of Fréchet Mean

4.1 Introduction

Privacy-preserving computing is an active area of research that is necessitated by ethics, regulations, requirements for protection of trade secrets, or possible lack of trust amongst distributed data siloes. Privacy preservation is desired across several topologies of data sharing, be it from client devices to powerful centralized entities or in a peer-to-peer fashion. Mistrust in data sharing carries over not only in the sharing of raw data but also in the sharing of results obtained from intermediate or complete computations. The need for stringent privacy protections is often fueled by many privacy leakages and attacks that continue to happen under various settings operating without the right level of privacy-protecting mechanisms.

In this context, differential privacy (DP)[158,154,160,153] has emerged as one of the leading mathematical definitions to ensure the preservation of privacy up to a chosen level. Privacy-preserving *mechanisms* that satisfy the definition of differential privacy were subsequently developed to privatize a wide range of statistical and machine learning computations. The earliest queries for which mechanisms have been proposed were for the privatization of sample means in statistics computed for data lying on linear spaces. When data belong to nonlinear manifolds, the Fréchet mean query[179] is the foundational building block of geometric statistics that needs to be privatized. Our work proposes a new, simpler, and faster mechanism for private Fréchet means on the manifold of symmetric positive definite (SPD) matrices endowed with log-Euclidean metric. This chapter is based on our work in[526].

4.1.1 Motivation

Fréchet mean: a building block in geometric statistics While traditional statistics studies data that lies on *linear spaces*, geometric statistics studies data that lies on *nonlinear spaces* such as Riemannian manifolds, affine connection spaces, or stratified spaces[406,355]. Such analysis is fruitful as data might have inherent constraints that are well captured by the geometry of a nonlinear space[356,381]. For instance, symmetric matrices constrained to have strictly positive eigenvalues are conveniently modeled as elements of the manifold of symmetric positive definite (SPD) matrices. Several extensions of traditional statistical analysis tools have thus been developed for the manifold setting: regression has been generalized to geodesic regression[177,507], principal component analysis (PCA) to principal geodesic analysis or geodesic PCA[176,482,236], and mean shift to Riemannian mean shift clustering[492,87]. In each of these algorithms, the computation of the *sample Fréchet mean* generalizes the computation of the *sample mean* and thus represents the most fundamental building block. The privatization of the Fréchet mean is, therefore, the key element required to privatize geometric statistical queries. Privacy-preserving geometric statistics is also crucial, as one of its main application areas is medical imaging and computational anatomy[406,355] for which privacy requirements are often desirable.

Importance of the SPD manifold with log-Euclidean metric Symmetric positive definite (SPD) matrices model a wide range of data, from medical images with Diffusion Tensor Imaging (DTI)[Basser et al.,405], to physiological signals with electroencephalography (EEG) signals from brain-computer interfaces (BCI)[585,595,111], to 3D shapes[501] to name a few. Given their central roles for medical data where privacy is of the utmost importance[332,315], private statistical computations on the SPD manifold are a worthy endeavour. The SPD manifold can be equipped with different *Riemannian metrics* that provide elementary operations such as distance computations. The log-Euclidean metric, originally proposed in[33], has numerous advantages over another popular Riemannian metric called the affine invariant metric[405]: (a) it is computationally faster, (b) it gives similar or better performances on several processing and learning tasks, (c) and quite importantly, it provides a *closed form* expression for the Fréchet mean - which otherwise requires solving an optimization problem.

Need for better and faster privacy mechanisms Despite its importance for the processing of a number of (medical) data, geometric statistics currently stand understudied from the lens of differential privacy. The very recent work by[425] provides the first differentially private mechanism for the Fréchet mean. However, its utility - a measure of the mechanism's deviation from non-privatized computations - makes it impracticable on the manifold of SPD matrices as soon as we consider matrices of moderate size, e.g. 20×20 matrices. Consequently, there is a need for better and faster privacy mechanisms on manifolds, starting with the SPD manifold.

4.1.2 Related Work and Contributions

Reimherr et al.[425] were first to consider differential privacy in manifold setting and developed *Riemannian Laplace mechanism* by extending the standard Laplace mechanism[160] for linear spaces to complete Riemannian manifolds. It is based on a Laplace distribution that was originally proposed for SPD matrices[211] based on the distance of the affine invariant metric[405], which they generalize to any manifold $\mathcal{M}$ equipped with a distance ρ:

$$p(x) \propto \exp\left(-\frac{\rho(x,m)}{\sigma}\right), \quad \forall x \in \mathcal{M} \tag{4.1}$$

where $m \in \mathcal{M}$, $\sigma \in \mathbb{R}_{>0}$(positive reals) are parameters of the probability density p. Reimherr et al.[425] show that the mechanism obtained achieves *pure* differential privacy and provides an upper bound for the expectation of its utility (a measure of the deviation from non-privatized computations) for the Fréchet mean query. Their method is applicable to various Riemannian manifolds that satisfy some regularity conditions.

Approximate differential privacy relaxes pure differential privacy (see Section 4.2) but provides significantly better utility for higher dimensions and is heavily used in real-world applications[2]. In the Euclidean case, the Gaussian mechanism, where noise is added from standard Gaussian, satisfies approximate differential privacy. To this end, we make use of log Gaussian distribution[454], an intrinsic distribution on SPD matrices, for deriving approximate differentially private mechanisms. This relaxation helps us obtain better utility compared to the Riemannian Laplace mechanism in terms of dimension, similar to the standard Euclidean case. We summarize our contributions as follows.

Mechanism $\mathbf{A}$	DP	$\mathbb{E}[\rho^2(f(\mathcal{D}), \mathbf{A}(\mathcal{D}))]$	Theoretical Results
Riemannian Laplace[425]	Pure DP	$\mathcal{O}(k^4)$	Expectation of $\rho^2(f(\mathcal{D}), \mathbf{A}(\mathcal{D}))$
tangent Gaussian (Ours)	Approx. DP	$\mathcal{O}(\ln(1/\delta)k^2)$	Exact Distribution of $\rho^2(f(\mathcal{D}), \mathbf{A}(\mathcal{D}))$

Table 4.1: Differences between existing[425] and proposed mechanisms for private Fréchet mean queries on the manifold of $k \times k$ SPD matrices endowed with the log-Euclidean metric. The notation $\rho^2(f(\mathcal{D}), \mathbf{A}(\mathcal{D}))$ represents the utility with $\mathcal{D}$ the dataset, $\mathbf{A}$ the mechanism under consideration, ρ the log-Euclidean distance, f the Fréchet mean and δ quantifies approximate differential privacy.

1. We propose a new and simple mechanism - called the *tangent Gaussian Mechanism* - that privatizes any statistical summary on the manifold of Symmetric Positive Definite (SPD) matrices endowed with the log-Euclidean metric. We prove that it achieves approximate differential privacy (Th. 8).

2. When the statistical summary is the Fréchet mean, we show that our mechanism obtains significant improvement in terms of utility over recent works - which we demonstrate theoretically and practically for data in higher dimensions. Further, our mechanism is computationally efficient and easily implementable.

3. We present the effectiveness of our mechanism on synthetic and real-world (medical) imaging data, the latter being represented via their covariance descriptors. To this aim, we also prove a theoretical bound on the radius of the log-Euclidean geodesic ball with the covariance descriptor pipeline[517] - required for the applicability of our mechanism (Th. 11).

Table 4.1 highlights the technical differences between[425] and our work.

4.2 Preliminaries and Notations

Elements of Riemannian Geometry Let $\mathcal{M}$ be a d-dimensional smooth connected manifold and $T_p\mathcal{M}$ be its tangent space at point $p \in \mathcal{M}$. A *Riemannian metric g* on M is a collection of inner products $g_p : T_p\mathcal{M} \times T_p\mathcal{M} \to \mathbb{R}$ that vary smoothly with p. A manifold $\mathcal{M}$ equipped with a Riemannian metric g is called a Riemannian manifold. Importantly, the metric g gives a distance ρ on $\mathcal{M}$. Let $\gamma : [0, 1] \to \mathcal{M}$ be a smooth parametrized curve on $\mathcal{M}$ with velocity vector at t denoted as $\dot{\gamma}_t \in T_{\gamma(t)}\mathcal{M}$. The length of γ is defined as $L_\gamma = \int_0^1 \sqrt{g_{\gamma(t)}(\dot{\gamma}_t, \dot{\gamma}_t)}dt$ and the distance ρ between any two points $p, q \in \mathcal{M}$ is: $\rho(p, q) = \inf_{\gamma:\gamma(0)=p,\gamma(1)=q} L_\gamma$.

If, in addition, $\mathcal{M}$ is complete for ρ, then any two points $p, q \in \mathcal{M}$ can be joined by a length-minimizing curve, called a geodesic. We refer the reader to[141,306,222] for a detailed exposition.

Elements of Differential Privacy (DP) Let $\mathcal{X}$ be an input data space and $\mathcal{M}$ the manifold under consideration. Let $f : \mathcal{X}^n \to \mathcal{M}$ be a manifold-valued statistical summary that requires privatization with respect to some sensitive dataset $\mathcal{D}$ of size n, *i.e.* $\mathcal{D} \in \mathcal{X}^n$. Two datasets $\mathcal{D}, \mathcal{D}' \in \mathcal{X}^n$ are said to be adjacent if they differ by at most one data point. We denote adjacency as $\mathcal{D} \sim \mathcal{D}'$. The *sensitivity* of the summary f with respect to the distance ρ on $\mathcal{M}$ is defined as:

$$\Delta_\rho = \sup_{\mathcal{D}\sim\mathcal{D}'} \rho(f(\mathcal{D}), f(\mathcal{D}')), \tag{4.2}$$

which is the maximum amount of deviation that can occur in the output of f for adjacent datasets.

A *mechanism* $\mathbf{A} : \mathcal{X}^n \to \mathcal{M}$ is a randomized algorithm that takes a dataset $\mathcal{D}$ as input and outputs a privatized version of the summary f on $\mathcal{D}$. The mechanism $\mathbf{A}$ satisfies $(\epsilon, 0)$ differential privacy (also *pure differential privacy*) if, for all adjacent datasets $\mathcal{D} \sim \mathcal{D}'$ and for all measurable sets S of $\mathcal{M}$ the following holds:

$$\mathbb{P}[\mathbf{A}(\mathcal{D}) \in S] \leq \exp(\epsilon)\, \mathbb{P}[\mathbf{A}(\mathcal{D}') \in S] \tag{4.3}$$

The intuition is that the change of a single element of the data space $\mathcal{X}$ does not significantly alter the output distribution of the mechanism. As a relaxation, the mechanism $\mathbf{A}$ satisfies (ϵ, δ)-differential privacy (also *approximate differential privacy*) if, for all adjacent datasets $\mathcal{D} \sim \mathcal{D}'$ and for all measurable sets S of $\mathcal{M}$:

$$\mathbb{P}[\mathbf{A}(\mathcal{D}) \in S] \leq \exp{(\epsilon)}\, \mathbb{P}[\mathbf{A}(\mathcal{D}') \in S] + \delta.$$

Intuitively, δ can be thought of as the probability of privacy failure when Eq. equation 4.3 is not guaranteed.

Let $p_{\mathbf{A}(\mathcal{D})}$ be the density of the random variable $Y = \mathbf{A}(\mathcal{D})$. Given adjacent datasets $\mathcal{D} \sim \mathcal{D}'$, the *privacy loss function* of $\mathbf{A}$ is defined as

$$\ell_{\mathbf{A},\mathcal{D},\mathcal{D}'}(y) = \ln\left(\frac{p_{\mathbf{A}(\mathcal{D})}(y)}{p_{\mathbf{A}(\mathcal{D}')}(y)}\right) \quad \forall y \in \mathcal{M}, \tag{4.4}$$

and the *privacy loss random variable* is $L_{\mathbf{A},\mathcal{D},\mathcal{D}'} = \ell_{\mathbf{A},\mathcal{D},\mathcal{D}'}(Y)$[42]. Importantly for our derivations, both sufficient and sufficient & necessary conditions for the mechanism $\mathbf{A}$ to be (ϵ, δ)-differentially private (DP) can be formulated in terms of $L_{\mathbf{A},\mathcal{D},\mathcal{D}'}$. The sufficient condition writes : $\forall \mathcal{D} \sim \mathcal{D}' : \mathbb{P}[L_{\mathbf{A},\mathcal{D},\mathcal{D}'} \geq \epsilon] \leq \delta \implies \mathbf{A}$ is(ϵ, δ)-DP. The sufficient & necessary condition is: $\forall \mathcal{D} \sim \mathcal{D}' : \mathbb{P}[L_{\mathbf{A},\mathcal{D},\mathcal{D}'} \geq \epsilon] - \exp{(\epsilon)}\mathbb{P}[L_{\mathbf{A},\mathcal{D},\mathcal{D}'} \leq -\epsilon] \leq \delta \iff \mathbf{A}$ is(ϵ, δ)-DP.

Fréchet Mean When the data space $\mathcal{X}$ is equal to the manifold $\mathcal{M}$, we will be interested in mechanisms that can privatize a specific statistical summary f called the Fréchet mean. The sample Fréchet mean $\overline{X}$[179] of the dataset $\mathcal{D} = \{X_1, \ldots X_n\}$ on the manifold $\mathcal{M}$ is defined as

$$\overline{X} \triangleq \left\{ p | p \in \arg\min_{q \in \mathcal{M}} \sum_{i=1}^{n} \rho^2(q, X_i) \right\},$$

i.e. we have in this case $\overline{X} = f(\mathcal{D})$ for $\mathcal{D} \in \mathcal{M}^n$. Intuitively, the Fréchet mean uses a property of the mean on linear spaces — namely, the fact that the mean minimizes the sum of squared distances to the data points - as a definition of mean on manifolds. Crucially, the Fréchet mean depends on the distance ρ and, therefore, on the Riemannian metric defined on $\mathcal{M}$. We also note that the Fréchet mean might not always exist, and if it exists, it might not be unique – see supplementary materials. In practice, computing $\overline{X}$ generally requires optimization algorithms such as gradient descent on manifolds[73].

4.3 Geometry of the SPD Manifold with Log Euclidean Metric

Manifold and vector space structures We now restrict $\mathcal{M}$ to be the manifold of symmetric positive definite (SPD) matrices:

$$\text{SPD}(k) = \left\{ X \in \mathbb{R}^{k \times k} | X^T = X \text{ and } \forall u \in \mathbb{R}^k \setminus \{0\}, u^T X u > 0 \right\}, \tag{4.5}$$

which has dimension $d = \frac{k(k+1)}{2}$. The tangent space of the manifold SPD(k) at any point $X \in$ SPD(k) is the vector space of symmetric matrices SYM(k). The mathematical construct $(\text{SPD}(k), +, .)$ is not a vector space under element-wise addition and element-wise scalar multiplication. This can be seen from the observation that $a \in \mathbb{R}_{\leq 0}, X \in \text{SPD}(k) \implies anX \notin$ SPD(k). Instead, SPD(k) is an open cone of $\mathbb{R}^{k \times k}$ and, as such, naturally possesses a smooth manifold structure which can further be equipped with different Riemannian metrics[506]. However, Arsigny *et al.*[34] showed in a surprising result that SPD(k) can be given a vector space structure $(\text{SPD}(k), \oplus, \odot)$ via the operations $\oplus, \odot$ defined in Table 4.2, where Expm, Logm denote the matrix exponential and matrix logarithm. This fact is central to the proofs provided in the present paper.

Riemannian structure Arsigny *et al.* further define a Riemannian metric on SPD(k), called

OPERATION	NOTATION	EXPRESSION
Addition	$X_1 \oplus X_2$	$\text{Expm}\left[\text{Logm}\, X_1 + \text{Logm}\, X_2\right]$
Subtraction	$X_1 \ominus X_2$	$\text{Expm}\left[\text{Logm}\, X_1 - \text{Logm}\, X_2\right]$
Scalar Multiplication	$a \odot X$	$\text{Expm}\left[a.\, \text{Logm}\, X\right]$

Table 4.2: Operations turning the manifold SPD(k) into a vector space. Expm and Logm denote the matrix exponential and logarithms, respectively. X_1, X_2 belong to SPD(k) while $a \in \mathbb{R}$ is a scalar.

the *log-Euclidean metric*, which induces the following distance:

$$\rho_{\text{LE}}(X_1, X_2) = \|\text{Logm}\, X_1 - \text{Logm}\, X_2\|_F, \quad \forall X_1, X_2 \in \text{SPD}(k), \tag{4.6}$$

where $\|.\|_F$ denotes the Frobenius norm on matrices. Importantly, the log-Euclidean metric[33] gives a unique and simple closed-form expression for the Fréchet mean in terms of matrix logarithm and matrix exponential

$$\overline{X}_{\text{LE}} = \text{Expm}\left[\frac{1}{n}\sum_{i=1}^{n} \text{Logm}\, X_i\right], \tag{4.7}$$

for the dataset $X_1, ..., X_n \in \text{SPD}(k)$.

Maps between spaces Lastly, we present maps that will help us define the differential privacy mechanism proposed in the next section. Consider the map $\text{vecd} : \text{SYM}(k) \rightarrow \mathbb{R}^{\frac{k(k+1)}{2}}$ defined as $\text{vecd}(X) = \left[\text{diag}(X)^T, \sqrt{2}\,\text{upperdiag}(X)^T\right]^T$, where $\text{diag} : \text{SYM}(k) \rightarrow \mathbb{R}^k$ and $\text{upperdiag} : \text{SYM}(k) \rightarrow \mathbb{R}^{\frac{k(k-1)}{2}}$ build vectors from the diagonal, and from the strictly upper diagonal entries, of the matrix X. The map vecd is invertible, and we denote by invvecd its inverse. Specifically, the spaces $\text{SPD}(k), \text{SYM}(k)$ and $\mathbb{R}^{\frac{k(k+1)}{2}}$ are now related as follows:

$$\begin{tikzcd} \text{SPD}(k) \arrow[r, shift left, "\text{Logm}"] & \text{SYM}(k) \arrow[l, shift left, "\text{Expm}"] \arrow[r, shift left, "\text{vecd}"] & \mathbb{R}^{\frac{k(k+1)}{2}} \arrow[l, shift left, "\text{invvecd}"] \end{tikzcd}.$$

4.4 Tangent Gaussian Mechanism on SPD manifolds

We can now introduce our differential privacy mechanism for statistical summaries on the $\mathrm{SPD}(k)$ manifold. Let $f : \mathcal{X}^n \to \mathrm{SPD}(k)$ be any $\mathrm{SPD}(k)$-valued summary that needs to be privatized. The proposed mechanism is based on the log Gaussian distribution on the SPD manifold[454], which is defined as follows. Consider a mean $M \in \mathrm{SPD}(k)$ and a tangent covariance $\Sigma \in \mathrm{SPD}\left(\frac{k(k+1)}{2}\right)$. We can (i) first map the mean M to the tangent space $\mathrm{SYM}(k)$ of $\mathrm{SPD}(k)$ at the identity using the matrix Logarithm Logm, then (ii) to $\mathbb{R}^{\frac{k(k+1)}{2}}$ using the map vecd introduced in the previous section, and (iii) consider whether the result follows a traditional Gaussian distribution.

Definition 4.4.1 (Log Gaussian Distribution on $\mathrm{SPD}(k)$[454])**.** *Given a mean $M \in \mathrm{SPD}(k)$, and a tangent covariance $\Sigma \in \mathrm{SPD}\left(\frac{k(k+1)}{2}\right)$, we say that $X \sim \mathcal{LN}(M, \Sigma)$ follows a log Gaussian distribution on $\mathrm{SPD}(k)$ if $\mathrm{vecd}[\mathrm{Logm}\, X] \sim \mathcal{N}(\mathrm{vecd}[\mathrm{Logm}\, M], \Sigma)$ follows a (regular) Gaussian distribution with mean $\mathrm{vecd}\,\mathrm{Logm}\, M$ and covariance matrix Σ on $\mathbb{R}^{\frac{k(k+1)}{2}}$.*

The density $p(X|M, \Sigma)$ is then given by

$$\frac{J(X)}{(2\pi)^{\frac{d}{2}}(\det \Sigma)^{\frac{1}{2}}} \exp\left(-\frac{1}{2}\,\mathrm{vecd}(\mathrm{Logm}\, X - \mathrm{Logm}\, M)^T \Sigma^{-1}\, \mathrm{vecd}(\mathrm{Logm}\, X - \mathrm{Logm}\, M)\right)$$

where $d = \frac{k(k+1)}{2}$, $J(X) = \frac{1}{\det X}\prod_{i<j} h(\lambda_i, \lambda_j)$, and $h(\lambda_i, \lambda_j) = \begin{cases} (\log \lambda_i - \log \lambda_j) & \lambda_i > \lambda_j \\ \frac{1}{\lambda_i} & \lambda_i = \lambda_j \end{cases}$, with λ_i, λ_j eigenvalues of the matrix X.

The definition of log Gaussian distribution on the $\mathrm{SPD}(k)$ manifold allows us to define our proposed tangent Gaussian mechanism.

```
Inputs : Dataset $\mathcal{D}$ of $k \times k$ SPD matrices of size $n$,
         sigma-type $\in$ {'classical', 'analytic' }, $\Delta_{\text{LE}}$ the log-Euclidean
         sensitivity of $f$, $\epsilon > 0$, $\delta \in (0, 1)$ and additionally $\epsilon < 1$ if sigma-type
         is 'classical', the noise calibration subroutines CLASSIC,
         ANALYTIC which take $\Delta_{LE}, \epsilon, \delta$ and provide $\sigma$.
Output : Private $f(\mathcal{D})$
if sigma-type is 'classical' then $\sigma = \text{CLASSIC}(\Delta_{\text{LE}}, \epsilon, \delta)$; else
 $\sigma = \text{ANALYTIC}(\Delta_{\text{LE}}, \epsilon, \delta)$;
Compute non private output : $f_{\text{np}} := f(\mathcal{D})$
Compute mean of Gaussian distribution: $M := \text{vecd}[\text{Logm}\, f_{\text{np}}]$, $M \in \mathbb{R}^{\frac{k(k+1)}{2}}$
Sample from the Gaussian distribution in $\mathbb{R}^{\frac{k(k+1)}{2}}$: $N \sim \mathcal{N}(M, \sigma^2 I)$
Map sample to the SPD manifold: $f_p := \text{Expm}[\text{invvecd}\, N]$
Return private $f_p$
```

Algorithm 1: tangent Gaussian Mechanism for $f : \mathcal{X}^n \to \text{SPD}(k)$

Definition 4.4.2 (tangent Gaussian Mechanism)**.** *Consider any statistical summary* $f : \mathcal{X}^n \to \text{SPD}(k)$ *on the manifold* $\text{SPD}(k)$ *equipped with log-Euclidean metric. Given* $\sigma^2 > 0$*, we define the tangent Gaussian mechanism* $\mathbf{A}_{\text{TG}} : \mathcal{X}^n \to \text{SPD}(k)$*, as*

$$\mathbf{A}_{\text{TG}}(\mathcal{D}) = X, \textit{where } X \sim \mathcal{LN}(f(\mathcal{D}), \sigma^2 I).$$

We now state our main theorem, which shows that the privacy loss of the tangent Gaussian mechanism is normally distributed with mean and variance parameterized by the log-Euclidean distance. Proof is given in Appendix 4.8.2

Theorem 7 (Distribution of Privacy Loss for the tangent Gaussian Mechanism). Let $\mathbf{A}_{\text{TG}}$ be a tangent Gaussian mechanism with variance σ^2. Its privacy loss is normally distributed as

$$L_{\mathbf{A}_{\text{TG}}, \mathcal{D}, \mathcal{D}'} \sim \mathcal{N}\left(\frac{\rho^2_{\text{LE}}(f(\mathcal{D}), f(\mathcal{D}'))}{2\sigma^2}, \frac{\rho^2_{\text{LE}}(f(\mathcal{D}), f(\mathcal{D}'))}{\sigma^2}\right).$$

This distribution is analogous to the distribution of the privacy loss for the Euclidean Gaussian mechanism but with the log-Euclidean sensitivity instead of the Euclidean sensitivity[160,42]. Consequently, our theoretical analysis of the tangent Gaussian mechanism - deriving privacy guarantees from the distribution of the privacy loss above - closely follows the steps of the analysis for the Euclidean Gaussian case. Specifically, we can proceed in two ways: with either a (1) classical approach where sufficient conditions are used to show the mechanism is (ϵ, δ)-DP as in[160], or with an (2) analytic approach where the utility is better by using sufficient and necessary conditions[42].

Theorem 8 (Privacy Guarantee of tangent Gaussian Mechanism). Consider $f : \mathcal{X}^n \to \text{SPD}(k)$ with log-Euclidean sensitivity Δ_{LE}.

1. (Classical) Given $\epsilon, \delta \in (0, 1)$, choosing $\sigma = \Delta_{\text{LE}}\sqrt{2\ln(1.25/\delta)}/\epsilon$, makes the tangent Gaussian mechanism (ϵ, δ)-differentially private.

2. (Analytic) Given $\epsilon \geq 0, \delta \in (0, 1)$ and Φ the cumulative distribution of the standard Gaussian, choosing any σ that satisfies $\Phi(\frac{\Delta_{\text{LE}}}{2\sigma} - \frac{\epsilon\sigma}{\Delta_{\text{LE}}}) - \exp(\epsilon)\Phi(\frac{\Delta_{\text{LE}}}{2\sigma} - \frac{\epsilon\sigma}{\Delta_{\text{LE}}}) \leq \delta$ makes the tangent Gaussian mechanism (ϵ, δ)-differentially private.

Proofs are given in Appendix 4.8.3. Algorithm 1 shows the implementation of the mechanism.

4.5 Privatizing the Fréchet mean

In the previous section, f is any function that outputs summary statistics on $\text{SPD}(k)$. In this section, we seek to privatize the Fréchet mean f of the log-Euclidean metric. We first compute its sensitivity and then provide its utility. In what follows, $\mathcal{B}_r(M) = \{X | \rho_{\text{LE}}(M, X) < r\}$ denotes an open geodesic ball of radius $0 < r < \infty$ centered at $M \in \text{SPD}(k)$.

Theorem 9 (Sensitivity of Log-Euclidean Fréchet Mean). Given data in $\mathcal{B}_r(M)$ for some $0 < r < \infty$ and $M \in \text{SPD}(k)$, the sensitivity of the log-Euclidean Fréchet mean verifies: $\Delta_{\text{LE}} \leq \frac{2r}{n}$.

Note that the above theorem can also obtained from[425] Theorem 2 by setting $\kappa = 0$. The utility of the tangent Gaussian mechanism for a Fréchet mean query is then given below.

Theorem 10 (Utility). Let $\mathbf{A}_{\text{TG}}$ be the (classical) tangent Gaussian mechanism, $\mathcal{B}_r(M)$ a geodesic ball of radius $0 < r < \infty$ and center $M \in \mathcal{M}$ containing the dataset $\mathcal{D}$ and f the Fréchet mean. The utility of the mechanism $\mathbf{A}_{\text{TG}}$ is given by:

$$\rho^2_{\text{LE}}(f(\mathcal{D}), \mathbf{A}_{\text{TG}}(\mathcal{D}))) \sim \sigma^2 \chi^2_d,$$
$$\mathbb{E}[\rho^2_{\text{LE}}(f(\mathcal{D}), \mathbf{A}_{\text{TG}}(\mathcal{D}))] = \frac{4r^2 \ln(1.25/\delta) d}{n^2 \epsilon^2} \qquad \text{with } d = dim(\text{SPD}(k)) = \frac{k(k+1)}{2},$$

where χ^2_d represents the chi squared distribution with d degree of freedoms.

Proofs of Th. 9 and Th. 10 are given in Appendix 4.8.4. We compare these results with those of the Riemannian Laplace mechanism[425], denoted $\mathbf{A}_{\text{RL}}$.

Utility: We compare the utility in terms of size k of spd matrices $k \times k$ because dependancy on other factors n, ϵ are same. Utility of the Riemannian Laplace mechanism has an expectation given by $\mathbb{E}[\rho^2_{\text{LE}}(f(\mathcal{D}), \mathbf{A}_{\text{RL}}(\mathcal{D}))] = \mathcal{O}(k^4)$. By contrast, our tangent Gaussian mechanism provides $\mathbb{E}[\rho^2_{\text{LE}}(f(\mathcal{D}), \mathbf{A}_{\text{TG}}(\mathcal{D}))] = \mathcal{O}(\ln(1/\delta)k^2)$. Hence our mechanism has significantly better utility in terms of dimension.

Pure DP vs Approx DP: It should be noted that our privacy guarantees are weaker than Riemannian Laplace. In practice, δ is chosen to be cryptographically small and typically $\delta \ll 1/n$[84].

Theoretical Results: The authors of[425] characterize the utility in terms of its expectation $\mathbb{E}[\rho^2_{\text{LE}}(f(\mathcal{D}), \mathbf{A}(\mathcal{D}))]$. By contrast, our results yield a more complete picture, as we derive the probability distribution of $\rho^2_{\text{LE}}(f(\mathcal{D}), \mathbf{A}(\mathcal{D})))$ given that we are tailoring mechanism for flat geometry of SPD matrices with log-Euclidean metric.

4.6 Experiments

We use the Riemannian Laplace mechanism as the baseline and recall that this mechanism uses the Riemannian Laplace distribution equation 4.27. Efficient sampling from the Riemannian Laplace distribution is only discussed for (i) SPDManifold with affine-invariant metric and (ii) Hypersphere with Euclidean metric in Reimherr et al.[425] and we didn't find any sampling procedure from this distribution on SPD manifold with log-Euclidean metric in[425,211] and hence we used MCMC sampling in our experiments.

4.6.1 Experiments on Synthetic Datasets

The utility depends on privacy parameters (ϵ, δ), the size k of the matrices, the dataset size n, and r, the radius of the geodesic ball containing the dataset. The utilities of the tangent Gaussian and Riemannian Laplace mechanisms have the same dependency on n, ϵ, r, such that their differentiating parameters are δ, k. Consequently, our experiments on synthetic data fix n, ϵ, r and vary δ, k.

We also consider Extrinsic approach suggested in[425] where Fréchet mean is seen to be belonging to Symmetric matrix and noise from Euclidean normal distribution is added, specifically $\mathbf{A}_{\text{EX}}(\mathcal{D}) = X, X \sim \operatorname{invvecd}\left(\mathcal{N}\left(\operatorname{vecd}\bar{\mathbf{X}}_{\text{LE}}, \sigma^2 I\right)\right)$ for appropriate σ. If r is the radius of the log-Euclidean geodesic ball of data, extrinsic sensitivity is given by $\Delta_{\text{EX}} = 2(\exp(r) - 1)/n$[425] Proposition 1. It should be emphasized that the resultant privatized Fréchet mean is *no longer a* SPD *matrix*. Hence, Reimherr et al.[425] compared the deviation between private and non-private Fréchet mean in the standard Euclidean norm.

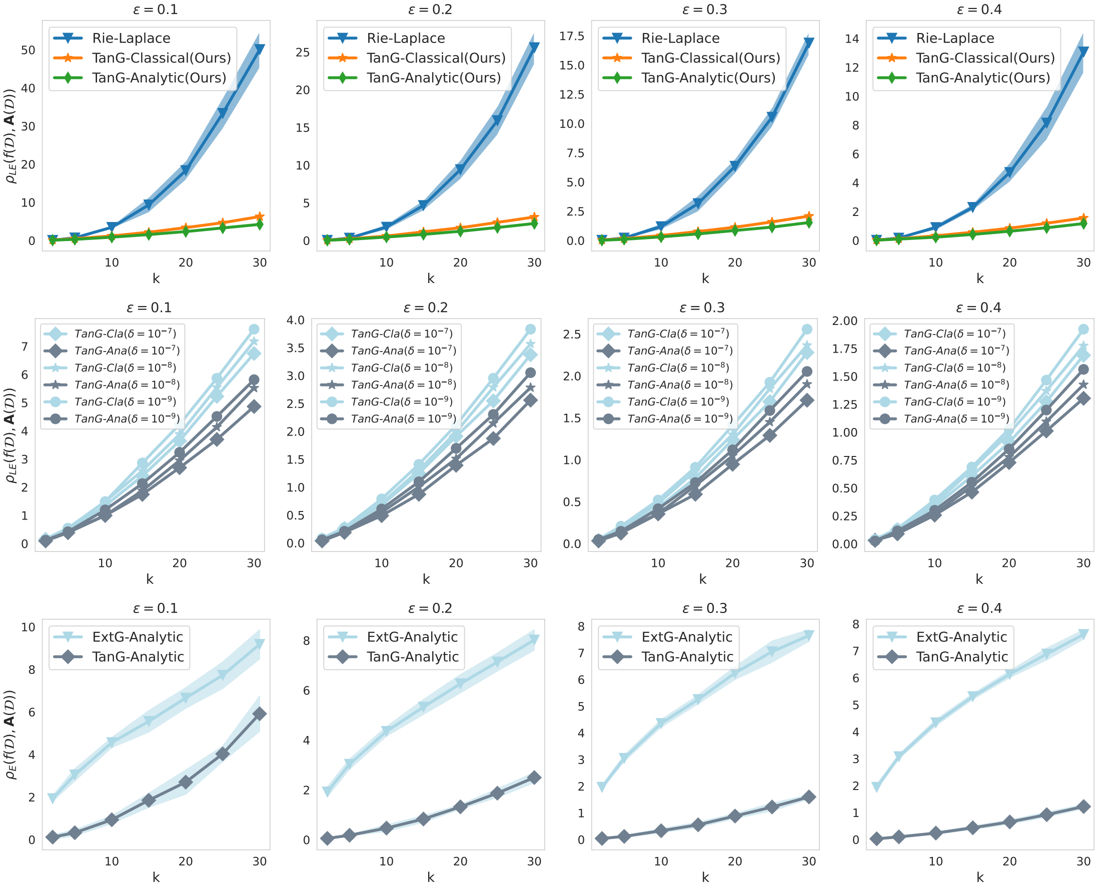


Figure 4.1: Utilities on synthetic data for *Rie-Laplace* the Riemannian Laplace mechanism[425], and *TanG Classical*, *TanG-Cla* and *TanG Analytic*, *TanG-Ana* our proposed tangent Gaussian mechanisms (classical and analytic versions), and *ExtG-Analytic* the Extrinsic analytic gaussian mechanism for different matrix sizes k and privacy parameter ϵ. ρ_{LE} and ρ_E denotes log-Euclidean and Euclidean distance respectively. Note that the output of the extrinsic mechanism is not an SPD matrix, and hence, deviation is measured in standard Euclidean distance.

We generate random $k \times k$ SPD matrices as follows: (i) generate k real values $(\lambda_1, \ldots, \lambda_k)$ uniformly in $[e^{-r}, e^r]$, (ii) build D the diagonal matrix with $D_{ii} = \lambda_i$, for $i \in \{1, \ldots, k\}$, (iii) generate a $k \times k$ random orthogonal matrix E with the Haar distribution, and (iv) build the SPD matrix as: $X = EDE^T$. This process generates SPD matrices that can be shown to belong to the geodesic ball $\mathcal{B}_{\sqrt{k}r}(I)$ with I the identity matrix: $\|\mathrm{Logm}\, X\|_F = \sqrt{\sum_{i=1}^{k} (\ln \lambda_i)^2} \leq$

$\sqrt{kr^2} = \sqrt{k}r$. We use $n = 500$ and $r = 1/4$ in our experiments and hence $\Delta_{\text{LE}} \leq \sqrt{k}/1000$.

Fig. 4.1 (first) compares utilities using a fixed $\delta = 10^{-6}$ for our mechanism, an MCMC burn-in of $50,000$ for the Riemannian Laplace mechanism, and different values of $k \in \{2, 5, 10, 15, 20, 25, 30\}$ and $\epsilon \in \{0.1, 0.2, 0.3, 0.4\}$. Each experiment is repeated 10 times, the results are averaged, and the band $(\mu - 2\sigma, \mu + 2\sigma)$ is shown, where μ and σ are the mean and standard deviation, respectively, of the associated result. The σ is small for our mechanism and does not appear on the plots. The tangent Gaussian mechanisms (ours) yield almost $\times 10$ utility improvement for larger k for each ϵ. Fig. 4.1 (middle) shows that, as expected, our utility is not significantly impacted by different values of $\delta \in \{10^{-7}, 10^{-8}, 10^{-9}\}$. Fig. 4.1 (bottom) compares utilities between Extrinsic Gaussian mechanism (analytic) and tangent Gaussian mechanism (analytic) in Euclidean distance and shows the proposed mechanism is better.

4.6.2 Experiments on Real-World Datasets

We run experiments on covariance descriptors of real-world images. Covariance descriptors[517] have been widely used for face and person recognition[518,603,399,292,336,82,596,346], action and gesture recognition[119,240,463], 3D shape analysis[501,336], medical imaging[279,120]; and even recently as layers in neural networks[588] - which makes them interesting data to privatize.

Let $\mathcal{I} \in \mathbb{R}^{h \times w \times c}$ be an image of height h, width w and with c channels, where c is 1 for grayscale images and 3 for RGB images. Let $\phi : \mathbb{R}^{h \times w \times c} \to \mathbb{R}^{hw \times k}$ be a feature extractor of dimension k, i.e. $\phi(\mathcal{I})(\mathbf{x})$ is a k-dimensional vector at each spatial coordinate $\mathbf{x}$ in the image's domain S. Given a small $\eta > 0$, the *covariance descriptor* $\mathsf{R}_\eta : \mathbb{R}^{h \times w \times c} \to \text{SPD}(k)$ associated with ϕ is defined as

$$\mathsf{R}_\eta(\mathcal{I}) = \left[\frac{1}{|\mathcal{S}|} \sum_{\mathbf{x} \in S} (\phi(\mathcal{I})(\mathbf{x}) - \mu)(\phi(\mathcal{I})(\mathbf{x}) - \mu)^T \right] + \eta . I, \tag{4.8}$$

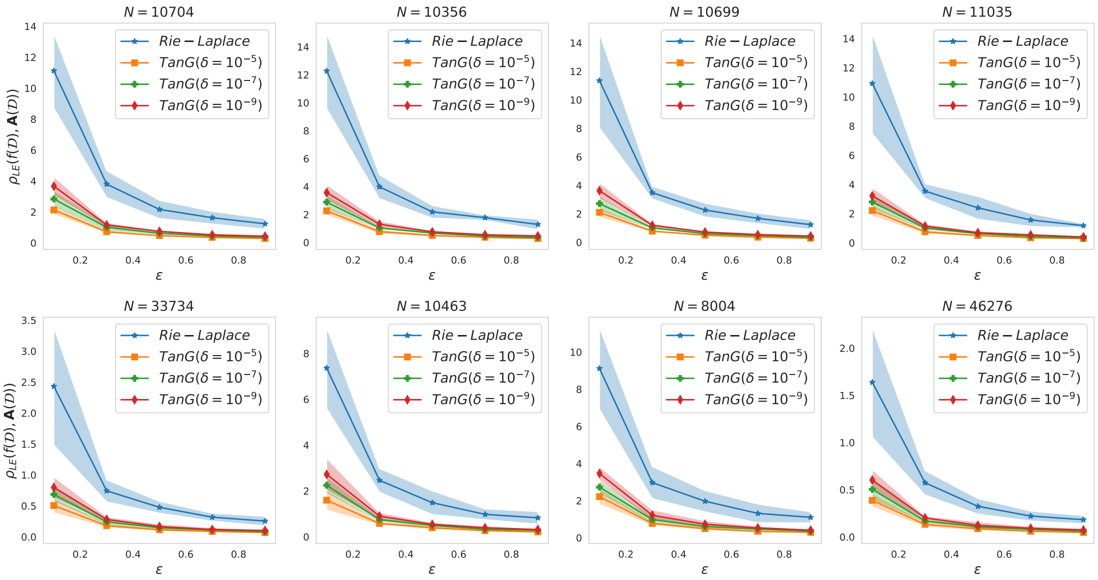


Figure 4.2: Utilities on the private Fréchet means for different privacy parameters ϵ, and real-world datasets of sizes N. Top: PathMNIST (RGB images yielding 11×11 SPD descriptors). Bottom: OctoMNIST (grayscale images yielding 9×9 SPD matrices). *Rie-Laplace* is the Riemannian Laplacian mechanism[425] and *TanG* the tangent Gaussian mechanism for different values of δ (ours). We also show the $(\mu - 2\sigma, \mu + 2\sigma)$ band.

where $\mu = |S|^{-1} \sum_{\mathbf{x} \in \mathcal{S}} \phi(\mathcal{I})(\mathbf{x})$, and $\eta.I$ ensures $\mathsf{R}_\eta(\mathcal{I}) \in \mathrm{SPD}(k)$ with η usually set to 10^{-6}. Our experiments follow[517,252] and use the covariance descriptors associated with the feature vector given as $\phi(\mathcal{I})(\mathbf{x}) = \left[x, y, \mathcal{I}, |\mathcal{I}_x|, |\mathcal{I}_y|, |\mathcal{I}_{xx}|, |\mathcal{I}_{yy}|, \sqrt{|\mathcal{I}_x|^2 + |\mathcal{I}_y|^2}, \arctan\left(\frac{|\mathcal{I}|_x}{|\mathcal{I}|_y}\right)\right]$, where $\mathbf{x} = (x, y)$, intensities derivatives are denoted by $\mathcal{I}_x, \mathcal{I}_y, \mathcal{I}_{xx}, \mathcal{I}_{yy}$ and we added the intensity values $\mathcal{I}$ for each channel compared to[517,252]. For gray scale images, $\phi(\mathcal{I})(\mathbf{x})$ is a 9-dimensional vector that makes $\mathsf{R}_\eta(\mathcal{I})$ a 9×9 SPD matrix, while for RGB images $\phi(\mathcal{I})(\mathbf{x})$ is a 11-dimensional vector that makes $\mathsf{R}_\eta(\mathcal{I})$ a 11×11 SPD matrix. We are within the assumptions of Th. 10 since such covariance descriptors belong to geodesic balls centered at I, as shown by the following theorem.

Theorem 11. Let $\mathsf{R}_\eta(\mathcal{I})$ be the covariance descriptor associated with the feature vector $\phi(\mathcal{I})$ above.

1. If $\mathcal{I}$ is a gray scale image, then $\|\mathrm{Logm}\,\mathsf{R}_\eta(\mathcal{I})\|_F \leq \sqrt{9}\max\{|\ln\eta|, |\ln(14+\eta)|\}$.
2. If $\mathcal{I}$ is a RGB image, then $\|\mathrm{Logm}\,\mathsf{R}_\eta(\mathcal{I})\|_F \leq \sqrt{11}\max\{|\ln\eta|, |\ln(16+\eta)|\}$.

Proof is given in Appendix 4.8.5

Experiments on medical imaging data

We use images from 4 classes of the medical imaging datasets PATHMNIST (grayscale) and OctoMNIST (RGB) from MedMNISTv2[577], compute the 4 class-wise Fréchet means of their covariance descriptors ($\eta = 10^{-6}$), which we privatize using the Riemannian Laplace and tangent Gaussian (analytical) mechanisms. We avoid using extrinsic approach because extrinsic sensitivity is extremely high Fig. 4.2 shows the utilities for different values of $\epsilon \in \{0.1, 0.3, 0.5, 0.7, 0.9\}$ and $\delta \in \{10^{-5}, 10^{-7}, 10^{-9}\}$. The dataset sizes N range from 8000 to 46276 images. The sensitivity of the Fréchet mean required for the mechanisms is calculated using Th. 11 and Th. 9. Each experiment is repeated 10 times and averaged, and the band $(\mu - 2\sigma, \mu + 2\sigma)$ is shown, where μ and σ are the mean and standard deviation, respectively, of the associated result. Our mechanism also outperforms the Riemannian Laplace on real-world datasets, and the utility gap is higher for smaller values of N and ϵ.

Experiments on standard imaging data

In this section, we perform additional experiments on standard image datasets. We choose MNIST, KMNIST[121] (grayscale images) and CIFAR10, FashionMNIST[572] (RGB images) as datasets. We extract images from 4 classes for each dataset and compute the corresponding class-wise Fréchet means of their covariance descriptors ($\eta = 10^{-6}$), which we privatize using the Riemannian Laplace Mechanism[425] and our proposed mechanism tangent Gaussian (Analytic). Fig. 4.3 shows the utilities for different values of $\epsilon \in \{0.1, 0.3, 0.5, 0.7, 0.9\}$ and

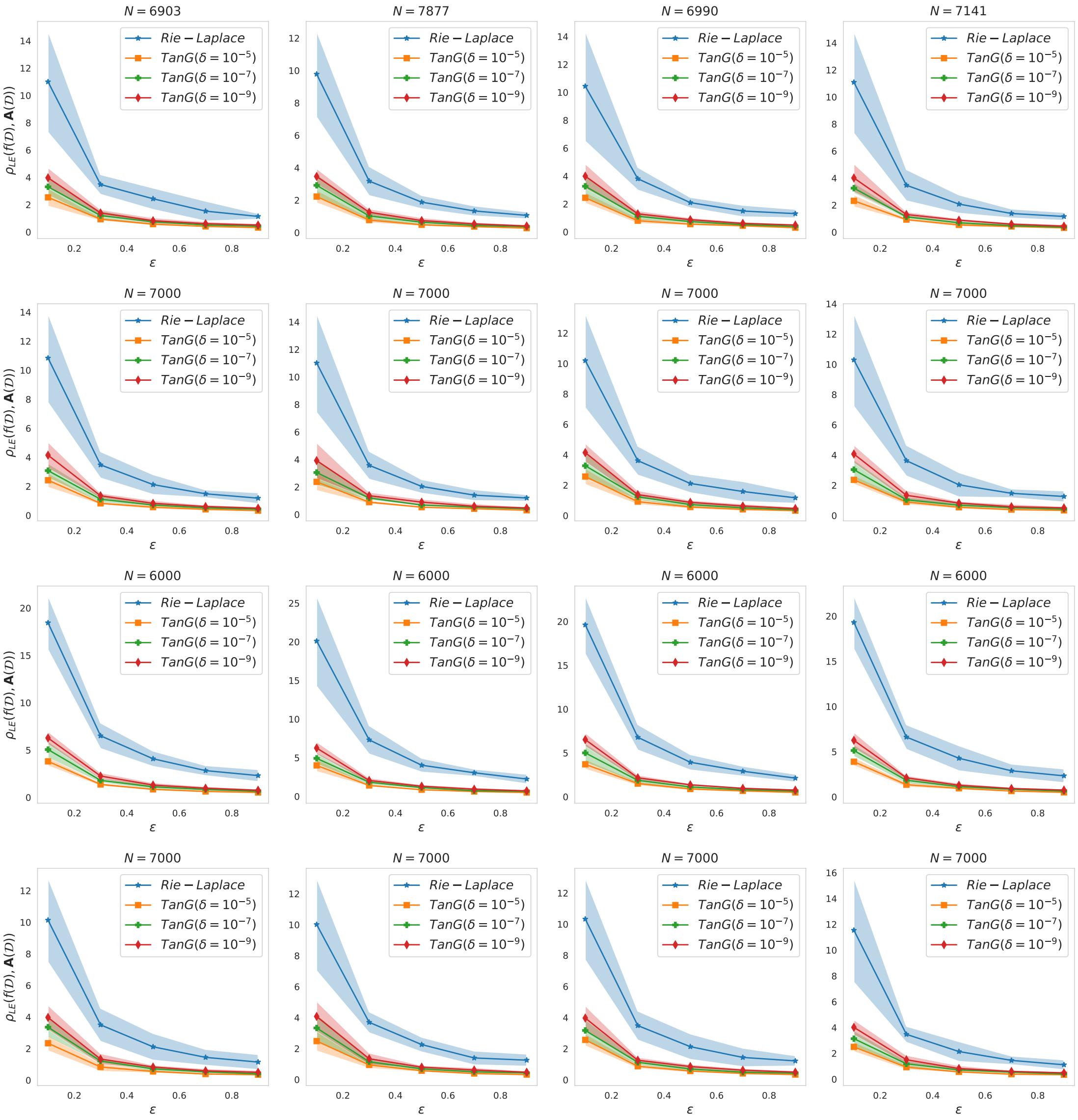


Figure 4.3: Utilities on the private Fréchet means for different privacy parameters ϵ, and real-world datasets of sizes N. First and Second Row: Fréchet mean from MNIST, KMNIST (Grayscale images yielding 9×9 SPD descriptors). Third and Fourth Row: Fréchet mean from CIFAR10, FashionMNIST (RGB images yielding 11×11 SPD descriptor). *Rie-Laplace* means the Riemannian Laplacian mechanism. *TanG* means the tangent Gaussian Mechanism for different values of δ (ours). We also show the mean-2$*$std, mean+2$*$std bands.

$\delta \in \{10^{-5}, 10^{-7}, 10^{-9}\}$. Each experiment is repeated 10 times, the results are averaged, and the band $(\mu - 2\sigma, \mu + 2\sigma)$ is shown, where μ and σ are the mean and standard deviation, respectively, of the associated result. Fig. 4.3 illustrates the better utility of our mechanism compared to the Riemannian Laplace mechanism.

4.7 Conclusion and Future Work

Differential privacy for geometric statistics and learning is at a very early stage. We proposed a tangent Gaussian mechanism that is specific to the SPD manifold equipped with the log-Euclidean metric and that outperforms the only existing baseline. One limitation of our work is that the proposed mechanism is restricted to one manifold with one specific metric. While the log-Euclidean metric is one of the most important metrics on the SPD manifold, future work should investigate how to build a Gaussian mechanism that works on any complete Riemannian manifold. We could define such as a mechanism using a Riemannian Gaussian distribution derived in[404]. The main challenge would be to show that the associated procedure is (ϵ, δ) differentially private. Future work can also seek to privatize other geometric statistical algorithms like geodesic regression or principal geodesic analysis.

4.8 Proofs

Consider $k \in \mathbb{N}^*$. In this supplementary material, $\|.\|_{\text{L2}}$ and $\langle , \rangle_2$ denote the standard Euclidean inner product and the Euclidean norm on vectors. i.e., for all $x, y \in \mathbb{R}^k$

$$\langle x, y \rangle_{\text{L2}} = \sum_{i=1}^{p} x_i y_i.$$
$$\|x\|_{\text{L2}} = \sqrt{\langle x, x \rangle_{\text{L2}}}.$$

Then, $\langle , \rangle_F, \|.\|_F$ denotes Frobenius inner product and Frobenius norm respectively, i.e., given $A, B \in \mathbb{R}^{k \times k}$

$$\langle A, B \rangle_F = \text{Tr}[A^T B].$$
$$\|A\|_F = \sqrt{\langle A, A \rangle_F}.$$

Lastly, $\|.\|_2$ denotes the spectral norm of matrices. i.e., for all $A \in \mathbb{R}^{k \times k}$

$$\|A\|_2 = \sup_{\|x\|_{\text{L2}} \neq 0} \frac{\|Ax\|_{\text{L2}}}{\|x\|_{\text{L2}}}.$$

4.8.1 Useful Lemmas

In this section, we derive the distribution of the privacy loss. Its proof requires us first to introduce the following definitions.

Definition 1 (Diffeomorphism and Isometry)**.** *A diffeomorphism between two manifolds $\mathcal{M}_1$ and $\mathcal{M}_2$ is an invertible smooth function whose inverse is also smooth. A diffeomorphism ϕ between two Riemannian manifolds $(\mathcal{M}_1, g_1)$, $(\mathcal{M}_2, g_2)$ is called an isometry if it preserves distances i.e., $\rho_{g_1}(p, q) = \rho_{g_2}(\phi(p), \phi(q))$ for all $p, q \in \mathcal{M}_1$.*

Note that Logm is a diffeomorphism from $\mathrm{SPD}(k)$ to $\mathrm{SYM}(k)$ and vecd is a diffeomorphism from $\mathrm{SYM}(k)$ to $\mathbb{R}^{\frac{k(k+1)}{2}}$, making $\mathrm{vecd}\,\mathrm{Logm}$ a diffeomorphism from $\mathrm{SPD}(k)$ to $\mathbb{R}^{\frac{k(k+1)}{2}}$. Importantly, for our derivations in the proofs of this Subsection, the operation $\mathrm{vecd}\,\mathrm{Logm}$ preserves the distances – making it an isometry.

Lemma 4.8.1 ($\mathrm{vecd}\,\mathrm{Logm}$ is an isometry)**.** *Let* $\mathrm{Logm} : \mathrm{SPD}(k) \to \mathrm{SYM}(k)$ *be the matrix logarithm and let* $\mathrm{vecd} : \mathrm{SYM}(k) \to \mathbb{R}^{\frac{k(k+1)}{2}}$ *be defined as* $\mathrm{vecd}(X) = \left[\mathrm{diag}(X)^T, \sqrt{2}\,\mathrm{upperdiag}(X)^T\right]^T$. *Then* $\mathrm{vecd}\,\mathrm{Logm} : \mathrm{SPD}(k) \to \mathbb{R}^{\frac{k(k+1)}{2}}$ *is an isometry from* $\mathrm{SPD}(k)$ *equipped with the log-Euclidean metric to standard Euclidean space* $\mathbb{R}^{\frac{k(k+1)}{2}}$ *with standard* $\mathrm{L2}$ *metric, i.e.,*

$$\rho_{\mathrm{LE}}(X_1, X_2) = \rho_{\mathrm{L2}}(\mathrm{vecd}\,\mathrm{Logm}\,X_1, \mathrm{vecd}\,\mathrm{Logm}\,X_2), \tag{4.9}$$

where $X_1, X_2 \in \mathrm{SPD}(k)$. *Hence we have that*

$$\|\mathrm{Logm}\,X\|_F = \|\mathrm{vecd}\,\mathrm{Logm}\,X\|_{\mathrm{L2}}. \tag{4.10}$$

Proof. Let X_1, X_2 be elements of SPD(k). We have:

$$\begin{aligned}
&\rho^2_{\text{LE}}(X_1, X_2)\\
&= \|\text{Logm}\, X_1 - \text{Logm}\, X_2\|^2_F\\
&= \sum_{i,j}^{k} (\text{Logm}\, X_1 - \text{Logm}\, X_2)^2_{ij}\\
&= \sum_{i<j}^{k} (\text{Logm}\, X_1 - \text{Logm}\, X_2)^2_{ij} + \sum_{i>j}^{k} (\text{Logm}\, X_1 - \text{Logm}\, X_2)^2_{ij} + \sum_{i=j}^{k} (\text{Logm}\, X_1 - \text{Logm}\, X_2)^2_{ij}\\
&= 2. \sum_{i<j}^{k} (\text{Logm}\, X_1 - \text{Logm}\, X_2)^2_{ij} + \sum_{i=j}^{k} (\text{Logm}\, X_1 - \text{Logm}\, X_2)^2_{ij}\\
&= \|\sqrt{2}\,\text{upperdiag}(\text{Logm}\, X_1 - \text{Logm}\, X_2)\|^2_{\text{L2}} + \|\text{diag}\,(\text{Logm}\, X_1 - \text{Logm}\, X_2)\|^2_{\text{L2}}\\
&= \|\text{vecd}(\text{Logm}\, X_1 - \text{Logm}\, X_2)\|^2_{\text{L2}}\\
&= \|\text{vecd}\,\text{Logm}\, X_1 - \text{vecd}\,\text{Logm}\, X_2\|^2_{\text{L2}}\\
&= \rho^2_{\text{L2}}(\text{vecd}\,\text{Logm}\, X_1, \text{vecd}\,\text{Logm}\, X_2).
\end{aligned}$$

from which we have Eq. equation 4.9. Eq. equation 4.10 follows as

$$\|\text{Logm}\, X\|_F = \rho_{\text{LE}}(X, I) = \rho_{\text{L2}}(\text{vecd}\,\text{Logm}\, X, \text{vecd}\,\text{Logm}\, I) = \|\text{vecd}\,\text{Logm}\, X\|_{\text{L2}}.$$

□

Now, we prove some useful properties of the Log Gaussian distribution, denoted $\mathcal{LN}$, that we will use later. Essentially, we show that the Log Gaussian distribution behaves “nicely” with a vector space structure of SPD(k). We recall that the vector space operations on the SPD manifold are defined as follows,

$$X_1 \oplus X_2 = \text{Expm}\left[\text{Logm}\, X_1 + \text{Logm}\, X_2\right]. \tag{4.11}$$

$$X_1 \ominus X_2 = \text{Expm}\left[\text{Logm}\, X_1 - \text{Logm}\, X_2\right]. \tag{4.12}$$

Lemma 4.8.2. *Take $k \in \mathbb{N}$. Let I denote the $k \times k$ identity matrix, and consider $M, C \in$ SPD(k), $\Sigma \in$ SPD$(\frac{k(k+1)}{2})$ and χ^2_d the chi-square distribution with d degrees of freedom. Then:*

$$X \sim \mathcal{LN}(I, \Sigma) \implies X \oplus M \sim \mathcal{LN}(M, \Sigma). \tag{4.13}$$

$$X \sim \mathcal{LN}(I, \sigma^2 I) \implies \langle \text{Logm}\, C, \text{Logm}\, X \rangle_F \sim \mathcal{N}(0, \sigma^2 \|\text{Logm}\, C\|_F^2). \tag{4.14}$$

$$X \sim \mathcal{LN}(I, \sigma^2 I) \implies \|\text{Logm}\, X\|_F^2 \sim \sigma^2 \chi^2_{\frac{k(k+1)}{2}}. \tag{4.15}$$

Proof. We first recall standard properties of multivariate normal distribution. Let $m, a \in \mathbb{R}^p$ and $\Sigma, I \in \mathbb{R}^{p \times p}$ then following properties hold true.

$$x \sim \mathcal{N}(m, \Sigma) \implies a + x \sim \mathcal{N}(a + m, \Sigma). \tag{4.16}$$

$$x \sim \mathcal{N}(m, \Sigma) \implies a^T x \sim \mathcal{N}(a^T m, a^T \Sigma a). \tag{4.17}$$

$$x \sim \mathcal{N}(0, \sigma^2 I) \implies \|x\|_{\text{L2}}^2 \sim \sigma^2 \chi^2_p. \tag{4.18}$$

where χ^2 denotes chi-square distribution. We prove the properties (a)-(c) below.

(a) Distribution of $X \oplus M$.

$$\begin{aligned}\operatorname{vecd}[\operatorname{Logm}[X \oplus M]] &\overset{(*)}{=} \operatorname{vecd}[\operatorname{Logm}[\operatorname{Expm}[\log X + \log M]]] \\ &= \operatorname{vecd}[\operatorname{Logm} X + \operatorname{Logm} M] \\ &= \operatorname{vecd}[\operatorname{Logm} X] + \operatorname{vecd}[\operatorname{Logm} M] \\ &\overset{(**)}{\sim} \mathcal{N}(\operatorname{vecd}[\operatorname{Logm} M], \Sigma).\end{aligned}$$

where in $(*)$ we used Eq. 4.11 and in $(**)$ Eq. equation 4.16.

(b) Distribution of $\langle \operatorname{Logm} C, \operatorname{Logm} X\rangle_F$.

$$\begin{aligned}\langle \operatorname{Logm} C, \operatorname{Logm} X\rangle_F &\overset{(*)}{=} \langle \operatorname{vecd}[\operatorname{Logm} C], \operatorname{vecd}[\operatorname{Logm} X]\rangle_{\mathrm{L2}} \\ &\overset{(**)}{\sim} \mathcal{N}\left(\langle \operatorname{vecd}[\operatorname{Logm} C], 0\rangle_{\mathrm{L2}}, \operatorname{vecd}[\operatorname{Logm} C]^T \sigma^2 I \operatorname{vecd}[\operatorname{Logm} C]\right) \\ &\sim \mathcal{N}(0, \sigma^2 \|\operatorname{vecd}[\operatorname{Logm} C]\|_{\mathrm{L2}}^2) \\ &\overset{(*)}{\sim} \mathcal{N}(0, \sigma^2 \|\operatorname{Logm} C\|_F^2).\end{aligned}$$

where we used Eq. 4.10 in $(*)$ and Eq. 4.17 in $(**)$.

(c) Distribution of $\|\operatorname{Logm} X\|_F^2$.

$$\|\operatorname{Logm} X\|_F^2 \overset{(*)}{=} \|\operatorname{vecd}[\operatorname{Logm} X]\|_{\mathrm{L2}}^2 \overset{(**)}{\sim} \sigma^2 \chi^2_{\frac{k(k+1)}{2}}.$$

where we used Eq. 4.10 in $(*)$ and Eq. 4.18 in $(**)$ with $p = \frac{k(k+1)}{2}$. □

As a corollary, we give an equivalent reformulation of the tangent Gaussian mechanism that will be useful in the rest of the proofs.

Corollary 4.8.1 (Equivalent Reformulation of tangent Gaussian)**.** *Let* $\mathbf{A}_{\text{TG}}$ *be a tangent Gaussian mechanism defined as* $\mathbf{A}_{\text{TG}}(f(\mathcal{D})) = X$, $X \sim \mathcal{LN}(f(\mathcal{D}), \sigma^2 I)$. *Then, it is equivalently defined as:*

$$\mathbf{A}_{\text{TG}}(f(\mathcal{D})) = f(\mathcal{D}) \oplus N, N \sim \mathcal{LN}(I, \sigma^2 I).$$

Proof. The proof comes from Eq. 4.13 of Lemma 4.8.2. □

Now, we are ready to prove the distribution of the privacy loss of the tangent Gaussian Mechanism, which is given Th. 7.

4.8.2 PROOF OF TH. 7

Theorem 4.8.1 (Distribution of the privacy loss of the tangent Gaussian). Let $\mathbf{A}_{\text{TG}}$ be a tangent Gaussian mechanism with variance σ^2. Its privacy loss is normally distributed as

$$L_{\mathbf{A}_{\text{TG}},\mathcal{D},\mathcal{D}'} \sim \mathcal{N}\left(\frac{\rho^2_{\text{LE}}(f(\mathcal{D}), f(\mathcal{D}'))}{2\sigma^2}, \frac{\rho^2_{\text{LE}}(f(\mathcal{D}), f(\mathcal{D}'))}{\sigma^2}\right).$$

Proof. Assume that $\mathcal{D}, \mathcal{D}'$ are adjacent datasets. Let $V = f(\mathcal{D}) \ominus f(\mathcal{D}')$. Consider the privacy loss random variable $L_{\mathbf{A}_{\mathrm{TG}},\mathcal{D},\mathcal{D}'}$. Let $Y = \mathbf{A}_{\mathrm{TG}}(\mathcal{D})$.

$$
\begin{aligned}
&\ln\left(\frac{p_{\mathbf{A}_{\mathrm{TG}}(\mathcal{D})}(Y)}{p_{\mathbf{A}_{\mathrm{TG}}(\mathcal{D}')}(Y)}\right)\\
&\overset{(1)}{=} \ln\left(\frac{p_{\mathbf{A}_{\mathrm{TG}}(\mathcal{D})}(f(\mathcal{D})\oplus N)}{p_{\mathbf{A}_{\mathrm{TG}}(\mathcal{D}')}(f(\mathcal{D})\oplus N)}\right)\\
&\overset{(2)}{=} -\frac{1}{2}\left[\operatorname{vecd}\Big(\operatorname{Logm}(f(\mathcal{D})\oplus N) - \operatorname{Logm} f(D)\Big)\right]^T \frac{I}{\sigma^2}\operatorname{vecd}\Big(\operatorname{Logm}(f(\mathcal{D})\oplus N) - \operatorname{Logm} f(D)\Big)\\
&\quad + \frac{1}{2}\left[\operatorname{vecd}\Big(\operatorname{Logm}(f(\mathcal{D})\oplus N) - \operatorname{Logm} f(D')\Big)\right]^T \frac{I}{\sigma^2}\operatorname{vecd}\Big(\operatorname{Logm}(f(\mathcal{D})\oplus N) - \operatorname{Logm} f(D')\Big)\\
&\overset{(3)}{=} -\frac{1}{2\sigma^2}\|\operatorname{vecd}\Big(\operatorname{Logm} N\Big)\|_{\mathrm{L2}}^2 + \frac{1}{2\sigma^2}\|\operatorname{vecd}\Big(\operatorname{Logm} f(D) - \operatorname{Logm} f(D') + \operatorname{Logm} N\Big)\|_{\mathrm{L2}}^2\\
&\overset{(4)}{=} -\frac{1}{2\sigma^2}\|\operatorname{vecd}\Big(\operatorname{Logm} N\Big)\|_{\mathrm{L2}}^2 + \frac{1}{2\sigma^2}\|\operatorname{vecd}\Big(\operatorname{Logm}(V\oplus N)\Big)\|_{\mathrm{L2}}^2\\
&\overset{(5)}{=} \frac{1}{2\sigma^2}\left[\|\operatorname{Logm}(V\oplus N)\|_F^2 - \|\operatorname{Logm} N\|_F^2\right]\\
&= \frac{1}{2\sigma^2}\left[\|\operatorname{Logm} V\|_F^2 + 2\langle \operatorname{Logm} V, \operatorname{Logm} N\rangle_F\right]\\
&\overset{(6)}{\sim} \frac{1}{2\sigma^2}\left[\|\operatorname{Logm} V\|_F^2 + 2\mathcal{N}\Big(0, \sigma^2\|\operatorname{Logm} V\|_F^2\Big)\right]\\
&\overset{(7)}{\sim} \mathcal{N}\left(\frac{\|\operatorname{Logm} V\|_F^2}{2\sigma^2}, \frac{\|\operatorname{Logm} V\|_F^2}{\sigma^2}\right)\\
&\overset{(8)}{\sim} \mathcal{N}\left(\frac{\rho_{\mathrm{LE}}^2(f(\mathcal{D}), f(\mathcal{D}'))}{2\sigma^2}, \frac{\rho_{\mathrm{LE}}^2(f(\mathcal{D}), f(\mathcal{D}'))}{\sigma^2}\right),
\end{aligned}
$$

where we used the following properties in each of the steps labeled above.

1. Equivalent reformulation of tangent Gaussian, Corollary. 4.8.1.

2. Density of Log Gaussian Distribution.

3. $f(\mathcal{D}) \oplus N = \operatorname{Expm}[\operatorname{Logm} f(\mathcal{D}) + \operatorname{Logm} N]$.

4. $\operatorname{Logm}(V \oplus N) = \operatorname{Logm} f(D) - \operatorname{Logm} f(D') + \operatorname{Logm} N$.

5. Isometry of the vecd operation, Eq.4.10

6. Eq. 4.14 in Lemma. 4.8.2.

7. standard Gaussian property, see Eq. 4.16.

8. $\|\text{Logm}\, V\|_F^2 = \|\text{Logm}\, f(\mathcal{D}) - \text{Logm}\, f(\mathcal{D}')\|_F^2 = \rho_{\text{LE}}^2(f(\mathcal{D}), f(\mathcal{D}'))$.

□

4.8.3 Proof of Th. 8

In this section, we give proof of privacy guarantee of the tangent Gaussian Mechanism.

Proof. The proof proceeds similarly to the proofs referenced below by only replacing the standard sensitivity Δ_{L2} with respect to the Euclidean L_2 metric, by Δ_{LE}:

1. (Classical). See Th. A.1 (Appendix A Page 261) in[160].

2. (Analytic). See Th. 5, Th. 8, Th. 9 (Section 3) in[42].

The fact that the mechanism is manifold valued comes into play while deriving privacy loss (Taken care by Theorem 1). Once privacy loss (which is *real valued scalar* random variable) is derived, going from privacy loss to actual privacy guarantee wouldn't be affected whether a mechanism is manifold-valued or not because both of the above proofs *entirely* rely on properties of one-dimensional euclidean Gaussian random variables. Specifically,

1. (Classical). Directly employs tail bound of one-dimensional Gaussian variable that $\mathbb{P}[x > t] < \frac{\sigma}{\pi} \exp(-\frac{t^2}{2\sigma^2})$

2. (Analytic). The method employs both the sufficient and necessary conditions of the (ϵ, δ) guarantee. Additionally, the algorithm avoids using tail bounds since they may be loose. Instead, it uses properties of Gaussian CDFs and employs binary search to

solve analytically for σ, given (ϵ, δ). See[42] Algorithm 1 and discussion therein for more details.

$\square$

4.8.4 Proof of Th. 9 and Th. 10

In this section, we prove the sensitivity of the Fréchet Mean in Theorem. 9 and then the utility of the tangent Gaussian Mechanism in Theorem. 10. First, we give the proof of 9.

Proof. Consider $k \in \mathbb{N}$, $0 < r < \infty$ and $M \in \mathrm{SPD}(k)$ such that $\mathcal{B}_r(M)$ is a geodesic ball of radius r and center M. Let $\mathcal{D} \sim \mathcal{D}'$ be adjacent datasets of size $n \in \mathbb{N}$ that lie in $\mathcal{B}_r(M)$. Without loss of generality, we can assume that they differ only by their last data point X_n and X_n': $\mathcal{D} = \{X_1, X_2, \dots, X_n\}$ and $\mathcal{D}' = \{X_1, X_2, \dots, X_n'\}$. Let $\overline{X}_{\mathcal{D}}, \overline{X}_{\mathcal{D}'}$ denote the Fréchet means of $\mathcal{D}$ and $\mathcal{D}'$ for the log-Euclidean metric, which can be expressed in closed forms as mentioned in the main text. The log-Euclidean distance between the Fréchet means writes:

$$
\begin{aligned}
&\rho_{\mathrm{LE}}(\overline{X}_{\mathcal{D}}, \overline{X}_{\mathcal{D}'})\\
&\overset{(*)}{=} \|\mathrm{Logm}\left(\mathrm{Expm}\left(\sum_{i=1}^{n} \frac{\mathrm{Logm}\, X_i}{n}\right)\right) - \mathrm{Logm}\left(\mathrm{Expm}\left(\sum_{i=1}^{n-1} \frac{\mathrm{Logm}\, X_i}{n} + \frac{\mathrm{Logm}\, X_n'}{n}\right)\right)\|_F\\
&= \|\frac{1}{n}\sum_{i=1}^{n-1} \mathrm{Logm}\, X_i - \frac{1}{n}\sum_{i=1}^{n-1} \mathrm{Logm}\, X_i + \frac{1}{n}\mathrm{Logm}\, X_n - \frac{1}{n}\mathrm{Logm}\, X_n'\|_F\\
&= \frac{1}{n}\|\mathrm{Logm}\, X_n - \mathrm{Logm}\, X_n'\|_F\\
&= \frac{1}{n}\rho_{\mathrm{LE}}(X_n, X_n').
\end{aligned}
$$

$$\Delta_{\text{LE}} = \sup_{\mathcal{D}\sim\mathcal{D}'} \rho_{\text{LE}}(\overline{X}_{\mathcal{D}}, \overline{X}_{\mathcal{D}'}) = \sup_{\mathcal{D}\sim\mathcal{D}'} \frac{1}{n}\rho_{\text{LE}}(X_n, X_n') \overset{(\dagger)}{\leq} \frac{1}{n}\left[\rho_{\text{LE}}(X_n, M) + \rho_{\text{LE}}(M, X_n')\right] \overset{(\ddagger)}{\leq} \frac{2r}{n},$$

where we use the closed form for the log-Euclidean Fréchet means in $(*)$, the triangle inequality in $(\dagger)$, and the assumption that data lies in $\mathcal{B}_r(M)$ in $(\ddagger)$. □

Proof of Th. 10 is given as follows,

Proof. Consider deviation $\rho^2_{\text{LE}}(f(\mathcal{D}), \mathbf{A}_{\text{TG}}(\mathcal{D})))$

$$\begin{aligned}
\rho^2_{\text{LE}}(f(\mathcal{D}), \mathbf{A}_{\text{TG}}(\mathcal{D}))) &= \|\text{Logm}\, f(\mathcal{D}) - \text{Logm}\, \mathbf{A}_{\text{TG}}(\mathcal{D})\|_F^2 \overset{(1)}{=} \|\text{Logm}\, f(\mathcal{D}) - \text{Logm}(f(\mathcal{D}) \oplus N)\|_F^2 \\
&\overset{(2)}{=} \|\text{Logm}\, N\|_F^2 \\
&\overset{(3)}{\sim} \sigma^2 \chi^2_d,
\end{aligned}$$

where we use the following properties at each step:

(1) Corollary. 4.8.1.

(2) $f(\mathcal{D}) \oplus N = \text{Expm}\,[\text{Logm}\, f(\mathcal{D}) + \text{Logm}\, N]$.

(3) Eq. 4.15 of Lemma. 4.8.2.

Now we derive expression for $\mathbb{E}[\rho^2_{\text{LE}}(f(\mathcal{D}), \mathbf{A}_{\text{TG}}(\mathcal{D}))]$

$$\begin{aligned}
\mathbb{E}[\rho^2_{\text{LE}}(f(\mathcal{D}), \mathbf{A}_{\text{TG}}(\mathcal{D}))] &\overset{(1)}{=} \sigma^2 d \\
&\overset{(2)}{=} \frac{2\Delta^2_{\text{LE}} \ln(1.25/\delta) d}{\epsilon^2} \\
&\overset{(3)}{\leq} \frac{8r^2 \ln(1.25/\delta) d}{n^2 \epsilon^2}.
\end{aligned}$$

where we use the following properties at each step:

1. $c \sim \chi^2_d \implies \mathbb{E}[c] = d$ i.e., the expectation of chi-squared distributed random variable is the number of degrees of freedom.

2. $\sigma = \Delta_{\text{LE}}\sqrt{2\ln(1.25/\delta)}/\epsilon$ for (ϵ, δ)-$\mathbf{A}_{\text{TG}}$ from Th. 8.

3. $\Delta_{\text{LE}} \leq \frac{2r}{n}$ from Th. 9.

□

4.8.5 Proof of Theorem 11

In this section, we derive the log-Euclidean geodesic radius of covariance descriptors. We first prove the following lemma that relates $||\operatorname{Logm} X||_F$ in terms of the lower bound on the least eigenvalue and upper bound on the largest eigenvalue of X.

Lemma 4.8.3. *If $X \in \text{SPD}(k)$ and let $\lambda_{\text{min}}(X), \lambda_{\text{max}}(X)$ be the minimum and maximum eigenvalues of X. If $\ell \leq \lambda_{\text{min}}(X)$ and $\lambda_{\text{max}}(X) \leq L$ Then, $\|\text{Logm}\, X\|_F \leq \sqrt{k}\max\{|\ln \ell|, |\ln L|\}$.*

Proof. Consider,

$$\begin{aligned} \|\text{Logm}\, X\|_F &\overset{(\dagger)}{\leq} \sqrt{k}\|\text{Logm}\, X\|_2 \\ &= \sqrt{k}\max_{i=1}^{n}|\ln \lambda_i| \\ &= \sqrt{k}\max\left\{|\min_{i=1}^{n}\ln\lambda_i|, |\max_{i=1}^{n}\ln\lambda_i|\right\} \\ &\overset{(\ddagger)}{=} \sqrt{k}\max\left\{|\ln\min_{i=1}^{n}\lambda_i|, |\ln\max_{i=1}^{n}\lambda_i|\right\} \\ &= \sqrt{k}\max\left\{|\ln\lambda_{\text{min}}|, |\ln\lambda_{\text{max}}|\right\}. \end{aligned} \tag{4.19}$$

where (†) uses the fact that $A \in \mathbb{R}^{k\times k}$, $\|A\|_F \leq \sqrt{k}\|A\|_2$ and (‡) uses the fact that ln is monotonically increasing. Now, we split the derivation into two cases.

1. CASE $\lambda_{\text{min}}(X) \geq 1$. For $x \geq 1$, $|\ln x|$ is an increasing function, which gives us: $|\ln \ell| \leq |\ln \lambda_{\text{min}}(X)| \leq |\ln \lambda_{\text{max}}| \leq |\ln L|$

$$\sqrt{k} \max\left\{|\ln \lambda_{\text{min}}|, |\ln \lambda_{\text{max}}|\right\} \leq \sqrt{k}|\ln L| = \sqrt{k} \max\left\{|\ln \ell|, |\ln L|\right\}. \tag{4.20}$$

2. CASE $\lambda_{\text{min}}(X) < 1$. For $x < 1$, $|\ln x|$ is a decreasing function: $|\ln \lambda_{\text{min}}| \leq |\ln \ell|$. We further split the derivation into two sub-cases here

   (a) SUB-CASE $\lambda_{\text{max}} \geq 1$. In this sub-case $|\ln \lambda_{\text{max}}| \leq |\ln L|$ and $\ln \lambda_{\text{min}} \leq |\ln \ell|$ from which we have that

   $$\sqrt{k} \max\left\{|\ln \lambda_{\text{min}}|, |\ln \lambda_{\text{max}}|\right\} \leq \sqrt{k} \max\left\{|\ln \ell|, |\ln L|\right\}. \tag{4.21}$$

   (b) SUB-CASE $\lambda_{\text{max}} < 1$. In this sub-case $|\ln L| \leq |\ln \lambda_{\text{max}}| \leq |\ln \lambda_{\text{min}}| \leq |\ln \ell|$.

   $$\sqrt{k} \max\left\{|\ln \lambda_{\text{min}}|, |\ln \lambda_{\text{max}}|\right\} \leq \sqrt{k}|\ln \ell| = \sqrt{k} \max\left\{|\ln \ell|, |\ln L|\right\}. \tag{4.22}$$

Based on Eq. 4.20, Eq. 4.21, Eq. 4.22 and Eq.4.19. We can conclude the lemma.

□

Lemma 4.8.4. *Let $R_\eta(\mathcal{I})$ denote the covariance descriptor for image $\mathcal{I}$ for given $\eta > 0$, which is defined as follows ,*

$$R_\eta(\mathcal{I}) = \left[\frac{1}{|\mathcal{S}|}\sum_{\boldsymbol{x}\in S}(\phi(\mathcal{I})(\boldsymbol{x}) - \mu)(\phi(\mathcal{I})(\boldsymbol{x}) - \mu)^T\right] + \eta.I,$$

with,

$$\phi(\mathcal{I}) = \left[x, y, \mathcal{I}, |\mathcal{I}_x|, |\mathcal{I}_y|, |\mathcal{I}_{xx}|, |\mathcal{I}_{yy}|, \sqrt{|\mathcal{I}_x|^2 + |\mathcal{I}_y|^2}, \arctan\left(\frac{|\mathcal{I}_x|}{|\mathcal{I}_y|}\right)\right].$$

where x, y are grid positions of Image $\mathcal{I}$, $\mathcal{I}$ denote pixel intensity values , $|\mathcal{I}_x|, |\mathcal{I}_y|$ denotes first order intensity derivatives and $|\mathcal{I}_{xx}|, |\mathcal{I}_{yy}|$ denotes the second-order intensity derivatives then following holds,

1. *If $\mathcal{I}$ is grayscale image, then $\|\mathsf{R}_\eta(\mathcal{I})\|_2 \leq 12 + \eta$.*

2. *If $\mathcal{I}$ is RGB image then $\|\mathsf{R}_\eta(\mathcal{I})\|_2 \leq 14 + \eta$.*

Proof. We have:

$$\begin{aligned}
\|\mathsf{R}_\eta(\mathcal{I})\|_2 &= \|\left[\frac{1}{|\mathcal{S}|}\sum_{\mathbf{x}\in S}(\phi(\mathcal{I})(\mathbf{x}) - \mu)(\phi(\mathcal{I})(\mathbf{x}) - \mu)^T\right] + \eta.I\|_2 \\
&\overset{(1)}{\leq} \frac{1}{|\mathcal{S}|}\sum_{\mathbf{x}\in S}\|(\phi(\mathcal{I})(\mathbf{x}) - \mu)(\phi(\mathcal{I})(\mathbf{x}) - \mu)^T\|_2 + \|\eta.I\|_2 \\
&\leq \max_{\mathbf{x}\in\mathcal{S}}\|(\phi(\mathcal{I})(\mathbf{x}) - \mu)(\phi(\mathcal{I})(\mathbf{x}) - \mu)^T\|_2 + \eta \\
&\overset{(2)}{=} \max_{\mathbf{x}\in\mathcal{S}}\|(\phi(\mathcal{I})(\mathbf{x}) - \mu)\|_{\mathrm{L2}}^2 + \eta \\
&\overset{(3)}{\leq} \max_{\mathbf{x}\in\mathcal{S}}\|\phi(\mathcal{I})(\mathbf{x})\|_{\mathrm{L2}}^2 + \eta,
\end{aligned} \tag{4.23}$$

where we used the following properties in each of the steps:

1. Triangle Inequality.

2. For all $a \in \mathbb{R}^p$, the spectral norm of 1-rank matrix aa^T is $\|a\|_{\mathrm{L2}}^2$.

3. Consider the descriptor $\phi(\mathcal{I}) = \Big[x, y, \mathcal{I}, |\mathcal{I}_x|, |\mathcal{I}_y|, |\mathcal{I}_{xx}|, |\mathcal{I}_{yy}|, \sqrt{|\mathcal{I}_x|^2 + |\mathcal{I}_y|^2}, \arctan\Big(\frac{|\mathcal{I}_x|}{|\mathcal{I}_y|}\Big)\Big]$. Then, $\phi(\mathcal{I})(\mathbf{x})_i \geq 0$ for each $\mathbf{x} \in \mathcal{S}$ and $i \in \{1, \ldots, k\}$. This yields: $(\mu)_i = \big(|\mathcal{S}|^{-1} \sum_{\mathbf{x} \in \mathcal{S}} \phi(\mathcal{I})(\mathbf{x})\big)_i \geq 0$. Hence it implies that $\|\phi(\mathcal{I})(\mathbf{x}) - \mu\|^2_{\mathrm{L2}} \leq \|\phi(\mathcal{I})(\mathbf{x})\|^2_{\mathrm{L2}}$.

Then, the following calculations provide an upper bound for $\|\phi(\mathcal{I})(\mathbf{x})\|^2_2$. Specifically, we bound each of the 6 elements constituting the descriptor $\phi(\mathcal{I}) = \Big[x, y, \mathcal{I}, |\mathcal{I}_x|, |\mathcal{I}_y|, |\mathcal{I}_{xx}|, |\mathcal{I}_{yy}|, \sqrt{|\mathcal{I}_x|^2 + |\mathcal{I}_y|^2}, \mathrm{arctan}$

1. Normalized grid positions : $\forall \mathbf{x} \in \mathcal{S},\ 0 \leq x, y \leq 1$.

2. Pixel intensity values C_i for $i \in [c]$: $\forall \mathbf{x} \in \mathcal{S},\ 0 \leq C_i[\mathbf{x}] \leq 1$.

3. First intensity derivatives $|\mathcal{I}_x|, |\mathcal{I}_y|$: The first intensity derivatives can be obtained by the convolution operation (denoted as $\star$):

$$\mathcal{I}_x = \mathcal{I} \star \frac{1}{4} \begin{bmatrix} +1 & 0 & -1 \\ +2 & 0 & -2 \\ +1 & 0 & -1 \end{bmatrix}, \mathcal{I}_y = \mathcal{I} \star \frac{1}{4} \begin{bmatrix} +1 & +2 & +1 \\ 0 & 0 & 0 \\ -1 & -2 & -1 \end{bmatrix}.$$

   Since $0 \leq \mathcal{I}(\mathbf{x}) \leq 1$, using the definition of the convolution operation yields $\forall \mathbf{x} \in \mathcal{S}$, $|\mathcal{I}_x(\mathbf{x})| \leq 1$, $|\mathcal{I}_y(\mathbf{x})| \leq 1$.

4. Second intensity derivatives $|\mathcal{I}_{xx}|, |\mathcal{I}_{yy}|$: The second intensity derivatives can be obtained by the convolution operation (denoted as $\star$)

$$\mathcal{I}_{xx} = \mathcal{I} \star \frac{1}{32} \begin{bmatrix} +1 & 0 & -2 & 0 & 1 \\ +4 & 0 & -8 & 0 & 4 \\ +6 & 0 & -12 & 0 & 6 \\ +4 & 0 & -8 & 0 & 4 \\ +1 & 0 & -2 & 0 & 1 \end{bmatrix}, \mathcal{I}_{yy} = \mathcal{I} \star \frac{1}{32} \begin{bmatrix} +1 & +4 & +6 & +4 & +1 \\ 0 & 0 & 0 & 0 & 0 \\ -2 & -8 & -12 & -8 & -2 \\ 0 & 0 & 0 & 0 & 0 \\ +1 & +4 & +6 & +4 & +1 \end{bmatrix}.$$

Since $0 \leq \mathcal{I}(\mathbf{x}) \leq 1$, using the definition of the convolution operation yields $|\mathcal{I}_{xx}(\mathbf{x})| \leq 1$, $|\mathcal{I}_{yy}(\mathbf{x})| \leq 1$.

5. Norm of first intensity derivatives : since $|\mathcal{I}_x(\mathbf{x})| \leq 1$, $|\mathcal{I}_y(\mathbf{x})| \leq 1$ we have that $\forall \mathbf{x} \in \mathcal{S}$, $\sqrt{\mathcal{I}_x(\mathbf{x})^2 + \mathcal{I}_y(\mathbf{x})^2} \leq \sqrt{2}$.

6. Angle of intensity derivatives : Note that for $a \geq 0$, $0 \leq \arctan a \leq \frac{\pi}{2}$. Hence we have that $\forall \mathbf{x} \in \mathcal{S}, \arctan\left(|\frac{\mathcal{I}_x(\mathbf{x})}{\mathcal{I}_y(\mathbf{x})}|\right) \leq \frac{\pi}{2}$.

These provide the following upper bounds on L2 norm of $\phi(\mathcal{I})(\mathbf{x})$,

$$\text{for a gray scale image}, \forall \mathbf{x} \in \mathcal{S} \|\phi(\mathcal{I})(\mathbf{x})\|_{\mathrm{L2}}^2 \leq 12, \tag{4.24}$$

$$\text{for RGB image}, \forall \mathbf{x} \in \mathcal{S} \|\phi(\mathcal{I})(\mathbf{x})\|_{\mathrm{L2}}^2 \leq 14. \tag{4.25}$$

The claim follows by using Eq. 4.24, Eq. 4.25 in Eq. 4.23 □

Theorem 4.8.2 (Geodesic Radius of Covariance Descriptors).

1. If $\mathcal{I}$ is a gray scale image, then $\|\operatorname{Logm} \mathsf{R}_\eta(\mathcal{I})\|_F \leq \sqrt{9} \max\{|\ln \eta|, |\ln(12+\eta)|\}$.

2. If $\mathcal{I}$ is a RGB image, then $\|\operatorname{Logm} \mathsf{R}_\eta(\mathcal{I})\|_F \leq \sqrt{11} \max\{|\ln \eta|, |\ln(14+\eta)|\}$.

Proof. We first note that

$$\begin{aligned}\lambda_{\min}(\mathsf{R}_\eta(\mathcal{I})) &= \lambda_{\min}\left[\frac{1}{|\mathcal{S}|}\sum_{\mathbf{x}\in S}(\phi(\mathcal{I})(\mathbf{x})-\mu)(\phi(\mathcal{I})(\mathbf{x})-\mu)^T+\eta.I\right] \\ &\overset{(1)}{\geq} \lambda_{\min}\left[\frac{1}{|\mathcal{S}|}\sum_{\mathbf{x}\in S}(\phi(\mathcal{I})(\mathbf{x})-\mu)(\phi(\mathcal{I})(\mathbf{x})-\mu)^T\right]+\lambda_{\min}[\eta I] \\ &\overset{(2)}{\geq} 0+\eta. \end{aligned} \tag{4.26}$$

where we used Weyl's inequality for symmetric matrices in (1) and $\lambda_{\min}$ of positive semi definite matrix is ≥ 0 and $\lambda_{\min}[\eta I]=\eta$ in (2).

For gray scale images, $\mathsf{R}_\eta(\mathcal{I})$ produces a 9×9 matrix:

$$\begin{aligned}\|\text{Logm}\,\mathsf{R}_\eta(\mathcal{I})\|_F &\overset{(*)}{\leq} \sqrt{9}\max\{|\ln\ell|,|\ln L|\} \\ &\overset{(**)}{=} \sqrt{9}\max\{|\ln\eta|,|\ln(12+\eta)|\},\end{aligned}$$

where we use Lemma. 4.8.3 in $(*)$ and Eq.4.26($\ell=\eta$) and Lemma. 4.8.4($L=12+\eta$) in $(**)$

For RGB images, $\mathsf{R}_\eta(\mathcal{I})$ produces a 11×11 matrix:

$$\begin{aligned}\|\text{Logm}\,\mathsf{R}_\eta(\mathcal{I})\|_F &\overset{(\dagger)}{\leq} \sqrt{11}\max\{|\ln\ell|,|\ln L|\} \\ &\overset{(\ddagger)}{=} \sqrt{11}\max\{|\ln\eta|,|\ln(14+\eta)|\},\end{aligned}$$

where we use Lemma. 4.8.3 in $(\dagger)$ and Eq.4.26 ($\ell=\eta$) and Lemma. 4.8.4($L=14+\eta$) in $(\ddagger)$, in a similar fashion. □

Note that in all of our experiments, we choose $\eta = 10^{-6}$ and hence $|\ln \eta| \approx 13.8$ dominates over $|\ln(12+\eta)| \approx 2.5$ and $|\ln(14+\eta)| \approx 2.6$.

4.9 Experiments

All experiments were run on Dell XPS 17 9710 Laptop which has 32GB of RAM, 11th Gen Intel(R) Core i9-11900H @ 2.50GHz Processor. No GPUs were used in the experiments.

4.9.1 Implementation Details

Let $k \in \mathbb{N}$, $M \in \text{SPD}(k)$, $\sigma > 0$ and ρ_{LE} denote log-Euclidean distance. The Riemannian Laplace distribution with log-Euclidean distance is given by

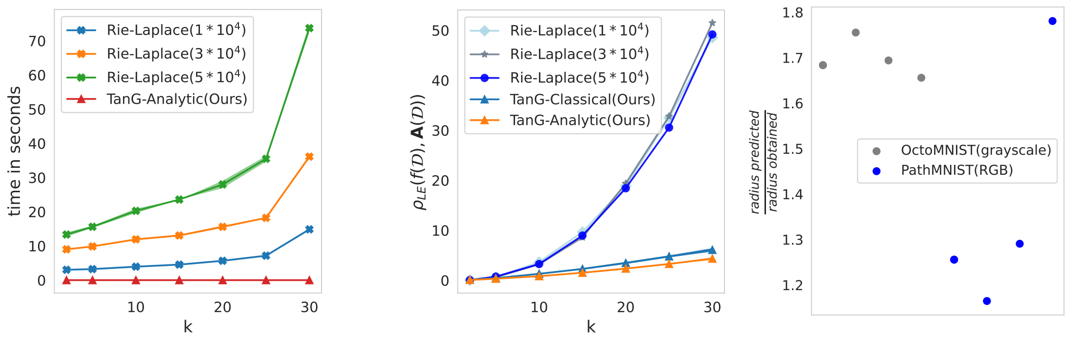
$$p(X|M,\sigma) = \frac{1}{\mathcal{C}_{M,\sigma}} \exp\left(-\frac{\rho_{\text{LE}}(X,M)}{\sigma}\right). \tag{4.27}$$

Note that sampling from Eq. equation 4.27 requires Markov Chain Monte Carlo (MCMC) methods[432], for which one needs to choose a proposal distribution that generates candidates on the SPD Manifold. We chose the Log Gaussian distribution as the proposal in our experiments given its simplicity and the fact that it is quick to sample from. In all experiments, we found that using the log Gaussian distribution as a proposal yields a stable acceptance ratio of 50% to 65%. To summarize,

1. Initialize X_{curr} at a random point of the manifold SPD(k).

2. For $1 \to n$ iterations

   (a) Draw a candidate from $X \sim \mathcal{LN}(X_{\text{curr}}, \sigma^2 I)$.



(a) **(b)** **(c)**

Figure 4.4: (a) Computational times for *Rie-Laplace*(x) the Riemannian Laplace mechanism with a MCMC burn-in of $x \in \{10,000; 30,000; 50,000\}$[425], and *TanG-Analytic* the proposed tangent Gaussian mechanism (analytic version). (b) Utility with varying burn-ins for *Rie-Laplace*. Plots (a, b) use different matrix sizes k. Plot (c) explores if the bound from Th. 11 is tight in practice.

(b) With probability $\exp(-\rho_{\text{LE}}(X_{\text{mean}}, X)/\sigma)/\exp(-\rho_{\text{LE}}(X_{\text{curr}}, X)/\sigma)$ accept the generated candidate X and set $X_{\text{curr}} = X$.

The final sample is chosen based on a burn-in period of 50,000 steps.

4.9.2 Additional Experiments

We compare the times required to privatize the Fréchet mean using both mechanisms and varying $k \in \{2, 5, 10, 15, 20, 25, 30\}$ in Fig. 4.4(a). Note that we used MCMC for Riemannian Laplace, and its time depends on the burn-in - that we choose in $\{10000, 30000, 50000\}$. For $k = 30$, Fig. 4.4(a) shows that Riemannian Laplace mechanism takes 14 sec (burn-in 10000), 36 sec (burn-in 30000) and 73 sec (burn-in 50000) - whereas our tangent Gaussian (Analytic) mechanism takes 1.3 *microsec*. Fig. 4.4(b) considers the effect of the burn-in on the Riemannian Laplace's utility and finds no significant difference for burn-ins in $\{10000, 30000, 50000\}$.

Fig. 4.4 (c) shows that the bound derived in Th. 11 is tight in practice, as illustrated by the ratio of the bound obtained in Th.11 and the practical bound.

4.9.3 Code

The code is attached with Supplementary Material. Both Riemannian Laplace and tangent Gaussian mechanisms can be easily implemented using existing libraries like `geomstats`[357], `tensorflow-riemopt`[480], `rieoptax`[525]. In all our experiments, we used `geomstats`[357].

"The combination of some data and an aching desire for an answer does not ensure that a reasonable answer can be extracted from a given body of data."

John Tukey

5

Variance-reduction with meta-estimation of private sketch data structures

5.1 Introduction

Distributed applications involving multiple client entities often have stringent privacy requirements that are governed by legal regulations such as HIPAA[354], GDPR[198], and PIPEDA[38].

Such requirements are also necessitated by individual preferences, ethical guidelines, national security interests, and enabling partnerships in a rapidly globalizing society. One such societal application that has recently come under the spotlight of privacy researchers, given the global advent of the recent COVID-19 pandemic, is that of private digital contact tracing and exposure notification [430,502,95,344,419,175,216,12,115,424,21,203]. As shown in Figure 5.1, this refers to the problem of privately ascertaining whether a querying client has come into close proximity of an infected patient to privately notify the querying user with an obtained result. Currently, cryptographic methods and differential privacy[160,154,351,161] is one of the widely accepted mathematical notions of formal privacy with varying levels of adoption for different applications. For example, the next U.S census[5,4] is being privatized via differential privacy while cryptographic techniques power several end-to-end encrypted messaging platforms. With respect to private digital contact tracing, several apps have recently been released. They are currently based on cryptographic schemes[430,95,55,513,478,53,106,327] such as secure multi-party computation (secure MPC), homomorphic encryption and public-key cryptosystems. In this paper, we propose one of the earliest solutions (to the best of our knowledge) for contact tracing that is instead based on differential privacy. This chapter is based on our work in[543].

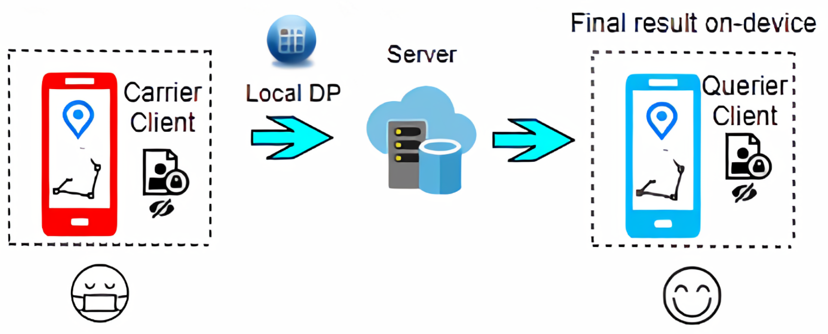
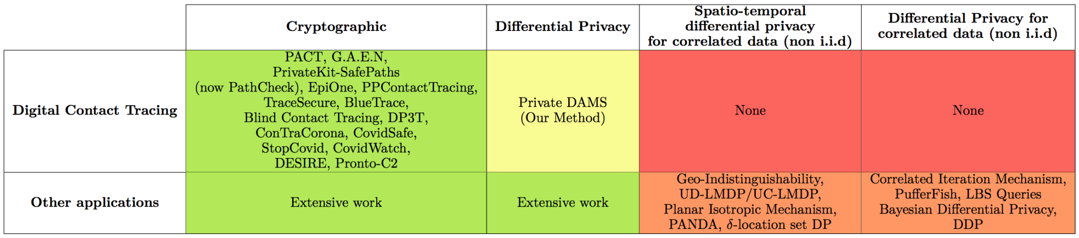


Figure 5.1: Private contact tracing refers to the problem of privately ascertaining whether a querying client has come into close proximity to any patient that is an infected carrier. Our proposed scheme involves a one-way upload of locally differentially private (local DP) information into the server that is downloaded by any client performing contact tracing.



	Cryptographic	Differential Privacy	Spatio-temporal differential privacy for correlated data (non i.i.d)	Differential Privacy for correlated data (non i.i.d)
Digital Contact Tracing	PACT, G.A.E.N, PrivateKit-SafePaths (now PathCheck), EpiOne, PPContactTracing, TraceSecure, BlueTrace, Blind Contact Tracing, DP3T, ConTraCorona, CovidSafe, StopCovid, CovidWatch, DESIRE, Pronto-C2	Private DAMS (Our Method)	None	None
Other applications	Extensive work	Extensive work	Geo-Indistinguishability, UD-LMDP/UC-LMDP, Planar Isotropic Mechanism, PANDA, δ-location set DP	Correlated Iteration Mechanism, PufferFish, LBS Queries Bayesian Differential Privacy, DDP

Figure 5.2: We compare and categorize the proposed method within the current landscape of works on the private digital contact tracing problem. Green refers to solutions that are already deployed or in an advanced stage of development. Red refers to methods that are non-existent (referred by none), or not deployed within the context of private digital contact tracing. Pastel yellow refers to our proposed method that is currently a prototype that has gone beyond the research stage as we plan to engineer it towards a controlled deployment while we move on to create, adopt, or build upon works currently in red for the contact tracing problem as part of future research. The red areas under other applications are very promising but need accelerated research for adopting them within the context of private digital contact tracing. The orange areas refer to differential privacy methods for non-i.i.d spatio-temporal data that exist but have not been adapted yet for contact tracing applications.

Works such as[206] have shown that differentially private technologies can drastically reduce the computational and communication costs of large-scale systems compared to cryptographic technologies, albeit at a weaker trade-off with privacy. A recent trend has been to build systems that depend on both differential privacy and cryptographic technologies at the same time[551,117] for better performance guarantees. Therefore, having a differentially private solution to contact tracing can have a downstream benefit from such efforts as well. All of our codebases will be made available as described in the ethics statement.

5.1.1 Contributions

1. We propose the first differentially private solution to COVID-19 contact tracing using sketching data structures.

2. We propose a new meta-estimator (DAMS) based on the private count-min sketch data structure and apply it to private digital contact tracing. We evaluate its performance

over important baselines on multiple real-life trajectory datasets of human mobility with respect to the classification metrics of private digital contact tracing. We empirically show that our meta-estimator performs at a drastically higher true positive rate (TPR) with a relatively much lower false positive rate (FPR) in comparison to these baselines.

3. We theoretically show that our meta-estimator (Private DAMS) is unbiased and has a lower variance than that of private count-mean-sketch (PCMS).

5.2 Related work

We categorize works related to this paper into three categories of: *private digital contact tracing, local differential privacy, and private sketching methods*.

5.2.1 Private digital contact tracing methods

There has been a rapid flurry of mobile apps released globally for digital contact tracing with varying levels of privacy protection. Within this space, a majority of deployed solutions or the ones that are undergoing rapid refinements are cryptography-based as categorized in Figure 2.

Differential privacy has been another popular approach for formal privacy. For example, it is being used to privatize the 2020 U.S census[5,4] that is currently underway. There has not been much work at the intersection of differential privacy and contact tracing as yet, as shown in this table. DAMS[543] for private contact tracing is instead based on differential privacy to help further the research on private digital contact tracing from a different viewpoint.

5.2.2 Local differential privacy

We employ the local differential privacy setting[126,127,258,267], where privacy is maintained locally at the client level. In this version, a privatized dataset is released from a client, and post-processing is applied remotely over this privatized dataset on a server or another client in order to complete analysis/model training/inference over that dataset. A weaker yet relatively similar setting to local differential privacy is called 'non-interactive private data release'[96,481]. The key difference is that in local differential privacy, each data owner, for e.g. an individual iPhone user, privatizes his/her data before sending it out for any post-processing as opposed to non-interactive differential privacy that requires a trusted centralized unit that sees the original data (not the privatized version); for e.g. everyones keyboard input data. Then, the trusted centralized unit privatizes the data before releasing it to the public.

5.2.3 Sketching methods

Sketching methods are popular for streaming data analysis, efficient information retrieval, and large-scale machine learning. These techniques typically involve a dictionary of multiple hash functions used to hash the dataset into a table or data structure. In order to obtain the solution to any specific query, such as frequency estimation, inner-product search, or range estimation, a post-processing function corresponding to that particular query is applied to the data structure in order to obtain the result efficiently. Bloom filters[78] is one of the earliest such randomized data structures. Other examples of sketching methods[243,77,23,413] include Hadamard sketch[505], Broder's Sketch[79], MinHash[476,244], AMS Sketch[20] and Count-Min-Sketch[125,128]. Differential private versions of some of these sketching methods like[505,169] exist. We modify this private count-mean-sketch data structure to obtain a better trade-off in terms of the true positive rate (TPR) and false positive rate (FPR) upon testing it on contact tracing use cases.

5.3 Preliminaries

Notation: The notation used in this paper is summarized for ease of reference in Table 5.1.

Depth	k
Width	m
Hash Dictionaries	$\mathcal{H}_1, \mathcal{H}_2, \ldots, \mathcal{H}_p$
Hash Functions	$h_1, h_2, \ldots, h_k$
Privacy Parameter	ϵ
Dataset Size (# of records)	n
Dataset	$D_n = \{d_1, d_2, \ldots d_n\}$
Hash Output	$v \in \mathbb{R}^m$
Post-Processing Function	ϕ
Count Estimator	$\tilde{f}(d)$
Bernoulli Noise Vector	$b \in \{-1, +1, \}^m$
# of Clients	w
Histograms	$F_1, F_2 \ldots F_p$
Sketch Matrix	$\mathbf{M}$

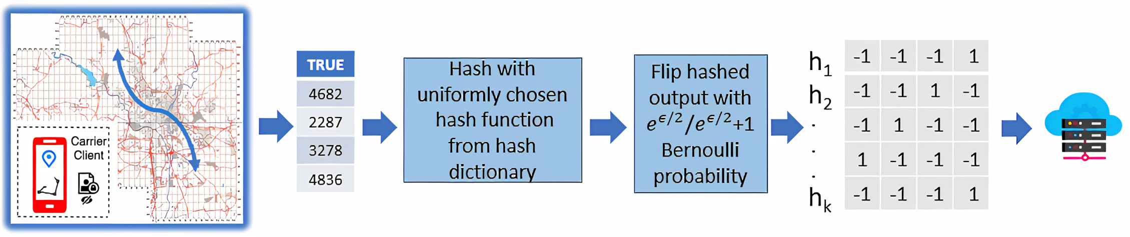
Table 5.1: This is the notation used in this paper.

Definition 2 (ϵ-Local Randomizer[160]). *Let $A : D \mapsto Y$ be a randomized algorithm mapping a data entry in data domain D to Y. The algorithm A is an ϵ-local randomizer if for all data entries $d, d' \in D$ and all outputs $y \in Y$, we have $-\epsilon \leq \ln\left(\frac{\Pr[A(d)=y]}{\Pr[A(d')=y]}\right) \leq \epsilon$.*

Definition 3 (Local Differential Privacy[160,505]). *Let $A : D_n \mapsto Z$ be a randomized algorithm mapping a dataset with n records to some arbitrary range Z. The algorithm A is ϵ-local differentially private if it can be written as $A(d_1, \ldots, d_n) = \phi(A_1(d_1), \ldots, A_n(d_n))$ where each $A_i : D \mapsto Y$ is an ϵ- local randomizer for each $i \in [n]$ and $\phi : Y_n \mapsto Z$ is some post-processing function of the privatized records $A_1(d_1), \ldots, A_n(d_n)$. Note that the post-processing function does not have access to the raw data records.*

5.3.1 Private count-mean-sketch

The work in[505] provides a locally differentially private mechanism called private count mean sketch (PCMS) for privately releasing histograms. It is based on a non-private version of this


Carrier: Client
TRUE
4682
2287
3278
4836
Hash with uniformly chosen hash function from hash dictionary
Flip hashed output with $e^{\epsilon/2}/e^{\epsilon/2}+1$ Bernoulli probability
h_1
h_2
h_k

h_1	-1	-1	-1	1
h_2	-1	-1	1	-1
.	-1	1	-1	-1
.	1	-1	-1	-1
h_k	-1	-1	-1	1

Figure 5.3: Sketch of the local DP scheme for standard private count-mean-sketch. Note that this is the client side of the scheme that is prior to applying the server's post-processing function. We contrast this with our proposed meta-estimator in Figure 3.

data structure (CMS) in[125]. PCMS has a client-side algorithm and a server-side algorithm. The client-side algorithm ensures the data that leaves the users device is ϵ-local differentially private. In PCMS, local differential privacy is achieved on a client via flipping the bits of any output v of a hash function applied to a data record d with a Bernoulli noise vector $b \in \{-1, +1\}^m$, whose elements are picked with a probability of $\frac{e^{\epsilon/2}}{e^{\epsilon/2}+1}$. This noised output is stored in a matrix of dimension $k \times m$ called the sketch matrix. Here, m (referred to as depth) is the dimension of the output of used hash functions, and k is the number of hash functions. A post-processing is applied on this table at the server to obtain the private histogram as follows. As noised vectors arrive from various clients, the server adds the privatized vector to the vector at row j of a server-side version of the sketch matrix $\mathbf{M}$, where j is the index of the hash function sampled by the device. The values of $\mathbf{M}$ are then scaled appropriately so that each row helps provide an unbiased estimator for the frequency of each element. To compute an estimate for any input $d \in D$, the server-side algorithm then averages the counts corresponding to each of the k hash functions in $\mathbf{M}$ for d.

5.4 Method

In this section, we propose a meta-estimator as an improvement to the private count-mean-sketch data structure in order to achieve a better trade-off between the true positive rate and false positive rates when applied to the problem of private contact tracing. In addition, we show theoretical results that our meta-estimator provides a variance reduction in comparison to the original private count-mean-sketch data structure-based estimator, and we substantiate this via empirical results as well. Before we describe the technical aspects of our proposed meta-estimator, we walk through a detailed example of any user's interaction with our proposed private contact tracing system.

5.4.1 Example roles of querying clients, infected clients, and server in the proposed system

- **Infected clients:** Infected clients upload a locally differentially private version of their trajectories of movement to a centralized server. To be precise, all clients share an indexing to a spatial grid overlaid on the map. There are efficient ways to maintain such a global grid indexing using technologies like geohashes (for square grid cells) or H3 geospatial grid indexing (for hexagonal grid cells). Every trajectory is represented by indexing corresponding to a discretized version of the trajectory to several grid cells. The set of indexes corresponding to each category is privatized using our proposed meta-estimator and shared with the server.

- **Server:** The server applies a post-processing function to obtain a locally differentially private histogram of counts of grid-cell indices traversed by all trajectories of a client within a chosen time window. It sums up all such private histograms obtained from each of the w infected clients to obtain a single private aggregated histogram.

Algorithm 1 DAMS

```
for Infected Clients $t \in [w]$ do
    For each $r \in [p]$,
compute $g_t^r$ = PrivateClientCMS($D_t, \epsilon_t, \mathcal{H}_r$) on-device.
    Send sketches $g_t^1, g_t^2, \ldots g_t^p$ to Server
ServerUpdate($g_t^1, g_t^2, \ldots g_t^p$):
for $t \in [w]$ do
    Estimate histograms $F_t^1 = \phi(g_t^1), \ldots F_t^p = \phi(g_t^p)$
    Compute average histogram $F_t^\mu = \frac{\sum_{i=1}^p F_t^i}{p}$
Compute aggregate of average histograms
$F_{agg} = F_1^\mu + F_2^\mu + \ldots F_w^\mu$
Send $F_{agg}$ to QuerierClient
QueryClient Check:
Matches its data with non-zero counts in $F_{agg}$ greater than a threshold for contact tracing result
```

Figure 5.4: DAMS Algorithm

- **Querier clients:** Any querier client would like to check if it came into contact with an infected client up to the resolution allowed by the grid cells. It downloads the private aggregated histogram from the server onto its device, matches it with its own trajectory data, and looks for counts beyond a threshold while also accounting for its own repeat visits.

5.4.2 Meta-estimator: Diversified averaging for meta-estimation of private sketches (DAMS)

We now describe our DAMS scheme. The steps can be summarized as follows.

- **Step 1:** Every infected client generates p private sketches of their raw data, where each version (or run) differs in terms of the dictionary of hash functions $\mathcal{H}_i$ used. Each private sketch is done using the private count-mean-sketch estimator. These p private sketches per client are sent to the server. Note that in addition, we also divide the spatio-temporal region under study into several large zones, where each zone has its own hash dictionary that changes from run to run. This helps filter and quantize the data record down to a zone before using its hash dictionary.

- **Step 2:** The server applies its post-processing function on each of these private sketches to generate a private histogram. These p private histograms are averaged to get a final private histogram per client. Since there are w clients, a total of w private histograms are obtained at the server. The server now adds these w histograms to obtain one aggregated private histogram.

- **Step 3:** This aggregated histogram is downloaded by any querying client that would like to check if it has come into contact (close proximity) with an infected client. The querying client checks if any of its movement trajectories match with the non-zero counts in the aggregated histogram beyond a threshold of counts after accounting for its own repeat visits. This helps the querying client obtain the final result of contact tracing on-device.

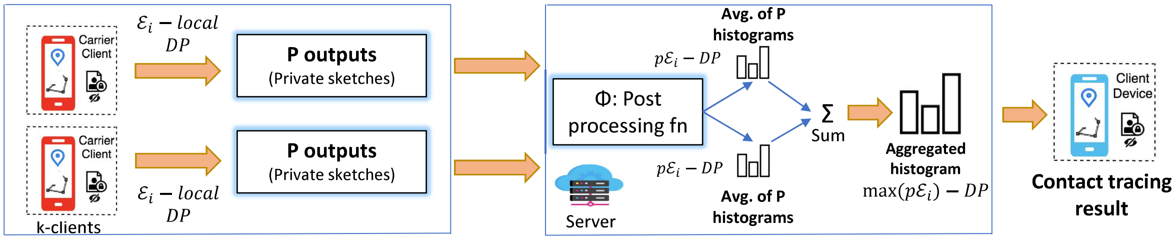


Figure 5.5: Illustration of our proposed meta-estimation scheme where each infected client device performs p sketches of its data using the private count-mean-sketch data structure, where each sketch is performed with a completely different dictionary of hash functions. The p intermediate result obtained from each client is said to the server, where they are post-processed to obtain p private histograms that are averaged to finally obtain one histogram per client. These are all aggregated to obtain one single histogram that is shared with the querying user for matching with its own data on-device to get the result of contact tracing.

These steps are presented in the Figure 5.5 presented above. Although we show empirically in the experimental section that our modified scheme improves the true positive rate of contact tracing while substantially reducing the false positive rate, it goes without saying that there is no free lunch. The trade-off of this increased utility happens at a reduction in privacy, precisely to the extent that we now describe. That said, we show that the constants that influence this utility-privacy trade-off are reasonably under control in empirical experiments. In step 2 above, if every client releases each one of the p histograms with ϵ_i- differential privacy, then due to the sequential composition property[160] of differential privacy, each averaged histogram from every client has $p\epsilon_i$- differential privacy. Similarly, due to the parallel composition property[160] of differential privacy, the aggregated private histogram has $max(p\epsilon_i)$-differential privacy, $\forall i \in \{1, 2, \dots, w\}$.

5.4.3 Variance reduction guarantees and important baselines

We now compare the variance under the following three scenarios

- **Scenario I** The scenario of using $\epsilon = p\epsilon'$ with the algorithm being run once with one set of hash functions. This is equivalent to the privacy level obtained when the same set of hash functions are used across p runs of the algorithm on the same dataset due to the sequential composition property[160] of differential privacy. This is an important baseline to compare against in order to confirm that changing the hash function dictionary across multiple runs (# of runs = p) is a better option than performing one single run with one hash function dictionary, yet with an equivalent level of privacy. We would like to note that, even when $p = 1$, there is a difference in hash dictionaries used across different zones that the region of interest is divided into, as explained in Step 1 of our method in the previous section.

- **Scenario II** The scenario of using $\epsilon = \epsilon'$, while the algorithm is run q times using a ***same*** dictionary of hash functions in each of the run as part of the private count-mean-sketch algorithm. The final result is obtained as the average of the estimate counts. This is an important baseline to compare against in order to confirm that changing the hash function dictionary across each of the p runs is a better option than keeping them same across the p runs.

- **Scenario III** The scenario of using $\epsilon = \epsilon'$, while the algorithm is run q times using a ***different*** dictionary of hash functions in each of the run as part of the private count-mean-sketch algorithm. The final result is obtained as the average of the estimated counts. We refer to this option as our proposed private DAMS estimator.

Theorem 5.4.1. The private DAMS estimator in scenario III has a lower variance than the estimator in scenario I when $\epsilon > 2$.

Proof. $\tilde{f}(d)$ is the estimated frequency of data element d and its variance in the standard differentially private count-mean-sketch scheme is given by[505].

$$\begin{aligned} \mathrm{Var}[\tilde{f}(d)] = n(c_\epsilon^2 - 1)/4 + \frac{n - f(d)}{m}\left(1 - \frac{1}{m} - \frac{1}{k} + \frac{1}{km}\right) \\ + \left(\frac{1}{km} - \frac{1}{km^2}\right)\left(\sum_{d^* \neq d} f(d^*)^2\right) \end{aligned} \tag{5.1}$$

Here $f(d^*) \in D$ is the original frequency of the element d^* and D is the dataset, n is the number of data points, k is the depth of the CMS-data structure, m is the width and $c_\epsilon = \frac{e^{\epsilon/2+1}}{e^{\epsilon/2}-1}$.

We now show the variance of the count estimator obtained in each of the above estimators. In scenario I, we have the following expression for the variance up to a constant C that is independent of ϵ.

$$\begin{aligned}\operatorname{Var}[\tilde{f}(d)] &= n(c_{p\epsilon'}^2 - 1)/4 + \frac{n - f(d)}{m}\left(1 - \frac{1}{m} - \frac{1}{k} + \frac{1}{km}\right) \\ &+ \left(\frac{1}{km} - \frac{1}{km^2}\right)\left(\sum_{d^* \neq d} f(d^*)^2\right) \qquad (5.2) \\ &= n(c_{p\epsilon'}^2 - 1)/4 + C\end{aligned}$$

In scenario III, since all the hash functions across the p runs are three-wise independent, we have $\operatorname{Var}\tilde{f}(d) = \frac{\operatorname{Var}[\tilde{f}_1(d)] + \operatorname{Var}[\tilde{f}_2(d)] + \ldots + \operatorname{Var}[\tilde{f}_p(d)]}{p^2}$ where $\operatorname{Var}\tilde{f}_i(d)$ is the variance of an individual run. But since we use the same k, n, m across runs although the hash function dictionaries are the same, we have

$$\operatorname{Var}\tilde{f}(d) = n(c_{\epsilon'}^2 - 1)/4 + C$$

Note that there is a reduction from $1/p^2$ to $1/p$ due to equality of variances. To complete the proof, we would need to show that

$$\frac{1}{p}\left[n(c_{p\epsilon'}^2 - 1)/4 + C\right] \leq n(c_{p\epsilon'}^2 - 1)/4 + C$$

Substituting $c_\epsilon = \frac{e^{\epsilon/2}+1}{e^{\epsilon/2}-1} = 1 + \frac{2}{e^{\epsilon/2}-1}$, we would need to show that

$$\frac{1}{p}\left[\frac{n}{4}\left[1 + \frac{2}{e^{\epsilon/2} - 1}\right]^2 + C\right] \leq \frac{n}{4}\left[1 + \frac{2}{e^{p\epsilon/2} - 1}\right]^2 + C.$$

For $\epsilon' > 2$, we have c_ϵ is approximately ≤ 4 and therefore

$$\frac{1}{p}\left[\frac{n}{4}\left[1 + \frac{2}{e^{\epsilon/2} - 1}\right]^2 + C\right] \leq \frac{1}{p}[4n + C]$$

Upon substituting the same into the r.h.s of the inequality we get

$$\frac{1}{p}[4n+C] \leq \frac{n}{4} + C$$

that can be trivially satisfied by choosing values of p that satisfy this inequality. □

Lemma 5.4.1. *Private DAMS estimator is unbiased.*

Proof. $\mathbb{E}\,\tilde{f}(d) = \frac{\mathbb{E}[\tilde{f}_1(d)]+\mathbb{E}[\tilde{f}_2(d)]+\ldots+\mathbb{E}[\tilde{f}_p(d)]}{p}$ where $\mathbb{E}\,\tilde{f}_i(d)$ is the expectation of an individual run. Each of the individual estimators in the numerator is unbiased as the differentially private count-mean-sketch estimator that was used is unbiased[505]. Therefore, the private DAMS estimator is unbiased. □

Theorem 5.4.2. The variance of the estimator of private DAMS in scenario III is less than the variance of the estimator in scenario II.

Proof. In scenario II, without loss of generality, when $p = 2$ we have

$$\operatorname{Var}\tilde{f}(d) = \frac{\sum_i \operatorname{Var}[\tilde{f}_i(d)]}{p^2} + 2\sum_{ij}\operatorname{Cov}(\tilde{f}_i(d), \tilde{f}_j(d))$$

where $\operatorname{Var}\tilde{f}_i(d)$ is the variance of an individual run and $\operatorname{Cov}(\tilde{f}_i(d), \tilde{f}_j(d))$ is the covariance. Since, we use 3-wise independent hash functions[505] as suggested in the standard differentially private count-mean-sketch estimator in[505], the covariance when $i \neq j$ is $\frac{1}{m} - \frac{1}{m^2}$ while it is 0, when $i = j$. Now, $\frac{1}{m} - \frac{1}{m^2}$ is always positive for non-zero integer values of m. Therefore, all the covariances show a positive correlation in this case, and the sum of covariances is of the order

$$q(\frac{1}{m} - \frac{1}{m^2})(2p-1)$$

Therefore, the variance of scenario III is always lesser than scenario II. Without loss of generality, this holds when $p > 2$ as well. □

5.5 Experiments

5.5.1 Microsoft GeoLife GPS Trajectory Dataset

This GPS trajectory dataset[606,607,608] is a massive dataset that was collected in (Microsoft Research Asia) Geolife project by 178 users in a period of over four years (from April 2007 to October 2011). A GPS trajectory of this dataset is represented by a sequence of time-stamped points, each of which contains information on latitude, longitude, and altitude. This dataset contains 17621 trajectories with a total distance of 1,251,654 kilometers and a total duration of 48,203 hours. A subset of this dataset was used for a detailed evaluation with 50 trajectories labeled as infected patient trajectories, and one was labeled as a querier trajectory. Each trajectory was of length 720. Therefore, $51 \times 720 = 36720$ datapoints were used to be processed through our private DAMS data structure.

5.5.2 GoTrack GPS trajectories dataset

This dataset is available on the UCI repository. We use a formatted subset of the dataset where querier trajectories intersect with some infected patient trajectories in 336 co-ordinates among 1123 co-ordinates. Unlike the above experiment, the trajectory length of each participant is not the same in this dataset.

5.5.3 Empirical evaluation

Private DAMS Vs. PCMS: We compare our approach of private DAMS with $p = 1$, where each zone has a different hash dictionary Vs. with the standard private count-mean-sketch

(PCMS-1 as in scenario I) with $p = 1$ as shown in Figures 4.6 and 4.9. In PCMS, each zone has the same dictionary. The comparison is in terms of the important metrics of true positive rates (TPR), false positive rates (FPR), F1 score, and MCC score of contact tracing received at querier client, with respect to the ground truth of intersections. Note that the x-axis refers to the different values of ϵ considered between 2.5 to 7, with increasing values of 0.5. That said, it is important to note that all the $\epsilon' s$ reported on the x-axis are the corresponding values obtained after accounting for the sequential and parallel composition laws of differential privacy in our scheme as described in Figure 5.5.

Effectiveness of $p > 1$ in DAMS: We also compare our approach private DAMS with $p = 5$ (DAMS-5) and $p = 10$ (DAMS-10) against scenario - II for $p = 5$ (CMS-5) and $p = 10$ (CMS-10) runs. These results are shown in Figures 4.7, 4.8, 4.10 and 4.11. We observe a greater TPR in each of the DAMS results in comparison to the CMS results, as desired. Similarly, we observe a lower FPR in each of the DAMS results in comparison to the CMS results as desired.

Variance reduction with DAMS: In addition, we observe that the variance across the obtained FPRs is significantly lower in the DAMS results in comparison to the CMS results, although the change in variances in the case of the TPRs is not as significant. We note that the denominator in computing the FPRs is way larger than that of the denominator in computing TPR over this dataset. Therefore, the overall variance reduction is significant. In Figure 4.12, we compare the effect of increasing p over our proposed DAMS scheme. We note that the TPR increases with increasing p, although the increase begins to flatten out with larger p's. That said, with increasing p, the FPRs mildly increase in the DAMS scheme, as shown in Figure 4.15. Note that regardless of this effect, the TPRs and FPRs of DAMS outperform CMS for all three p's that were tried, as in for $p = 1, p = 5$, and $p = 10$.

Data imbalance As the datasets are highly imbalanced (and so is the use case of contact tracing), in terms of having a much smaller number of intersections as against the number of non-

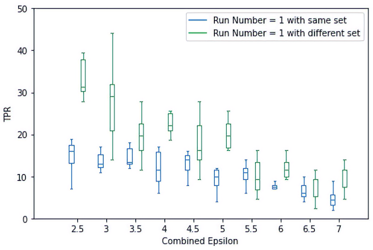


Figure 5.6: GeoLife GPS: TPR of DAMS Vs. PCMS for $p = 1$

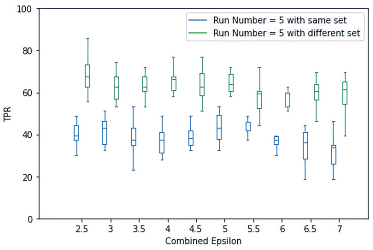


Figure 5.7: GeoLife GPS: TPR of DAMS Vs. PCMS for $p = 5$

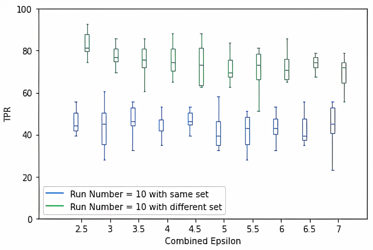


Figure 5.8: GeoLife GPS: TPR of DAMS Vs. PCMS for $p = 10$

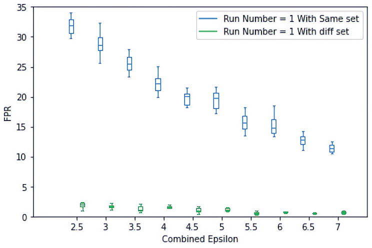


Figure 5.9: GeoTrack GPS: FPR of DAMS Vs. PCMS for $p = 1$

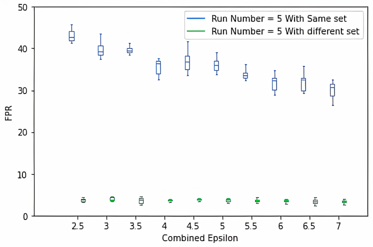


Figure 5.10: GeoTrack GPS: FPR of DAMS Vs. PCMS for $p = 5$

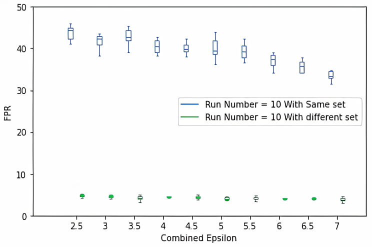


Figure 5.11: GeoTrack GPS: FPR of DAMS Vs. PCMS for $p = 10$

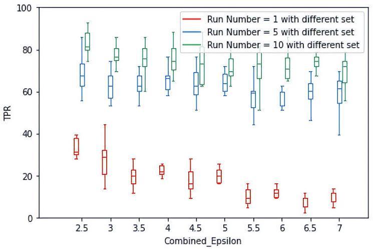


Figure 5.12: GeoLife GPS: TPR trend in DAMS for $p = 1, 5$ and 10

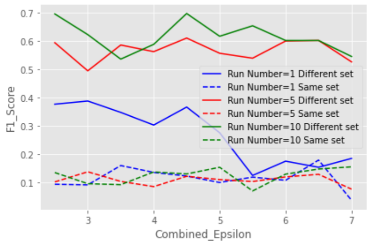


Figure 5.13: GeoLife GPS: F1 score across DAMS and PCMS for $p = 1, 5$ and 10

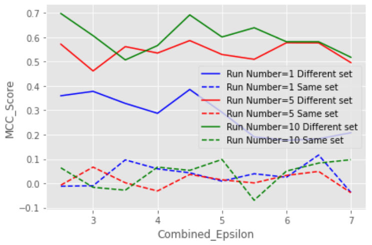


Figure 5.14: GeoLife GPS: MCC score across DAMS and PCMS for $p = 1, 5$ and 10

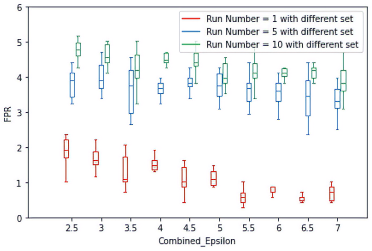


Figure 5.15: GeoTrack GPS: FPR trend in DAMS for $p = 1, 5$ and 10

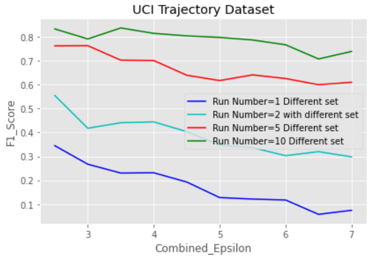


Figure 5.16: GeoTrack GPS: F1 score across DAMS and PCMS for $p = 1, 5$ and 10

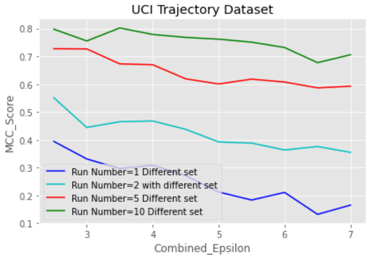


Figure 5.17: GeoTrack GPS: MCC score across DAMS and PCMS for $p = 1, 5$ and 10

intersections between the trajectories of querier clients and infected clients, we therefore also compare the different versions of DAMS and CMS in terms of F1 scores and MCC scores, that are better suited for such settings. These results are presented in Figures 4.13, 4.14, 4.16 and 4.17.

5.6 Future research

As part of suggested future work, we give credence to the non-i.i.d (non-independent and identically distributed) nature of the problem in contact tracing, as our proposed solution could be further improved using differential privacy primitives that are well-suited for dependent/correlated data. These notions of modified differential privacy for non-i.i.d data[101,323,24] are currently at an early stage of the research horizon. We believe that first investigating the digital private contact tracing problem through the lens of differential privacy under the relatively simpler assumption of i.i.d data is beneficial to carry forward the learnings obtained into the more stringent settings of non-i.i.d data, as shown in Table 1 in red. Other location-based COVID-19 privacy projects, such as[426] by Facebook, also assume i.i.d'ness to support solutions with simplistic assumptions at first.

"The idea of concentration of measure is arguably one of the great ideas of analysis in our times."

Michel Talagrand

6

Effects of Privacy on Welfare and Influence of Referendum Systems

6.1 Introduction

This chapter is based on our work in[171]. Differential privacy[159] provides a compelling privacy guarantee to ensure that the outcome of a query over any dataset is substantially not influenced by the presence or absence of an individual's record. This form of privacy has recently been studied in the context of social choice theory[462,305,217]. A predominant strategy to achieve differential privacy in general, even outside the context of social choice theory, is to introduce noise or some sort of randomization into the system. One of the issues that has been widely studied in this context of noising is the specific loss of accuracy in releasing the true output of the non-privatized query as caused by increasing levels of privacy preservation. This has been commonly referred to as the *privacy-accuracy* or *privacy-utility trade-off*. Recent work has involved the formalization of other trade-offs, such as the trade-off between privacy and fairness[132]. In this work, we analyze two other trade-offs. We show that introducing noise to privatize systems that aggregate the preferences of individuals may affect several other fundamental phenomena such as *influence* and *welfare*.

In this context, does an increase in the level of privacy for releasing the outputs of social choice functions increase or decrease the level of influence and welfare, and at what rate? In this paper, we mainly address this question in more precise terms and affirmatively answer that this relation is inversely proportional and shares specific corresponding rates for the popular *ρ-correlated randomized response* mechanism of privatization when used in a referendum setting with two candidates.

The noisy mechanism that we propose[170] and analyze with regard to influence and welfare in this paper is based on a simple coin-flipping perturbation of the input as follows. Let ρ be an exogenous constant in $[0,1]$ and let each original vote made in the ballot take a value of either 1 or -1. The randomized response records each original vote in the ballot as it is with a probability ρ while with probability $1-\rho$, it ignores the original vote. Instead, it records it as either a 1 or -1 with a uniformly random pick. The resulting probability space is known as *ρ-correlated distribution* or *noisy distribution* in the field of analysis of Boolean functions, and it is referred to as the *randomized response* mechanism in the field of differential privacy. * We show that this mechanism preserves ordinal relations between the influences of voters for 'any' social choice function. Therefore, if Alice had more influence before than Bob, she would still continue to have more influence.

In the field of analysis of Boolean functions, the notion of the *influence* of a voter is used to measure the power of an individual on the final result of a social choice function. We extend this definition of influence to our probabilistic setting where noise is introduced for privacy and term this new notion of influence as *probabilistic influence*. Similarly, we define *welfare* to address the second issue of capturing how *ideal* a voting rule is. First, we define it for deterministic functions, and then we extend this definition to any probabilistic mechanism. We then show the effect of our privacy-inducing randomized response on the welfare of the system. In particular, we show that it preserves the ordinal relations between the welfare of voting systems. That is, if a social choice function f had greater welfare than g in the deterministic setting after the randomized response M_ρ is applied based on the exogenous parameter ρ, the welfare of $M_\rho f$ will continue to be greater than that of $M_\rho g$.

In this context, we share precise statements connecting the noising probabilities ρ used in the mechanism M_ρ, their effect on the level of privacy ϵ, which in turn results in a specific level of influence and welfare expressed in terms of ρ. We precisely show that as the level of privacy increases, the welfare and influence happen to decrease at correspondingly specific rates. Arguably, having a higher welfare in a voting system is desirable, and therefore, we shine a light on this new trade-off between privacy and welfare. In terms of influence, it is questionable whether a decrease in influence with an increase in privacy is desirable or not. We believe it depends on the context, and therefore, in this case, we do not refer to it as a trade-off

*For a survey of the field of analysis of Boolean functions, see[392]. For a survey of the field of differential privacy, see[160].

but instead call it a scaling law. However, as we show in Section 6.5, the welfare of the society is equal to the total influence of the society.

6.1.1 Contributions

We contribute towards bridging differential privacy and social choice theory by deriving the following results on the effect of randomized response over influence, welfare, and accuracy.

1. **The privacy-influence relationship:** A notion of *influence* is widely used in the analysis of Boolean functions to study social choice functions. We extend the notion of influence to the noisy setting and call it *probabilistic influence*. We then show a result relating the trade-off between $\rho-$correlated distribution-based differential privacy and probabilistic influence. We show that such privatization changes the influence of every single voter by a factor of $\frac{1+\rho^2}{2}$. Thus, the randomized response preserves the ordinal relations between influences of agents while scaling them by a factor depending on ρ while still ensuring their privacy is preserved.

2. **The Privacy-welfare trade-off:** We define *welfare* $W(f)$ of a social choice function f and extend the definition to probabilistic mechanisms. Then, we show that $W(M_\rho f) = \rho \cdot W(f)$, i.e. the randomized response scales the welfare by a factor of ρ, whereby preserving the ordinal relations between the welfare of social choice functions.

3. **Accuracy analysis:** We restrict the analysis of *accuracy* (or *utility*) of our mechanism to social choice functions, i.e. the functions with range $\{-1, 1\}$. We give the accuracy for Dictatorship, Majority, AND, and OR functions. For dictatorship, AND, and OR functions, we provide a theoretical analysis of accuracy. For the Majority function, we give an asymptotic accuracy when n goes to ∞ based on the existing results in the literature.

We also give an exact analysis of accuracy for the Majority function for small n by using a computational method that involves dynamic programming.

6.1.2 ORGANIZATION

The rest of the paper is organized as follows. In Section 6.2, we provide further motivation and background. In Section 6.3, we formally describe the differentially private randomized response mechanism. In Section 6.4, we introduce the notion of probabilistic influence and give one of our main results that influence scales down by the same constant for every individual. In Section 6.5, we introduce the concept of welfare for general probabilistic mechanisms and analyze it for randomized response. We shed light on the connection between influence and welfare and give our second main result, which is that randomized response scales down welfare by the same factor for any given social choice function. In Section 6.6, we provide an analysis of the accuracy of the randomized response mechanism. In Chapter 16, we discuss the possible future work and the limitations of this paper, and we conclude. Some preliminaries from social choice theory are provided in Section 6.9. All of the proofs are relegated to Section 6.8.

6.2 MOTIVATION

To intuitively expand on the potential relation between privacy and influence, consider an instance where it might be the case that the introduction of noise for the sake of obtaining privacy results in undesired shifts of the power held by different individuals in deciding the final selected outcome. For example, say that a voter, Alice, would have had more impact on the outcome than Bob in a case where there is no privatization. It could also be the case that the power balance shifts to Bob having more impact than Alice after a privacy-inducing noise is introduced. We conclusively show that this cannot be the case as the influence scales down for every voter with the increasing level of privacy by the same constant in the case of the popular randomized response privacy mechanism.

Secondly, regarding the potential relation between privacy and welfare, consider an instance where it may be the case that upon the introduction of noise, the chosen social choice function that was originally used to aggregate the individual preferences into an outcome ends up not being ideal anymore. Hence, it may instead be desirable to switch to another social choice function. For example, suppose that a system uses the majority function to decide which one of the two candidates is elected in the deterministic case. However, the majority function could be severely affected in some instances upon introduction of noise, and another function could end up being a *better* choice. We show that as privacy increases in the randomized response mechanism, the welfare of each social choice function scales down proportionally. This implies that if a function is a welfare maximizer before introducing noise, it still is a welfare maximizer after the introduction of the noisy mechanism. These two results are especially useful, as they imply that the designers of the initial deterministic social choice mechanism do not have to be concerned about whether their design is robust to the introduction of noise in terms of influence and welfare.

We now discuss the work that has been done regarding influence and welfare in the context of social choice theory. Influences have long been studied in discrete Fourier analysis and theoretical computer science. The notion of influence was first introduced by[408] and it was first systematically studied by[50]. Some other novel works related to influences in the context of social choice theory include, but are not limited to, KKL Theorem[263] and the Majority is Stablest Theorem[374]. We extend the notion of influence to the noisy setting and call it *probabilistic influence*, and prove a direct linear relation between deterministic influence and probabilistic influence.

The question of the ideal voting rule has long been a matter of discussion in social choice theory. When there are only two candidates, the answer is relatively simple, as the majority function seems to be the most ideal voting rule.[347] showed that majority is the only social choice function that is anonymous and monotone among all two-candidate voting rules. For more than two candidates, different objectives may result in different voting rules or even in impossibility results[31,32,208,190,196].[225] studies various aspects of utilitarian voting. Finding the best function in computationally efficient ways has been studied in the recent field of computational social choice theory. The works of[341] and[342] aim to maximize welfare given each voter's utility for candidates in a 'distortion framework' in which there is a lack of information about voter's utilities. In that framework, a typical approach is to attempt to maximize the worst-case objective.

To the best of our knowledge, a definition of welfare that is closest to ours is the one given by O'Donnell (2014, page 51). Although they do not explicitly define the welfare of a social choice function, there is a linear relation between the expected value of their objective function and the way we define welfare. However, our main conceptual contribution is that our definitions are extended to hold for probabilistic mechanisms, and we analyze the effects of privacy on influence and welfare.[392] proved that among all two-candidate voting rules, the majority is the unique maximizer of welfare, whose proof is essentially based on[509]. Our main objective is not to find the function that maximizes the welfare; that is rather a simple question. In fact, we show that the majority is the unique welfare maximizer as well in an almost identical way to O'Donnell. The primary motivation of the paper is to show that if a voting rule is better in the deterministic setting, it is still better after the privacy-inducing noise is introduced.

6.3 Model: Randomized Response and Privacy Guarantee

There are three main reasons why we chose the randomized response as the privacy-preserving mechanism to focus our attention. First, it is simple, in addition to being one of the earliest and yet one of the most popularly used privacy-preserving mechanisms to date, be it in the classic form or as a variant of it. As an example, RAPPOR[169] is a recent popular real-world use-case of randomized response, otherwise classically used a few decades ago[556,343]. Second, the mechanism is based on perturbations of the input, which allows it to be applied to 'any' social choice function. This enables us to talk about the ordinal relations between the welfare of potential social choice functions before and after the mechanism is applied. Third, ρ-correlated distributions are well studied in mathematical social choice theory[392].

Our randomized mechanism is an input-perturbing mechanism. That is, the mechanism introduces noise to the votes in the ballot so that one can use any social function afterward, yet the same privacy guarantee will continue to hold due to the post-processing property[156] of differential privacy. Randomized response introduces noise by utilizing a simple coin-flip scheme that is based on the following distribution that is widely used in the analysis of Boolean functions.

Definition 4. *Let $\rho \in [0,1]$ and $x \in \{-1,1\}^n$ be fixed. y is called ρ-*correlated with *x if for every $i \in [n]$, $y_i = x_i$ with probability ρ and uniformly distributed with probability $1-\rho$, and it is denoted by $y \sim N_\rho x$.*

Note the symmetry in the definition of ρ-correlation. We formalize this symmetry in the following facts, which we will often use in the proofs of our results.

Fact 1. *$x \sim \{-1,1\}^n, y \sim N_\rho x$ if and only if $y \sim \{-1,1\}^n, x \sim N_\rho y$. If $x \sim \{-1,1\}^n, y \sim N_\rho x$, we say (x, y) is a ρ-*correlated uniformly random pair.

In the literature, ρ-correlated distribution is sometimes referred to as *noisy distribution*. A famous analogy for this definition is as follows. Suppose the votes are recorded by a *noisy* machine. That is, the machine records each ballot correctly with probability ρ and blurs the ballot with probability $1-\rho$, and instead records it at uniform random. As a result, the vote gets misrecorded with probability $(1-\rho)/2$. In fact, our mechanism corresponds to this noisy machine. Hence, we will call it by the generic name *randomized response*, or *ρ-correlated randomized response* when we need to specify ρ and denote a mechanism that applies it by M_ρ as defined below. [†] It is worth noting that *ρ-correlated randomized response* is in essence just like *randomized response*[556], a classic scheme that inspired several privacy mechanisms.

Definition 5. *Let $f : \{-1,1\}^n \to \mathbb{R}$ be any function. For every $x \in \{-1,1\}^n$, the* randomized response *$M_\rho f(x)$ outputs $f(y)$ where $y \sim N_\rho x$.*

Now that we have formally defined the randomized response mechanism, we can give the formal definition of differential privacy in our context.

Definition 6 (ϵ-Differential Privacy[160])**.** *A randomized voting mechanism $\mathcal{A} : \{-1,1\}^n \to \{-1,1\}$ is ϵ-differentially private if for all pair of neighboring voting profiles $\mathbf{x}, \mathbf{x}' \in \{-1,1\}^n$*

[†] Note the subtle distinction between M_ρ and N_ρ.

that differ in exactly one bit and for all $\mathbf{s} \in \{-1, 1\}$,

$$\Pr[\mathcal{A}(\mathbf{x}) = \mathbf{s}] \leq e^{\epsilon} \Pr[\mathcal{A}(\mathbf{x}') = \mathbf{s}]$$

The above definition of differential privacy is specific to our context. For the general definition of differential privacy and a broad survey of the field, see[160]. The randomized response mechanism preserves ε-differential privacy. The following result holds for any Boolean function f.

Proposition 1. *For any* $\rho \in [0, 1]$, *randomized response* $M_\rho f$ *preserves* $\log(\frac{1+\rho}{1-\rho})$*-differential privacy regardless of the function* $f : \{-1, 1\}^n \to \mathbb{R}$. *(or, (*$\varepsilon$*,0)-differential privacy when* $\rho \leq 1 - \frac{2}{\exp(\varepsilon)+1}$*).*

Proof. The proof is relegated to Appendix 6.8.1. □

Remark 6.3.1. *The equality case is satisfied if* f *is a dictatorship, which implies that the bound* $\log(\frac{1+\rho}{1-\rho})$ *is tight. That is, when* f *is a dictatorship,* $M_\rho f$ *is not* ε*-differentially private for any* $\varepsilon < \log(\frac{1+\rho}{1-\rho})$*. In fact, it can be shown that a social choice function* f *satisfies the equality case if and only if there is a triple* (r, b, i) *where* $r \in \mathbb{R}, b \in \{-1, 1\}, i \in [n]$ *such that* $\emptyset \neq \{z \in \{-1, 1\}^n | f(z) = r\} \subseteq \{z \in \{-1, 1\}^n | z_i = b\}$.

The reason our mechanism preserves differential privacy for any Boolean function f is that the mechanism is input-perturbing. In this sense, we could instead present the mechanism as $M_\rho : \{-1, 1\}^n \to \{-1, 1\}^n$ and write $f \circ M_\rho$ instead of $M_\rho f$. Then, we could prove the analogous version of Proposition 1, and by using the post-processing property of differential privacy, we would again obtain Proposition 1. In fact, one can see that in the proof, we also prove the post-processing property, seemingly for no reason. However, the reason we choose to give the mechanism altogether after post-processing with f is to make all equality cases in the above remark apparent. Once post-processing is applied black-box, whether the privacy result is robust is not clear anymore. For example, consider any constant function f, e.g. $f(x) = 1$ for any $x \in \{-1, 1\}^n$. In this case, $M_\rho f$ is not only $\log(\frac{1+\rho}{1-\rho})$-differentially private but 0-differentially private.

6.4 Probabilistic Influence

The influence of a voter is a notion that is used to measure the power of an individual on a deterministic social choice function. Influences of Boolean functions have long been studied in computer science and the field of analysis of Boolean functions starting with[50]. The *influence* of a voter in a voting system is defined to be the probability of the change in outcome when the voter changes their vote *ceteris paribus*. For example, in the case of a dictatorship, the dictator has influence 1 while every other voter has influence 0. In the majority function with $n = 2k+1$ voters, each voter's influence is the same and equal to $\binom{2k}{k}/2^{2k}$.

We use $x_{i\to 1} = (x_1, \cdots, x_{i-1}, 1, x_{i+1}, \cdots, x_n)$ to denote the case where the i-th voter chooses to vote for 1, and every other voter follows x. Similarly, we denote the alternate case where the i-th voter chooses to vote for -1 and every other voter follows x by $x_{i\to -1} = (x_1, \cdots, x_{i-1}, -1, x_{i+1}, \cdots, x_n)$. Using this notation, influence in the deterministic setting is defined as follows.

Definition 7. *For $f : \{-1,1\}^n \to \{-1,1\}$, the influence of elector i is defined as*

$$I_i[f] = \mathbb{P}_{x\in\{-1,1\}^n}[f(x_{i\to 1}) \neq f(x_{i\to -1})]$$

The total influence of the function f is defined to be

$$I[f] = \sum_{i=1}^{n} I_i[f]$$

A similar notion can be introduced in the probabilistic setting where the randomized response $M_\rho f(x)$ is applied. To do so, we consider the case where everybody casts their votes, following which $M_\rho f(x)$ is applied, and the voter i changes their vote. That is, we leave all the noisy versions of the votes cast by everyone as is except for the elector i's vote. For this particular vote, we re-run the randomized response on coordinate i. The probability of the result being different is called the *probabilistic influence* of coordinate i. We now introduce the formal definition of the proposed probabilistic influence, which applies not only to social choice functions with range $\{-1,1\}$ but to all Boolean functions with range in $\mathbb{R}$ as follows. In the notation of the following definition, $y_i \sim N_\rho(1)$ refers to the case where voter i chooses to vote for 1 while $z_i \sim N_\rho(-1)$ refers to the case where voter i chooses to vote for -1.

Definition 8. *Let $f : \{-1,1\}^n \to \mathbb{R}$ and the probabilistic influence of coordinate i in a mechanism $M_\rho f(x)$ is defined as*

$$I_i[M_\rho f] = \mathbb{E}_{x\sim\{-1,1\}^n, \forall j\neq i\; z_j=y_j=x_j, y_i\sim N_\rho(1), z_i\sim N_\rho(-1)}[\left(\frac{f(y)-f(z)}{2}\right)^2]$$

The total influence of the mechanism $M_\rho f$ is defined to be

$$I[M_\rho f] = \sum_{i=1}^{n} I_i[M_\rho f]$$

We showed in Proposition 1 that our probabilistic voting mechanism preserves ε-differential privacy. Inducing such privacy requires probabilistic mechanisms as opposed to using deterministic functions. For example, in the majority voting with $2k+1$ voters, if the votes are split k to $k+1$, then changing only one bit in the input may change the outcome of the voting mechanism. Thus, it is not differentially private. Similarly, no deterministic Boolean function can preserve differential privacy unless it is a constant function.

On the other hand, introducing noise may cause several issues in the voting system, one of which is the accuracy of the mechanism, which we will discuss in more detail in Section 6.6. Another possible issue is that when noise is introduced, we might be altering the voting system in favor of a particular voter. For example, voter A might have more influence relative to voter B in the system now, even if that was not the case before. For symmetric social choice functions, it is natural to expect that the randomized response mechanism would have the same effect for any voter since the noise is also symmetric. However, it is not as trivial for arbitrary social choice functions. Yet, we show that each voter's probabilistic influence is proportional to her influence in the deterministic setting. Therefore, the randomized response preserves the ordinal relations between the influences of the voters regardless of the original social choice function being used. In other words, if voter A had greater influence than another voter B, she would still have a greater influence on the system after the noise is introduced.

Theorem 6.4.1. Let $\rho \in [0,1]$ be any real number and $f : \{-1,1\}^n \to \mathbb{R}$ be any function. For every $i \in [n]$, $I_i[M_\rho f] = \frac{1+\rho^2}{2} I_i[f]$.

Proof. The proof is relegated to Section 6.8.2. □

6.5 Welfare

In this section, we introduce a formal definition of *welfare* of social choice functions. Then, we extend this definition to probabilistic mechanisms, and we show that the randomized response preserves the ordinal relations between the welfare of social choice functions.

6.5.1 Welfare of Deterministic Voting Systems

[434] argues in his *Social Contract* that an ideal voting rule should maximize the number of votes that agree with the outcome.[392] proves that the majority function is the unique ideal function based on Rousseau's perception of the ideal voting rule without formally introducing welfare. Perhaps, when he proved this result, he had some form of welfare in his mind, especially because he used the letter w to denote the number of votes that agree with the outcome. In this section, we will formally define welfare, which will be slightly different than what the w notation of O'Donnell describes. In particular, we define *welfare* of a social choice function $f : \{-1, 1\}^n \to \{-1, 1\}$ as the average difference between the number of votes that agree with the outcome and the number of votes that do not agree with the outcome under the impartial culture assumption.

Definition 9. *Let $f : \{-1, 1\}^n \to \{-1, 1\}$ and $x \in \{-1, 1\}^n$, and let $w_x(f) = |\{i; x_i = f(x)\}| - |\{i; x_i \neq f(x)\}|$. Welfare of the social choice function f is defined to be*

$$W(f) = \mathbb{E}_x[w_x(f)].$$

We can still prove that the majority function is the unique maximizer of welfare when n is odd by using a similar method as in the proof of Theorem 2.33 in[392].

Proposition 2. *When n is odd, the unique maximizer of $W(f)$ is the majority function.*

Proof. The proof is relegated to 6.8.3. □

Without further assessment, it is not possible to say whether we prefer total influence to be larger or smaller for the welfare of society in a voting system. As we show in the following result, if the social choice function is monotone — that is, if a voter changes her vote in favor of a candidate, then this candidate should be weakly better off — then these two notions collide with each other. This result has implications beyond being a simple identity, making the case that if we want to achieve greater social welfare while adhering to monotone social choice functions, we must choose a function with a greater total influence.

Proposition 3. *Let f be any monotone social choice function $f : \{-1,1\}^n \to \{-1,1\}$. Then, $W(f) = I[f]$.*

Proof. The proof is relegated to Section 6.8.4. □

6.5.2 Welfare of Noisy Mechanisms

To capture the same notion for the probabilistic functions as well, we similarly define welfare of a randomized mechanism applied on a social choice function as follows. Note that the following definition is not only for the randomized response M_ρ, but any mechanism defined on social choice functions.

Definition 10. *Let $f : \{-1,1\}^n \to \{-1,1\}$, $x \in \{-1,1\}^n$, and M be any mechanism. Let $w_x(Mf) = |\{i; x_i = Mf(x)\}| - |\{i; x_i \neq Mf(x)\}|$. The welfare of the mechanism M with the social choice function f is defined to be*

$$W(Mf) = \mathbb{E}_{x,M}[w_x(Mf)]$$

where the expectation is both over x and the mechanism M.

We showed in Theorem 6.4.1 that although introducing ρ-correlated noise in a voting system has negative effects on influences, it does not provide an unfair advantage to any agent. Another possible undesired byproduct of a randomized mechanism could be that the effect of randomization on the welfare of a particular voting system is more severe compared to the other voting systems. For example, we showed in Proposition 2 that the majority function is the unique welfare maximizer. It could be the case that after we introduce noise, it is more likely in the majority function that the outcome will change. Within this context, the following result implies that every voting system is equally affected by the input-perturbing randomized response mechanism. Therefore, the randomized response preserves the ordinal relations between the welfare of two-candidate voting systems.

Theorem 6.5.1. Let f be any social choice function $f : \{-1,1\}^n \to \{-1,1\}$. Then, $W(M_\rho f) = \rho \cdot W(f)$.

Proof. The proof is relegated to Section 6.8.5 □

This result, together with Proposition 2, implies that the majority function is the unique welfare maximizer also after the noise is introduced by applying the randomized response mechanism.

6.6 Accuracy Analysis

There is one significant drawback of the randomized response privatization mechanism in consideration. It is hard to analyze the accuracy of releasing the output of social choice functions upon privatizing it with the randomized response. Although our main objective in this work is not about the analysis of accuracy, we will dedicate a section to the analysis of accuracy for the sake of completeness. As a first pass, we easily find a *generic* lower-bound on the accuracy of the randomized response, but it ends up being so low that it makes it redundant. Therefore, we restrict our analysis to *specific* social choice functions. We theoretically provide results on accuracy for dictatorship, AND, and OR functions.[‡] In addition, we give a tight lower bound as well as an upper bound for the accuracy of the majority function. We also give an algorithm to calculate the exact accuracy of the majority function by using dynamic programming via memoization. The dynamic programming approach avoids the need to make calculations over every entry in the power-set. It is much more efficient while still resulting in an exact solution for computing the accuracy. Our definition of accuracy is, in fact, the average of accuracy under the impartial culture assumption. That is,

$$Acc(M_\rho f) = \mathbb{P}_{\substack{x \sim \{-1,1\}^n \\ M_\rho}}[M_\rho f(x) = f(x)].$$

Now, we define the *noise operator*, also referred to as the noisy Markov operator, which is a linear operator on the set of Boolean functions. This operator will be useful for accurate calculations.

[‡] For formal definitions of these widely known social choice functions, see Appendix 6.9.

Definition 11. *For any $\rho \in [0,1]$, the noise operator T_ρ is the linear operator on the set of functions $f : \{-1,1\} \to \mathbb{R}$ defined by*

$$T_\rho f(x) = \mathbb{E}_{y \sim N_\rho x}[f(y)].$$

Before we start our analysis, let us also give the definition of *noise stability*.

Definition 12. *For any $\rho \in [0,1]$ and $f : \{-1,1\}^n \to \mathbb{R}$, ρ-correlated noise stability of f is given by*

$$Stab_\rho(f) = \mathbb{E}_{\substack{x \sim \{-1,1\}^n \\ y \sim N_\rho(x)}}[f(x) \cdot f(y)]$$

There is a linear relation between the noise stability of a function and the accuracy of the randomized response on this function. Note that $M_\rho f(x) \cdot f(x) = 1$ if $M_\rho f(x) = f(x)$, $M_\rho f(x) \cdot f(x) = -1$ otherwise. Thus,

$$2 \cdot Acc(M_\rho f) - 1 = 2 \cdot \mathbb{P}_{\substack{x \sim \{-1,1\}^n \\ y \sim N_\rho(x)}}[f(y) = f(x)] - 1 = \mathbb{E}_{\substack{x \sim \{-1,1\}^n \\ y \sim N_\rho(x)}}[f(y) \cdot f(x)] = Stab_\rho(f). \tag{6.1}$$

Also, note that

$$Stab_\rho(f) = \mathbb{E}_{\substack{x \sim \{-1,1\}^n \\ y \sim N_\rho(x)}}[f(x) \cdot f(y)] = \mathbb{E}_{x \sim \{-1,1\}^n}[f(x) T_\rho f(x)]. \tag{6.2}$$

The reason we feel the need to write accuracy in terms of stability is that in the field of Analysis of Boolean functions, most results are given in terms of stability for convenience. Yet, we use stability explicitly only when we analyze the accuracy of the majority function.

6.6.1 MAJORITY

In this section, we will give the asymptotic accuracy for the Maj_n function where n is an odd number that goes to infinity.

Lemma 6.6.1 (Proposition 10,[391]). *For any $\rho \in [0,1)$, $Stab_\rho[Maj_n]$ is a decreasing function of n where n is an odd number, with*

$$\frac{2}{\pi}\arcsin(\rho) \leq Stab_\rho[Maj_n] \leq \frac{2}{\pi}\arcsin(\rho) + O(\frac{1}{\sqrt{1-\rho^2}\sqrt{n}}).$$

By using the fact that accuracy is equal to $\frac{1}{2} + \frac{1}{2}Stab_\rho(f)$ due to Equation (6.1), we get that

$$\frac{1}{2} + \frac{1}{\pi}\arcsin(\rho) \leq Acc[M_\rho(Maj_n)] \leq \frac{1}{2} + \frac{1}{\pi}\arcsin(\rho) + O(\frac{1}{\sqrt{1-\rho^2}\sqrt{n}}). \tag{6.3}$$

Despite this fact being quite useful, there is no convenient way to calculate the exact value of accuracy of the randomized response on the Majority function. Hence, we compute it using dynamic programming via memoization in the following section.

ALGORITHM TO COMPUTE THE EXACT ACCURACY FOR SMALL n

We now provide a dynamic programming algorithm with memoization to compute the accuracy of the randomized response. In particular, we give the algorithm to calculate the accuracy of the *threshold functions*, that are of the form

$$f_\theta(x) = \begin{cases} 1 & \text{if } \sum_{i\in[n]} x_i > \theta \\ -1 & \text{if } \sum_{i\in[n]} x_i \leq \theta \end{cases}$$

Note that $Maj_n = f_0(\cdot)$ where it takes care of ties by considering them as if -1 is the winner. In general, we work with the odd number of voters when we talk about the majority function. But as a simple trick, we will compute it for any n based on the generic definition of the threshold function we gave above since it makes the algorithm less involved.

We now state the noise operator $T_\rho f_{\theta_0}(x)$ as introduced in Definition 11 when applied to threshold functions as a way to quantify the expected accuracy as

$$T_\rho f_{\theta_0}(x) = \mathbb{E}_{y\sim N_\rho x}\left[1\left(y_1 + \dots y_n > \theta_0\right)\right].$$

Let x_{-n} denote x without the last bit. In particular, if $x = (x_1, x_2, \cdots, x_{n-1}, x_n)$, then $x_{-n} = (x_1, x_2, \cdots, x_{n-1})$. Note that $x_{-n} \in \{-1,1\}^{n-1}$ while $x \in \{-1,1\}^n$. Then, the stability can be defined using two calls of recursion as follows

$$T_\rho f_{\theta_0}(x) = \frac{1+\rho}{2} T_\rho f_{\theta_0 - x_n}\left(x_{-n}\right) + \frac{1-\rho}{2} T_\rho f_{\theta_0 + x_n}\left(x_{-n}\right)$$

That is because

$$\begin{aligned}
&\mathbb{E}_{y\sim N_\rho x}\left[1\left(y_1 + \dots + y_n > \theta_0\right)\right] \\
&\quad = \mathbb{E}_{y_n \sim N_\rho x_n}\left[\mathbb{E}_{y_{-n}\sim N_\rho x_{-n}}\left[\mathbb{1}\left(y_1 + \dots + y_{n-1} > \theta_0 - y_n\right) \mid y_n\right]\right] \\
&\quad = \frac{1+\rho}{2} \mathop{\mathbb{E}}_{y_{-n}\sim N_\rho (x_{-n})}\left[\mathbb{1}\left(y_1 + \dots + y_{n-1} > \theta_0 - x_n\right)\right] \\
&\qquad\quad + \frac{1-\rho}{2} \mathop{\mathbb{E}}_{y_{-n}\sim N_\rho (x_{-n})}\left[\mathbb{1}\left(y_1 + \dots + y_{n-1} > \theta_0 + x_n\right)\right] \\
&\quad = \frac{1+\rho}{2} T_\rho f_{\theta_0 - x_n}\left(x_{-n}\right) + \frac{1-\rho}{2} T_\rho f_{\theta_0 + x_n}\left(x_{-n}\right)
\end{aligned}$$

To summarize, this dynamic programming with memoization algorithm is as shown Figure 6.1. In terms of notation we denote a specific dictionary (in terms of popular programming terminology of dictionary data types) as Dictionary: $\{(\rho, n, s, \theta) = T_\rho f_{\theta_0}(x)$ for some x s.t $sum(x) = s\}$.

Our approach is to use this proposed recursive relation with an appropriate initial condition to exactly compute the noise operator $T_\rho f(x)$. Then, by using Equation (6.2), we calculate the Stability of the function. Finally, by using the linear relation between stability and accuracy from Equation (6.1), we compute the exact accuracy. This dynamic programming approach avoids having to make 2^n computations, given that $x \sim \{-1, 1\}^n$. Note that, $T_p f_{\theta_0}(x) = T_p f_{\theta_0}(z)$ if $sum(x) = sum(z)$. Therefore we iterate over i from 1 to n to represent vectors with i number of $1's$. Then, as the rest of the entries are -1, and since the length of the array is n, this approach can model the exact sum of all possible vectors. Since the calculation of the stability is one-to-one with respect to sums, we store the intermediate results in a dictionary indexed by this sum. As there are $\binom{n}{i}$ vectors that can be represented this way, we just compute once per each i and multiply it by $\binom{n}{i}$. This enables us to model all possible vectors efficiently but allows us not to have to compute the intermediate results every time via our recursive approach.

```
ALGORITHM 1: Proposed dynamic programming algorithm with memoization
            Define Dictionary: $\{(\rho, n, s, \theta) = T_\rho f_{\theta_0}(x)$ for some $x$ s.t $sum(x) = s\}$
def $T_\rho f_{\theta_0}(x)$:
s = $sum(x)$;
if $(p, n, s, \theta_0)$ is in dictionary then
    return dictionary $[(p, n, s, \theta_0)]$;
end
else
    Using 2 recursive calls in summands, compute:
     $\alpha = \frac{1+\rho}{2} T_\rho f_{\theta_0 - x_n}(x_{-n}) + \frac{1-\rho}{2} T_\rho f_{\theta_0 + x_n}(x_{-n})$
     Save $(p, n, s, \theta_0) = \alpha$ to dictionary
    return $\alpha$
end

def $Acc_\rho(f_\theta)$:
total = 0;
for $i$ in range $n + 1$ do
    total += $\binom{n}{i} \cdot f_{\theta_0}(x) \cdot T_\rho f_\theta(x)$ for some $x$ s.t. $x$ has $i$ different +1 bits
end
return $\frac{1}{2} + \frac{total/2^n}{2}$
```

Figure 6.1: Proposed dynamic programming algorithm with memoization

In Figure 6.2, we plot the accuracy curves of the randomized response mechanism with varying values of ρ applied to the majority function as the number of voters increases. Note that as n goes to ∞, the accuracy asymptotically approaches to $\frac{1}{2} + \frac{1}{\pi}\arcsin(\rho)$ as implied by Equation (6.3).

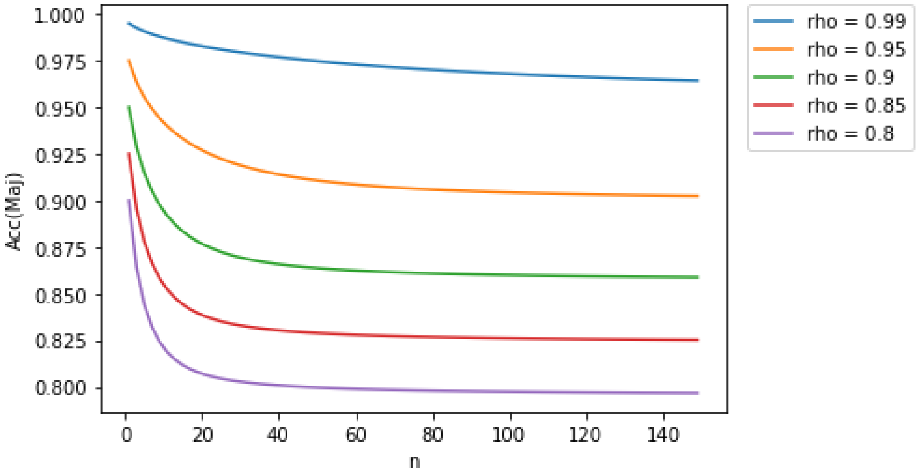


Figure 6.2: The accuracy curves of the randomized response mechanism with varying values of ρ applied to the majority function as the number of voters increases.

6.6.2 Dictatorship

Let $f : \{-1,1\}^n \to \{-1,1\}$ be the dictatorship of voter-i, that is $f(x) = 1$ if and only if $x_i = 1$.

Then, for any given $x \in \{-1,1\}^n$,

$$\mathbb{P}[M_\rho f(x) = f(x)] = \mathbb{P}_{y \sim N_\rho(x)}[f(y) = f(x)] = \mathbb{P}_{y_i \sim N_\rho(x_i)}[y_i = x_i] = \frac{1+\rho}{2}.$$

Hence, the average accuracy is also equal to $\frac{1+\rho}{2}$.

6.6.3 AND_n AND OR_n

We will first make the calculations for AND_n and the results will be analogous due to symmetry. We will make use of Fact 1 in the analysis.

First, we start with a generic calculation that holds for any social choice function f. In the calculations in this section, our probability space is $x \sim \{-1,1\}^n, M_\rho f(x) \sim f(y)$ where $y \sim N_\rho x$.

Note that by Fact 1,

$$\mathbb{P}_{x,M_\rho}[M_\rho f(x) = 1] = \mathbb{P}_x[f(x) = 1].$$

$$\mathbb{P}\left[M_\rho f(x) = f(x)\right] = \mathbb{P}\left[M_\rho f(x) = 1 \wedge f(x) = 1\right] + \mathbb{P}\left[M_\rho f(x) = -1 \wedge f(x) = -1\right]$$

and

$$\begin{aligned}\mathbb{P}\left[M_\rho f(x) = -1 \wedge f(x) = -1\right] &= 1 - \mathbb{P}\left[M_\rho f(x) = 1 \vee f(x) = 1\right] \\ &= 1 - \mathbb{P}\left[M_\rho f(x) = 1\right] - \mathbb{P}\left[f(x) = 1\right] + \mathbb{P}\left[M_\rho f(x) = 1 \wedge f(x) = 1\right] \\ &= 1 - 2 \cdot \mathbb{P}\left[f(x) = 1\right] + \mathbb{P}\left[M_\rho f(x) = 1 \wedge f(x) = 1\right].\end{aligned}$$

Thus for any social choice function f,

$$\mathbb{P}\left[M_\rho f(x) = f(x)\right] = 1 - 2 \cdot \mathbb{P}\left[f(x) = 1\right] + 2 \cdot \mathbb{P}\left[M_\rho f(x) = 1 \wedge f(x) = 1\right]$$

For $f = AND_n$,

$$\mathbb{P}[f(x) = 1] = \prod_{i \in [n]} \mathbb{P}[x_i = 1] = 2^{-n},$$

and

$$\mathbb{P}\left[M_\rho f(x) = 1 \wedge f(x) = 1\right] = \mathbb{P}[f(x) = 1] \cdot \mathbb{P}[M_\rho f(x) = 1 | f(x) = 1] = 2^{-n} \cdot (\frac{1+\rho}{2})^{-n}.$$

Hence, the accuracy of M_ρ for AND_n function is equal to $1 - 2^{-n+1}(1 - (\frac{1+\rho}{2})^n)$, whose limit goes to 1 as n goes to ∞. Due to symmetry, accuracy analysis is the same for OR_n function.

6.7 Conclusion

The main objective of this work is to study the privacy-welfare trade-off and the relation between privacy and probabilistic influence. The proposed definition of welfare happens to hold for any mechanism, while on the other hand, the defined probabilistic influence is only specific to the randomized response mechanism. In fact, a more general definition of influence could be coined, and a similar property could potentially be observed. We leave out this potential generalization of influence to future work. In terms of welfare, the analysis done in this paper can be replicated in a similar style to other popular privatization schemes such as the Laplace and exponential mechanisms. The privacy-accuracy trade-off of the current mechanism for the majority function may also be further improved. Note that Dictatorship, AND, and OR functions satisfy the equality condition in Proposition 1 as discussed in Remark 6.3.1. Thus, the accuracy-privacy analyses for these functions are tight. On the other hand, for a given ρ, the asymptotic accuracy of the majority is tight, whereas the privacy result is a possibly loose upper bound.

Also, our definitions of influence and welfare assume that the votes are unbiased. They consider everybody to be equally likely to vote for -1 or $+1$. In fact, these definitions can be further generalized to cover the same concept, but for the case of biased voting. For example, one can extend the definitions to be *p-biased* for a given $p \in [-1, 1]$; that is, the expected value of each vote is p instead of 0. p-biased distribution is also well-studied in the field of Analysis of Boolean functions.

Finally, our voting model in this paper is a classical referendum model with two candidates. However, in most real-world applications, we generally have multiple candidates, and we have to aggregate the rankings. If there is a Condorcet winner in a voting system, then the results regarding two-candidate elections can be directly applied in the multiple-candidate setting. Yet, in many cases, there is no Condorcet winner. Restricting the number of candidates to two has the primary advantage that both the definitions and analyses of welfare and influence naturally follow. We believe that extending the definitions and the tools developed in this paper to multiple-candidate settings would be interesting.

From a broader perspective, we study the effect of using privacy-inducing randomized responses in the voting process. We construct a relation between the level of privacy and the resulting level of influence of voters involved in the voting system and the welfare of the chosen social choice function. An insightful takeaway that we can deduce from the derived relationships in this paper is that the ordering of voters' influences and the ordering of welfare amongst the considered social choice functions remain unchanged upon introducing noise via the celebrated randomized response mechanism. Existing works have extensively studied the relationship between privacy and the resulting accuracy in preserving the output of the query that was privatized. At a high level, we are the first to shed light on the relationship between privacy and other important phenomena of influence and welfare. We hope that this bridge we have proposed between the two important fields of differential privacy and social choice theory will be further studied and extended as part of future works.

6.8 Proofs

6.8.1 Proof of Proposition 1

Proof. Let r be any element in the range of $M_\rho f$. Let $Z = \{z \in \{-1,1\}^n | f(z) = r\}$. Let x and x' differ only at x_i for some $i \in [n]$.

$$\frac{\mathbb{P}[M_\rho f(x) = r]}{\mathbb{P}[M_\rho f(x') = r]} = \frac{\sum_{z\in Z} \mathbb{P}_{y \sim N_\rho x}[y = z]}{\sum_{z\in Z} \mathbb{P}_{y \sim N_\rho x'}[y = z]} = \frac{\sum_{z\in Z} \prod_{j\in[n]} \mathbb{P}_{y_j \sim N_\rho x_j}[y_j = z_j]}{\sum_{z\in Z} \prod_{j\in[n]} \mathbb{P}_{y_j \sim N_\rho x'_j}[y_j = z_j]}.$$

The first equality is upon considering all cases of the output of the randomized response resulting in a $z \in Z$. Then, by definition, that would result in the function f evaluated on this output z to be r. The second equality is due to the independence assumption across the voter's choices. Now, for any $z \in Z$,

$$\mathbb{P}_{y_j \sim N_\rho x_j}[y_j = z_j] = \begin{cases} \frac{1+\rho}{2} & \text{if } x_j = z_j \\ \frac{1-\rho}{2} & \text{if } x_j \neq z_j \end{cases} \quad \text{and} \quad \mathbb{P}_{y_j \sim N_\rho x'_j}[y_j = z_j] = \begin{cases} \frac{1+\rho}{2} & \text{if } x'_j = z_j \\ \frac{1-\rho}{2} & \text{if } x'_j \neq z_j \end{cases}$$

This is because $\frac{1-\rho}{2}$ is the probability of a misrecorded vote and $1 - \frac{1-\rho}{2} = \frac{1+\rho}{2}$ is the probability otherwise. More explicitly, with probability $1-\rho$, it chooses to blur the ballot, and the blurring is then done by picking uniformly out of the two options of $\{-1, 1\}$ with probability 0.5 each, out of which one pick would result in no change to the vote and the other would result in a misrecorded vote. Also, for any $j \neq i$,

$$\mathbb{P}_{y_j \sim N_\rho x_j}[y_j = z_j] = \mathbb{P}_{y_j \sim N_\rho x'_j}[y_j = z_j].$$

Thus,

$$\frac{1-\rho}{1+\rho} \leq \frac{\sum_{z\in Z}\prod_{j\in[n]}\mathbb{P}_{y_j\sim N_\rho x_j}[y_j = z_j]}{\sum_{z\in Z}\prod_{j\in[n]}\mathbb{P}_{y_j\sim N_\rho x'_j}[y_j = z_j]} \leq \frac{1+\rho}{1-\rho},$$

which completes the proof. □

6.8.2 PROOF OF THEOREM 6.4.1

Proof. Using conditional probability, we get that

$$\begin{aligned} I_i[M_\rho f] &= \mathbb{E}_{x\sim\{-1,1\}^n, \forall j\neq i\; z_j=y_j=x_j, y_i\sim N_\rho(1), z_i\sim N_\rho(-1)}\left[\left(\frac{f(y)-f(z)}{2}\right)^2\right] \\ &= \mathbb{P}_{y_i\sim N_\rho(1), z_i\sim N_\rho(-1)}[y_i = 1, z_i = -1]\cdot\mathbb{E}_{x\sim\{-1,1\}^n}\left[\left(\frac{f(x_{i\to 1})-f(x_{i\to -1})}{2}\right)^2\right] \\ &+ \mathbb{P}_{y_i\sim N_\rho(1), z_i\sim N_\rho(-1)}[y_i = -1, z_i = 1]\cdot\mathbb{E}_{x\sim\{-1,1\}^n}\left[\left(\frac{f(x_{i\to 1})-f(x_{i\to -1})}{2}\right)^2\right] \end{aligned}$$

Noting that

$$\mathbb{P}_{y_i\sim N_\rho(1), z_i\sim N_\rho(-1)}\left[y_i = 1, z_i = -1\right] = \left(\frac{1+\rho}{2}\right)^2,$$

$$\mathbb{P}_{y_i\sim N_\rho(1), z_i\sim N_\rho(-1)}\left[y_i = -1, z_i = 1\right] = \left(\frac{1-\rho}{2}\right)^2,$$

and that

$$\mathbb{E}_{x\sim\{-1,1\}^n}\left[\left(\frac{f(x_{i\to 1})-f(x_{i\to -1})}{2}\right)^2\right] = I_i[f],$$

we get that

$$I_i[M_\rho f] = \frac{1+\rho^2}{2} I_i[f].$$

□

6.8.3 PROOF OF PROPOSITION 2

Proof. First, let us fix x. Note that

$$w_x(f) = f(x) \cdot \sum_{i \in [n]} x_i.$$

Since $f(x) \in \{-1, 1\}$, $f(x) \cdot \sum_{i \in [n]} x_i$ is maximized when $f(x) = sign(\sum_{i \in [n]} x_i)$. Hence, $W(f)$ is maximized if $\forall x \in \{-1,1\}^n$, $f(x) = sign(\sum_{i \in [n]} x_i)$, which is exactly the definition of the majority function. □

Remark 6.8.1. *Note that we used the condition that n is odd to ensure that the $sign$ function is well-defined. If n was even, then the maximizers of $W(f)$ are again the majority functions where it does not matter who is elected if it is tied.*

6.8.4 PROOF OF PROPOSITION 3

To prove this result, we use discrete Fourier analysis. It is a well-known result from the field of analysis of Boolean functions that every function $f : \{-1, 1\}^n \to \mathbb{R}$ can be uniquely expressed as a multilinear polynomial,

$$f(x) = \sum_{S \subseteq [n]} \widehat{f}(S) \chi_S(x)$$

where for any $S \in [n]$

$$\chi_S(x) = \prod_{i \in S} x_i.$$

This expression is called the Fourier expansion of f, and the real number $\widehat{f}(S)$ is called the Fourier coefficient of f on S. Collectively, the coefficients are called the Fourier spectrum of f. The following is an essential result from discrete Fourier Analysis.

Lemma 6.8.1 (Plancherel's Theorem)**.** *For any functions* $f, g : \{-1,1\}^n \to \mathbb{R}$*,*

$$E_{x\sim\{-1,1\}^n}[f(x)g(x)] = \sum_{S\subseteq[n]} \widehat{f}(S)\widehat{g}(S).$$

It is possible to neatly calculate many features of f, including the influences in terms of Fourier coefficients.

Lemma 6.8.2 (Proposition 2.21,[392])**.** *Let* $f : \{-1,1\}^n \to \{-1,1\}$ *be a monotone function and let the Fourier spectrum of* f *be* $f(x) = \sum_{S\subseteq[n]} \widehat{f}(S)\chi_S(x)$*. Then, for any* $i \in [n]$*,*

$$I_i[f] = \widehat{f}(\{i\}).$$

It is also possible to calculate the welfare in terms of the Fourier coefficients by taking one step further from the proof of Proposition 2.

Lemma 6.8.3. *Let* f *be any social choice function* $f : \{-1,1\}^n \to \{-1,1\}$*. Then,* $W(f) = \sum_{i\in[n]} \widehat{f}(\{i\})$*.*

Proof. By the definition of welfare,

$$W(f) = \mathbb{E}_x[w_x(f)] = \mathbb{E}_x[f(x)\cdot\sum_{i\in[n]} x_i] = \sum_{i\in[n]} \widehat{f}(\{i\})$$

where the last equation follows from Lemma 6.8.1. □

We are ready to finish the proof.

Proof of Proposition 3. The proof follows immediately from Lemma 6.8.2 and Lemma 6.8.3. □

6.8.5 Proof of Theorem 6.5.1

Proof. We prove this identity by using a double-counting method and linearity of expectation. Fix f. For any $i \in [n]$, let $1_{i,x,\rho}$ be the indicator random variable defined as follows:

$$1_{i,x,\rho} = \begin{cases} 1 & \text{if } M_\rho f(x) = x_i \\ -1 & \text{if } M_\rho f(x) \neq x_i \end{cases}$$

Where the randomization is due to the randomized response. Note then when x is given and $\rho = 1$, there is no randomization because $M_\rho f(x) = f(x)$ with probability 1. Therefore, $1_{i,x,1}$ is a deterministic function. For the sake of simplicity, we will abuse the notation and write $1_{i,x}$ instead of $1_{i,x,1}$ in the deterministic case. Then,

$$w_x(M_\rho f) = \sum_{i \in [n]} 1_{i,x,\rho} \quad \text{and} \quad w_x(f) = \sum_{i \in [n]} 1_{i,x}.$$

Thus,

$$W(M_\rho f) = \mathbb{E}_{M_\rho,x}[w_x(f)] = \mathbb{E}_{x,M_\rho}[\sum_{i \in [n]} 1_{i,x,\rho}] = \sum_{i \in [n]} \mathbb{E}_{x,M_\rho}[1_{i,x,\rho}]$$

and so

$$W(f) = \sum_{i \in [n]} \mathbb{E}_x[1_{i,x}].$$

Now, we will show that for any $i \in [n]$,

$$\mathbb{E}_{x,M_\rho}[1_{i,x,\rho}] = \rho \cdot \mathbb{E}_x[1_{i,x}].$$

First, note that

$$\mathbb{E}_{x,M_\rho}[1_{i,x,\rho}] = \mathbb{P}_{\substack{x\sim\{-1,1\}^n\\ y\sim N_\rho x}}[f(y) = x_i] - \mathbb{P}_{\substack{x\sim\{-1,1\}^n\\ y\sim N_\rho x}}[f(y) \neq x_i].$$

By using

$$\mathbb{P}_{\substack{x\sim\{-1,1\}^n\\ y\sim N_\rho x}}[f(y) = x_i] + \mathbb{P}_{\substack{x\sim\{-1,1\}^n\\ y\sim N_\rho x}}[f(y) \neq x_i] = 1,$$

we get that

$$\mathbb{E}_{x,M_\rho}[1_{i,x,\rho}] = 2 \cdot \mathbb{P}_{\substack{x\sim\{-1,1\}^n\\ y\sim N_\rho x}}[f(y) = x_i] - 1.$$

By Fact 1, we can replace $x \sim \{-1,1\}^n, y \sim N_\rho x$ with $y \sim \{-1,1\}^n, x \sim N_\rho y$. Thus, by using conditional probability,

$$\begin{aligned}
\mathbb{E}_{x,M_\rho}[1_{i,x,\rho}] &= 2 \cdot \mathbb{P}_{\substack{y\sim\{-1,1\}^n\\ x\sim N_\rho y}}[f(y) = x_i] - 1\\
&= 2(\mathbb{P}_{x\sim N_\rho y}[x_i = y_i] \cdot \mathbb{P}_{y\sim\{-1,1\}^n}[f(y) = y_i] + \mathbb{P}_{x\sim N_\rho y}[x_i = -y_i] \cdot \mathbb{P}_{y\sim\{-1,1\}^n}[f(y) = -y_i]) - 1\\
&= (1+\rho) \cdot \mathbb{P}_{y\sim\{-1,1\}^n}[f(y) = y_i] + (1-\rho) \cdot \mathbb{P}_{y\sim\{-1,1\}^n}[f(y) \neq y_i] - 1\\
&= \rho \cdot (\mathbb{P}_{y\sim\{-1,1\}^n}[f(y) = y_i] - \mathbb{P}_{y\sim\{-1,1\}^n}[f(y) \neq y_i])\\
&= \rho \cdot \mathbb{E}_x[1_{i,x}]
\end{aligned}$$

which completes the proof. □

6.9 SOCIAL CHOICE FUNCTIONS

In this paper, we exclusively focus on social choice functions with two alternatives. There are many ways to interpret these functions. It can be considered as a two-candidate election or as a referendum in the context of political science. It can also be interpreted as a classifier in the context of Machine Learning. In this paper, we will generally give the interpretations in the context of two-candidate elections.

In general, we work with the Boolean functions defined as $f : \{-1,1\}^n \to \mathbb{R}$, and we denote the bit i of the input x by x_i for any $i \in [n]$. However, we define welfare only for *social choice functions*, that is, the Boolean functions whose ranges are $\{-1,1\}$. We analyze accuracy only for the following specific social choice functions.

- **Majority:** Suppose that n is an odd number. The majority function of n agents/voters is denoted by Maj_n and defined as

  $$f(x) = sign(\sum_{i \in [n]} x_i)$$

  for any $x \in \{-1,1\}^n$ where $sign : \mathbb{R} \to \{-1,0,1\}$ is the function such that

  $$sign(a) = \frac{a}{|a|}$$

  for any $a \in \mathbb{R}, a \neq 0$ and sign(0)=0.

- **Dictatorship:** For a given number n and $i \in [n]$, the dictatorship of voter-i is defined as

  $$f(x) = x_i$$

  for any $x \in \{-1,1\}^n$.

- **AND$_n$:** The AND_n function outputs 1 if there is unanimity on 1, outputs -1 otherwise. Namely,

  $$f(x) = \begin{cases} 1 & \text{if } \forall i \in [n], x_i = 1 \\ -1 & \text{otherwise} \end{cases}$$

- **OR$_n$:** The OR_n function outputs 1 if at least one voter votes for 1, and outputs -1 otherwise. In other words, it outputs -1 if there is unanimity on -1 and outputs 1 otherwise. Namely,

$$f(x) = \begin{cases} -1 & \text{if } \forall i \in [n], x_i = -1 \\ 1 & \text{otherwise} \end{cases}$$

Note that, in this paper, we assume *the impartial culture assumption*, that is, the voters are not affected by each other, and they vote independently uniform at random between two candidates.

Part II

Distributed and Private Machine Learning

“Taking a model too seriously is really just another way of not taking it seriously at all.”

George E.P. Box

7

Private Generation and Visualization of Embeddings in Supervised Manifold Learning

7.1 Introduction

Privacy-preserving computation enables distributed hosts with 'siloed' away data to query, analyse or model their sensitive data and share findings in a privacy-preserving manner. As a motivating problem, in this paper, we focus on the task of privately retrieving the nearest matches to a client's target image with respect to a server's database of images. Consider the setting where a client would like to obtain the k-nearest matches to its target from an external distributed database. State-of-the-art image retrieval machine learning models such as[345,107,611,149] exist for feature extraction prior to obtaining the neighbors to a given match in the learnt space of deep feature representations. Unfortunately, this approach is not private. The goal of our approach is to be able to use these useful features for the purpose of image retrieval in a manner that is formally differentially private. The seminal idea for a mathematical notion of privacy, called differential privacy, along with its foundations, is introduced quite well in[160]. In our approach, we geometrically embed the image features via a supervised manifold learning query that we propose. Our query falls within the framework of supervised manifold learning as formalized in[549]. We then propose a differentially private mechanism[536] to release the outputs of

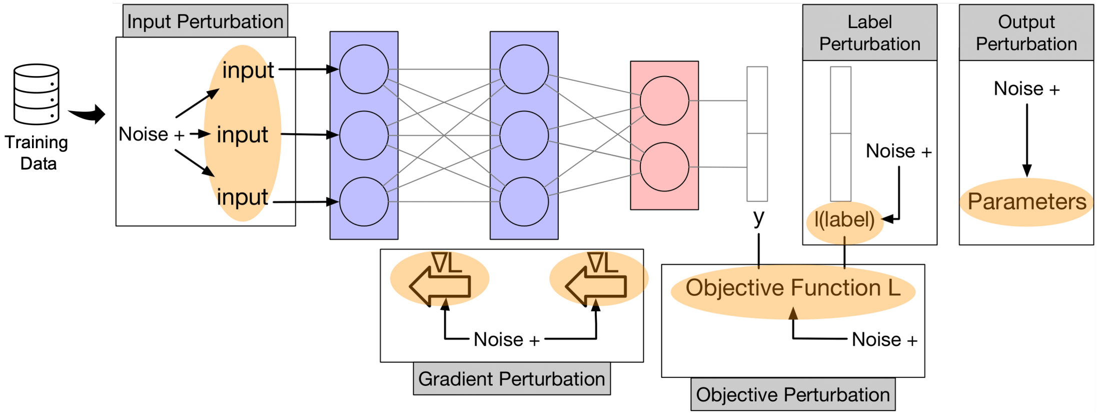


Figure 7.1: Categorization of most existing methods for differentially private training of machine learning models

this query. The privatized outputs of this query are used to perform the matching and retrieval of the nearest neighbors in this privatized feature space. Differential privacy aims to prevent membership inference attacks[475,515,319,485,472]. It has been shown that differential privacy mechanisms can also prevent reconstruction attacks under a constraint on the level of utility that can be achieved, as shown in[162,189]. Currently, cryptographic methods for the problem of information retrieval were studied in works like[571]. These methods ensure the protection of the client's data via homomorphic encryption and oblivious transfer. However, they also come with an impractical trade-off of computational scalability, especially when the size of the server's database is large and the feature size is high-dimensional, as is always the case in practice[167,308,582].

7.1.1 Motivation

Currently available differential privacy solutions for biometric applications where content-based matching of records is performed[488,94] is based on a small number of hand-crafted features. We instead consider state-of-the-art feature extraction used by recent deep learning architectures specialized for image retrieval such as[259]. We privatize these features and share them in the

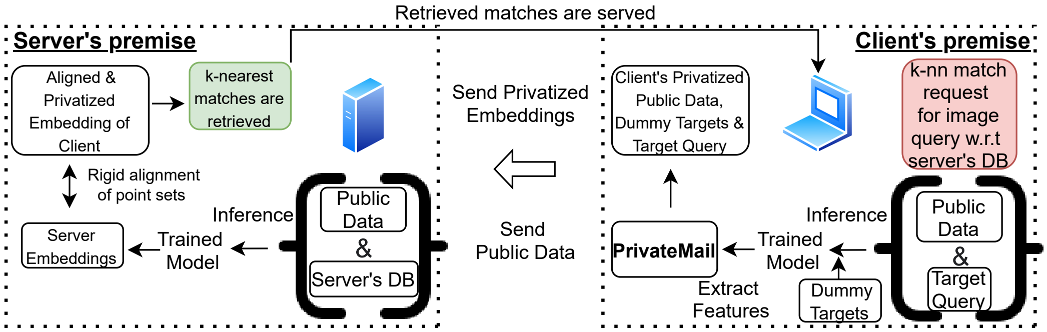

Retrieved matches are served
Server's premise
Aligned & Privatized Embedding of Client
k-nearest matches are retrieved
Rigid alignment of point sets
Server Embeddings
Trained Model
Inference
Public Data
&
Server's DB
Send Privatized Embeddings
Send Public Data
Client's Privatized Public Data, Dummy Targets & Target Query
Client's premise
k-nn match request for image query w.r.t server's DB
PrivateMail
Extract Features
Trained Model
Inference
Dummy Targets
Public Data
&
Target Query

Figure 7.2: This illustration shows the lifecycle of interactions between client and server-side entities for private image retrieval. The interaction starts from the red bubble on the right. At first, the client and server train an on-premise machine learning model that is tailored for image retrieval. The client extracts features from this model on the target query image, dummy targets that are only known to the client, and a public dataset known to the client and server. The extracted features go through the proposed Private-Mail for embedding them via locally differentially private supervised manifold learning. These private embeddings are aligned at the server prior to performing the nearest-neighbor retrieval of matches that are served back to the client. The privatized representation of the public dataset is used as an anchor in order to align the feature embeddings between the client and server.

form of differentially private embeddings that are, in turn, used for the image retrieval task. Furthermore, cryptographic methods with strong security guarantees are currently not scalable computationally for secure KNN queries[167,308,582] especially when the server-side database is large, as is typically the case in real-life scenarios.

7.2 Contributions

The main contribution of our paper is a differentially private method called *PrivateMail* for the private release of outputs from a supervised manifold learning query that embeds data into a lower dimension. We test our scheme for differentially private 'content-based image retrieval', where the matches to a target image requested by a client are retrieved from a server's database while maintaining differential privacy. We also show a substantial improvement in the utility-privacy trade-off of our embeddings over five existing baselines. Finally, the supervised manifold learning query that we propose to embed features extracted from deep networks geometrically is novel in itself. That said, we would only consider this as a secondary contribution to this paper.

7.3 Related Work

Non-private image search and retrieval: The current state-of-the-art pipelines for content-based image retrieval under the non-private setting are fairly mature and based on nearest-neighbor queries performed over specialized deep feature representations of these images. The query image and the database of images are compared in this learnt representation space. A detailed set of tutorials and surveys on this problem in the non-private setting is provided in[345,107,611,149].

Private manifold learning: There have been recent developments in learning private geometric embeddings with differentially private unsupervised manifold learning. Notable examples include distributed and differentially private versions of t-SNE[531] called DP-dSNE[441,439] and[29] for differentially private Laplacian Eigenmaps[46,48]. Furthermore, the work in[116] provides a method for differentially private random projection trees to perform unsupervised private manifold learning. The work in[524] also studies Riemannian manifold learning with differential privacy for manifolds with a bounded condition number and geodesic covering regularity. However, none of these works consider differentially private manifold learning in the supervised setting that we explore in this paper. We show a substantial improvement in privacy-utility trade-offs of the supervised manifold embedding approach over existing baselines that include private and non-private methods in the supervised and unsupervised paradigms.

Motivated by the supervised manifold learning framework in[549] that is based on a difference between two unsupervised manifold learning objectives, we present an iterative update to optimize it efficiently. We refer to this iterative optimization as the *supervised manifold learning query (SMLQ)*. We then provide a privacy mechanism called *PrivateMail* to perform this supervised manifold learning query with a guarantee of differential privacy. To do that, we derive the sensitivity of our query that is required to calibrate the amount of noise needed to attain differential privacy. As part of the experimental results, we apply our approach to a novel task of differentially private image retrieval that has not been well-studied in current literature as opposed to the non-private image retrieval task, which is a widely studied problem.

Notation	Description
n	Sample size
d	Data dimension
k	Embedded dimension
$\mathbf{X}_{n\times d}$	Data matrix
$\mathbf{Y}_{n\times 1}$	Labels
f	Manifold learning map
σ	Gaussian kernel bandwidth
σ_q	std. dev. of entries in $\mathbf{Q}$
α	regularization in $\mathbf{L}_{\mathbf{X}} - \alpha\mathbf{L}_{\mathbf{Y}}$
$\mathbf{Q}$	$Q_{i,j} \sim N(0, \sigma_q^2)$

Table 7.1: Notations

7.4 Moving from Unsupervised to Supervised Manifold Learning

We first briefly introduce some preliminaries for unsupervised manifold learning in order to build upon it to introduce supervised manifold learning.

7.4.1 Preliminaries for Unsupervised Manifold Learning

This problem is a discrete analogue of the continuous problem of learning a map $f : \mathcal{M} \mapsto \mathbb{R}^k$ from a smooth, compact high dimensional Riemannian manifold such that for any two points x_1, x_2 on $\mathcal{M}$, the geodesic distance on the manifold $d_{\mathcal{M}}(x_1, x_2)$ is approximated by the Euclidean distance $\|f(x_1) - f(x_2)\|$ in $\mathbb{R}^k$. Different manifold learning techniques vary in the tightness of this approximation on varying datasets. Manifold learning techniques like Laplacian Eigenmaps[47], Diffusion Maps[123], and Hessian Eigenmaps[144] aim to find a tighter approximation by trying to minimize a relevant bounding quantity **B** such that $\|f(x_1) - f(x_2)\| \leq B \cdot d_{\mathcal{M}}(x_1, x_2) + o(d_{\mathcal{M}}(x_1, x_2))$. Different techniques propose different possibilities for such a B. For example, Laplacian Eigenmaps uses $B = ||\nabla f(x_1)||$ for which it is shown that this relation holds as

$$||f(x_1) - f(x_2)|| \leq ||\nabla f(x_1)|| \cdot ||x_1 - x_2|| + o(||x_1 - x_2||)$$

Hence, controlling $||\nabla f||_{L^2(\mathcal{M})}$ preserves geodesic relations on the manifold in the Euclidean space after the embedding.

7.4.2 From Continuous to Discrete

This quantity of $||\nabla f||_{L^2(\mathcal{M})}$ in the continuous domain can be optimized by choosing the eigenfunctions of the Laplace-Beltrami operator in order to get the optimal embedding. This is explained in a series of papers by[197,48,257]. From a computational standpoint, we note that, for a specific graph defined on all pairs of data points with an adjacency matrix $\mathbf{W_X}$ and corresponding graph Laplacian $\mathbf{L_X}$, the following quantity

$$\Sigma_{i,j}(||\mathbf{f}(\mathbf{X_i}) - \mathbf{f}(\mathbf{X_j})||^2 \cdot [\mathbf{W_X}]_{ij}) = \mathrm{Tr}(\mathbf{f}(\mathbf{X})^T \mathbf{L_X} \mathbf{f}(\mathbf{X})) \tag{7.1}$$

is the discrete version of $||\nabla f||^2_{L^2(\mathcal{M})}$ under the assumption that the dataset $\mathbf{X}$ is a sample lying on the manifold $\mathcal{M}$. Here, $\mathbf{f}(\mathbf{X_i})$ and $\mathbf{f}(\mathbf{X_j})$ refer to the k dimensional real-valued output of the manifold learning map f at two single points represented by i and j rows in the data matrix

$\mathbf{X}_{n\times d}$. Similarly, $\mathbf{f}(\mathbf{X})$ refers to mapping the points indexed by each row in $\mathbf{X}$ to $\mathbb{R}^k$. That is, the output of $\mathbf{f}(\mathbf{X})$ is a real-valued matrix of dimension $n\times k$. Therefore, the equivalent solution to map $\{\mathbf{X}_1, ...\mathbf{X}_n\} \subset \mathbb{R}^d$ while preserving local neighborhood into $\{f(\mathbf{X}_1), ...f(\mathbf{X}_n)\} \subset \mathbb{R}^k$ is to minimize this objective function in equation 7.1 for a specific graph Laplacian $\mathbf{L}_\mathbf{X}$ that we describe below. This popular graph Laplacian, under which the above results were studied, is that of graphs whose adjacency matrices are represented by the Gaussian kernel given by

$$\mathbf{L}(\mathbf{X},\sigma)_{ik} = \begin{Bmatrix} \sum_{k\neq i} e^{(-\frac{\|\mathbf{x_i}-\mathbf{x_k}\|^2}{\sigma})} & \text{if } i = k \\ -e^{(-\frac{\|\mathbf{x_i}-\mathbf{x_k}\|^2}{\sigma})} & \text{if } i \neq k \end{Bmatrix} \tag{7.2}$$

where the scalar σ here is also referred to as kernel bandwidth. The seminal work in[197,47,48] showed that this discrete Graph Laplacian converges to the Laplace-Beltrami operator. Minimizing this objective of Equation 7.1 under the constraint $\mathrm{Tr}(\mathbf{f}(\mathbf{X})^\mathbf{T}\mathbf{D}\mathbf{f}(\mathbf{X})) = \mathbf{I}$ where $\mathbf{I}$ is identity matrix, to avoid a trivial solution of $\mathrm{Tr}(\mathbf{f}(\mathbf{X})^\mathbf{T}\mathbf{L}_\mathbf{X}\mathbf{f}(\mathbf{X})) = 0$ is equivalent to setting the solution for the embedding $\mathbf{f}(\mathbf{X})$ to be the d smallest eigenvectors of $\mathbf{L}_\mathbf{X}$.

7.4.3 Supervised Manifold Learning Queries (SMLQ)

It has been shown in[549] that this formulation for unsupervised manifold learning of minimizing equation equation 7.2 can be extended to the case of supervised manifold learning by posing the objective function as a difference of the terms in equation 7.1 as shown below.

$$v(\mathbf{f}(\mathbf{X})) = \mathrm{Tr}(\mathbf{f}(\mathbf{X})^\mathbf{T}\mathbf{L}_\mathbf{X}\mathbf{f}(\mathbf{X})) - \alpha\,\mathrm{Tr}(\mathbf{f}(\mathbf{X})^\mathbf{T}\mathbf{L}_\mathbf{Y}\mathbf{f}(\mathbf{X})) \tag{7.3}$$

Note that the formula for computing $\mathbf{L}_\mathbf{Y}$ over $\mathbf{Y}$, is the same as the one used in equation 7.2 to compute $\mathbf{L}_\mathbf{X}$ from $\mathbf{X}$. They provide results explaining the effect of optimizing such a loss for the purposes of learning an embedding $\mathbf{f}(\mathbf{X})$ for supervised learning. Their results are agnostic

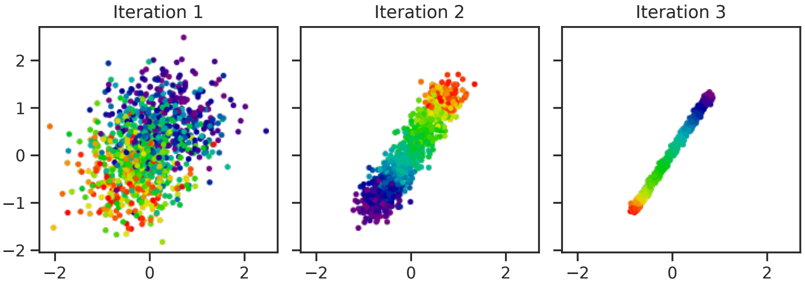


Figure 7.3: Embeddings of our supervised manifold learning query on CUB-200-2011 for 3 iterations with input features extracted from state-of-the-art CGD[259] deep image retrieval architecture with ResNet 50 backbone and G type global descriptors. The colors indicate different class labels. We show that these embeddings preserve information about the class separation and the locality structure required for classification.

to the choice of neighborhood graphs defined on $\mathbf{X}, \mathbf{Y}$ to obtain the corresponding Laplacians used in this objective. An example of such an embedding when applied to features extracted from state-of-the-art CGD (Jun et al. 2019) deep image retrieval architecture with ResNet 50 backbone is shown in Figure 7.3.

Separation-Regularity Trade-Off

The intuition is that since equation equation 7.3 is a discrete version of a difference of terms of the kind in equation 7.1, this formulation looks for a function that has a slow variation on the manifold $\mathcal{M}_X$ in order to smoothly preserve neighborhood relations between the input features. It does this while ensuring the function has a fast variation on a manifold $\mathcal{M}_Y$ with regards to $\mathbf{Y}$, therefore encouraging larger separation with regards to the label manifold. Therefore, this second term acts as a regularizer to make sure similar features are not embedded way closer than needed. This is mathematically substantiated by Theorem 9 in[549] (restated in Appendix D) as it shows that this regularization is required in order to minimize the generalization error of a classifier applied on the output of supervised manifold learning obtained via minimization of equation equation 7.3 for any choice of positive semidefinite $\mathbf{L_X}, \mathbf{L_Y}$.

Theorem 7.4.1. For a fixed α, the iterate

$$\mathbf{X_t} = \frac{\mathbf{Diag(L_X)^{-1}}}{\mathbf{2}}[\alpha\mathbf{L_Y} - \mathbf{L_X}]\mathbf{X_{t-1}} + \mathbf{X_{t-1}} \tag{7.4}$$

monotonically minimizes the objective

$$v(\mathbf{X}_t) = \operatorname{Tr}(\mathbf{X_t^T L_X X_t}) - \alpha \operatorname{Tr}(\mathbf{X_t^T L_Y X_t})$$

Proof Sketch. The full proof, along with the required background, is in the appendix. The proof strategy involves using the majorization-minimization[237,302,610] procedure in order to obtain this iterative update. We first derive a majorization function, which always upper bounds the objective everywhere except at the current iterate, where it touches it. We then note that this majorization function is a sum of convex and concave functions. This makes the minimization of the majorization function to be equivalent to using the concave-convex procedure[593]. As the update is based on majorization-minimization (MM) and CCCP, which itself is a special case of MM, it thereby guarantees monotonic convergence[237]. We refer to this iterate as the *Supervised Manifold Learning Query (SMLQ)*, and the rest of the paper focuses on releasing the outputs of SMLQ with differential privacy. □

As shown in Figure 7.4, our iterative update converges in just 5 to 7 iterations to embed deep feature representations needed for an image retrieval task tested on 3 datasets, as further detailed in the experimental section.

Complexity Analysis

The graph Laplacian based on the Gaussian kernel in our method is sparse, and computing the sparse matrix-vector product for this specific graph Laplacian has been studied to take $\mathcal{O}(n)$

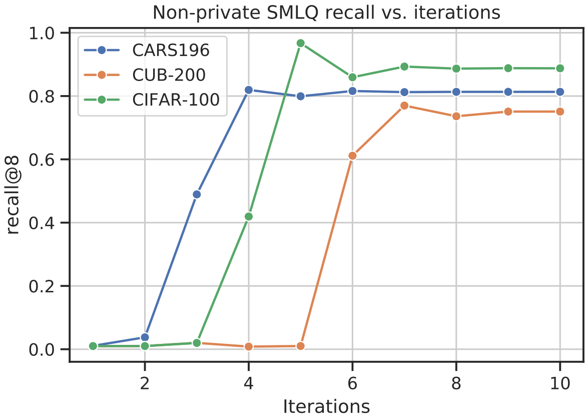


Figure 7.4: The convergence of our SMLQ across three datasets is shown with respect to image recall based on feature embeddings over the iterations. All three datasets reasonably converge in as quick as 7 iterations. The image recall metric is discussed in the Experiments section.

time[15]. Since in the term $\mathbf{L_Y X}_{t-1}$, the number of columns in $\mathbf{X}_{t-1}$ is k, we have an overall time complexity of $\mathcal{O}(nk)$ as the addition of $n \times k$ matrices also takes $\mathcal{O}(nk)$. That said, this does not include the complexity required to construct the Laplacian. This has been studied in[448].

7.5 Privatization of the Supervised Manifold Learning Query

Post-Processing Invariance

Differential privacy is immune to post-processing, meaning that an adversary without any additional knowledge about the dataset $\mathbf{X}$ cannot compute a function on the output $\mathcal{A}(\mathbf{X})$ to violate the stated privacy guarantees.

Gaussian Noise Mechanism

A query on a dataset can be privatized by adding controlled noise from a predetermined distribution. One popular private mechanism is the Gaussian mechanism[155], which adds Gaussian noise depending on the query's *sensitivity*.

Definition 13 (l_2-sensitivity)**.** *Let $f : \mathcal{X} \to \mathbb{R}^k$. The l_2-sensitivity of f is*

$$\Delta_2^{(f)} = \max_{\mathbf{X},\mathbf{X}' \in \mathcal{X}} \|f(\mathbf{X}) - f(\mathbf{X}')\|_2$$

where $\mathbf{X}, \mathbf{X}'$ are neighboring databases.

Definition 14 (Gaussian Mechanism[160])**.** *Let $f : \mathcal{X} \to \mathbb{R}^k$. The Gaussian mechanism is defined as $\mathcal{M}_G(\mathbf{X}) = f(\mathbf{X}) + \mathbf{Y}$, where $\mathbf{Y} \sim \mathcal{N}^k(0, \sigma^2)$ with $\sigma \geq \frac{\sqrt{2\ln(1.25\ \delta)}\Delta_2^{(f)}}{\epsilon}$. The Gaussian mechanism is (ϵ, δ)-differentially private.*

We use the above mechanism to privatize the SMLQ, for which we derive the sensitivity. Note that the query's utility could be improved even further via the more recent analytical Gaussian mechanism in[42].

7.5.1 Derivation of SMLQ Sensitivity

We derive a bound on the sensitivity for the first iteration of the SMLQ, $f(\mathbf{X}) = \frac{1}{2}\text{diag}(\mathbf{L_X})^\dagger \left[\alpha \mathbf{L_Y} - \mathbf{L_X}\right] \mathbf{Q} + \mathbf{Q}$, where we initialize $\mathbf{X_0}$ to a matrix $\mathbf{Q}$ such that each entry is distributed as $\mathbf{Q}_{ij} \sim \mathcal{N}(0, \sigma_q^2)$, for which σ_q is a hyperparameter chosen by the user. It is typical to use random initialization for iterative optimization. We also assume that $\mathbf{X} \in \mathbb{R}^{n \times k}$ is normalized to have unit norm rows. Under all possible cases of adding one additional unit norm record to $\mathbf{X}$ to produce a neighboring dataset $\tilde{\mathbf{X}} \in \mathbb{R}^{(n+1) \times k}$ (denoted by the constraint $d(\mathbf{X}, \tilde{\mathbf{X}}) = 1$), the sensitivity

PrivateMail

1. **Client's input:** Raw data (or activations) $\mathbf{X}$ normalized to have unit norm rows and integer labels $\mathbf{Y}$, Gaussian kernel bandwidth σ, regularizing parameter α, variance σ_q^2 for random embedding initialization.
2. **Client computes embedding:** $\mathbf{X_t} = \frac{1}{2}\,\mathrm{diag}(\mathbf{L_X})^{\dagger}[\alpha\mathbf{L_Y} - \mathbf{L_X}]\mathbf{X_{t-1}} + \mathbf{X_{t-1}}$ with initialization $\mathbf{X_0} = \mathbf{Q}$ such that $\mathbf{Q}_{ij} \sim \mathcal{N}(0, \sigma_q^2)$, $\mathbf{L_X}$ and $\mathbf{L_Y}$ are graph Laplacians formed over adjacency matrices upon applying Gaussian kernels to $\mathbf{X}, \mathbf{Y}$ with bandwidth σ.
3. **Client side privatization:** The client takes the following actions:
   (a) **Initialization:** Compute constant M that depends on chosen α, σ and data size n as defined in appendix.
   (b) **Computation of global-sensitivity:** Compute upper bound on global sensitivity as $\Delta = \frac{M\sqrt{n+1}}{2}\|\mathbf{Q}\|_F$
   (c) **Add differentially private noise** Release $\mathbf{X_t}$ with the global sensitivity upper bound in step $3(b)$ via the (ϵ, δ)- differentially private multi-dimensional Gaussian mechanism: $\mathbf{X_t} + \mathcal{N}^{n\times k}\left(\mu = 0, \sigma^2 = \frac{2\ln(1.25/\delta)\cdot\Delta^2}{\epsilon^2}\right)$

Figure 7.5: Protocol for the proposed PrivateMail mechanism

of our query is defined as $\Delta_2^{(f)} = \max_{\mathbf{X},\tilde{\mathbf{X}}:d(\mathbf{X},\tilde{\mathbf{X}})=1} \|f(\mathbf{X}) - f(\tilde{\mathbf{X}})\|_F$. Note that we append an extra row of zeroes to $\mathbf{X}$ and $\mathbf{Y}$ such that the matrix dimensions agree with $\tilde{\mathbf{X}}$ and $\tilde{\mathbf{Y}}$ when evaluating $f(\mathbf{X}) - f(\tilde{\mathbf{X}})$. To simplify further calculations, we let $\mathbf{M}$ denote the matrix defined by

$$\mathbf{M}(\mathbf{X}, \tilde{\mathbf{X}}) = \mathrm{diag}(\mathbf{L_X})^{\dagger}\left[\alpha\mathbf{L_Y} - \mathbf{L_X}\right] - \mathrm{diag}(\mathbf{L}_{\tilde{\mathbf{X}}})^{\dagger}\left[\alpha\mathbf{L}_{\tilde{\mathbf{Y}}} - \mathbf{L}_{\tilde{\mathbf{X}}}\right] \tag{7.5}$$

and let $\mathbf{M_i}$ denote the ith row of $\mathbf{M}$.

Theorem 7.5.1. **SMLQ sensitivity bound** We have that, $\Delta_2^{(f)} \leq \frac{M\sqrt{n+1}}{2}\|\mathbf{Q}\|_F$. where M is a constant defined in appendix such that $M \geq \|\mathbf{M_i}\|$ for all $\mathbf{X}$ and $\tilde{\mathbf{X}}$.

Proof. Note that $f(\mathbf{X}) - f(\tilde{\mathbf{X}})$ may be expressed as the product $\frac{1}{2}\mathbf{MQ}$. Thus, by submultiplicativity of the Frobenius norm, the global sensitivity is bounded by

$$\begin{aligned}\Delta_2^{(f)} &= \max_{\mathbf{X},\tilde{\mathbf{X}}:d(\mathbf{X},\tilde{\mathbf{X}})=1} \left\| \frac{1}{2}\mathbf{MQ} \right\|_F \\ &\leq \frac{1}{2}\|\mathbf{Q}\|_F \cdot \max_{\mathbf{X},\tilde{\mathbf{X}}:d(\mathbf{X},\tilde{\mathbf{X}})=1} \|\mathbf{M}\|_F \qquad (7.6)\end{aligned}$$

Since $\|\mathbf{M}\|_F = \sqrt{\sum_{i=1}^{n+1} \|\mathbf{M_i}\|^2}$, then if M is a constant as defined in the theorem, we have $\|\mathbf{M}\|_F \leq \sqrt{\sum_{i=1}^{n+1} M^2} = M\sqrt{n+1}$. Substituting this expression into the above inequality, we obtain the bound in the theorem. The derivation of a constant M relies on expanding the definition of the Laplacian matrices in equation 7.4 and applying the law of cosines for the difference of vectors. For the full derivation, see the appendix. □

The above bound on $\Delta_2^{(f)}$ is computed for the sensitivity parameter when adding differentially private noise to the data embedding. Figure 4 summarizes the procedure for privatization, which we call *PrivateMail*.

7.5.2 Private Iteration-Distribute-Recursion Framework

We show that the proposed SMLQ, fortunately, can be applied under a specific framework that we propose so that it can be used in conjunction with the post-processing property of differential privacy to its advantage in obtaining a much better trade-off of utility and privacy. In addition, it allows for distributing the work required to complete the iterative embedding across multiple distributed entities while still preserving privacy. This helps further reduce the computational requirements of the client device prior to distributing the work. The framework still holds in improving the utility-privacy trade-off even if used without distributing the computation. We notice that the only term

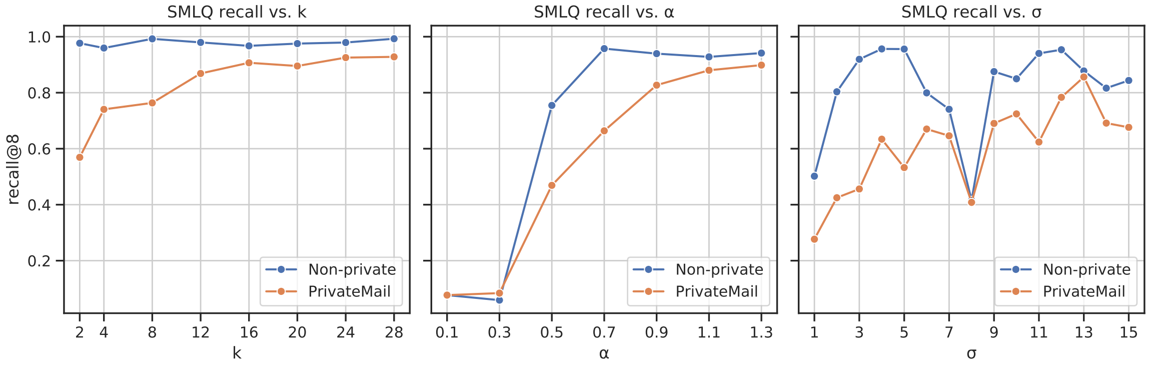


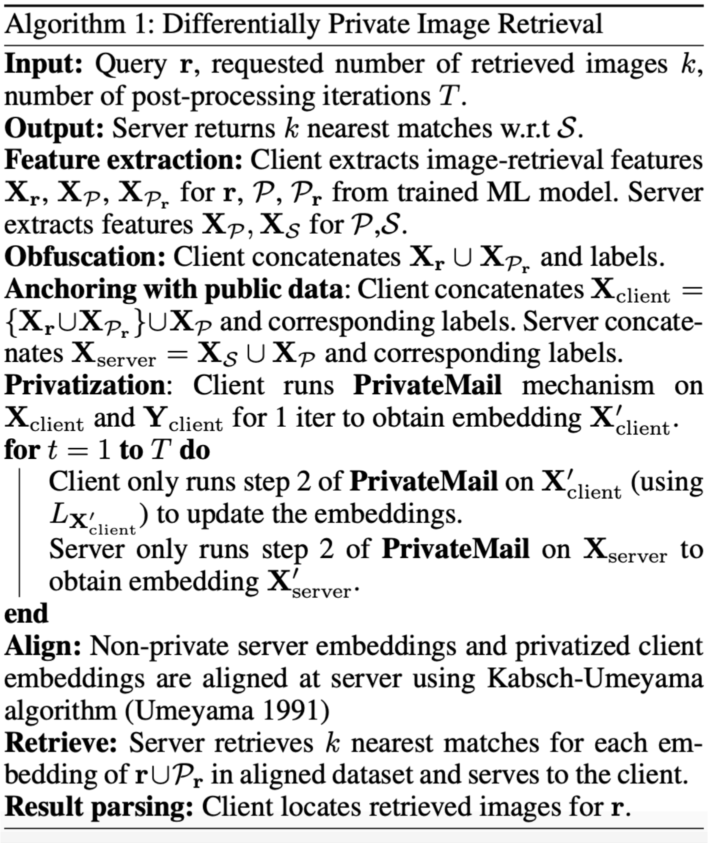
Figure 7.6: The effect of k, α, and σ on retrieval performance with the non-private SMLQ and the private version of PrivateMail.

that requires accessing the sensitive raw dataset is $\mathbf{L_X}$. Still, the good thing is that this term does not change over iterations and hence is not sub-scripted by iteration t as we show in equation 7.4. Therefore, we first apply our proposed differentially private release of PrivateMail to just the first iteration. The privately obtained embedding is instead used this time to rebuild the graph Laplacian $\mathbf{L_X}$. From the next iteration onwards, this modified Laplacian is used instead, and the post-processing property of differential privacy now holds as no iteration from now on needs access to the raw dataset. For this reason, these iterations can be continued over the server or another device as opposed to the original client device that runs the first PrivateMail iteration.

7.5.3 PrivateMail for Image Retrieval

We apply the proposed PrivateMail mechanism to the task of private content-based image retrieval, where a client seeks to retrieve the k-nearest neighbors of their target image $\mathbf{r}$ from a server's database $\mathcal{S}$ based on the feature embedding of their target which is sent to the server. The objective is to preserve the privacy of the client's target image.

We assume the setting in which the client and server have access to a relevant public database $\mathcal{P}$ of images. We propose a differentially private image retrieval algorithm where we first generate feature vectors for $\mathbf{r}$, $\mathcal{P}$, and $\mathcal{S}$ using any feature extraction model of choice. We then generate low-dimensional embeddings for these features using the SMLQ in equation 7.4. Since the query relies on the graph Laplacian of a dataset, a single target image feature is insufficient to generate its embedding. Therefore, the client concatenates $\mathbf{r}$ with the public dataset $\mathcal{P}$. The client runs one iteration of PrivateMail, where noise is added via the Gaussian mechanism before recomputing the Laplacian over the private embedding. This makes the next iterations that we run to be differentially private due to the post-processing invariance property, as the iteration is now functionally independent of the raw features. We then run post-processing embeddings for a varying number of iterations depending on the dataset. Furthermore, since the client and server have access to different data, the embedding of $\mathbf{r} \cup \mathcal{P}$ on the client is not guaranteed to align with that of $\mathcal{S}$ on the server. We thus also concatenate $\mathcal{S}$ with $\mathcal{P}$ so the public data serves as a common "anchor" for the embeddings, which is used to align the embeddings of $\mathbf{r}$ and $\mathcal{S}$ via the Kabsch-Umeyama rigid-transformation algorithm[520]. Once the server retrieves the k-nearest neighbors of the client's privatized embedding of $\mathbf{r}$ with respect to the server's non-private embedding of $\mathcal{S}$, the server gains additional information about $\mathbf{r}$ based on its neighbors. To obfuscate $\mathbf{r}$, we append a dataset $\mathcal{P}_{\mathbf{r}}$ of dummy queries to $\mathbf{r} \cup \mathcal{P}$ on the client side. $\mathcal{P}_{\mathbf{r}}$ is generated by uniformly sampling images from the public dataset such that $\mathcal{P}_{\mathbf{r}}$ contains one image of every class besides the class of $\mathbf{r}$. The client's target image class is equally likely to be any of the possible classes in the dataset, so the server cannot directly infer the target class. The



```
Algorithm 1: Differentially Private Image Retrieval
Input: Query $\mathbf{r}$, requested number of retrieved images $k$, number of post-processing iterations $T$.
Output: Server returns $k$ nearest matches w.r.t $\mathcal{S}$.
Feature extraction: Client extracts image-retrieval features $\mathbf{X}_{\mathbf{r}}$, $\mathbf{X}_{\mathcal{P}}$, $\mathbf{X}_{\mathcal{P}_{\mathbf{r}}}$ for $\mathbf{r}$, $\mathcal{P}$, $\mathcal{P}_{\mathbf{r}}$ from trained ML model. Server extracts features $\mathbf{X}_{\mathcal{P}}, \mathbf{X}_{\mathcal{S}}$ for $\mathcal{P}$,$\mathcal{S}$.
Obfuscation: Client concatenates $\mathbf{X}_{\mathbf{r}} \cup \mathbf{X}_{\mathcal{P}_{\mathbf{r}}}$ and labels.
Anchoring with public data: Client concatenates $\mathbf{X}_{\text{client}} = \{\mathbf{X}_{\mathbf{r}} \cup \mathbf{X}_{\mathcal{P}_{\mathbf{r}}}\} \cup \mathbf{X}_{\mathcal{P}}$ and corresponding labels. Server concatenates $\mathbf{X}_{\text{server}} = \mathbf{X}_{\mathcal{S}} \cup \mathbf{X}_{\mathcal{P}}$ and corresponding labels.
Privatization: Client runs PrivateMail mechanism on $\mathbf{X}_{\text{client}}$ and $\mathbf{Y}_{\text{client}}$ for 1 iter to obtain embedding $\mathbf{X}'_{\text{client}}$.
for $t = 1$ to $T$ do
    Client only runs step 2 of PrivateMail on $\mathbf{X}'_{\text{client}}$ (using $L_{\mathbf{X}'_{\text{client}}}$) to update the embeddings.
    Server only runs step 2 of PrivateMail on $\mathbf{X}_{\text{server}}$ to obtain embedding $\mathbf{X}'_{\text{server}}$.
end
Align: Non-private server embeddings and privatized client embeddings are aligned at server using Kabsch-Umeyama algorithm (Umeyama 1991)
Retrieve: Server retrieves $k$ nearest matches for each embedding of $\mathbf{r} \cup \mathcal{P}_{\mathbf{r}}$ in aligned dataset and serves to the client.
Result parsing: Client locates retrieved images for $\mathbf{r}$.
```

Figure 7.7: Differentially private retrieval

client is then able to filter out the retrieved images for the dummy targets. This process is visualized in Figure 1 and described in greater detail in Algorithm 1.

7.6 Experiments

Datasets

In this section we present experimental results on three important image retrieval benchmark datasets of i) Caltech-UCSD Birds-200-2011 (CUB-200-2011)[562], ii) Cars196[291], and iii) CIFAR-100[294].

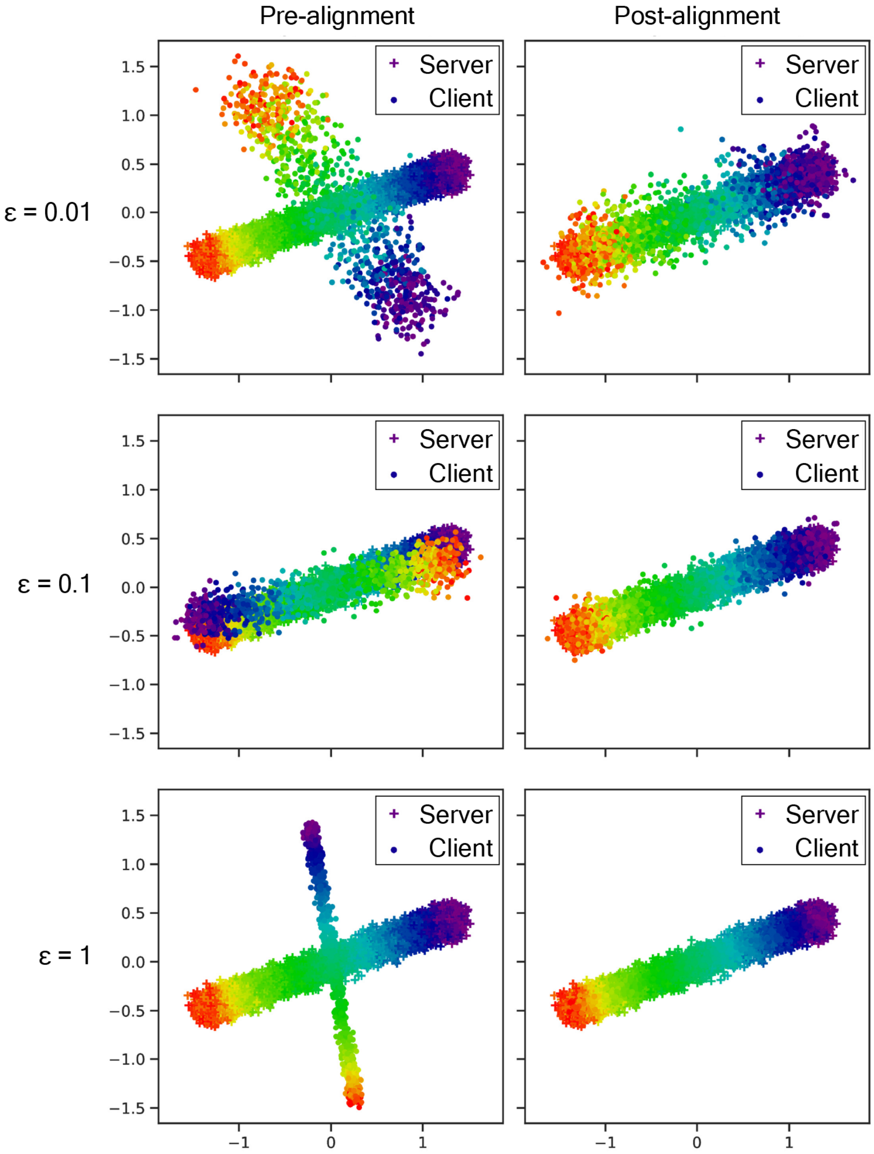


Figure 7.8: Embeddings for CARS196 data (with $\alpha = 0.5$ and parameters in appendix A) at varying privacy levels ε. We show that alignment improves as less noise is added. The privacy-induced noise can be seen at various levels of ε.

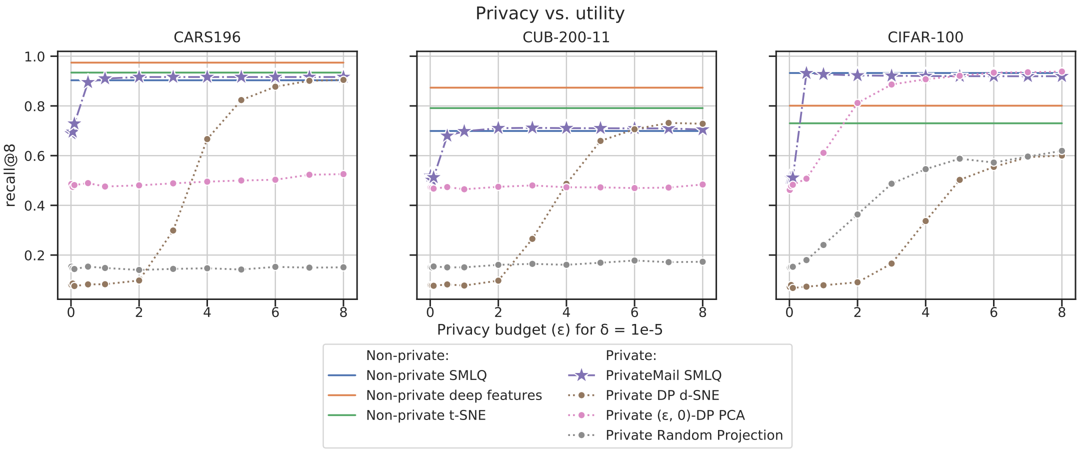


Figure 7.9: We compare the privacy-utility trade-off of PrivateMail with Recall@k experiments with $k = 8$ for three datasets on 6 baselines that include private and non-private methods. The lower values of ϵ refer to higher levels of privacy.

Methodology

We use the state-of-the-art image retrieval method of *'combination of multiple global descriptors'* (CGD)[259] with ResNet-50[220] backbone to generate features for the Cars196 and CUB-200-2011 datasets. CIFAR-100 features are extracted directly from ResNet-50 pre-trained on ImageNet[136]. We run Algorithm 1 on each dataset with the parameters outlined in appendix A.

Quantitative metrics

We measure retrieval performance using the Recall@k metric as used in this popular non-private image retrieval paper[259]. As our proposed work is a differentially private algorithm, we study the *utility-privacy trade-off* by looking at the recalls obtained at varying levels of ϵ. Note that lower ϵ refers to higher privacy.

7.6.1 Baselines

We compare the utility of our proposed PrivateMail mechanism against several important baselines as below.

Non-private state of the art for image retrieval We compare against the non-private method of CGD that unfortunately does not preserve privacy, and see how close we get to its performance while also preserving privacy. Note that there exists a trade-off of privacy vs utility, and the main goal is to preserve privacy while attempting to maximize utility.

Differentially private unsupervised manifold embedding A comparison with the differentially private unsupervised manifold embedding method of DP-dSNE[440,441,439] is done as this is one of the most recent manifold embedding methods with differential privacy.

Non-private supervised manifold embedding We compare against non-private supervised manifold embedding to show how close our differentially private version fares in terms of achievable utility when the privacy is not at all preserved.

Non-private unsupervised manifold embedding We compare against the non-private unsupervised manifold embedding method of t-SNE[531] to show the benefit of a supervised manifold embedding over an unsupervised embedding in terms of the utility.

Differentially private classical projections We compare against differentially private versions of more classical methods such as private PCA[105] and private random projections[276].

7.6.2 Evaluation

As shown in Figure 7.9, PrivateMail SMLQ obtains a substantially better privacy-utility trade-off over all the considered private baselines on all the datasets. It also reaches closer to the methods that do not preserve privacy on CARS196. It even meets the non-private performance on CIFAR-100 at much higher levels of privacy (lower ϵ's). DP-dSNE reaches the performance of PrivateMail only at low levels of privacy on 2 out of the 3 datasets, while PrivateMail does substantially better at high levels of privacy preservation. A similar phenomenon happens again with respect to private PCA on CIFAR-100.

Sensitivity to Hyper-Parameters

In Figure 7.6, we study the sensitivity of our method's performance with respect to various parameters such as choice of embedding dimension k, the weighting parameter α which acts as a regularizer for the embedding by weighting the graph Laplacians in the term $\mathbf{L}_\mathbf{X} - \alpha \mathbf{L}_\mathbf{Y}$ in our embedding update, and the σ parameter used in defining the Gaussian kernels used to build $\mathbf{L}_\mathbf{X}, \mathbf{L}_\mathbf{Y}$. As shown, tuning of k, α is stable while tuning of σ requires a bit of a grid search. However, since we are in a supervised setting, standard methods for tuning could be used for practical purposes.

Qualitative Visualizations

Example of PrivateMail embeddings are given in Figure 7.8 for different values of privacy parameter ϵ pre- and post- server-client alignment.

7.7 Conclusion

We proposed a differentially private supervised manifold learning method and applied it to the private image retrieval problem. That said, there is a broad range of applications for manifold learning beyond that of image retrieval. Therefore, it would be interesting to investigate the potential benefits of doing these other tasks in a privacy-preserving manner. We would like to extend the derived global sensitivity results to smooth sensitivities[388] in order to potentially further improve the privacy-utility trade-off.

7.8 Experiment parameters

Unless noted otherwise, we use the following parameters for the SMLQ experiments.

Parameter	**CARS196**	**CUB-200-2011**	**CIFAR-100**
σ	6	5	6
α	0.6	0.5	0.6
k	2	2	2
σ_q	10^{-8}	10^{-8}	10^{-8}
T	5	5	5
ϵ	0.1	0.1	0.1
δ	10^{-5}	10^{-5}	10^{-5}

Table 7.2: Default experiment parameters

7.9 Global sensitivity derivation

The proof of Theorem 7.5.1 relies on deriving a constant bound M such that $M \geq \|\mathbf{M_i}\|$ for all $\mathbf{X}$ and $\tilde{\mathbf{X}}$, where $\mathbf{M_i}$ is the i-th row of $\mathbf{Z}$ as defined in equation equation 7.5.

Lemma 7.9.1. *For all* $\mathbf{X}$ *and* $\tilde{\mathbf{X}}$*,* $M = nM_{ij} + M_{ii} \geq \|\mathbf{M_i}\|$*, where* $M_{ii}^{\max}$ *and* $M_{ij}^{\max}$ *are defined as*

$$M_{ii} = \alpha^2 \left[\left(\frac{n}{ne^{-\frac{2}{\sigma^2}} + e^{-\frac{1}{2\sigma^2}} - 1} \right)^2 + \left(\frac{n}{(n+1)e^{-\frac{2}{\sigma^2}} - 1} \right)^2 - \frac{2\left((n+1)e^{-\frac{c^2}{2\sigma^2}} - 1 \right)^2}{n\left(n + e^{-\frac{1}{2\sigma^2}} - 1 \right)} \right] \tag{7.7}$$

$$\begin{aligned} M_{ij} = {} & \frac{\alpha^2 + 1}{\left(ne^{-\frac{2}{\sigma^2}} + e^{-\frac{1}{2\sigma^2}} - 1 \right)^2} - \frac{2\alpha e^{-\frac{c^2+4}{2\sigma^2}}}{\left(n + e^{-\frac{1}{2\sigma^2}} - 1 \right)^2} + \frac{\alpha^2 + 1}{\left((n+1)e^{-\frac{2}{\sigma^2}} - 1 \right)^2} - \frac{2\alpha e^{-\frac{c^2+4}{2\sigma^2}}}{n^2} \\ & -2 \cdot \frac{\alpha^2 e^{-\frac{c^2}{\sigma^2}} + e^{-\frac{4}{\sigma^2}}}{n\left(n + e^{-\frac{1}{2\sigma^2}} - 1 \right)} + \frac{4\alpha}{\left(ne^{-\frac{2}{\sigma^2}} + e^{-\frac{1}{2\sigma^2}} - 1 \right)\left((n+1)e^{-\frac{2}{\sigma^2}} - 1 \right)} \end{aligned} \tag{7.8}$$

Proof. Recall that we denote the i-th row of $\mathbf{X}$ by $\mathbf{X_i}$. $\mathbf{L}_{\tilde{\mathbf{X}}}$, $\mathbf{L_Y}$, and $\mathbf{L}_{\tilde{\mathbf{Y}}}$ are defined similarly for $\tilde{\mathbf{X}}$, $\mathbf{Y}$, and $\tilde{\mathbf{Y}}$ respectively. Expanding the definition of $\mathbf{M_i}$,

$$\begin{aligned} \|\mathbf{M_i}\| &= \sum_{j=1}^{n+1} \left[\operatorname{diag}(\mathbf{L_X})\dagger \left[\alpha \mathbf{L_Y} - \mathbf{L_X}\right] - \operatorname{diag}(\mathbf{L}_{\tilde{\mathbf{X}}})\dagger \left[\alpha \mathbf{L}_{\tilde{\mathbf{Y}}} - \mathbf{L}_{\tilde{\mathbf{X}}}\right]\right]_{\mathbf{i,j}}^2 \\ &= \sum_{j=1}^{n+1} \left(\left[\operatorname{diag}(\mathbf{L_X})\dagger \left[\alpha \mathbf{L_Y} - \mathbf{L_X}\right]\right]_{\mathbf{i,j}} - \left[\operatorname{diag}(\mathbf{L}_{\tilde{\mathbf{X}}})\dagger \left[\alpha \mathbf{L}_{\tilde{\mathbf{Y}}} - \mathbf{L}_{\tilde{\mathbf{X}}}\right]\right]_{\mathbf{i,j}} \right)^2 \\ &= \sum_{j=1}^{n+1} \begin{pmatrix} \left[\operatorname{diag}(\mathbf{L_X})\dagger \left[\alpha \mathbf{L_Y} - \mathbf{L_X}\right]\right]_{\mathbf{i,j}}^2 + \left[\operatorname{diag}(\mathbf{L}_{\tilde{\mathbf{X}}})\dagger \left[\alpha \mathbf{L}_{\tilde{\mathbf{Y}}} - \mathbf{L}_{\tilde{\mathbf{X}}}\right]\right]_{\mathbf{i,j}}^2 \\ -2 \left[\operatorname{diag}(\mathbf{L_X})\dagger \left[\alpha \mathbf{L_Y} - \mathbf{L_X}\right]\right]_{\mathbf{i,j}} \left[\operatorname{diag}(\mathbf{L}_{\tilde{\mathbf{X}}})\dagger \left[\alpha \mathbf{L}_{\tilde{\mathbf{Y}}} - \mathbf{L}_{\tilde{\mathbf{X}}}\right]\right]_{\mathbf{i,j}} \end{pmatrix} \end{aligned}$$

Since the off-diagonal entries of $\mathrm{diag}(\mathbf{L}_{\mathbf{X}})$ and $\mathrm{diag}(\mathbf{L}_{\tilde{\mathbf{X}}})$ are zero, we have

$$\begin{aligned}
[\mathrm{diag}(\mathbf{L}_{\mathbf{X}})\dagger\,[\alpha\mathbf{L}_{\mathbf{Y}}-\mathbf{L}_{\mathbf{X}}]]_{\mathbf{i,j}} &= \sum_{k=1}^{n}\mathrm{diag}(\mathbf{L}_{\mathbf{X}})\dagger_{\mathbf{i,k}}\,[\alpha\mathbf{L}_{\mathbf{Y}}-\mathbf{L}_{\mathbf{X}}]_{\mathbf{k,j}}\\
&= \mathrm{diag}(\mathbf{L}_{\mathbf{X}})\dagger_{\mathbf{i,i}}\,[\alpha\mathbf{L}_{\mathbf{Y}}-\mathbf{L}_{\mathbf{X}}]_{\mathbf{i,j}} = \frac{\alpha\mathbf{L}_{\mathbf{Y}\mathbf{i,j}}-\mathbf{L}_{\mathbf{X}\mathbf{i,j}}}{\mathbf{L}_{\mathbf{X}\mathbf{i,i}}}\\
[\mathrm{diag}(\mathbf{L}_{\tilde{\mathbf{X}}})\dagger\,[\alpha\mathbf{L}_{\tilde{\mathbf{Y}}}-\mathbf{L}_{\tilde{\mathbf{X}}}]]_{\mathbf{i,j}} &= \frac{\alpha\mathbf{L}_{\tilde{\mathbf{Y}}\mathbf{i,j}}-\mathbf{L}_{\tilde{\mathbf{X}}\mathbf{i,j}}}{\mathbf{L}_{\tilde{\mathbf{X}}\mathbf{i,i}}}
\end{aligned}$$

Therefore, the norm of $\mathbf{M_i}$ is given by

$$\|\mathbf{M_i}\| = \sum_{j=1}^{n+1}\left(\left(\frac{\alpha\mathbf{L}_{\mathbf{Y}\mathbf{i,j}}-\mathbf{L}_{\mathbf{X}\mathbf{i,j}}}{\mathbf{L}_{\mathbf{X}\mathbf{i,i}}}\right)^2+\left(\frac{\alpha\mathbf{L}_{\tilde{\mathbf{Y}}\mathbf{i,j}}-\mathbf{L}_{\tilde{\mathbf{X}}\mathbf{i,j}}}{\mathbf{L}_{\tilde{\mathbf{X}}\mathbf{i,i}}}\right)^2-2\frac{\left(\alpha\mathbf{L}_{\mathbf{Y}\mathbf{i,j}}-\mathbf{L}_{\mathbf{X}\mathbf{i,j}}\right)\left(\alpha\mathbf{L}_{\tilde{\mathbf{Y}}\mathbf{i,j}}-\mathbf{L}_{\tilde{\mathbf{X}}\mathbf{i,j}}\right)}{\mathbf{L}_{\mathbf{X}\mathbf{i,i}}\mathbf{L}_{\tilde{\mathbf{X}}\mathbf{i,i}}}\right) \tag{7.9}$$

We bound the above summation by bounding each summand,

$$M_{ij} = \left(\frac{\alpha\mathbf{L}_{\mathbf{Y}\mathbf{i,j}}-\mathbf{L}_{\mathbf{X}\mathbf{i,j}}}{\mathbf{L}_{\mathbf{X}\mathbf{i,i}}}\right)^2+\left(\frac{\alpha\mathbf{L}_{\tilde{\mathbf{Y}}\mathbf{i,j}}-\mathbf{L}_{\tilde{\mathbf{X}}\mathbf{i,j}}}{\mathbf{L}_{\tilde{\mathbf{X}}\mathbf{i,i}}}\right)^2-2\frac{\left(\alpha\mathbf{L}_{\mathbf{Y}\mathbf{i,j}}-\mathbf{L}_{\mathbf{X}\mathbf{i,j}}\right)\left(\alpha\mathbf{L}_{\tilde{\mathbf{Y}}\mathbf{i,j}}-\mathbf{L}_{\tilde{\mathbf{X}}\mathbf{i,j}}\right)}{\mathbf{L}_{\mathbf{X}\mathbf{i,i}}\mathbf{L}_{\tilde{\mathbf{X}}\mathbf{i,i}}} \tag{7.10}$$

Recall that the $(n+1)$-th row of $\mathbf{X}$ and $\mathbf{Y}$ is $\mathbf{0}$. By the definition of the Laplacian in equation 7.2,

$$\mathbf{L}_{\mathbf{X}\mathbf{i,j}} = \begin{cases}\sum_{k=1}^{n}\exp\left(-\frac{\|\mathbf{X_i}-\mathbf{X_k}\|^2}{2\sigma^2}\right)+\exp\left(-\frac{\|\mathbf{X_i}\|^2}{2\sigma^2}\right)-1 & \text{if } i=j\\ -\exp\left(-\frac{\|\mathbf{X_i}-\mathbf{X_j}\|^2}{2\sigma^2}\right) & \text{otherwise}\end{cases} \tag{7.11}$$

$$\mathbf{L}_{\mathbf{Y}\mathbf{i,j}} = \begin{cases}\sum_{k=1}^{n}\exp\left(-\frac{\|\mathbf{Y_i}-\mathbf{Y_k}\|^2}{2\sigma^2}\right)+\exp\left(-\frac{\|\mathbf{Y_i}\|^2}{2\sigma^2}\right)-1 & \text{if } i=j\\ -\exp\left(-\frac{\|\mathbf{Y_i}-\mathbf{Y_j}\|^2}{2\sigma^2}\right) & \text{otherwise}\end{cases} \tag{7.12}$$

Let $\mathbf{v_X}$ and $\mathbf{v_Y}$ be the additional records in the $(n+1)$-th rows of $\mathbf{L}_{\tilde{\mathbf{X}}}$ and $\mathbf{L}_{\tilde{\mathbf{Y}}}$ respectively. Then similarly to the above definitions of $\mathbf{L}_{\mathbf{Xi,j}}$ and $\mathbf{L}_{\mathbf{Yi,j}}$, we have

$$\mathbf{L}_{\tilde{\mathbf{X}}\mathbf{i,j}} = \begin{cases} \sum_{k=1}^{n} \exp\left(-\frac{\|\mathbf{X_i}-\mathbf{X_k}\|^2}{2\sigma^2}\right) + \exp\left(-\frac{\|\tilde{\mathbf{X}}_\mathbf{i}-\mathbf{v_X}\|^2}{2\sigma^2}\right) - 1 & \text{if } i = j \\ -\exp\left(-\frac{\|\tilde{\mathbf{X}}_\mathbf{i}-\tilde{\mathbf{X}}_\mathbf{j}\|^2}{2\sigma^2}\right) & \text{if } i \neq j \end{cases} \tag{7.13}$$

$$\mathbf{L}_{\tilde{\mathbf{Y}}\mathbf{i,j}} = \begin{cases} \sum_{k=1}^{n} \exp\left(-\frac{\|\mathbf{Y_i}-\mathbf{Y_k}\|^2}{2\sigma^2}\right) + \exp\left(-\frac{\|\tilde{\mathbf{Y}}_\mathbf{i}-\mathbf{v_Y}\|^2}{2\sigma^2}\right) - 1 & \text{if } i = j \\ -\exp\left(-\frac{\|\tilde{\mathbf{Y}}_\mathbf{i}-\tilde{\mathbf{Y}}_\mathbf{j}\|^2}{2\sigma^2}\right) & \text{if } i \neq j \end{cases} \tag{7.14}$$

We proceed to find upper and lower bounds for M_{ij} by separately analyzing two cases: when $i = j$ and when $i \neq j$.

Case 1: $i = j$**.** By equation equation 7.10, we have

$$M_{ii} = \left(\frac{\alpha\mathbf{L}_{\mathbf{Yi,i}} - \mathbf{L}_{\mathbf{Xi,i}}}{\mathbf{L}_{\mathbf{Xi,i}}}\right)^2 + \left(\frac{\alpha\mathbf{L}_{\tilde{\mathbf{Y}}\mathbf{i,i}} - \mathbf{L}_{\tilde{\mathbf{X}}\mathbf{i,i}}}{\mathbf{L}_{\tilde{\mathbf{X}}\mathbf{i,i}}}\right)^2 - 2\frac{\left(\alpha\mathbf{L}_{\mathbf{Yi,i}} - \mathbf{L}_{\mathbf{Xi,i}}\right)\left(\alpha\mathbf{L}_{\tilde{\mathbf{Y}}\mathbf{i,i}} - \mathbf{L}_{\tilde{\mathbf{X}}\mathbf{i,i}}\right)}{\mathbf{L}_{\mathbf{Xi,i}}\mathbf{L}_{\tilde{\mathbf{X}}\mathbf{i,i}}} \tag{7.15}$$

This equation further simplifies to

$$\begin{aligned} M_{ii} &= \frac{\alpha^2\mathbf{L}_{\mathbf{Yi,i}}^2 - 2\alpha\mathbf{L}_{\mathbf{Xi,i}}\mathbf{L}_{\mathbf{Yi,i}} + \mathbf{L}_{\mathbf{Xi,i}}^2}{\mathbf{L}_{\mathbf{Xi,i}}^2} + \frac{\alpha^2\mathbf{L}_{\tilde{\mathbf{Y}}\mathbf{i,i}}^2 - 2\alpha\mathbf{L}_{\tilde{\mathbf{X}}\mathbf{i,i}}\mathbf{L}_{\tilde{\mathbf{Y}}\mathbf{i,i}} + \mathbf{L}_{\tilde{\mathbf{X}}\mathbf{i,i}}^2}{\mathbf{L}_{\tilde{\mathbf{X}}\mathbf{i,i}}^2} \\ &\quad - 2\frac{\alpha^2\mathbf{L}_{\mathbf{Yi,i}}\mathbf{L}_{\tilde{\mathbf{Y}}\mathbf{i,i}} - \alpha\mathbf{L}_{\tilde{\mathbf{X}}\mathbf{i,i}}\mathbf{L}_{\mathbf{Yi,i}} - \alpha\mathbf{L}_{\mathbf{Xi,i}}\mathbf{L}_{\tilde{\mathbf{Y}}\mathbf{i,i}} + \mathbf{L}_{\mathbf{Xi,i}}\mathbf{L}_{\tilde{\mathbf{X}}\mathbf{i,i}}}{\mathbf{L}_{\mathbf{Xi,i}}\mathbf{L}_{\tilde{\mathbf{X}}\mathbf{i,i}}} \\ &= \alpha^2\left(\frac{\mathbf{L}_{\mathbf{Yi,i}}^2}{\mathbf{L}_{\mathbf{Xi,i}}^2} + \frac{\mathbf{L}_{\tilde{\mathbf{Y}}\mathbf{i,i}}^2}{\mathbf{L}_{\tilde{\mathbf{X}}\mathbf{i,i}}^2} - \frac{2\mathbf{L}_{\mathbf{Yi,i}}\mathbf{L}_{\tilde{\mathbf{Y}}\mathbf{i,i}}}{\mathbf{L}_{\mathbf{Xi,i}}\mathbf{L}_{\tilde{\mathbf{X}}\mathbf{i,i}}}\right) \end{aligned} \tag{7.16}$$

Since each row in $\mathbf{X}$, $\tilde{\mathbf{X}}$ is the unit norm, by the law of cosines, we have

$$\begin{aligned} &\|\mathbf{X_i} - \mathbf{X_k}\|^2 = \|\mathbf{X_i}\|^2 + \|\mathbf{X_k}\|^2 - 2\|\mathbf{X_i}\|\|\mathbf{X_k}\| \cos\theta_{\mathbf{X_i},\mathbf{X_k}} = 2 - 2\cos\theta_{\mathbf{X_i},\mathbf{X_k}} \\ &\|\tilde{\mathbf{X}}_\mathbf{i} - \tilde{\mathbf{X}}_\mathbf{k}\|^2 = 2 - 2\cos\theta_{\tilde{\mathbf{X}}_\mathbf{i},\tilde{\mathbf{X}}_\mathbf{k}} \end{aligned} \tag{7.17}$$

The cosine of the angle between two unit vectors falls between -1 and 1. We use this property to bound $\mathbf{L}_{\mathbf{X}\mathbf{i},\mathbf{i}}$,

$$\begin{aligned} \sum_{k=1}^{n} \exp\left(-\frac{2-2(-1)}{2\sigma^2}\right) + \exp\left(-\frac{1}{2\sigma^2}\right) - 1 \leq \mathbf{L}_{\mathbf{X}\mathbf{i},\mathbf{i}} \leq \sum_{k=1}^{n} \exp\left(-\frac{2-2(1)}{2\sigma^2}\right) + \\ \exp\left(-\frac{1}{2\sigma^2}\right) - 1 \\ ne^{-\frac{2}{\sigma^2}} + e^{-\frac{1}{2\sigma^2}} - 1 \leq \mathbf{L}_{\mathbf{X}\mathbf{i},\mathbf{i}} \leq n + e^{-\frac{1}{2\sigma^2}} - 1 \end{aligned} \tag{7.18}$$

as well as $\mathbf{L}_{\tilde{\mathbf{X}}\mathbf{i},\mathbf{i}}$,

$$\begin{aligned} \sum_{k=1}^{n} \exp\left(-\frac{2-2(-1)}{2\sigma^2}\right) + \exp\left(-\frac{2-2(-1)}{2\sigma^2}\right) - 1 \leq \\ \mathbf{L}_{\tilde{\mathbf{X}}\mathbf{i},\mathbf{i}} \leq \sum_{k=1}^{n} \exp\left(-\frac{2-2(1)}{2\sigma^2}\right) + \exp\left(-\frac{2-2(1)}{2\sigma^2}\right) - 1 \\ (n+1)e^{-\frac{2}{\sigma^2}} - 1 \leq \mathbf{L}_{\tilde{\mathbf{X}}\mathbf{i},\mathbf{i}} \leq n \end{aligned} \tag{7.19}$$

$\mathbf{Y}, \tilde{\mathbf{Y}}$ are vectors of integer labels in $\{0, \ldots, c\}$, where $c+1$ is the number of unique classes in the dataset. We then have the constraints $0 <= \|\mathbf{Y_i} - \mathbf{Y_j}\|^2 <=$

c^2, which generate the following bounds for $\mathbf{L}_{\mathbf{Y}\mathbf{i},\mathbf{i}}$,

$$\sum_{k=1}^{n} \exp\left(-\frac{c^2}{2\sigma^2}\right) + \exp\left(-\frac{c^2}{2\sigma^2}\right) - 1 \leq \mathbf{L}_{\mathbf{Y}\mathbf{i},\mathbf{i}} \leq \sum_{k=1}^{n} \exp\left(-\frac{0}{2\sigma^2}\right) + \exp\left(-\frac{0}{2\sigma^2}\right) - 1$$
$$(n+1)e^{-\frac{c^2}{2\sigma^2}} - 1 \leq \mathbf{L}_{\mathbf{Y}\mathbf{i},\mathbf{i}} \leq n \tag{7.20}$$

and similarly for $\mathbf{L}_{\tilde{\mathbf{Y}}\mathbf{i},\mathbf{i}}$,

$$\sum_{k=1}^{n} \exp\left(-\frac{c^2}{2\sigma^2}\right) + \exp\left(-\frac{c^2}{2\sigma^2}\right) - 1 \leq \mathbf{L}_{\tilde{\mathbf{Y}}\mathbf{i},\mathbf{i}} \leq \sum_{k=1}^{n} \exp\left(-\frac{0}{2\sigma^2}\right) + \exp\left(-\frac{0}{2\sigma^2}\right) - 1$$
$$(n+1)e^{-\frac{c^2}{2\sigma^2}} - 1 \leq \mathbf{L}_{\tilde{\mathbf{Y}}\mathbf{i},\mathbf{i}} \leq n \tag{7.21}$$

Combining these bounds with those in equations equation 7.18 and equation 7.19, we bound M_{ii} from above by

$$M_{ii} \leq \alpha^2 \left[\left(\frac{n}{ne^{-\frac{2}{\sigma^2}} + e^{-\frac{1}{2\sigma^2}} - 1}\right)^2 + \left(\frac{n}{(n+1)e^{-\frac{2}{\sigma^2}} - 1}\right)^2 - \frac{2\left((n+1)e^{-\frac{c^2}{2\sigma^2}} - 1\right)^2}{n\left(n + e^{-\frac{1}{2\sigma^2}} - 1\right)} \right]$$
$$= M_{ii} \tag{7.22}$$

Now that we have derived upper bounds for summands of the form M_{ii} in equation 7.10, we bound M_{ij} where $i \neq j$.

Case 2: $i \neq j$. Expanding equation 7.10, we have

$$\begin{aligned} M_{ij} = &= \frac{\alpha^2 \mathbf{L}_{\mathbf{Y}\mathbf{i,j}}^2 + \mathbf{L}_{\mathbf{X}\mathbf{i,j}}^2}{\mathbf{L}_{\mathbf{X}\mathbf{i,i}}^2} - \frac{2\alpha \mathbf{L}_{\mathbf{X}\mathbf{i,j}} \mathbf{L}_{\mathbf{Y}\mathbf{i,j}}}{\mathbf{L}_{\mathbf{X}\mathbf{i,i}}^2} + \frac{\alpha^2 \mathbf{L}_{\tilde{\mathbf{Y}}\mathbf{i,j}}^2 + \mathbf{L}_{\tilde{\mathbf{X}}\mathbf{i,j}}^2}{\mathbf{L}_{\tilde{\mathbf{X}}\mathbf{i,i}}^2} - \frac{2\alpha \mathbf{L}_{\tilde{\mathbf{X}}\mathbf{i,j}} \mathbf{L}_{\tilde{\mathbf{Y}}\mathbf{i,j}}}{\mathbf{L}_{\tilde{\mathbf{X}}\mathbf{i,i}}^2} \\ &- 2 \cdot \frac{\alpha^2 \mathbf{L}_{\mathbf{Y}\mathbf{i,j}} \mathbf{L}_{\tilde{\mathbf{Y}}\mathbf{i,j}} + \mathbf{L}_{\mathbf{X}\mathbf{i,j}} \mathbf{L}_{\tilde{\mathbf{X}}\mathbf{i,j}}}{\mathbf{L}_{\mathbf{X}\mathbf{i,i}} \mathbf{L}_{\tilde{\mathbf{X}}\mathbf{i,i}}} + 2 \cdot \frac{\alpha \mathbf{L}_{\tilde{\mathbf{X}}\mathbf{i,j}} \mathbf{L}_{\mathbf{Y}\mathbf{i,j}} + \alpha \mathbf{L}_{\mathbf{X}\mathbf{i,j}} \mathbf{L}_{\tilde{\mathbf{Y}}\mathbf{i,j}}}{\mathbf{L}_{\mathbf{X}\mathbf{i,i}} \mathbf{L}_{\tilde{\mathbf{X}}\mathbf{i,i}}} \end{aligned} \tag{7.23}$$

Similarly to the previous case, we use the law of cosines in equation 7.17 to bound each term in equation 7.10. Recall that we have already derived bounds for $\mathbf{L}_{\mathbf{X}\mathbf{i,i}}$ and $\mathbf{L}_{\tilde{\mathbf{X}}\mathbf{i,i}}$. The bounds for $\mathbf{L}_{\mathbf{X}\mathbf{i,j}}$ are given by

$$\begin{gathered} -\exp\left(-\frac{2-2(1)}{2\sigma^2}\right) \leq \mathbf{L}_{\mathbf{X}\mathbf{i,j}} \leq -\exp\left(-\frac{2-2(-1)}{2\sigma^2}\right) \\ -1 \leq \mathbf{L}_{\mathbf{X}\mathbf{i,j}} \leq -e^{-\frac{2}{\sigma^2}} \end{gathered} \tag{7.24}$$

and for $\mathbf{L}_{\tilde{\mathbf{X}}\mathbf{i,j}}$ by

$$\begin{gathered} -\exp\left(-\frac{2-2(1)}{2\sigma^2}\right) \leq \mathbf{L}_{\tilde{\mathbf{Y}}\mathbf{i,j}} \leq -\exp\left(-\frac{2-2(-1)}{2\sigma^2}\right) \\ -1 \leq \mathbf{L}_{\tilde{\mathbf{X}}\mathbf{i,j}} \leq -e^{-\frac{2}{\sigma^2}} \end{gathered} \tag{7.25}$$

By the constraint $0 \leq \|\mathbf{Y_i} - \mathbf{Y_j}\|^2 \leq c^2$, bounds for $\mathbf{L}_{\mathbf{Y}\mathbf{i,j}}$ are given by,

$$\begin{gathered} -\exp\left(-\frac{0}{2\sigma^2}\right) \leq \mathbf{L}_{\mathbf{Y}\mathbf{i,j}} \leq -\exp\left(-\frac{c^2}{2\sigma^2}\right) \\ -1 \leq \mathbf{L}_{\mathbf{Y}\mathbf{i,j}} \leq -e^{-\frac{c}{2\sigma^2}} \end{gathered} \tag{7.26}$$

and for $\mathbf{L}_{\tilde{\mathbf{Y}}\mathbf{i},\mathbf{i}}$,

$$\begin{aligned}
-\exp\left(-\frac{0}{2\sigma^2}\right) \leq \mathbf{L}_{\tilde{\mathbf{Y}}\mathbf{i},\mathbf{j}} \leq -\exp\left(-\frac{c^2}{2\sigma^2}\right) \\
-1 \leq \mathbf{L}_{\tilde{\mathbf{Y}}\mathbf{i},\mathbf{j}} \leq -e^{-\frac{c}{2\sigma^2}}
\end{aligned} \tag{7.27}$$

Substituting these bounds into equation 7.23, we bound M_{ij} from above by

$$\begin{aligned}
M_{ij} \leq\ & \frac{\alpha^2(-1)^2 + (-1)^2}{(ne^{-\frac{2}{\sigma^2}} + e^{-\frac{1}{2\sigma^2}} - 1)^2} \\
& - \frac{2\alpha(-e^{-\frac{2}{\sigma^2}})(-e^{-\frac{c^2}{2\sigma^2}})}{(n + e^{-\frac{1}{2\sigma^2}} - 1)^2} + \frac{\alpha^2(-1)^2 + (-1)^2}{((n+1)e^{-\frac{2}{\sigma^2}} - 1)^2} - \frac{2\alpha(-e^{-\frac{2}{\sigma^2}})(-e^{-\frac{c^2}{2\sigma^2}})}{n^2} \\
& - 2 \cdot \frac{\alpha^2(-e^{-\frac{c^2}{2\sigma^2}})^2 + (-e^{-\frac{2}{\sigma^2}})^2}{n(n + e^{-\frac{1}{2\sigma^2}} - 1)} + 2 \cdot \frac{\alpha(-1)^2 + \alpha(-1)^2}{(ne^{-\frac{2}{\sigma^2}} + e^{-\frac{1}{2\sigma^2}} - 1)((n+1)e^{-\frac{2}{\sigma^2}} - 1)} \\
=\ & \frac{\alpha^2 + 1}{(ne^{-\frac{2}{\sigma^2}} + e^{-\frac{1}{2\sigma^2}} - 1)^2} - \frac{2\alpha e^{-\frac{c^2+4}{2\sigma^2}}}{(n + e^{-\frac{1}{2\sigma^2}} - 1)^2} + \frac{\alpha^2 + 1}{((n+1)e^{-\frac{2}{\sigma^2}} - 1)^2} - \frac{2\alpha e^{-\frac{c^2+4}{2\sigma^2}}}{n^2} \\
& - 2 \cdot \frac{\alpha^2 e^{-\frac{c^2}{\sigma^2}} + e^{-\frac{4}{\sigma^2}}}{n(n + e^{-\frac{1}{2\sigma^2}} - 1)} + \frac{4\alpha}{(ne^{-\frac{2}{\sigma^2}} + e^{-\frac{1}{2\sigma^2}} - 1)((n+1)e^{-\frac{2}{\sigma^2}} - 1)} \\
=\ & M_{ij}^{\max}
\end{aligned} \tag{7.28}$$

Therefore, an upper bound for $\|\mathbf{M_i}\|$ is given by

$$\begin{aligned}
\|\mathbf{M_i}\| &\leq \sum_{j \neq i} M_{ij} + M_{ii} \\
&= nM_{ij} + M_{ii} \\
&= M
\end{aligned} \tag{7.29}$$

□

Applying this bound, which holds for all $\mathbf{X}$ and $\tilde{\mathbf{X}}$, to equation equation 7.6 in the proof of Theorem 2, we obtain a bound on the sensitivity of the SMLQ.

7.10 Optimization

Solution without matrix inverses or a step size parameter In this section, we formulate an efficient monotonically convergent solution for the proposed supervised embedding loss where the update does not require a matrix inverse or a step size parameter. In empirical results, we saw that even a few iterations of our solution were good enough to give a great embedding. For brevity, we refer to the embedding $\mathbf{f}(\mathbf{X})$ by $\mathbf{Z}$ in this appendix.

Concave-convex procedure: Special case of majorization minimization

A function $g(\mathbf{Z}_{t+1}, \mathbf{X}_t)$ is said to majorize the function $v(\mathbf{Z})$ at $\mathbf{Z}_t$ provided $v(\mathbf{Z}_t) = g(\mathbf{Z}_t, \mathbf{Z}_t)$ and $v(\mathbf{X}_t) \leq g(\mathbf{X}_t, \mathbf{Z}_{t+1})$ always holds true. The MM iteration guarantees monotonic convergence[237,566,302,610] because of this sandwich inequality that directly arises due to the above definition of majorization functions.

$$v(\mathbf{Z}_{t+1}) \leq g(\mathbf{Z}_{t+1}, \mathbf{X}_t) \leq g(\mathbf{X}_t, \mathbf{X}_t) = v(\mathbf{X}_t)$$

The concave-convex procedure to solve the difference of convex (DC) optimization problems is a special case of MM algorithms as follows. For objective functions $v(\mathbf{X})$ which can be written as a difference of convex functions as $v_{vex}(\mathbf{Z}) + v_{cave}(\mathbf{Z})$ we have the following majorization function that satisfies the two properties described in the beginning of this subsection.

$$v(\mathbf{Z}) \leq v_{vex}(\mathbf{Z}) + v_{cave}(\mathbf{X}) + (\mathbf{Z} - \mathbf{X})^T \nabla v_{cave}(\mathbf{X}) = g(\mathbf{Z}, \mathbf{X}) \tag{7.30}$$

where $g(\mathbf{Z}, \mathbf{Z}) = v(\mathbf{Z})$ and $g(\mathbf{Z}, \mathbf{X}) \geq v(\mathbf{Z})$ when $\mathbf{Z} \neq \mathbf{X}$.

Therefore the majorization minimization iterations are

1. Solve $\frac{\partial g(\mathbf{Z}_{t+1}, \mathbf{X}_t)}{\partial \mathbf{Z}_t + 1} = 0$ for $\mathbf{X}_t$
2. Set $\mathbf{X}_t = \mathbf{Z}_t$ and continue till convergence.

This gives the iteration known as the concave-convex procedure.

$$\nabla v_{vex}(\mathbf{Z}_{t+1}) = -\nabla v_{cave}(\mathbf{Z}_t) \tag{7.31}$$

ITERATIVE UPDATE WITH MONOTONIC CONVERGENCE FOR SMLQ

Proof. We denote by $\mathrm{diag}(\mathbf{L_X})$, a diagonal matrix whose diagonal is the diagonal of $\mathbf{L_X}$. Now, we can build a majorization function over $\mathrm{Tr}\,(\mathbf{X^T L_X X})$, based on the fact that $2\,\mathrm{diag}(\mathbf{L_X}) - \mathbf{L_X}$ is diagonally dominant. This leads to the following inequality for any matrix $\mathbf{M}$ with real entries and of the same dimension as $\mathbf{X}$.

$$\mathrm{Tr}((\mathbf{X} - \mathbf{M})^{\mathbf{T}}[\mathbf{2}\,\mathrm{diag}(\mathbf{L_X}) - \mathbf{L_X}](\mathbf{X} - \mathbf{M})) \geq 0$$

as also used in[547]. We now get the following majorization inequality over $\mathrm{Tr}\,(\mathbf{X^T L_X X})$ by separating it from the above inequality.

$$\mathrm{Tr}\,(\mathbf{X^T L_X X}) + \mathbf{b}(\mathbf{Y}) \leq \mathrm{Tr}\,[\mathbf{2X^T}\,\mathrm{diag}(\mathbf{L_X})\mathbf{X}] -$$
$$2\,\mathrm{Tr}\,[\mathbf{X^T}(\mathbf{2}\,\mathrm{diag}(\mathbf{L_X}) - \mathbf{L_X})\mathbf{M}] = \lambda(\mathbf{X}, \mathbf{M})$$

which is quadratic in $\mathbf{X}$ where, $\mathbf{b}(\mathbf{M}) = \mathrm{Tr}\,(\mathbf{M^T L_X M}) - \mathrm{Tr}\,(\mathbf{M^T 2}\,\mathrm{diag}(\mathbf{L_X})\mathbf{M})$. Let $h(\mathbf{X}, \mathbf{M}) = \lambda(\mathbf{X}, \mathbf{M}) - \alpha\,\mathrm{Tr}\,(\mathbf{X^T L_Y X})$

This leads to the following bound over our loss function with $const(\mathbf{M})$ being a function that only depends on $\mathbf{M}$:

$$\mathbf{G}(\mathbf{X}) + const(\mathbf{M}) \leq h(\mathbf{X}, \mathbf{M})\ \forall \mathbf{X} \neq \mathbf{M}$$
$$= h(\mathbf{X}, \mathbf{X}), \text{ when } \mathbf{X} = \mathbf{M}$$

that satisfies the supporting point requirement, and hence $h(\cdot)$ touches the objective function at the current iterate and forms a majorization function. Now, the following majorization minimization iteration holds true for an iteration t:

$$\mathbf{X}_{t+1} = \underset{\mathbf{X}}{\mathrm{argmin}}\, h(\mathbf{X}, \mathbf{M_t}) \text{ and } \mathbf{M_{t+1}} = \mathbf{X_t}$$

It is important to note that these inequalities occur amongst the presence of additive terms, $\mathbf{const}(\mathbf{M})$ that is independent of $\mathbf{X}$ unlike a typical majorization-minimization framework, and hence, it is a relaxation. The majorization function $\mathbf{h}(\mathbf{X}, \mathbf{M_t})$ can be expressed as a sum of a convex function $e_{vex}(\mathbf{X}) = \lambda(\mathbf{X}, \mathbf{M_t})$ and a concave function $e_{cave}(\mathbf{X}) = -\alpha\,\mathrm{Tr}\,(\mathbf{X^T L_Y X})$. By the concave-convex formulation, we get the iterative solution by solving for $\nabla e_{vex}(\mathbf{X}_t) = -\nabla e_{cave}(\mathbf{X}_{t-1})$ which gives us

$$\mathbf{X_t} = \frac{\alpha}{2}\,\mathrm{diag}(\mathbf{L_X})^{\dagger}\mathbf{L_Y}\mathbf{X_{t-1}} + \mathbf{M_t} - \frac{1}{2}\,\mathrm{diag}(\mathbf{L_X})^{\dagger}\mathbf{L_X}\mathbf{M_t}$$

and on applying the majorization update over $\mathbf{M}_t$, we finally get the supervised manifold learning update that does not require a matrix inversion or a step-size parameter while guaranteeing monotonic convergence. □

If the concave Hessian has small curvature compared to the convex Hessian in the neighborhood of an optima, then CCCP will generally have a superlinear convergence like quasi-Newton methods. This and other characterizations for convergence of CCCP, under various settings, have been studied in great detail in[442].

7.11 Separation-regularity trade-off in supervised manifold learning[549]

Theorem 7.11.1. Let $X = \{x_i\}_{i=1}^{N} \subset \mathbb{R}^n$ be a set of training samples such that each x_i is drawn i.i.d. from one of the probability measures $\{\nu_m\}_{m=1}^{M}$, with ν_m denoting the probability measure of the m-th class. Let $Z = \{y_i\}_{i=1}^{N}$ be an embedding of X in $\mathbb{R}^d$ such that there exist a constant $\gamma > 0$ and a constant A_δ depending on $\delta > 0$ satisfying

$$\|z_i - z_j\| < A_\delta, \text{ if } \|x_i - x_j\| \leqslant 2\delta \text{ and } C\left(x_i\right) = C\left(x_j\right)$$
$$\|z_i - z_j\| > \gamma, \text{ if } C\left(x_i\right) \neq C\left(x_j\right)$$

For given $\epsilon > 0$ and $\delta > 0$, let $f : \mathbb{R}^n \to \mathbb{R}^d$ be a Lipschitz continuous interpolation function with constant L, which maps each x_i to $f\left(x_i\right) = y_i$, such that

$$L\delta + \sqrt{d}\epsilon + A_\delta \leqslant \frac{\gamma}{2}$$

Consider a test sample x randomly drawn according to the probability measure ν_m of class m. For any $Q > 0$, if X contains at least N_m training samples from the m-th class drawn i.i.d. from ν_m such that

$$N_m > \frac{Q}{\eta_{m,\delta}}$$

then the probability of correctly classifying x with 1-NN classification in $\mathbb{R}^d$ is lower bounded as

$$P(\hat{C}(x) = m) \geq 1 - \exp\left(-\frac{2\left(N_m \eta_{m,\delta} - Q\right)^2}{N_m}\right) - 2d \exp\left(-\frac{Q\epsilon^2}{2L^2\delta^2}\right)$$

*“The mathematician needs no laboratories or supplies.
A piece of paper, a pencil, and creative powers form
the foundation of his work.”*

Aleksandr Yakovlevich Khinchin

8

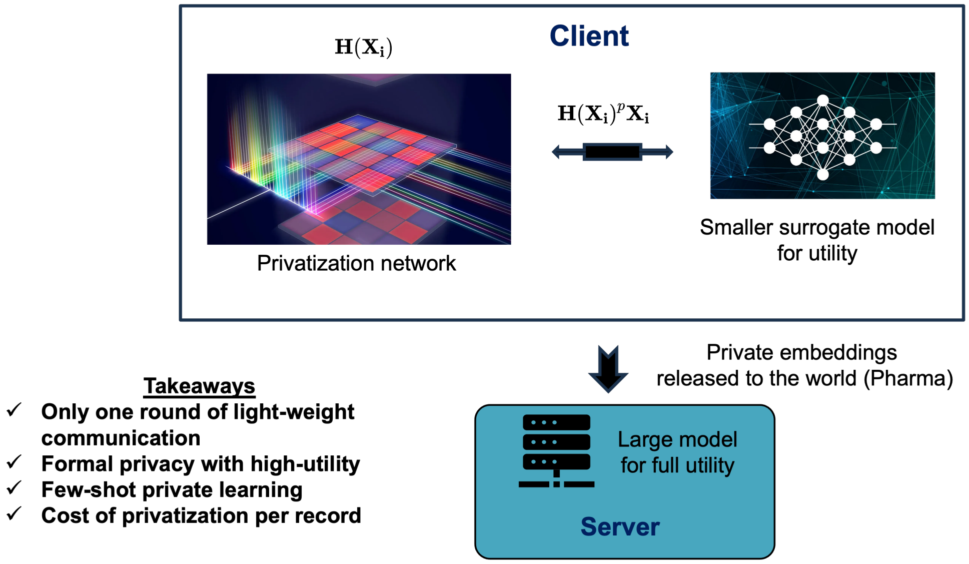
Power Learning for Private Distributed Learning with One Round of Communication

8.1 Introduction

In this chapter, we focus on understanding the requirements for instilling formal guarantees of privacy when the query under consideration is a neural network. We focus on the problem of releasing a privatized version of a sensitive data set (private embeddings) with another entity. The the downstream goal of the entity that, in turn, consumes the released private embeddings for it to perform supervised machine learning with downstream predictive applications. This helps to privatize variants of split learning, a popular form of distributed machine learning that involves the sharing of intermediate activations between clients and the server, as illustrated in Figure 8.2. This chapter is based on our work in[542]. In this method, each client trains the network up to a certain layer known as the cut layer and sends the weights to the server. The server then trains the network for the rest of the layers. This completes the forward propagation. The server then generates the gradients for the final layer and back-propagates the error until the cut layer. The gradient is then passed over to the client. The client completes the rest of the back-propagation. This is continued till the network is trained. In this framework as well, there is no explicit sharing of raw data, but this does not guarantee privacy and hence requires a method like the proposed power mechanism[541] as a variant of this setting that is mathematically (formally) guaranteed to be private.

There are three settings for this problem, namely the interactive setting, semi-interactive setting , and non-interactive setting, as follows. In the interactive setting, a private model is trained over a labeled training set within a trusted entity, and test points need to be sent to this model by the querying user in order to get private predictions. In the semi-interactive setting, a private model is trained over a labeled training set within a trusted entity, just like in the previous setting. Still, the querying user sends a smaller subset of its test dataset to the entity hosting the private model. This model hosting entity then, in turn, sends back a privatized model (i.e. model weights, if a deep learning model) to the querying user. The user then infers from this privatized model to get the predictions. In the non-interactive setting, the model is once again trained just like in the above two settings. But in order to infer, the querying user never sends his test dataset to this private model hosting entity. Instead, upon training, the model hosting entity sends a privatized model (i.e. weights in the case of the deep learning model) to the querying user. The user then infers with this model over his test dataset in order to get the private predictions.

The primary question we consider before proposing an end-to-end method, that we build upon it is as follows. Given a data set with n samples $\mathbf{X}_{k\times n} \in \mathbb{R}^k$, a positive integer $p \in \mathbb{Z}^+$, and an operator $\mathbf{H}_{k\times k} : \mathbf{X_{k\times 1}} \mapsto \mathbf{Z_{k\times 1}}$ such that $\mathbf{Z} = \mathbf{H}(\mathbf{X_i})^p\mathbf{X_i}$; what are the required conditions that need to be satisfied by $\mathbf{H}(X)$ and $\mathbf{p}$ to formally guar-


Client
$\mathbf{H}(\mathbf{X_i})$
Privatization network
$\mathbf{H}(\mathbf{X_i})^p\mathbf{X_i}$
Smaller surrogate model for utility
Private embeddings released to the world (Pharma)
Large model for full utility
Server

Takeaways

- ✓ Only one round of light-weight communication
- ✓ Formal privacy with high-utility
- ✓ Few-shot private learning
- ✓ Cost of privatization per record

Figure 8.1: Systems interactions and takeaways of the power mechanism.

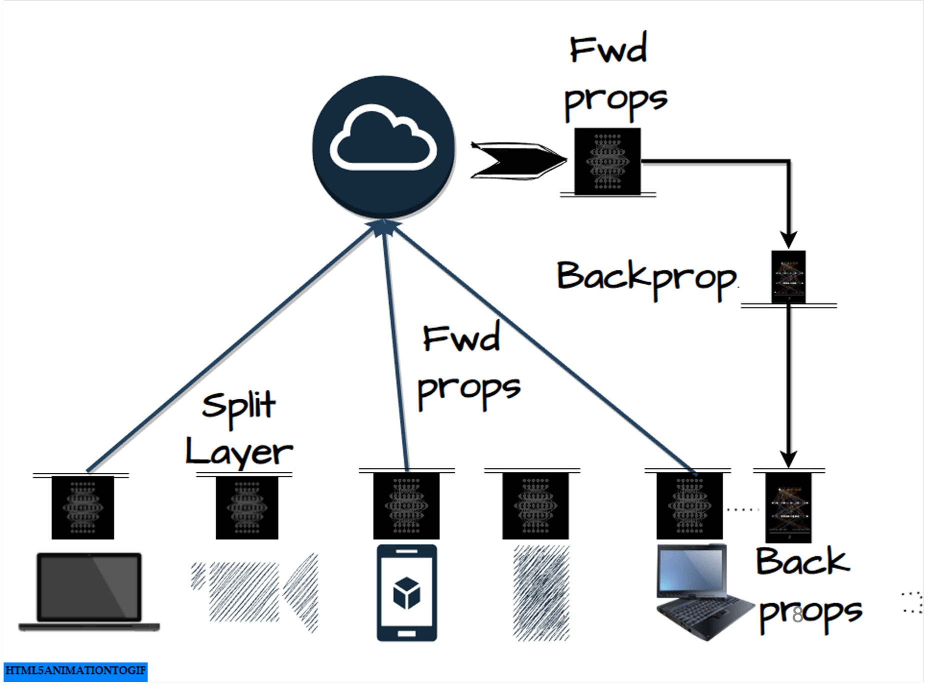


Figure 8.2: Split learning

antee that $\mathbf{Z}$ is ϵ-Lipschitz private with respect to the dataset $\mathbf{X}$? To avoid confusion, we clarify that $\mathbf{H}(\mathbf{X})$ denotes a matrix whose entries depend on $\mathbf{X}$. In the rest of the paper, we use $\mathbf{H}(\mathbf{X})$ and $\mathbf{H}$ interchangeably to mean the same thing without any loss of generality.

8.2 Motivation

A solution to this main question helps us design newer paradigms of distributed machine learning beyond the current approaches of federated learning and split learning. Such a newer paradigm would allow for private distributed machine learning with only one round of communication to the server.

8.2.1 Systems interactions

The client that holds the sensitive data would have a privatization network (solely based on our proposed theoretical result) that helps release private embeddings to the server that trains a complete network for the sake of utility. The privatization network on the client's side generates the private (formally differentially private) embedding in conjunction with a smaller surrogate utility network on the client side to squeeze some of the utility to complete a machine learning task within the formal privatization process of generating private embeddings. These private embeddings would then be sent to the server in one shot that attains complete utility upon training a larger network based on these embeddings. We illustrate this at a high level along with the following benefits of this approach in Figure 8.1.

8.2.2 Benefits of power learning

1. **Train data and test data privacy** Currently, a majority of privacy-preserving deep learning methods for the private release of model weights provide privacy guarantees with regards to training data, as opposed to post-production test data during predictive inference. These methods include the popular DP-SGD[3] and its recent variants. Our method primarily allows for both train-time and test-time privacy, with a better utility, although in the case of privatizing training data.

2. **One round of communication** As our approach generates a privacy-preserving embedding in conjunction with a smaller surrogate network on the client side, just one round of communication from the client to server suffices as opposed to traditional approaches of federated learning or split learning.

3. **Few-shot private learning** Upon applying the power mechanism, we obtain a distribution of ϵ's that indicates the ease of privatization of any given record. This allows us to choose a cut-off threshold based on the needed privacy level that results in a subset of records that maintain that privacy level. Our experiments show that we often perform with competitive test accuracy using just those privatized records. In addition to enabling few-shot private learning, this further reduces communication and potentially acts as a utility function for the cost of privatization per record in the setting of private data markets.

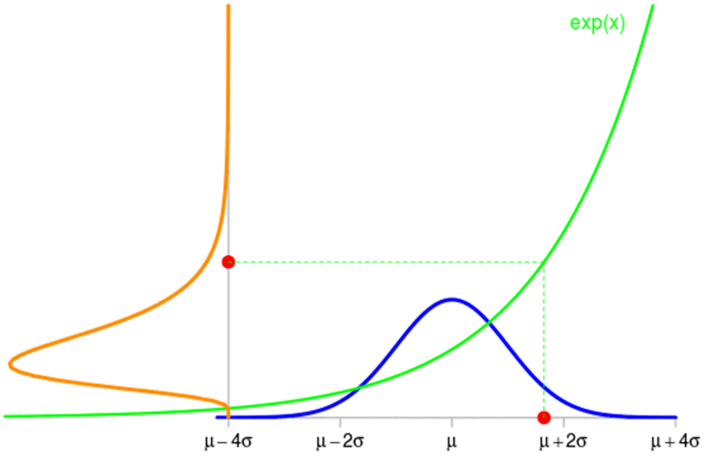


Figure 8.3: Example of change of variable theorem in action for probability distributions.

8.3 LIPSCHITZ PRIVACY

The notion of privacy that we use in this work is that of Lipschitz privacy[286], which is equivalent to differential privacy (details of equivalence in the original paper). It is a Lipschitz bound on the log density of the output of the query as follows.

$$|\ln \mathcal{P}(Q(u) \in S) - \ln \mathcal{P}\left(Q\left(u'\right) \in S\right)| \leq \epsilon d\left(u, u'\right), \forall u, u' \in \mathcal{U}$$

This has an equivalent definition that is, at times, more practical to work within the context of machine learning and is given below.

$$\|\nabla_u \ln \mathcal{P}(Q(u) = y)\| \leq \epsilon$$

8.4 Main Proposed result

Theorem 8.4.1 (Power mechanism theorem). For $\mathbf{X} \in \mathbb{R}^k$ distributed as $f_X(x)$, applying $\mathbf{Z_{p_i}} = g_p \circ g_{p-1} \circ \cdots \circ g_1(\mathbf{X_i})$ guarantees ϵ-Lipschitz privacy through $\mathbf{Z}$ using $\mathbf{J}(\mathbf{X})$, the Jacobian matrix of the composition when we have

$$\left\| \frac{f'(\mathbf{X})}{f(\mathbf{X})} - \sum_{p=1}^{p} \log(|\det(J)) \right\| \leq \epsilon$$

.

Proof. The equation $\mathbf{Z}_{p_i} = [\mathbf{H}(\mathbf{X_i})]^p \mathbf{X}$ can be unrolled as

$$\mathbf{Z_{p_i}} = g_p \circ g_{p-1} \circ \cdots \circ g_1(\mathbf{X_i}) \tag{8.1}$$

where $g_p \circ g_{p-1}(\cdot) = \mathbf{H}(\mathbf{X}).g_{p-1}(\cdot)$. If g is a one-to-one function on the support of $\mathbf{X}$ whose pdf is given by $f_X(x)$ where $x \in \mathbb{R}^k$, then the pdf of $\mathbf{Z} = \mathbf{g}(\mathbf{X})$ is

$$h(\mathbf{Z}) = f_{\mathbf{X}}(g^{-1}(\mathbf{Z}))|\det(\mathbf{J}(g^{-1}(\mathbf{Z})))|$$

for $\mathbf{Z}$ in the range of g, where $\mathbf{J}(\mathbf{X})$ is the Jacobian matrix of g that is evaluated at $\mathbf{X}$. This is classically known as the multidimensional change of variable theorem in the context of probability density functions (an illustration is provided in Figure 8.3). But since we have $g_p \circ g_{p-1} \circ \cdots \circ g_1(\mathbf{X})$ instead of a single $g(\cdot)$, this can be written as

$$h_p(\mathbf{Z}_p) = h_{p-1}(g_p^{-1}(\mathbf{Z}_p)) \left| \det \frac{dg_p^{-1}}{d\mathbf{Z}_p} \right|$$

We can rearrange the Jacobian of our iteration as follows

$$\frac{\partial \mathbf{H}^p \mathbf{X}}{\partial \mathbf{Z}_{p-1}} = \frac{\partial \mathbf{H}^p X}{\partial \mathbf{H}^{p-1} X} = \frac{\partial \mathbf{Z}^p}{\partial \mathbf{Z}^{p-1}} = \frac{\partial \mathbf{H}\mathbf{H}^{p-1}\mathbf{X}}{\partial \mathbf{H}^{p-1}\mathbf{X}} = \mathbf{J}(\mathbf{Z}_{(\mathbf{p-1})_\mathbf{i}})$$

Now, in order to find this specific Jacobian matrix $\mathbf{J}$, consider the equation

$$\mathbf{Z}_{p_i} = \sum_{j=1}^{k} \mathbf{H}(\mathbf{Z}_{p-1})_{ij} \mathbf{Z}_{p-1_j}$$

Since the Jacobian matrix $\mathbf{J}$ is

$$\mathbf{J}_{ij} = \frac{\partial \mathbf{Z}_{p_i}}{\partial \mathbf{Z}_{p-1_j}}$$

$$\mathbf{J}_{ij} = \frac{\partial \sum_{l=1}^{k} \mathbf{H}(\mathbf{Z}_{p-1})_{il} \mathbf{Z}_{p-1_l}}{\partial \mathbf{Z}_{p-1_j}}$$

$$\mathbf{J}_{ij} = \sum_{l=1}^{k} \frac{\partial \mathbf{H}(\mathbf{Z}_{p-1})_{il}}{\partial \mathbf{Z}_{p-1_j}} \mathbf{Z}_{p-1_l} + \mathbf{H}(\mathbf{Z}_{p-1})_{ij}$$

Upon applying a logarithm to the result of the change of variable theorem within our setup, we get

$$\left|\det\left(\frac{dg_p}{d\mathbf{Z}_{p-1}}\right)^{-1}\right| = \left|\det \frac{dg_p}{d\mathbf{z}_{p-1}}\right|^{-1}$$

$$= \log h_{p-2}(\mathbf{Z}_{p-2}) - \log\left|\det \frac{dg_{p-1}}{d\mathbf{Z}_{p-2}}\right| - \log\left|\det \frac{dg_p}{d\mathbf{Z}_{p-1}}\right|$$

$$= \dots$$

$$= \log h_0(\mathbf{Z}_0) - \sum_{i=1}^{p} \log \left| \det \frac{dg_i}{d\mathbf{Z}_{i-1}} \right|$$

Therefore we have that the logarithm of the ratio of the probability densities before and after p iterations as

$$\log\left(\frac{h(\mathbf{Z})}{f(\mathbf{X})}\right) = -\sum_{p=1}^{p} \log|\det \mathbf{J}(\mathbf{Z_{(p-1)_i}}| = -\log(\prod_{i=1}^{p} |\det \mathbf{J}(\mathbf{Z_{(p-1)_i}}|)$$

Upon applying the derivative to the log probability and taking its norm and setting it to be less than ϵ, we get the following relations followed by the required condition in order to satisfy Lipschitz privacy.

$$\left\|\frac{\partial}{\partial \mathbf{X}} \log h(\mathbf{Z})\right\| = \left\| \frac{f'(\mathbf{X})}{f(\mathbf{X})} - \sum_{p=1}^{p} \frac{\frac{\partial|\det \mathbf{J}(\mathbf{Z_{(p-1)_i}}|}{\partial X_i}}{|\det \mathbf{J}(\mathbf{Z_{(p-1)_i}}|} \right\|$$

Now, to differentiate the determinant of a matrix, we use Jacobi's formula to get,

$$\left\|\frac{\partial}{\partial \mathbf{X}} \log h(\mathbf{Z})\right\| = \left\| \frac{f'(\mathbf{X})}{f(\mathbf{X})} - \sum_{p=1}^{p} \frac{\det(\mathbf{J}(\mathbf{Z}_{(p-1)_i})tr(\mathbf{J}^{-1}\frac{\partial \mathbf{J}(\mathbf{Z}_{(p-1)_i})}{\partial \mathbf{X}_i})}{|\det \mathbf{J}(\mathbf{Z_{(p-1)_i}}|} \right\|$$

Finally, we evaluate the term $\mathbf{J}' = \frac{\partial \mathbf{J}(\mathbf{Z}_{(p-1)_i})}{\partial \mathbf{X}_i}$ as follows.

$$\mathbf{J}'_{lm} = \frac{\partial \mathbf{J}(\mathbf{Z}_{(p-1)_i})_{lm}}{\partial \mathbf{X}_i} = \frac{\partial(\sum_{n=1}^{k} \frac{\partial \mathbf{H}(\mathbf{Z}_{p-1})_{ln}}{\partial \mathbf{Z}_{p-1_m}} \mathbf{Z}_{p-1_n} + \mathbf{H}(\mathbf{Z}_{p-1})_{lm})}{\partial \mathbf{X}_i}$$

$$\mathbf{J}'_{lm} = \sum_{n=1}^{k} \left(\frac{\partial^2 \mathbf{H}(\mathbf{Z}_{p-1})_{ln}}{\partial \mathbf{X}_i \partial \mathbf{Z}_{p-1_m}} \mathbf{Z}_{p-1_n} + \frac{\partial \mathbf{H}(\mathbf{Z}_{p-1})_{ln}}{\partial \mathbf{Z}_{p-1_m}} \frac{\partial \mathbf{Z}_{p-1_n}}{\partial \mathbf{X}_i} \right)$$

Therefore, for obtaining $\epsilon-$Lipschitz privacy, we need to have

$$\left\| \frac{\partial}{\partial \mathbf{X}} \log h(\mathbf{Z}) \right\| = \left\| \frac{f'(\mathbf{X})}{f(\mathbf{X})} - \sum_{p=1}^{p} \log(|\det(\mathbf{J}))\right\| \leq \epsilon$$

$$\left\| \frac{\partial}{\partial \mathbf{X}} \log h(\mathbf{Z}) \right\| = \left\| \frac{f'(\mathbf{X})}{f(\mathbf{X})} - p\mathbf{J}^{-1}\frac{\partial \mathbf{J}}{\partial \mathbf{X}} \right\| \leq \left\| \frac{f'(\mathbf{X})}{f(\mathbf{X})} \right\| + \left\| p\mathbf{H}^{-1}\frac{\partial \mathbf{J}}{\partial \mathbf{X}} \right\| \leq \epsilon$$

□

8.5 Pracical calibration of privacy level

In order to be able to use the main result above practically, we need an estimate for the probability densities and their first derivatives, along with corresponding confidence intervals around the true densities or their first derivatives. We use a kernel density estimation-based approach for estimating the probability density of each sample as follows using a valid kernel function K, and we have $\hat{f}(x) = \frac{1}{nh^d}\sum_{i=1}^{n} K\left(\frac{x-X_i}{h}\right)$ where the Gaussian kernel is given by $K(u) = \frac{e^{-||u||^2}}{(2\pi)^{d/2}}$. However, as mentioned, we need to find confidence intervals for these probability density estimates. The range in which the true probability density lies with $1-\alpha$ probability is given by

$$CI_{1-\alpha} = [\hat{f}(x) - z_{1-\alpha/2}\sqrt{\frac{\mu_K \hat{f}(x)}{nh^d}}, \hat{f}(x) + z_{1-\alpha/2}\sqrt{\frac{\mu_K \hat{f}(x)}{nh^d}}]$$

where the term μ_K is given by $\mu_K = \int K^2(x)dx$. For the Gaussian kernel, this evaluates to $\mu_K = 1/(2^d \pi^{d/2})$. Similarly, the confidence bound for the gradient of probability density is given by

$$\frac{\partial f(x)}{\partial x_i} - \frac{\partial \hat{f}(x)}{\partial x_i} = O(h^2) + O_P\Big(\sqrt{\frac{1}{nh^{d+2}}}\Big)$$

where,

$$f(x) = \hat{f}(x) + \sqrt{K\hat{f}(x)}\mathcal{N}(0,1)$$

and

$$\frac{\partial f(x)}{\partial x_i} = \frac{\partial \hat{f}(x)}{\partial x_i} + \sqrt{\frac{K}{4\hat{f}(x)}}\hat{f}(x)\mathcal{N}(0,1)$$

As this condition, in turn, requires a density estimator, we need to use the confidence interval on $f(X)$ in the above condition to get a final estimate of the privacy level as follows. Using,

$$\epsilon' = \max\Bigg(\Bigg\|\frac{\hat{f}'(\mathbf{X})}{\hat{f}(x) - z_{1-\alpha/2}\sqrt{\frac{\mu_K \hat{f}(x)}{nh^d}}} - \sum_{p=1}^{p}\log(|\det(J))\Bigg\|, \quad (8.2)$$
$$\Bigg\|\frac{\hat{f}'(\mathbf{X})}{\hat{f}(x) + z_{1-\alpha/2}\sqrt{\frac{\mu_K \hat{f}(x)}{nh^d}}} - \sum_{p=1}^{p}\log(|\det(J))\Bigg\|\Bigg)$$

the final ϵ is given by $\epsilon = \epsilon' + \left\|\frac{f'(\mathbf{X}) - \hat{f}'(\mathbf{X})}{f(\mathbf{X})}\right\|$.

8.6 Experiments

We illustrate an instance of applying a power mechanism to generate a resulting histogram that enables few-shot private learning in Figure 8.4 over the Kaggle churn prediction dataset. In Figure 8.5a, we show the resulting test-accuracies upon applying power mechanism over the forest cover data set with four different ϵ privacy levels of $0.5, 0.65, 1$ and 1.5 denoted by the four curves from bottom to top respectively. This shows the obtained privacy-accuracy trade-off. Similarly, in Figure 8.5b, we show the same trade-off upon applying the popular baseline of DP-SGD for the same privacy levels.

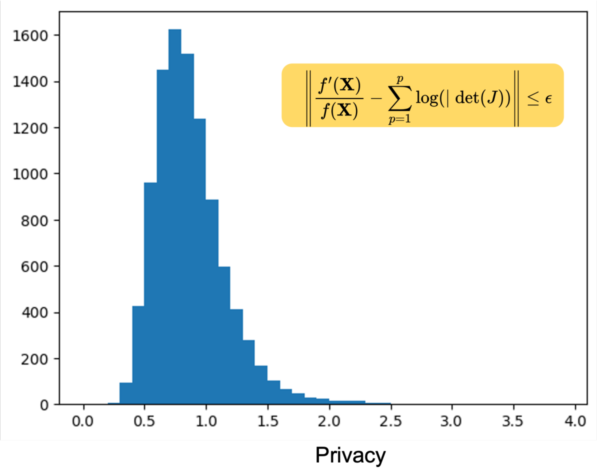
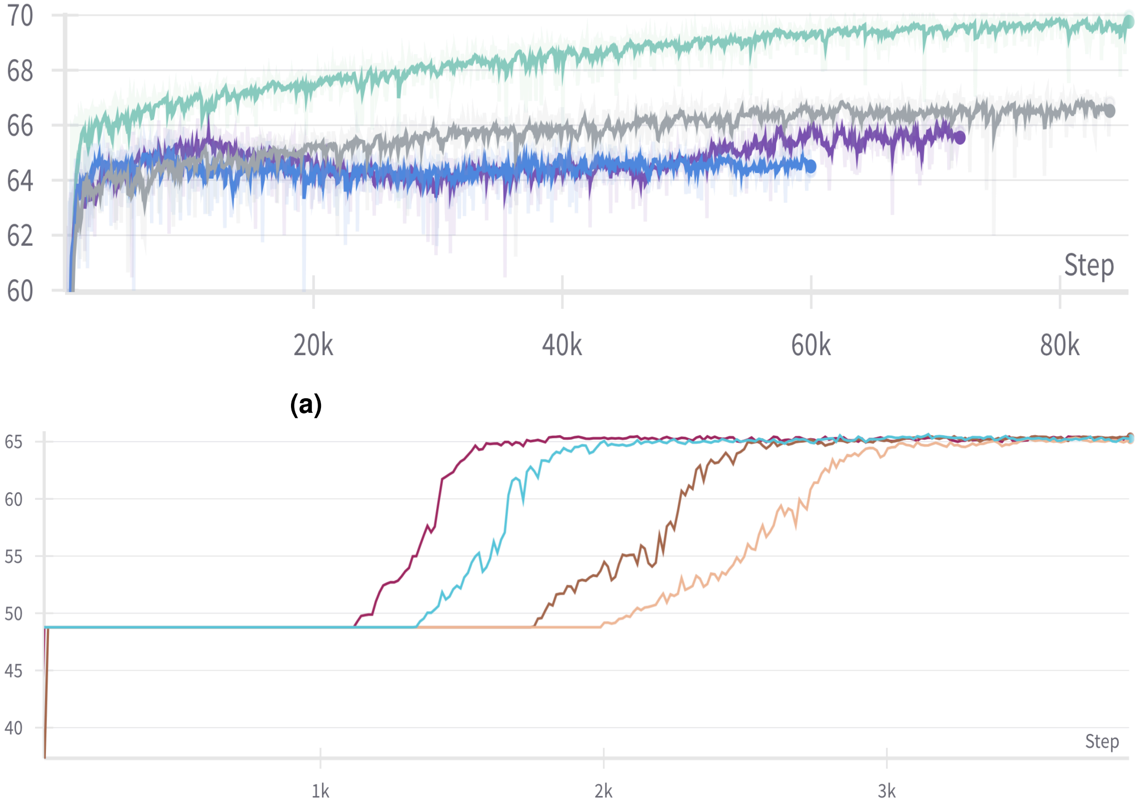


Figure 8.4: Example of change of variable theorem in action for probability distributions.



(a)

(b)

Figure 8.5: (a) The resulting test-accuracies upon applying power mechanism over the forest cover data set with four different ϵ privacy levels of $0.5, 0.65, 1$ and 1.5 denoted by the four curves from bottom to top respectively. This shows the obtained privacy-accuracy trade-off. (b) Similarly, here, we show the same trade-off upon applying the popular baseline of DP-SGD for the same privacy levels. To conclude, the power mechanism provides comparable (and at times better) privacy-utility tradeoffs while enabling resource-efficient distributed machine learning, unlike DP-SGD-based methods that enable federated learning, which do not have such compelling resource efficiency. The power mechanism also enables private few-shot learning, train-time, and inference-time privacy based on operating with corresponding privacy histograms.

“The scientific method’s central motivation is the ubiquity of error - the awareness that mistakes and self-delusion can creep in absolutely anywhere and that the scientist’s effort is primarily expended in recognizing and rooting out error.”

David Donoho

9

Structured mixture distributions that effectively poison ML on large n small p tabular data

9.1 Introduction

Data poisoning attack methods have propped up in plenty[415,455] to damage the efficacy of training machine learning models. Their mode of operation is based on either modifying existing training data records via attacks such as one-pixel attacks[491] or via the addition of a subsample of poisoned data points[460] to the training datasets. These methods attempt to evade detection by models that screen the datasets or ML pipelines and anomaly detectors to detect data poisoning. Post the filtering of any detected points (typically with false alarms or false negatives), the rest of the undetected points produce a degradation in model performance on otherwise genuine data points upon which model predictions are to be obtained post-deployment of the model. These methods are currently based on adversarial training[13,93,493,602,334,313,330,130,375]. We provide an alternative attack scheme for data poisoning that is instead based on structured learning of Gaussian mixtures with low KL-divergence from target mixture models that in turn model the raw data. We show that samples from these structured distributions are highly effective and evasive in evaluating training datasets of popular machine learning training pipelines such as neural networks, XGBoost, and random forests. In the current day and age of machine learning, Gaussian mixtures are perceived to be an older/classical technique. Therefore it is quite interesting to see that they can be highly effective in performing data poisoning attacks if learned with the right structural constraints. This chapter is based on our work in[Balla et al.].

9.2 Structured Decoy Distribution Learning

We now present our proposed results that help in structured distribution learning of Gaussian mixtures such that the KL-divergence between the learnt mixture and the target mixture is minimized. This helps in learning distributions from which the poisoned data points can be sampled from.

9.3 Bounds on KL-Divergence between two Gaussian Mixtures

For the distribution learning problem motivated in the previous section, the key is to be able to learn a τ-close Gaussian mixture to a given target Gaussian mixture. We, therefore, share some results on KL-divergences between Gaussian mixtures[151]. This helps exploit lower bounds in distribution testing problems that attempt to distinguish two distributions based on their samples. Let f and g be two PDFs in $\mathbb{R}^d$, where d is the dimension of the observed vectors x. The KL-divergence between f and g is defined as

$D_{KL}(f||g) = \int_{\mathbb{R}^d} f(x) \log \frac{f(x)}{g(x)} dx$. When f and g are PDFS of multivariate normals:

$$D_{KL}(f||g) = \frac{1}{2} \log \frac{|\Sigma^g|}{|\Sigma^f|} + \frac{1}{2} \mathbf{Tr}((\Sigma^g)^{-1} \Sigma^f) + \frac{1}{2} (\mu^f - \mu^g)^T (\Sigma^g)^{-1} (\mu^f - \mu^g) - \frac{d}{2} \tag{9.1}$$

When f and g are PDFs for GMMs, the expression for f is (with an analogous expression for g):

$$f(x) = \sum_{a=1}^{A} \omega_a^f f_a(x) = \sum_{a=1}^{A} \omega_a^f N\left(x; \mu_a^f, \Sigma_a^f\right) \tag{9.2}$$

A practical upper-bound on KL-divergence between two Gaussian mixtures $D_{\text{avg}}(f||g)$ is given by

$$\frac{1}{2} \sum_a \omega_a^f \left[\log \sum_\alpha \omega_\alpha^f e^{-D_{KL}(f_a||f_\alpha)} + \log \sum_\alpha \omega_\alpha^f z_{a\alpha} - \log \sum_b \omega_b^g t_{ab} - \log \sum_b \omega_b^g e^{-D_{KL}(f_a||g_b)} \right]$$

as can be obtained via the details shared later on in this work.

9.4 Upper bounds on KL-divergence between Gaussian mixtures

[151] defines the upper and lower bounds for KL-Divergence between GMMs to be:

$$D_{\text{lower}}(f||g) = \sum_a \omega_a^f \log \frac{\sum_\alpha \omega_\alpha^f e^{-D_{KL}(f_a||f_\alpha)}}{\sum_b \omega_b^g t_{ab}} - \sum_a \omega_a^f H(f_a) \tag{9.3}$$

$$D_{\text{upper}}(f||g) = \sum_a \omega_a^f \log \frac{\sum_\alpha \omega_\alpha^f z_{a\alpha}}{\sum_b \omega_b^g e^{-D_{KL}(f_a||g_b)}} + \sum_a \omega_a^f H(f_a) \tag{9.4}$$

where $H(f_a)$ is the entropy of f_a, and the normalization constants of the product of the individual Gaussians are given by:

$$\log t_{ab} = -\frac{d}{2}\log 2\pi - \frac{1}{2}\log|\Sigma_a^f + \Sigma_b^g| - \frac{1}{2}(\mu_b^g - \mu_a^f)^T(\Sigma_a^f + \Sigma_b^g)^{-1}(\mu_b^g - \mu_a^f) \quad (9.5)$$

$$\log z_{a\alpha} = -\frac{d}{2}\log 2\pi - \frac{1}{2}\log|\Sigma_a^f + \Sigma_\alpha^f| - \frac{1}{2}(\mu_\alpha^f - \mu_a^f)^T(\Sigma_a^f + \Sigma_\alpha^f)^{-1}(\mu_\alpha^f - \mu_a^f) \quad (9.6)$$

We will focus on optimizing the following average of the lower and upper bounds of the KL-Divergence between GMMs as it was shown to be a good estimate of the KL-Divergence between GMMs in[151].

9.5 Route 1: Modified EM algorithms for our structured distribution learning problem

We substituted the KL terms with distance covariance in order to minimize the above average bound on KL-divergence $D_{\text{avg}}(f||g)$ between Gaussian mixtures and optimized it to see the efficacy of the learnt poisoned samples.

This is, fortunately, possible because the KL divergence between Gaussian mixtures is expressed via separable KL terms between components of Gaussian mixtures. Note that two terms are constant here with respect to the target mixture distribution as follows

$$D_{\text{avg}}(f||g) = \frac{1}{2}\sum_a \omega_a^f \left[C_1 + C_2 - \log \sum_b \frac{\omega_b^g}{\sqrt{|\Sigma_a^f + \Sigma_b^g|}} - \log \sum_b \omega_b^g e^{\text{DCov}(\Sigma_f^a, \Sigma_b^g)} \right] \tag{9.7}$$

With $\Sigma_g^f = \frac{1}{N-1} Z_b^T Z_b$, where N is the number of samples, our problem is equivalent to minimizing the following for each component a

$$\omega_a^f \log \sum_\alpha \omega_\alpha^f e^{\text{DCov}(\Sigma_a^f, \Sigma_\alpha^f)} + \omega_a^f \log \sum_\alpha \frac{\omega_\alpha^f}{\sqrt{|\Sigma_a^f + \Sigma_\alpha^f|}} \tag{9.8}$$

$$- \omega_a^f \log \sum_b \omega_b^g e^{\text{DCov}(\Sigma_a^f, \frac{1}{N-1} Z_b^T Z_b)} - \omega_a^f \log \sum_b \frac{\omega_b^g}{\sqrt{|\Sigma_a^f + \frac{1}{N-1} Z_b^T Z_b|}} \tag{9.9}$$

$$= \omega_a^f (C_1 + C_2) - \omega_a^f \log \sum_b \omega_b^g e^{\text{DCov}(\Sigma_a^f, \frac{1}{N-1} Z_b^T Z_b)} - \omega_a^f \log \sum_b \frac{\omega_b^g}{\sqrt{|\Sigma_a^f + \frac{1}{N-1} Z_b^T Z_b|}} \tag{9.10}$$

$$- \omega_a^f \log \sum_b \omega_b^g e^{\text{DCov}\left(\Sigma_a^f, \frac{1}{N-1}\left(Z_b - \mu_b^g\right)^T \left(Z_b - \mu_b^g\right)\right)} \tag{9.11}$$

$$- \omega_a^f \log \sum_b \frac{\omega_b^g e^{-\frac{1}{2}\left(\mu_b^g - \mu_a^f\right)^T \left(\Sigma_a^f + \frac{1}{N-1}\left(Z_b - \mu_b^g\right)^T \left(Z_b - \mu_b^g\right)\right)^{-1} \left(\mu_b^g - \mu_a^f\right)}}{\sqrt{|\Sigma_a^f + \frac{1}{N-1}(Z_b - \mu_b^g)^T (Z_b - \mu_b^g)|}} \tag{9.12}$$

$$+ \omega_a^f (C_1 + C_2) + \lambda.\text{EMLoss}$$

where the EMLoss in the last term is the standard EM loss; here, the objective function is regularized with the standard loss used in EM algorithms for estimating Gaussian mixtures. Therefore, we now have a modified EM algorithm that learns Gaussian mixtures with respect to a target distribution while satisfying the closeness constraints with respect to KL-divergence.

E-step updates: For each component b at step t, compute

$$\gamma_{ib}^{(t+1)} = \frac{\omega_b^{g(t)} p\left(y_i | \mu_b^{g(t)}, \Sigma_b^{g(t)}\right)}{\sum_{b'=1}^{B} w_{b'}^{g\,(t)} p\left(y_i | \mu_{b'}^{g\,(t)}, \Sigma_{b'}^{g\,(t)}\right)}, \quad i = 1, \ldots, N$$

and finally

$$n_b^{(t+1)} = \sum_{i=1}^{N} \gamma_{ib}^{(t+1)}$$

M-step updates: For each component b, compute the following update

$$\omega_b^{g(t+1)} = \frac{n_b^{(t+1)}}{N}$$

The rest of the updates for the mean vector and covariances are omitted here for brevity.

Theorem 9.5.1. The function $\log \sum_b \frac{\omega_b^g}{\sqrt{|\Sigma_a^f + \Sigma_b^g|}}$ is convex if

$$\omega_b^g \sum_b \left(\frac{\omega_b^g}{\sqrt{|\Sigma_a^f + \Sigma_b^g|}} - \omega_b^g \right) \geq 0$$

as this results in a positive semi-definite Hessian.

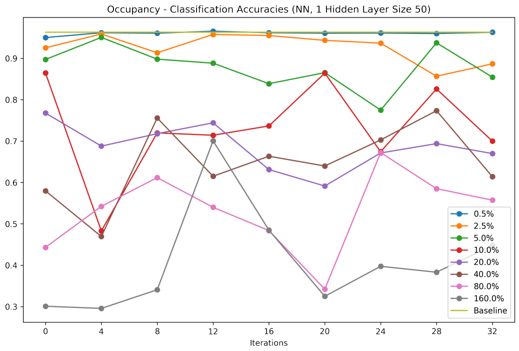
Proof. The proof is in Appendix 9.7. □

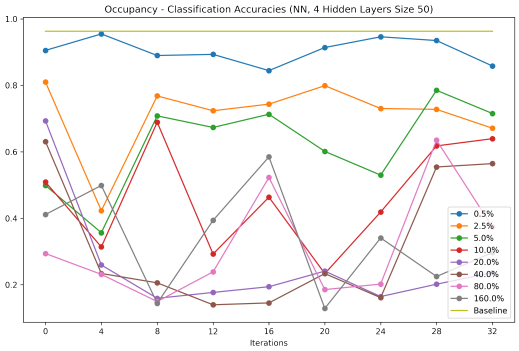
Theorem 9.5.2. The function $\log \sum_b \omega_b^g e^{\mathrm{DCov}(\Sigma_f^a, \Sigma_b^g)}$ is convex.

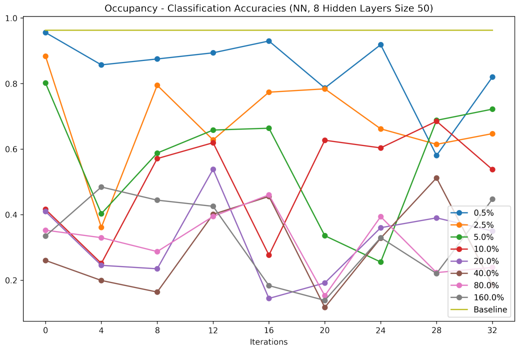
Proof. The proof is in the Appendix. □

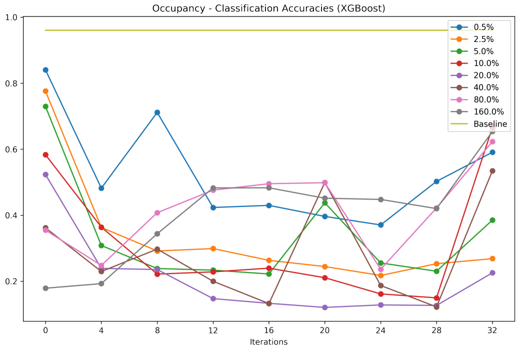
9.6 Numerical Experiments

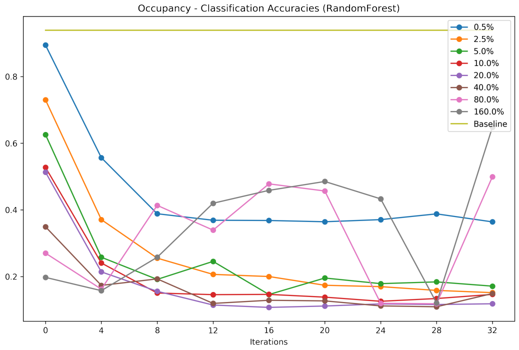
We performed numerical experiments on 3 UCI-ML repository datasets of EEG eye state, occupancy, and Avila with their dimensions and specifications detailed in Table 1 above. We show in a series of captioned figures in the appendix below that the well-tuned classification models, such as neural networks with increasing hidden layers of 1, 4, 8, and 12, as well as models such as XGBoost and Random Forests, cannot distinguish between the real and poisoned samples generated by our scheme, thereby making it really hard for an attacker that is dependent on machine learning to estimate the pair of mixture distributions used to model the real samples and to obtain poisoned samples respectively. Our pipeline consists of a model to detect a decoy Vs. non-decoy and in addition, we also perform a label reconstruction attack to reconstruct the raw labels of the client. The poisoned samples are generated only using raw features. We see a spin-off empirical benefit that upon adding poisoned samples, not only do we prevent their detection, but we also make it extremely hard for the attacker to reconstruct the raw labels corresponding to the raw data via a second model. We use default SciPy parameters for Powell minimization to optimize mu and parameters of `ftol = 0.001,` `xtol = 0.001, maxfev = 4000` for optimizing $\mathbf{Z_b}$ in our modified EM algorithm. In contrast, the rest of the steps in our algorithm are trivial to compute. We show the efficacy and evasiveness of data poisoning with structured learning of Gaus-

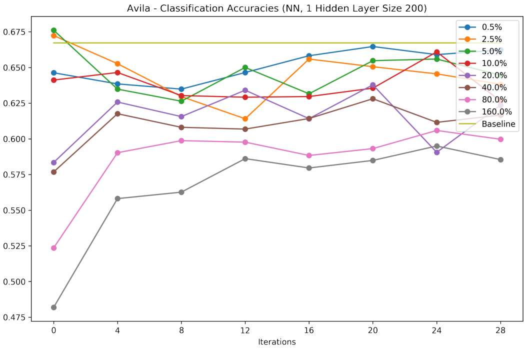

Occupancy - Classification Accuracies (NN, 1 Hidden Layer Size 50)

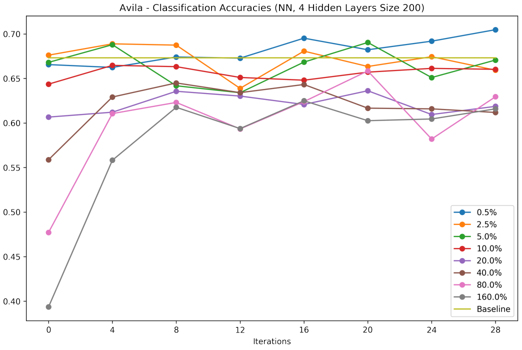

Occupancy - Classification Accuracies (NN, 4 Hidden Layers Size 50)

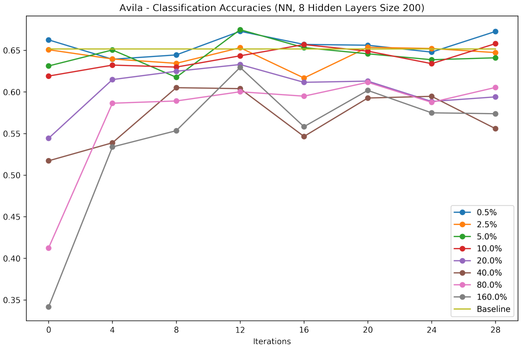

Occupancy - Classification Accuracies (NN, 8 Hidden Layers Size 50)

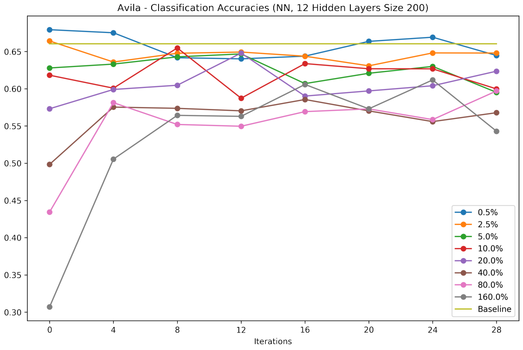

Occupancy - Classification Accuracies (XGBoost)

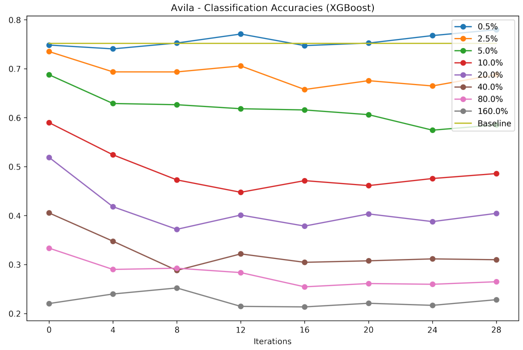

Occupancy - Classification Accuracies (RandomForest)

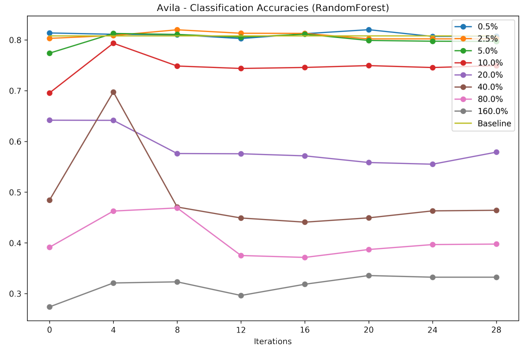

Avila - Classification Accuracies (NN, 1 Hidden Layer Size 200)

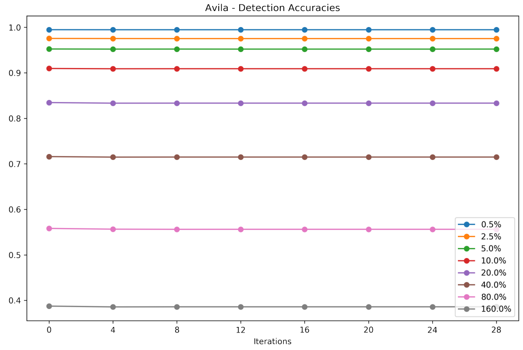

Avila - Classification Accuracies (NN, 4 Hidden Layers Size 200)


Avila - Classification Accuracies (NN, 8 Hidden Layers Size 200)









Occupation and Avila datasets: Classification of decoy Vs. non-decoy splinters using NNs, XGBoost and Random Forest shows that the models are unable to distinguish them when the sample size of decoy splinters is twice that of the non-decoy splinters. Our pipeline is a standard one used in data-poisoning schemes with two models: one to detect and one to classify. Our pipeline consists of a model to detect a decoy Vs. non-decoy and in addition, we also perform a label reconstruction attack to reconstruct the raw labels of the client. The splinters are generated only using raw features. We see a spin-off empirical benefit that upon adding decoy splinters, not only do we prevent their detection, but we also make it extremely hard for the attacker to be able to reconstruct the raw labels corresponding to the raw data via a second model.

Dataset	**Sample Size**	**Attributes**	**Balanced**	**# of Classes**
EEG Eye State	14,980	15	Yes	2
Avila	20,867	10	Yes	12
Skin Segmentation	245,057	4	No	2

Table 9.1: A listing of datasets that we used for empirical investigations is provided in this table along with their dimensions.

We show that our approach also reduces the needed KL-divergence as shown below.

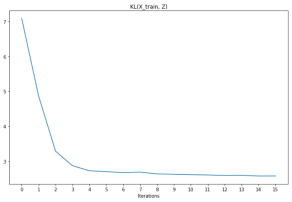


Figure 9.1: We obtain KL-divergence reduction between the data-related mixture and the poisoning mixture learnt through our approach.

sian mixtures with low KL-divergence from target mixture models that, in turn, model the raw data. We also provide new results connecting RKHS and distance statistics like distance correlation to information theoretic measures like KL-divergence, and employ these results in optimizing for KL-divergence between Gaussian mixtures.

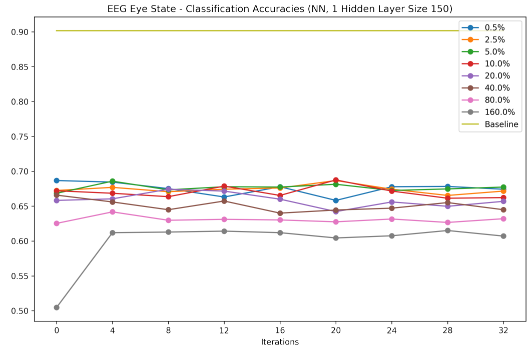
9.7 Appendix A: Proof of Theorem 9.5.1

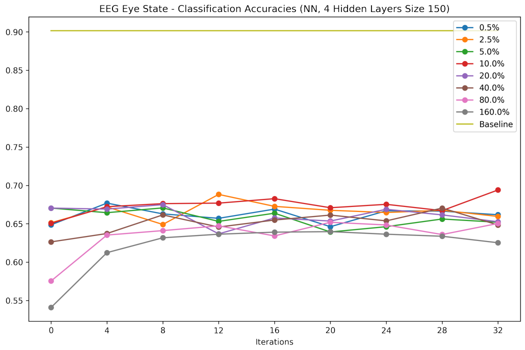
Proof. This condition simplifies to requiring

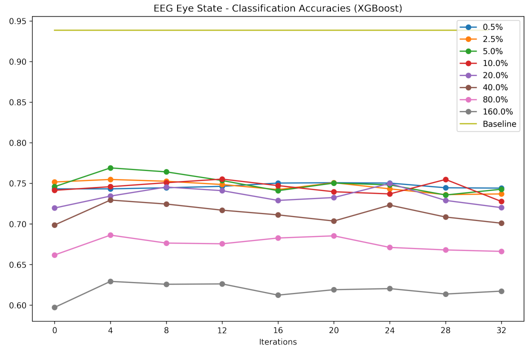
$$\sqrt{|\Sigma_a^f + \Sigma_b^g|} \leq \omega_b^g, \forall b$$

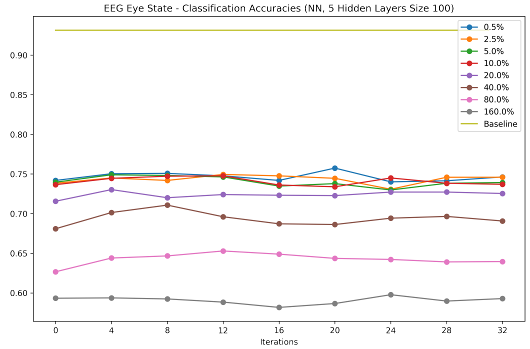
By the arithmetic-geometric-mean (A.G.M) inequality we have,

$$\prod_{k=1}^{n} \lambda_k \leq \frac{1}{n^n} \left(\sum_{k=1}^{n} \lambda_k \right)^n$$


EEG Eye State - Classification Accuracies (NN, 1 Hidden Layer Size 150)
0.90
0.85
0.80
0.75
0.70
0.65
0.60
0.55
0.50
0 4 8 12 16 20 24 28 32
Iterations
0.5%
2.5%
5.0%
10.0%
20.0%
40.0%
80.0%
160.0%
Baseline


EEG Eye State - Classification Accuracies (NN, 4 Hidden Layers Size 150)
0.90
0.85
0.80
0.75
0.70
0.65
0.60
0.55
0 4 8 12 16 20 24 28 32
Iterations
0.5%
2.5%
5.0%
10.0%
20.0%
40.0%
80.0%
160.0%
Baseline


EEG Eye State - Classification Accuracies (XGBoost)
0.95
0.90
0.85
0.80
0.75
0.70
0.65
0.60
0 4 8 12 16 20 24 28 32
Iterations
0.5%
2.5%
5.0%
10.0%
20.0%
40.0%
80.0%
160.0%
Baseline


EEG Eye State - Classification Accuracies (NN, 5 Hidden Layers Size 100)
0.90
0.85
0.80
0.75
0.70
0.65
0.60
0 4 8 12 16 20 24 28 32
Iterations
0.5%
2.5%
5.0%
10.0%
20.0%
40.0%
80.0%
160.0%
Baseline

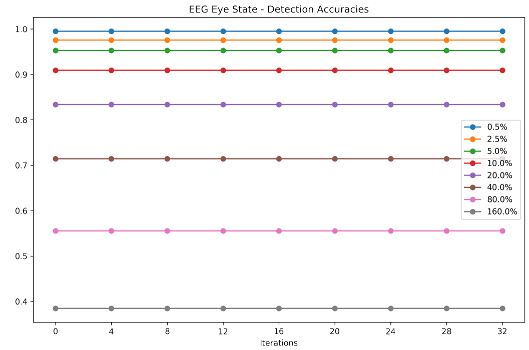


Figure 9.2: EEG: Classification of decoy Vs. Non-decoy splinters using NNs, XGBoost, and Random Forest show that the models are unable to distinguish them when the sample size of decoy splinters is twice that of the non-decoy splinters. Our pipeline is a standard one used in data-poisoning schemes with two models: one to detect and one to classify. We obtained similar results using anomaly detectors such as isolation forests. The pipeline consists of a model to detect a decoy Vs. non-decoy, and in addition, we also perform a label reconstruction attack to reconstruct the raw labels of the client. The splinters are generated only using raw features. We see a spin-off empirical benefit that upon adding decoy splinters, not only do we prevent their detection, but we also make it extremely hard for the attacker to able to reconstruct the raw labels corresponding to the raw data via a second model.

Therefore $\sum_b |\Sigma_a^f + \Sigma_b^g| \leq \frac{\sum_b [\mathrm{Tr}(\Sigma_a^f + \Sigma_b^g)]^n}{n^n}$ This implies that if,

$$\sum_b \mathrm{Tr}(\Sigma_a^f + \Sigma_b^g) \leq n \sqrt[n]{\omega_b^g}, \forall b$$

then the condition for convexity $\sum_b \sqrt{|\Sigma_a^f + \Sigma_b^g|} \leq n \sqrt[n]{\omega_b^g}, \forall b$ will be satisfied. □

9.8 Appendix B: Proof of Theorem 9.5.2

Proof. We now show that the LogSumExp function $\log \sum_b \omega_b^g e^{\mathrm{DCov}(\Sigma_f^a, \Sigma_b^g)}$ is convex as well. In fact, $LogSumExp(f(z))$ happens to be convex for any convex function $f(z)$

as shown below.

$$\frac{\partial^2}{\partial z^2} log \sum e^{f_i(z)} = \frac{\partial}{\partial z}\left[\frac{\sum(e_i^f(z)\frac{\partial}{\partial z}f_i(z))}{\sum e^{f_i(z)}}\right] \tag{9.13}$$

which is equal to

$$\frac{\sum e_i^f \frac{\partial^2}{\partial z^2} f_i(z)}{\sum e^{f_i(z)}} + \frac{\sum e^{f^i(z)}[\frac{\partial}{\partial z}f_i(z)]^2}{\sum e^{f_i(z)}} - \frac{(\sum e^{f_i(z)}\frac{\partial}{\partial z}f_i(z))^2}{(\sum e^{f_i(z)})^2} \tag{9.14}$$

The first term is positive. The difference of the next two terms is positive due to Jensen's inequality as

$$\sum\left[a_i\left(\frac{\partial}{\partial z}f_i(z)\right)^2\right] \geq \left[\sum a_i \frac{\partial}{\partial z}f_i(z)\right]^2 \tag{9.15}$$

This proves convexity of $\log \sum_b \omega_b^g e^{\mathrm{DCov}(\Sigma_f^a,\Sigma_b^g)}$. □

"It has been asserted - and this is no overstatement - that whereas other sciences draw their conclusions from what we know, the science of probability derives its most important results from what we do not know."

Richard von Mises

10

Empirical heuristics for preventing face reconstruction attacks in distributed inference after on-premise training

10.1 Introduction

Data sharing and distributed computation while preserving privacy and safety have been identified amongst important current trends in the adoption of computer vision and machine learning technologies. In this setting, with several client-server entities interacting in a distributed fashion, there is a need for privacy-preserving technologies to handle face and gesture data such that attackers residing in one or more entities cannot reconstruct face data belonging to genuine clients. This would help to deploy powerful face recognition technologies such as biometric authentication, facial expression analysis, and consumer attention/engagement analysis in a truly distributed fashion across a wide array of device types while maintaining privacy.

We now elaborate on the sub-problem of private collaborative inference, that is, the setting in which this paper proposes a method to prevent face reconstruction attacks. With rapid advances in computing, organizations are now able to train ultra-large machine learning models on huge data sets with massive computing resources. This opens up a new set of problems for external clients that intend to predict with these models on their query test data. The client would not like to download these large models in their entirety on their device, given that they often have billions of parameters. Generation of predictions with these trained models is computationally intensive if solely performed on-device by the client.

In this setting, we propose a method (NoPeek-Infer) for the client to share activations from a chosen intermediate layer such that reconstruction attacks of raw face data can be prevented. In contrast, the rest of the prediction after this layer is performed on a server. We test this on several face datasets to measure the efficacy of NoPeek-Infer in preventing face reconstruction attacks within the task of distributed predictive inference. This chapter is based on our work in[545].

10.1.1 Motivation

Activation sharing: This setting of distributed learning with communication of intermediate activations upon splitting the deep learning model such that some layers lie with the client and the rest with the server is popular in *split learning*[210,538], an important variant of *federated learning*[283,348,266]. Sharing of activations from intermediate layers is also relevant in distributed learning approaches of local parallelism[304], features replay[239], divide and conquer quantization[168] and in task-independent privacy-respecting data crowdsourcing[310]. The client's data records on which the predictions need to be obtained are private. Therefore, the model's intermediate representations (or activations) that are communicated in this setting need to be desensitized to prevent reconstruction attacks. This opens up the relatively new problem of private collaborative inference

(PCI), where the model is split across the client and server.

This is in contrast to an alternate setting of federated learning with considerable existing work where the server intends to share the weights of a trained model privately. The privacy desired is with regard to the server's own training data. Traditionally, two standard modes of machine learning deployment exist for practical applications: a.) on-device prediction and b.) machine learning as a service (MLaaS). In the MLaaS setup, the service provider is assumed to be trusted by the client using the service. The assumption is not valid if the clients data is sensitive.

The following issues motivate the design of practical PCI algorithms and systems for on-device prediction:

1) Computation efficiency: Recent state-of-the-art models require a lot of computation even during inference. These models cannot fit into hardware-limited devices such as smartphones and other edge/IoT devices.

2) Secrecy of the models: Parameters of a model or architecture can be a secret or intellectual property of the server. In such cases, it is not possible to ship the models locally.

3) Shipping updates to the model: To update the model parameters, the server needs to apply updates to all clients in on-device machine learning. Within PCI, the server can update server-side parameters and treat the client's model as frozen.

Privacy preserving ML for faces: Recent privacy-preserving machine learning techniques applied to face data include blurring techniques such as[393]. In the experimental section, we compare the performance of NoPeek-Infer against this method. Earlier works on blurring, such as[386], have shown how earlier approaches of blurring fail to preserve privacy in home-based video conferencing setups. Other recent baselines we compare our method against include siamese embedding-based privacy[395] and adversarial baselines such as DeepObfuscator[312] that was originally benchmarked on faces and with Privacy Adversarial Networks[326] that was benchmarked on non-face images. We note that recent works such as[71] were applied to faces for preventing reconstruction of specifically chosen attributes about the face as opposed to altogether preventing reconstruction of the entire face. We note that NoPeek-Infer deals with this latter problem of preventing face reconstruction attacks as opposed to any attribute-specific reconstruction. In addition, we also compare against other face reconstruction defenses, such as noising and blur-based approaches.

10.2 Contributions

This paper proposes a way to mitigate reconstruction attacks on raw data in the distributed machine learning settings of private collaborative inference. To this end, the contributions of this work can be summarized as follows:

1) We introduce NoPeek-Infer to prevent reconstruction attacks during activation sharing in PCI via minimization of a statistical dependency measure called distance correlation[500,458]between raw data and any intermediary communications across the clients or server.

2) We evaluate the performance of our method on face datasets and share detailed results upon applying two state-of-the-art reconstruction attacks: i) supervised decoder

Attack name	**Time of attack**	**Mode of training**	**Mode of prediction**	**Input for attacker**	**Target of attack**
Feature space hijacking attack	Training	Distributed	Distributed/On-premise	Intermediate activations	Training/Test Data
Federated/Client-side attack	Training	P2P/Distributed	Distributed/On-premise	Client weights	Training/Test Data
Attribute attack	Train/Test	Distributed	Distributed/On-Premise	Intermediate activations/weights	Specific attributes
Decoder and Likelihood attacks	Test	On-premise	Distributed	Intermediate activations/weights	Test Data

(a) We categorize several forms of reconstruction attacks within the context of distributed machine learning. The last row shows the attacks that are relevant to the setting of private collaborative inference considered in this paper.

Method	**Sensitive Input**	**Sensitive Attribute**	**No client arch. alteration**	**Adversary Free**
Osia et al[396]	✗	✓	✗	✓
Min-max filters[212]	✗	✓	✓	✗
DeepObfuscator[312]	✓	✓	✓	✗
Shredder[360]	✗	✓	✗	✓
Mitigating information[436]	✗	✓	✓	✗
Kernelized ARL[438]	✗	✓	✓	✗
PrivacyNet[364]	✗	✓	✗	✗
IdentityDP[563]	✗	✓	✗	✗
NoPeek-Infer (Ours)	✓	✗	✓	✓

(b) Different defense mechanisms for private inference. The third column *no alteration of client architecture* refers to techniques where additional operations or layers are not required for removing sensitive information from data. The last column *adversary free* refers to techniques that require a proxy adversary during training. *Sensitive input* refers to protection of entire raw data, and *sensitive attribute* refers to techniques that protect only a given subset of attributes.

Table 10.1: Reconstruction attacks and defences studied within the context of split learning and its variants.

attacks and ii) likelihood maximization attacks in addition to some standard baselines. The likelihood maximization attack has not received attention in current works on private activation sharing, while it has been widely used[519] in the computer vision community of late.

3) In order to promote rigorous benchmarking in the PCI domain, we introduce a dataset of privatized activations using different PCI techniques for two face datasets, Fairface[299] and CelebA[329]. This dataset will act as a benchmark for the evaluation of existing and future attack and defense techniques.

10.2.1 Benefits of NoPeek-Infer

1) A key benefit of the NoPeek-Infer defense over other existing defenses is that it does not require any additional adversarial network for it to be learnt, unlike the rest. This reduces the number of parameters that need to be trained in NoPeek-Infer in comparison to other existing defense methods.

2) NoPeek-Infer does not require any modification to the client side architecture, which holds the network up to an intermediate layer, unlike existing methods, thereby making it highly suitable for the machine learning as a service (MLaaS) mode of deployments.

10.3 Related work:

10.3.1 Attacks

Attacks in distributed machine learning can be categorized as shown in Table 10.1a based on time of attack (during train/test) and mode of training (distributed, peer to peer, on-premise). Other factors include the type of input (entire dataset/specific attributes) that the malicious attacker has access to and the target dataset that it aims to reconstruct. Attackers can reside in any client or server that receives communications from another client. We now enumerate various reconstruction attacks. We compare the performance of NoPeek-Infer in defending against the supervised decoder and likelihood maximization-based reconstruction attacks. These two are the most relevant to our settings from this list of attacks.

1) **Feature space hijacking attack** is applied for distributed training of neural networks to reconstruct private data samples from the shared activations[403]. As opposed to their setup, our focus in NoPeek-Infer is to protect the client's query data in the distributed prediction/inference phase.

2) **Federated/client-side attack:** In federated learning[348,283,266], the untrusted party has access to the averaged weights of all the clients. Similarly, in split learning[210,538], the local weights of the client-side network need to be shared peer-to-peer with one other adjacent client.

3) **Attribute attack:** In this setting the attacker attempts to reconstruct only a subset of input data attributes that are considered to be sensitive[212,360,436,438,364,563] as opposed to the entire input sample as in NoPeek-Infer.

4) **Offline supervised decoder attack:** In a worst-case reconstruction attack setting, the attacker has access to a leaked subset of samples of training data x along with corresponding transformed activations z at a given layer, which are always exposed to other clients/server for distributed training of the network to be possible. The attacker could reside in any untrusted client or server that is part of the distributed training setup. The attacker also has access to the rest of the activations corresponding to unleaked training data at the same layer. This is also, by design, for distributed training to be possible. The attacker tries to learn an image-to-image translation model from the transformed activations to the leaked raw data. The attacker can then use this model to reconstruct raw data from activations corresponding to unleaked training data or unleaked test/validation data. This offline attack is also illustrated in Figure 10.1.

5) **Likelihood maximization attack:** Unlike the above scheme, this attack does not require pairs of raw images and corresponding activations, (z, x), in order to reconstruct the sensitive input. Instead, the attacker uses weights θ_1 of the client-side network. The attacker randomly initializes a network $\hat{f}(\hat{\theta}; \cdot)$ such that it generates an image $\hat{x}$ to produce $\hat{z} = f_1(\theta_1, \hat{x})$. Then the loss $\ell_2(\hat{z}, z)$ between random and sensitive activations is minimized by optimizing for the weights $\hat{\theta}$. This attack scheme is inspired by deep image prior[519] for feature inversion. One drawback of this attack is that it is only applicable to sensitive input protection and not sensitive attributes. This attack setting is stronger and also harder to defend against because it does not require access to the (z, x) pairs.

10.3.2 Defenses

Defenses that are relevant to our work are categorized in Table 10.1. We categorize them based on a.) the type of sensitive data under consideration and b.) whether additional privatizing operations and/or an additional adversarial model is required. Our proposed method of NoPeek-Infer is the only method to the best of our knowledge that does not have either of the requirements stated in b.). We also note that NoPeek-Infer focuses on preventing the reconstruction of input data, as opposed to specific attributes that have been the focus of the majority of the defense schemes.

1) **Noisy perturbations:** Differential privacy[160] is a popular notion of privacy for various queries to prevent membership inference attacks. In the context of model training, it is implemented via noisy perturbations of gradient updates as in[2,560,568,9,56,563]. Our proposed mechanism of No-Peek Infer is instead for the setting of private collaborative inference rather than training. In the context of split learning,[396] and[360] learn informal noisy perturbations to prevent reconstruction attacks but require altering the architecture of the client network that is being privatized. These works are also specific to preventing the reconstruction of a target attribute as opposed to the input dataset itself. Typically, adding noise to the activations leads to a costly trade-off of privacy versus accuracy.

2) **Siamese defense:** In this defense, a contrastive loss is used to nudge points from the same class label to be closer to each other in a learnt representation space. This loss is used in combination with an accuracy loss for prediction purposes. Such siamese embeddings have been used in various works outside the realm of privacy prior to being introduced by[395] solely for privacy purposes within the distributed setting involving intermediate activation sharing.

3) **Adversarial defenses:**[312,436,438,320,364,446] attempt to learn activations of a given network at chosen layers while attempting to protect against an adversary that attempts to reconstruct raw data or partial attributes of raw data from these activations. These

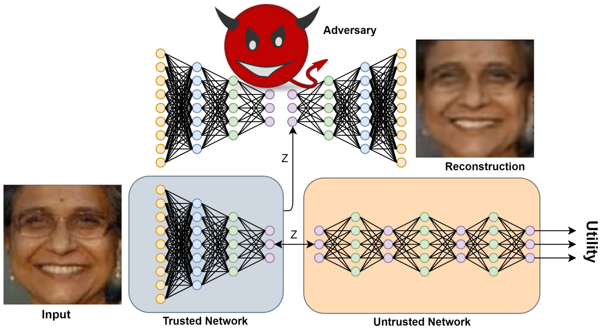


Figure 10.1: Face reconstruction attack The attack is possible when activations are shared for distributed predictive inference if a proper defense is not in place. Information about sensitive raw input data can get leaked through intermediate activations even after input data passes through multiple layers. Upon sending these intermediate activations from a trusted network on a client to an untrusted network for computing the rest of the task, an adversary on the server side can reconstruct the original raw face data from the activations.

methods require an adversarial deep network to be trained in addition to the original deep network that is used for prediction. This is in contrast to our method, which does not require any other additional network and sharply reduces the number of parameters to be trained in our case.

10.4 Method

Key idea: The key idea of our proposed method is to reduce information leakage by adding an additional loss term to the commonly used classification loss term, categorical cross-entropy. The information leakage reduction loss term we use is distance correlation[500]; a powerful measure of non-linear (and linear) statistical dependence between random variables. The distance correlation loss is minimized between raw input data and the output of any chosen layer whose outputs need to be communicated from the client to another untrusted client or untrusted server. This setting is crucial to some

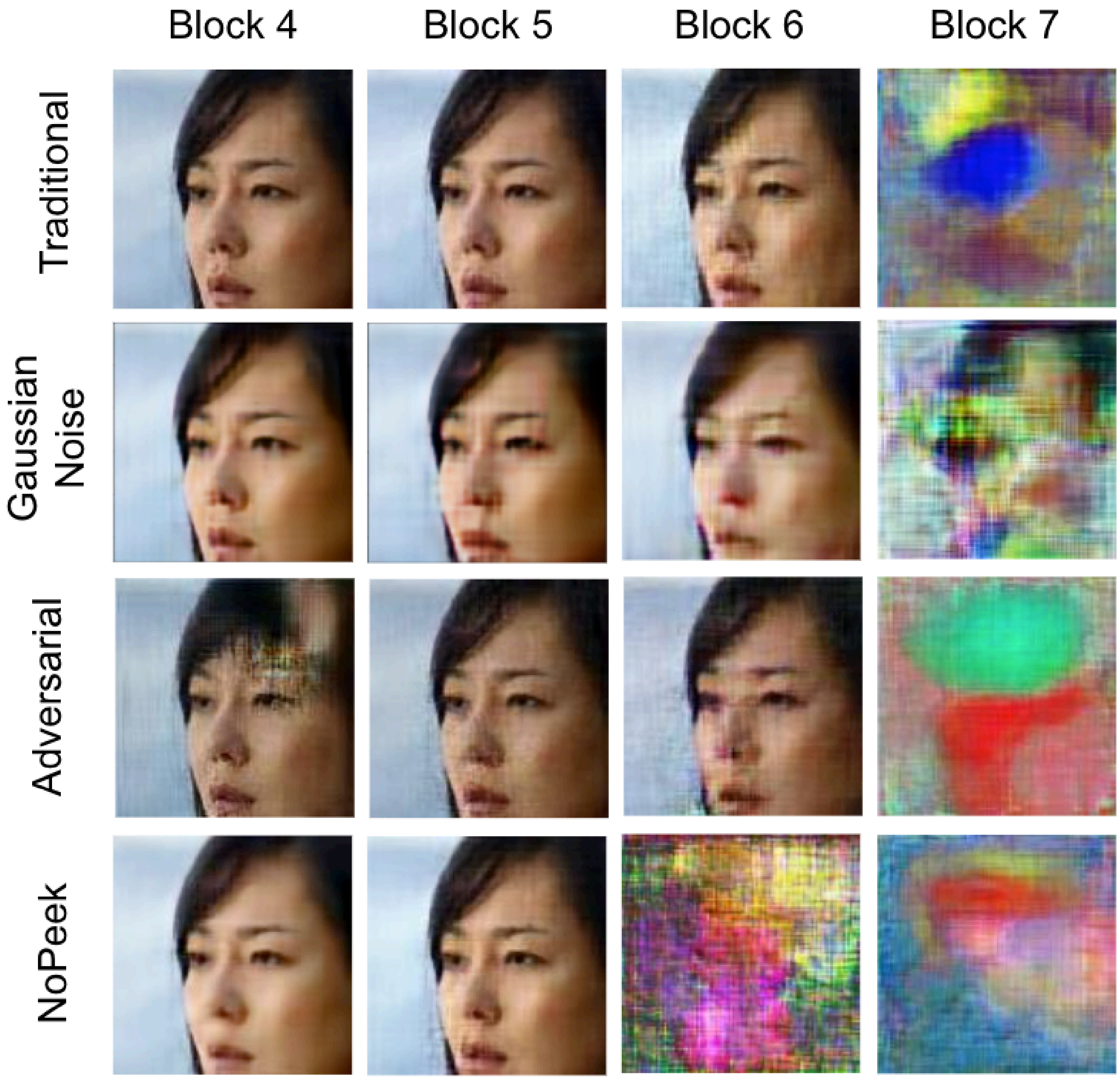


Figure 10.2: Reconstruction results on CelebA: We apply the likelihood maximization attack on activations obtained from different blocks of ResNet-18[219] for different mechanisms. For brevity, we only show blocks 4-7 since blocks before 4 get full reconstruction and blocks after 8 do not obtain a reasonable reconstruction.

popular forms of distributed machine learning that require the sharing of activations from an intermediate layer. This has been motivated under the ‘activation sharing’ subsection in the motivation section.

Optimization of this combination of losses helps ensure the activations resulting from the protected layer have minimal information for reconstructing raw data while still being useful enough to achieve reasonable classification accuracies upon post-processing. The quality of preventing reconstruction of raw input data while maintaining reasonable classification accuracies is qualitatively and quantitatively substantiated in the experiments section. The joint minimization of distance correlation with cross entropy leads to a specialized feature extraction or transformation such that it is imperceptible to leak information about the raw dataset with respect to both the human visual system and more sophisticated reconstruction attacks as we show later in the experiments section.

Loss function: The total loss function using n samples of input data $\mathbf{X}$, activations from protected layer $\mathbf{Z}$, true labels $\mathbf{Y}_{true}$, predicted labels $\mathbf{Y}$, and scalar weight α is given along with distance correlation being *DCOR* and categorical cross entropy being *CCE* as

$$\alpha DCOR(\mathbf{X}, \mathbf{Z}) + (1 - \alpha) CCE(\mathbf{Y}_{true}, \mathbf{Y}) \tag{10.1}$$

The following subsections introduce the definition of distance correlation. In contrast, the gradient of distance correlation is provided for optimization purposes in the Appendix, although we optimize our loss using *Autograd*, thereby not requiring this gradient in an explicit manner. That said, useful deep learning-friendly code for computing distance correlation is also provided in the Appendix for reproducibility.

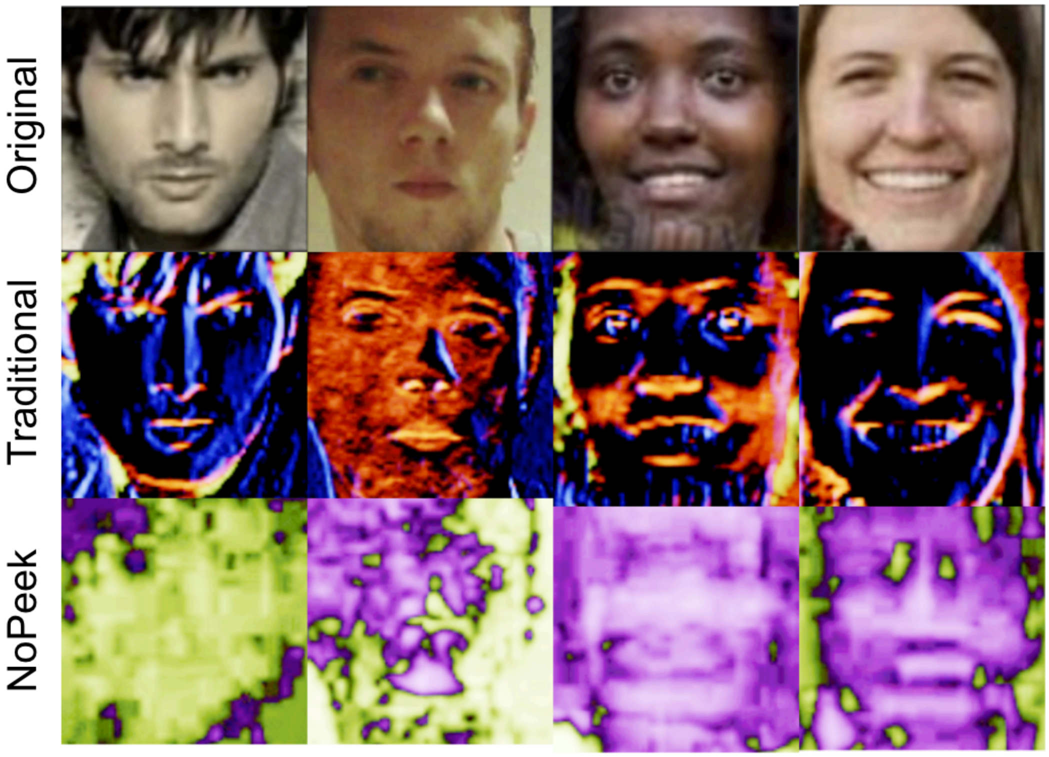


Figure 10.3: Visualization of the activations of the first layer of a ResNet. In the activation maps in the second row, subtle facial features can be observed from the activations about the raw image. In contrast, in the third row, the NoPeek-Infer-Infer method forces the network to decorrelate the features with respect to raw data, hence making it hard to interpret.

10.4.1 Advantages of using distance correlation

Estimation of classical information theoretic measures as used in[363,597,570] is a known hard problem. Recent approaches to estimate it effectively, like[45], are based on iterative optimization. A recent data-efficient version of it requires 3 nested for loops of optimization[321]. In the context of deep learning, every epoch of learning the weights is dependent on this iterative optimization. In contrast, our approach uses distance correlation. Fast estimators of distance correlation requires $\mathcal{O}(nlogn)$[102,238] computational complexity for univariate and $\mathcal{O}(nKlogn)$ complexity[234] for multivariate settings with $\mathcal{O}(\max(n, K))$ memory, where K is the number of random projections required as part of the estimation. Distance correlation has been shown to be a simpler special case of other recent popular measures of dependence such as Hilbert-Schmidt Independence Criterion (HSIC), Maximum Mean Discrepancy (MMD) and Kernelized Mutual Information (KMI) that have been extensively studied and used in the machine learning and statistics community[458,510]An advantage of using a simpler alternative is that in addition to being differentiable and easily computable with a closed-form, it requires no other tuning of parameters and is self-contained, unlike HSIC, MMD, and KMI that depend on a choice of separate kernels for features as well as labels along with their respective tuning parameters.

10.5 Distance covariance Bounds

We propose some bounds on distance covariance as follows. Using Cauchy-Bunkowski inequality

$$
\begin{aligned}
|\phi_{X,Y}(t,s) - \phi_X(t)\phi_Y(s)|^2 &= \left[E\left(e^{i\langle t,X\rangle} - \phi_X(t)\right)\left(e^{i\langle s,Y\rangle} - \phi_Y(s)\right)\right]^2 \\
&\leq E\left[e^{i\langle s,X\rangle} - \phi_X(t)\right]^2 E\left[e^{i\langle s,Y\rangle} - \phi_Y(s)\right]^2 \\
&= \left(1 - |\phi_X(t)|^2\right)\left(1 - |\phi_Y(s)|^2\right).
\end{aligned}
$$

Using Fubini's theorem and Lemma 1 in[500], we have

$$
\begin{aligned}
&\int_{\mathbb{R}^{p+q}} |\phi_{X,Y}(t,s) - \phi_X(t)\phi_Y(s)|^2 \, d(t) \\
&\qquad \leq \int_{\mathbb{R}^p} \frac{1 - |\phi_X(t)|^2}{c_p|t|_p^{1+p}} dt \int_{\mathbb{R}^q} \frac{1 - |\phi_Y(s)|^2}{c_q|s|_q^{1+q}} ds
\end{aligned}
$$

Upon rearranging the Heisenberg uncertainty principle, we get

$$
\frac{\left[\int |f(x)|^2 dx\right]^2}{4\pi \int |xf(x)|^2 \, dx} \leq \int |t\phi_X(t)|^2 dt
$$

where, $\phi_x(t) = \mathbb{E}\left(e^{itx}\right) = \int_{-\infty}^{\infty} e^{itx} f_x(x) dx$.

We now need to upper-bound $\boxed{\int_{\mathbb{R}^p} \frac{1 - |\phi_X(t)|^2}{c_p|t|_p^{1+p}} dt}$ by simplifying the above inequality.

Observe that we have

$$
\int |t\phi_X(t)|^2 dt = \int |t\mathbb{E}(\cos(tx) + i\sin(tx))|^2 dt
$$

and

$$\int |t\mathbb{E}(\cos(tx) + i\sin(tx))|^2 dt = \int |t^2||\mathbb{E}(\cos(tx)) + i\mathbb{E}(\sin(tx))|^2 dt.$$

Now, we observe that in the complex plane, for each fixed t, we essentially have a unit circle with an infinite distribution of points on it. As a result, the weighted average, e.g., the expected value of the points, must be inside this circle as it is convex, and the median of a set of points forming a convex figure is inside this figure.

This means that we have that since any point in or on the unit circle has distance at most 1 from the origin, then

$$|\mathbb{E}(\cos(tx)) + i\mathbb{E}(\sin(tx))| \leq 1 \to |\mathbb{E}(\cos(tx)) + i\mathbb{E}(\sin(tx))|^2 \leq 1$$

$$\to \int |t^2||\mathbb{E}(\cos(tx)) + i\mathbb{E}(\sin(tx))|^2 dt \leq \int |t^2||1|^2 dt = \int t^2 dt = \frac{t^3}{3}.$$

This gives us an upper bound on the such value.

10.6 Smoothness-independence-variance (SIV) phenomenon

In this section, we propose an interesting phenomenon that in order to reduce statistical dependency via distance correlation between $\mathbf{X}$ and a $\mathbf{Z}$ learnt through a map $f : \mathbf{X} \mapsto \mathbf{Z}$, the product of the Lipschitz smoothness of the map and a ratio of total variations of $\mathbf{X}$ and $\mathbf{Z}$ needs to be minimized. We call this phenomenon the *smoothness-independence-variance* (SIV) phenomenon, and it helps explain the requirements in order to learn a $\mathbf{Z}$ through a Lipschitz smooth map such that it is reasonably decorrelated with raw data

X.

SIV Inequality

We mathematically characterize this SIV phenomenon in this section via an upper bound that we derive on distance correlation. Based on the definition of Lipschitz continuity, we have the following bound where L is the Lipschitz constant of the map that learns $\mathbf{Z}$ from $\mathbf{X}$,

$$||\mathbf{Z_i} - \mathbf{Z_j}||^2 \leq L||\mathbf{X_i} - \mathbf{X_j}||^2 \tag{10.2}$$

Multiplying by $\langle \mathbf{X_i}, \mathbf{X_j} \rangle$ on both sides and summing over all points we have

$$\sum_{ij} ||\mathbf{Z_i} - \mathbf{Z_j}||^2 \langle \mathbf{X_i}, \mathbf{X_j} \rangle \leq L \sum_{ij} ||\mathbf{X_i} - \mathbf{X_j}||^2 \langle \mathbf{X_i}, \mathbf{X_j} \rangle$$

Now dividing on both sides by
$\sqrt{\sum_{ij} ||\mathbf{Z_i} - \mathbf{Z_j}||^2 \langle \mathbf{Z_i}, \mathbf{Z_j} \rangle} \sqrt{\sum_{ij} ||\mathbf{X_i} - \mathbf{X_j}||^2 \langle \mathbf{X_i}, \mathbf{X_j} \rangle}$ we get

$$DCOR(\mathbf{X}, \mathbf{Z}) \leq \frac{L\sqrt{\sum_{ij} ||\mathbf{X_i} - \mathbf{X_j}||^2 \langle \mathbf{X_i}, \mathbf{X_j} \rangle}}{\sqrt{\sum_{ij} ||\mathbf{Z_i} - \mathbf{Z_j}||^2 \langle \mathbf{Z_i} \mathbf{Z_j} \rangle}} \tag{10.3}$$

But $\frac{\sqrt{\sum_{ij} ||\mathbf{X_i} - \mathbf{X_j}||^2 \langle \mathbf{X_i}, \mathbf{X_j} \rangle}}{\sqrt{\sum_{ij} ||\mathbf{Z_i} - \mathbf{Z_j}||^2 \langle \mathbf{Z_i} \mathbf{Z_j} \rangle}}$ is the ratio of distance standard deviations which is the square root of distance variance which is in turn distance covariance between a variable and itself. It has been shown in[164] that the distance standard deviation can be upper bounded by the trace of the covariance matrix. Therefore we have

$$DCOR(\mathbf{X}, \mathbf{Z}) \leq \frac{L.Tr(\Sigma_{\mathbf{X}})}{Tr(\Sigma_{\mathbf{Z}})} \tag{10.4}$$

Therefore, the distance correlation can be bounded by the product of the Lipschitz constant of the map $f : \mathbf{X} \mapsto \mathbf{Z}$ and the ratio of total variations of the covariance matrices of $\mathbf{X}$, $\mathbf{Z}$ respectively. Therefore, for a decorrelation function, if the total variation of its output is not much larger (if not closer) to the total variation of its input, then the decorrelation function has to be relatively smoother in order to be able to effectively decorrelate the dataset in order to compensate for the larger ratio of the total variations of input and output.

Population SIV bound on distance covariance

The work in[458] shows an equivalence between distance correlation and another popular measure of statistical dependence called Hilbert Schmidt independence criterion (HSIC) by just a constant. Equation 10.5 is based on[204] and gives an Hoeffding bound on the quality of the sample estimator $DCOV(X, Z)$ of distance covariance in estimating the population distance covariance $DCOV(\mathcal{F}_{\mathcal{X}}, \mathcal{G}_{\mathcal{Z}})$ where F_X, G_Z represent the true distributions of the samples $\mathbf{X}, \mathbf{Z}$. The bound is given below

$$|DCOV(\mathcal{F}_{\mathcal{X}}, \mathcal{G}_{\mathcal{Z}}) - DCOV(X, Z)| \lessapprox \sqrt{\frac{log(6/\delta)}{0.24n}} + \frac{C}{n} \tag{10.5}$$

with probability at least $1 - \delta$.

Population SIV inequality Therefore, combining our sample SIV inequality with this concentration Hoeffding bound on the quality of estimating population distance covariance from sample distance covariance, we get the population SIV inequality as

$$DCOV(p_{xy}, \mathcal{F}, \mathcal{G}) \lessapprox \sqrt{\frac{log(6/\delta)}{0.24n}} + \frac{C}{n} + \frac{Lc_x}{\sqrt{\sum_{ij} \|\mathbf{Z_i} - \mathbf{Z_j}\|^2}} \tag{10.6}$$

Theorem 10.6.1. Under Gaussianity assumptions on $\mathbf{Z}$, minimizing $DCOV(\mathbf{X_j}, \mathbf{Z})$ is equivalent to minimizing mutual information $MI(\mathbf{X_j}, \mathbf{Z})$. This is also equivalent to maximizing the error in linear regression with respect to the attributes $\mathbf{Z}$ instead of coefficients as

$$\underset{\mathbf{Z}}{\text{argmax}} \; \|\mathbf{X_j} - \mathbf{Z}\beta\| \text{ s.t, } \beta = (\mathbf{Z^T Z})^{-1}\mathbf{Z^T X_j}$$

Proof. Maximizing $KL(P_{\mathbf{Z},\mathbf{X}}||P_{\mathbf{Z}}P_{\mathbf{X}}) \equiv$ minimizing $MI(\mathbf{Z}, \mathbf{X}) \equiv$ maximizing $\mathbb{E}(i(\mathbf{Z}(\mathbf{X}); \mathbf{X}))$ where KL refers to KL-divergence, MI refers to mutual information and $i(.)$ refers to information density[232].

For obfuscation of a chosen sensitive feature j, the goal is to minimize 10.9

$$i(\mathbf{Z}(\mathbf{X_1}, \mathbf{X_2} \ldots \mathbf{X_{j-1}}); \mathbf{X_j})|(\mathbf{X1}_, \mathbf{X_2}, \ldots \mathbf{X_{j-1}}) \tag{10.7}$$

For ease of notation, denote $\mathbf{X_1}, \mathbf{X_2} \ldots \mathbf{X_{j-1}}$ as $\mathbf{X^{-j}}$ and the goal above can be simply restated as being

$$\underset{\mathbf{Z}}{\text{argmin}} \; \mathbf{i}(\mathbf{Z}(\mathbf{X^{-j}}; \mathbf{X_j})) - \mathbf{i}(\mathbf{Z}(\mathbf{X^{-j}}); \mathbf{X^{-j}})$$

$$\equiv \underset{Z}{\text{argmin}} \; MI(\mathbf{Z}(\mathbf{X^{-j}}); \mathbf{X_j}) - \mathbf{MI}(\mathbf{Z}(\mathbf{X^{-j}}); \mathbf{X^{-j}})$$

and in the Gaussian case

$$MI(\mathbf{Z}, \mathbf{X_j}) = \frac{-1}{2} log[\frac{det(\mathbf{C})}{det(\mathbf{Z^T Z})\mathbf{det}(\mathbf{X_j^T X_j})}] \tag{10.8}$$

where $\mathbf{C} = \begin{pmatrix} \mathbf{Z^T Z} & \mathbf{Z^T X_j} \\ \mathbf{X_j^T Z} & \mathbf{X_j^T X_j} \end{pmatrix}$. Since minimizing mutual information is equivalent to maximizing

$$e^{-2MI} = \frac{\det(\mathbf{C})}{\det(\mathbf{Z^T Z X_j^T X_j})} \tag{10.9}$$

We show that minimizing regularized distance covariance maximizes the determinant $\det(\mathbf{C})$ while minimizing the determinant $\det(\mathbf{Z^T Z})$. As the denominator of 10.9 decreases while the numerator increases with the reduction of distance covariance, we also have that the mutual information between the smashed data $\mathbf{Z}$ and hidden attribute $\mathbf{X_j}$ decreases as a result. In addition, we know that equation 10.9 is equivalent to

$$\equiv \underset{\mathbf{Z}}{\operatorname{argmax}}\, \mathbf{X_j^T X_j} - \mathbf{X_j^T Z}(\mathbf{Z^T Z})^{-1}\mathbf{Z^T X_j} \equiv \underset{\mathbf{Z}}{\operatorname{argmin}}\, \mathbf{X_j^T \hat{X}_j}$$

This is in turn equivalent to doing the opposite of classical linear regression by maximizing the error with respect to learning covariates $\mathbf{Z}$ as

$$\underset{\mathbf{Z}}{\operatorname{argmax}}\, \|\mathbf{X_j} - \mathbf{Z}\beta\| \text{ s.t, } \beta = (\mathbf{Z^T Z})^{-1}\mathbf{Z^T X_j}$$

This provides an additional interpretation as well as a classical connection with respect to distance correlation for attribute privacy. □

10.7 Experiments

Reconstruction attack testbed: We empirically examine the privacy aspects of our method by designing a testbed that performs feature inversion[145] under different threat models for PCI. The goal of the testbed is to emulate attackers in order to examine information leakage both qualitatively and quantitatively. We use the attack testbed for both supervised decoder and likelihood maximization attacks as described in the attacks part of section 10.3.

The decoder attack architecture consists of upsampling layers composed of transpose convolutions. Similar architectures have been used in generative models for generating images from low-dimensional latent codes. Under the threat model for a decoder attack, the attacker has access to a dataset consisting of multiple samples of (z_l, x). Input to the testbed is the intermediate activations, z_l from any arbitrary layer l of the target model, and output is the generated image $\hat{x}$. After the training of the defense component (NoPeek or baselines), we use a held-out validation set to generate intermediate activations using the client network of the defense component. We thereby generate a paired dataset of activations and corresponding images. We use this paired dataset to train the reconstruction testbed to emulate the attacker. We use 90% of the original validation dataset for training the reconstruction testbed and the remaining 10% as the test set for qualitative evaluation of reconstruction quality. The training is a standard supervised decoder training on a dataset of z_l, x pairs with a loss function of the Euclidean norm between x and $\hat{x}$. We want to emphasize that there may potentially be a better design for architectures of the reconstruction testbed and better loss functions,

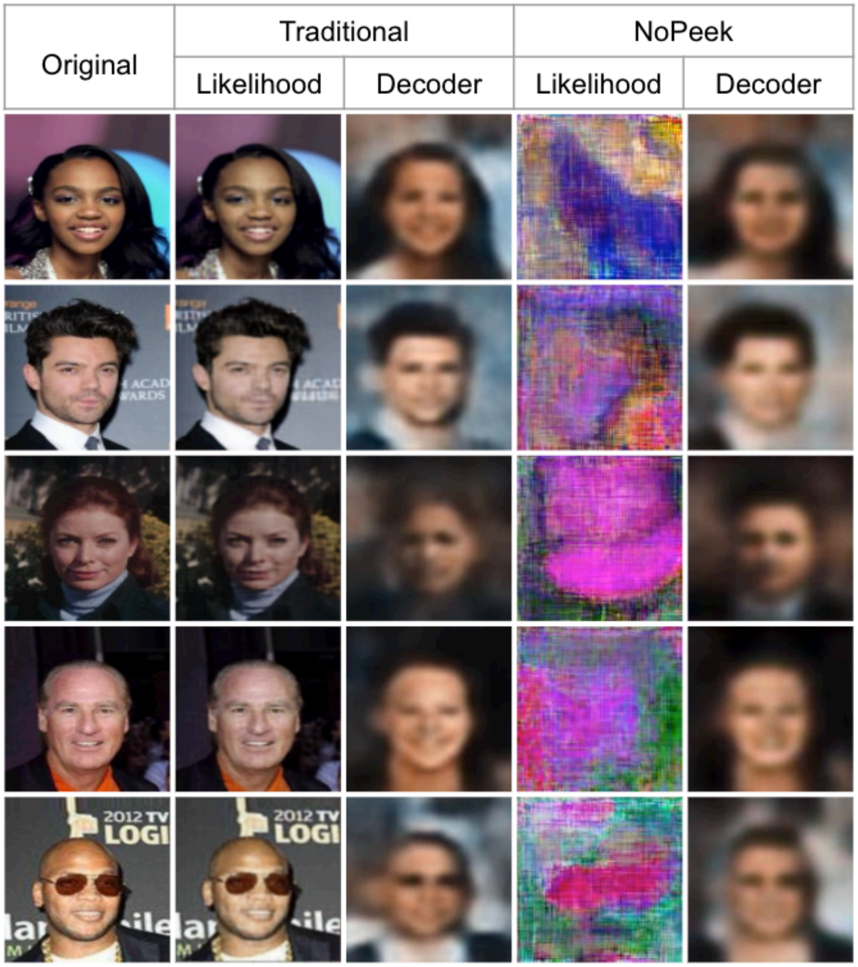


Figure 10.4: Likelihood vs. Decoder Reconstruction Attacks: A qualitative comparison between likelihood and supervised decoder reconstruction attacks on traditional and NoPeek methods. While the likelihood attack performs a visually similar reconstruction for the traditional approach, the decoder attack gets a better reconstruction result for NoPeek. However, in the case of NoPeek, the attack results in a blurred and average face image across a certain set of facial attributes. The purpose of this result is to illustrate the relative benefit of using different types of adversaries when evaluating NoPeek and other baselines.

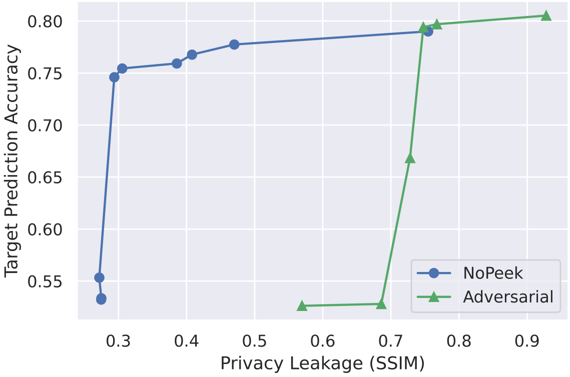


Figure 10.5: Privacy-Utility Trade-off: We vary the value of α to display the relationship between privacy leakage and task utility. Leakage is measured as the SSIM score between input and reconstructed images from the likelihood attack scheme.

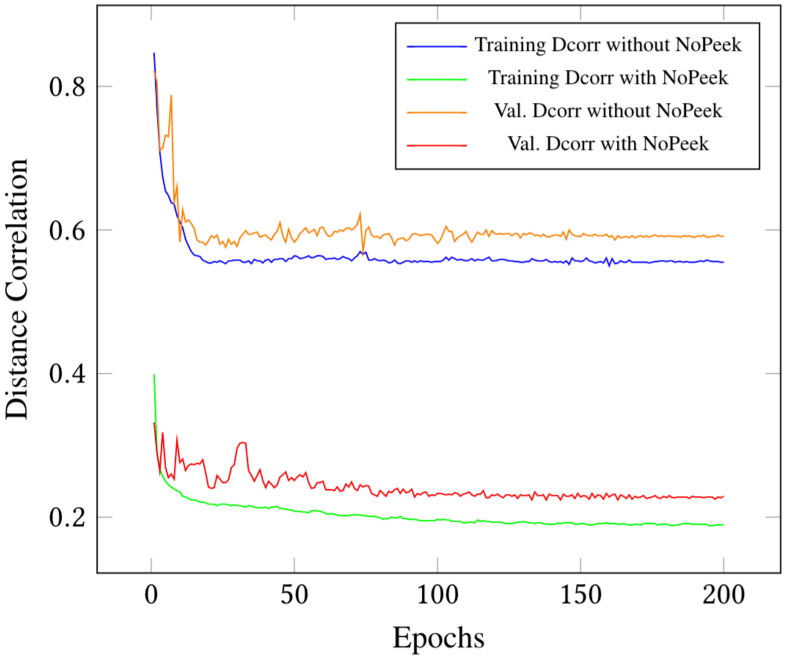


Figure 10.6: We plot distance correlation during training and testing as the network gets trained on UTK faces with and without NoPeek-Infer.

but the goal of this paper is just to have a fair comparison between NoPeek-Infer-based training and regular training of deep networks using a reasonable reconstruction architecture. The number of upsampling layers in the architecture of the testbed varies depending upon the difference in the dimensionality of z_l and x. Next, we evaluate the performance of the likelihood maximization attack. The threat model for likelihood maximization attacks requires the adversary to have access to client network weights and the architecture. The details of the likelihood maximization attack inspired by the work on deep image priors[519] is described in section 10.3. This attack has not been used in the privacy community looking at the feature inversion problem but used for several vision tasks like super-resolution, denoising, and feature inversion.

10.7.1 DATASETS

CELEBA

CelebA[329] is a large-scale celebrity face dataset with 202,599 face images that are well aligned and centered. These faces span 10,177 identities, each of which is associated with 40 different binary attributes.

FAIRFACE

Fairface[299] is a dataset of 108,501 face images with three attributes – gender, race, and ethnicity. The images are centered but contain different poses and lighting. We evaluate our approach using gender as the target attribute for both datasets.

Baselines: Our experiments consist of four categories of activation sharing methods - traditional (no defense), adversarial defense, siamese embedding defense, and noise-based defense, as detailed in section 10.3.2. Traditional refers to the setup where the client shares activations with the server without any specific defense. Adversarial refers to the set of techniques[312,54] that jointly trains a proxy adversary, resulting in a min-max optimization between the adversary and client network. Siamese embedding-based privacy is via a combination of a contrastive loss, and an accuracy loss as detailed in[395]. Noise is the category of baseline where we add Gaussian noise to the intermediate activations. While not related to activation sharing, many differentially private mechanisms add similar noise calibrated to sensitivity[158,160]. Even though we do not calibrate the noise, we try a broad range of noise spanning across the highest and lowest attainable utility. In all of our experiments, we train a standard ResNet-18[219] to minimize the loss on the main task. In all of our reported experiments, we use Adam optimizer with an initial learning rate of $1 \times e^{-3}$ and exponential decay for training. In the first experiment, we study the role of intermediate layer l by evaluating privacy and utility across different blocks of ResNet-18 for different methods. Figure 10.2 shows the qualitative results for different approaches. For the first five blocks, all techniques fail to defend against the likelihood of attack. However, NoPeek-Infer provides adequate protection at block-6. In order to prevent any selection bias for the qualitative result, we also show reconstruction for six random samples from the dataset in Figure 10.8. We compared the baselines and NoPeek-Infer on different metrics of image reconstruction quality and predictive utility of the model as shown in Table 10.2. We compared defenses of NoPeek-Infer & various baselines on the reconstruction of sensitive input with respect to likelihood maximization attack & observed that the defense of NoPeek-Infer

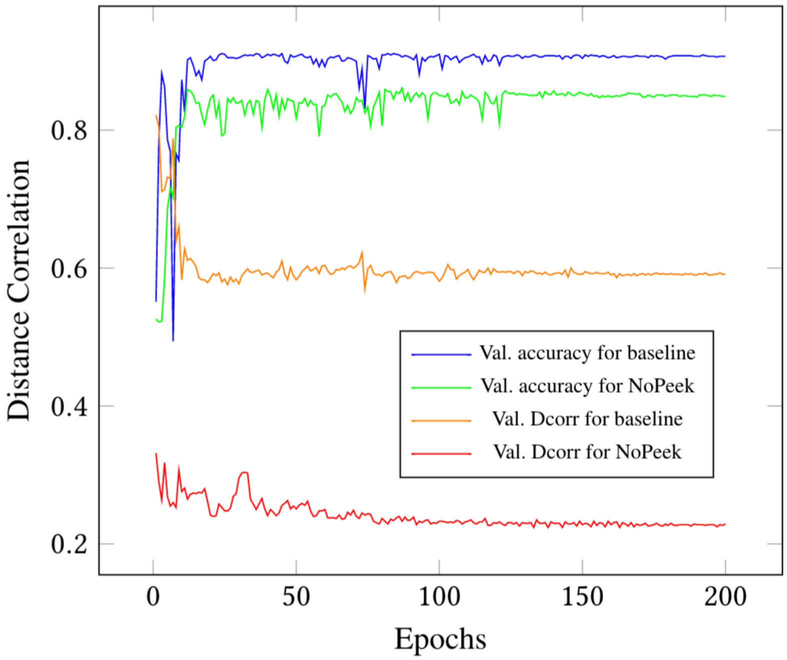


Figure 10.7: By introducing NoPeek-Infer in the training of the network, we obtain a major decrease in the distance correlation from 0.6 (baseline) to 0.22 (NoPeek-Infer). In contrast, the decrease in the accuracies is relatively much lesser.

performs the best by achieving the worst reconstruction when attacked, which indicates that NoPeek-Infer is a better method for preventing reconstruction attacks. In terms of the broader trend, we observe that NoPeek-Infer fared the best, followed by DeepObfuscator and then followed by Siamese Embedding, PAN, and Noise (& Blur) approaches in preventing the reconstruction attack in terms of SSIM score, PSNR, and l_1 metrics. We also compare against a primitive baseline that is based on the reduction of linear correlation as opposed to our proposed approach of nonlinear correlation minimization to show that the distance correlation (or nonlinear correlation) based approach is substantially better. While this comes at the cost of a small drop in accuracy, we note that the improvement in privacy is much higher than the corresponding reduction in utility. To further examine the privacy-utility trade-off, we vary the trade-off parameter for both adversarial and NoPeek-Infer and plot different points along the privacy-utility trade-off in Figure 10.5. As we reach higher privacy, the utility performance drops faster for adversarial in comparison to NoPeek-Infer. This makes NoPeek-Infer amenable to high privacy regimes without any significant loss in the utility. It is an accepted standard that privacy-utility trade-offs exist in privacy-preserving machine learning, and thereby, the above tradeoff observed in NoPeek-Infer is competitive.

10.8 Discussion

For comparing with the noise baseline, we add Gaussian noise to every component of the z_l vector with varying standard deviation σ of the noise for different experiments. We empirically observe that even for $\sigma = 400$ the reconstruction happens successfully using the likelihood attack while the utility gets close to chance accuracy. This

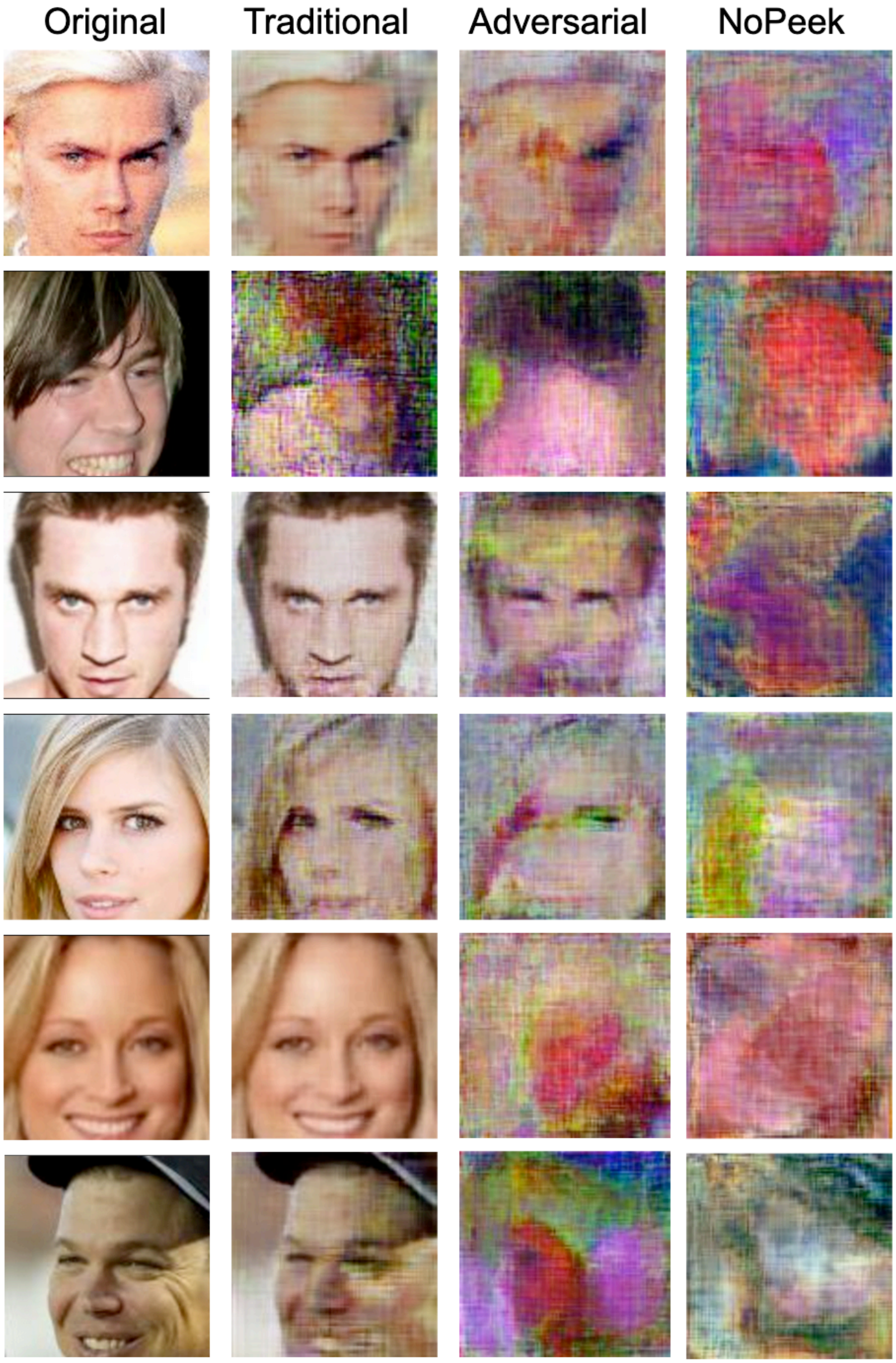


Figure 10.8: Reconstruction across different samples

Dataset	Method	SSIM ↓	PSNR ↓	ℓ_1 ↑	Utility ↑
Fairface	Traditional[210]	0.915 ± 0.110	72.982 ± 6.682	0.066 ± 0.051	**0.9912**
	PAN[326]	0.777 ± 0.218	69.585 ± 7.403	0.097 ± 0.069	0.9864
	NoPeek-Infer (Ours)	**0.306 ± 0.141**	**60.453 ± 2.813**	**0.206 ± 0.057**	0.9803
	Blur[393]	0.893 ± 0.884	61.2864 ± 2.5906	0.1066 ± 0.045	0.9881
	Gaussian Noise	0.842 ± 0.233	70.235 ± 2.672	0.0771 ± 0.045	0.8857
	Laplacian Noise	0.733 ± 0.1495	69.488 ± 5.539	0.0701 ± 0.0858	0.8568
	DeepObfuscator[312]	0.4467 ± 0.107	61.19 ± 3.935	0.191 ± 0.0894	0.9811
	Siamese Embedding[395]	0.484 ± 0.117	61.712 ± 1.169	0.198 ± 0.066	0.9511
	Linear Correlation	0.585 ± 0.02	67.789 ± 3.283	0.0625 ± 0.01	0.9115
CelebA	Traditional[210]	0.563 ± 0.237	65.655 ± 4.968	0.123 ± 0.067	**0.9759**
	PAN[326]	0.646 ± 0.168	64.650 ± 4.485	0.121 ± 0.056	0.9513
	NoPeek-Infer (Ours)	**0.239 ± 0.081**	58.901 ± 1.835	**0.240 ± 0.053**	0.9488
	Blur[393]	0.524 ± 0.168	60.248 ± 5.15	0.1373 ± 0.0669	0.9452
	Gaussian Noise	0.656 ± 0.187	63.584 ± 2.896	0.1348 ± 0.0352	0.9608
	Laplacian Noise	0.6276 ± 0.168	61.868 ± 5.011	0.1487 ± 0.0572	0.966
	DeepObfuscator[312]	0.2874 ± 0.0436	**56.3463 ± 1.479**	0.2189 ± 0.032	0.9531
	Siamese Embedding[395]	0.539 ± 0.249	59.243 ± 3.5206	0.185 ± 0.085	0.9376
	Linear Correlation	0.4154 ± 0.0913	60.342 ± 4.17	0.203 ± 0.0745	0.944

Table 10.2: Comparison for sensitive input leakage: We compare defenses of NoPeek-Infer & baselines on reconstruction of sensitive input with respect to likelihood maximization attack & observe that the defense of NoPeek-Infer performs the best by achieving a worst reconstruction when attacked.

illustrates that Gaussian noise mechanism is approximately the same as the *traditional* category due to its inability to provide any privacy-utility trade-off despite adjusting $\sigma = 0$.

To show the trade-off between privacy and utility via choice of α we plot the distance correlation of a given intermediate activation during training a NoPeek-Infer network and a traditional network without NoPeek-Infer in Figure 10.6. This demonstrates that the network without NoPeek-Infer naturally reduces distance correlation during training. Our proposed method can be seen as an additional regularization, which forces the network to regularize for the reduction in distance correlation at a much higher rate between raw data and activations. The consistency between training and testing distance correlation in Figure 10.6 also demonstrates the capability of weights learnt by NoPeek-Infer in generalizing the decorrelation phenomenon to prevent reconstruction attacks.

The first row of Figure 10.3 shows some raw input images, and the output of the first layer of the trained network when NoPeek-Infer is not used is shown in the second row. The third row shows the output at the first layer in the case when NoPeek-Infer is used. We restrict it to only three output channels to visualize only the RGB component as part of a qualitative investigation. As seen, the second row visually leaks a lot of information about the raw image in comparison to the third row. This demonstrates semantically meaningful obfuscation performed by the layers of the client network when trained with NoPeek-Infer. In Figure 10.7, we observe that the accuracy dropped by a relatively small amount compared to the drop in distance correlation (or leakage of sensitive information). This relative difference can be controlled by tuning α. The important aspect to note from the figure is that the distance correlation between the samples and activations can be reduced significantly without any significant drop in accuracy.

10.9 Conclusion

The proposed NoPeek-Infer schemes based on distance correlation seem to have versatile applicability in the space of privacy, computer vision, and machine learning, given that they do not require major changes in the model setup and architectures except for the proposed modification to the loss function. It would be great to realize on-device implementations of the NoPeek-Infer scheme. With regards to human visual perception of bias and privacy, we would also like to conduct a large-scale crowdsourced survey to compare the performance of human participants in deciphering the true sensitive image upon looking at NoPeek-Infer results in comparison to a uniform random choice.

10.10 Appendix A: Gradient of distance correlation

Distance correlation between centered data can be represented as $\frac{Tr(\mathbf{X^TXZ^TZ})}{\sqrt{Tr(\mathbf{X^TX})^2 Tr(\mathbf{Z^TZ})^2}}$ in a graph-theoretic dual space[547]. Distance covariance in the numerator can be written as $Tr(\mathbf{X^TZX}) = \sum_{ij}\langle z_i, z_j\rangle(\|x_i - x_j\|)^2$. This can be written in matrix form using basis vectors e_i, e_j as

$$\sum_{ij}[Tr(\mathbf{Z^Te_ie_j^TZ})\mathbf{Tr}(\mathbf{X^T(e_i-e_j)(e_i-e_j)^TX})] \tag{10.10}$$

Simplifying the notation with $M_{ij} = e_i e_j^T$ and $A_{ij} = (e_i - e_j)(e_i - e_j)^T$ we have $\frac{\partial Tr(\mathbf{Z^TL_ZZ})}{\partial \mathbf{Z}} = \sum_{ij}(\mathbf{2M_{ij}Z})\mathbf{Tr}(\mathbf{X^TA_{ij}X})$. On the lines of 10.10, we have $Tr(\mathbf{Z^TL_ZZ}) = \sum_{ij}[Tr(\mathbf{Z^TM_{ij}Z})\mathbf{Tr}(\mathbf{Z^TA_{ij}Z})]$. Therefore, utilizing these identities, the derivative of squared distance correlation w.r.t $\mathbf{Z}$ can be written as

$$\frac{c_x Tr(\mathbf{Z^TL_ZZ})\frac{\partial Tr(\mathbf{X^TL_ZX})}{\partial \mathbf{Z}} - [Tr(\mathbf{X^TL_ZX})]^2 c_x \frac{\partial Tr(\mathbf{Z^TL_ZZ})}{\partial \mathbf{Z}}}{[Tr(\mathbf{Z^TL_ZZ})]^2}$$

up to a constant.

10.10.1 Deep-learning friendly source code for sample distance correlation

```python
def pairwise_dist(A):
    r = tf.reduce_sum(A*A, 1)
    r = tf.reshape(r, [-1, 1])
    D = tf.maximum(r - 2*tf.matmul(A, tf.transpose(A)) + tf.transpose(r), 1e-7)
    D = tf.sqrt(D)
    return D

def dist_corr(X, Y):
    n = tf.cast(tf.shape(X)[0], tf.float32)
    a = pairwise_dist(X)
    b = pairwise_dist(Y)
    A = a - tf.reduce_mean(a, axis=1) -\
    tf.expand_dims(tf.reduce_mean(a,axis=0),axis=1)+\
    tf.reduce_mean(a)
    B = b - tf.reduce_mean(b, axis=1) -\
    tf.expand_dims(tf.reduce_mean(b,axis=0),axis=1)+\
    tf.reduce_mean(b)
    dCovXY = tf.sqrt(tf.reduce_sum(A*B) / (n ** 2))
    dVarXX = tf.sqrt(tf.reduce_sum(A*A) / (n ** 2))
    dVarYY = tf.sqrt(tf.reduce_sum(B*B) / (n ** 2))

    dCorXY = dCovXY / tf.sqrt(dVarXX * dVarYY)
    return dCorXY
```

11

Split learning on Vertically Partitioned Data

11.1 Introduction

Leading banks and financial services are currently using deep learning algorithms to optimize their processes on targeted tasks, such as approving loans, assessing risk, and carrying out credit scores, among others. The financial sector generates huge amounts of data daily and is always in need of better ways to assess risk detect fraud, and utilize the data efficiently. Even if considerable progress has been made in a data-intensive industry, financial services companies face challenges to efficiently adapt to the latest data processing techniques, and there is an increasing pressure to access third-party data to improve their operational efficiency. On top of these challenges, there are issues like regulatory compliance costs, competition, legacy infrastructures, and security concerns that prevent financial services companies from effectively using data. Unlocking data silos and uncovering novel sources of data will provide a competitive advantage, and companies in regulated sectors will have higher network effects than their competitors.

Currently, each company works in isolation, where they keep their data private and use that to build their own proprietary models. On the other hand, banks utilize third-party data and resources pulled from several companies as services to build customized models for their targeted tasks. However, with the recent rise of a new distributed deep learning: SplitNN[210,538,465] architecture, a new way to process data collaboratively has emerged without conceding ownership or loosening privacy requirements. Furthermore, using SplitNN also enables the use of distributed sources of data, which results in improved and generalized robust models. Tapping into different sources of data, which is often private and owned by several companies, poses a new set of challenges that vertical SplitNN can address.

Motivated by the above observations, we are interested in a setting where multiple entities (clients) collaborate for a targeted task at hand under the coordination of a central server or service provider. Each clients raw data is stored locally and is not exchanged or transferred; instead, focused updates intended for immediate aggregation are used to achieve the learning objective. Usually, the data is split horizontally, meaning that each company holds a unique set of features over a non-overlapping set of users. As mentioned earlier, in an industry with huge amounts of data on every client, such as the financial services industry, companies often have enough data of a particular type. In that scenario, SplitNN provides more value since it allows the utilization of data from several parties[465,466]. The other variants to SplitNN for distributed private training can also be achieved with Federated Learning[349,266], but unlike SplitNN, it is necessary to share the complete model with all the clients. In essence, the goal of our work is to learn a shared model using vertically partitioned data coming from several sources while preserving data privacy. This chapter is based on our work in[89].

11.2 Related Work

In this section, we share related works on several techniques for vertically partitioned machine learning. We categorize these works under the following categories.

1. **Vertically partitioned linear and logistic regression** The work in[191] proposes a multi-party computation (MPC) scheme based on garbled circuits for secure linear regression in the vertically partitioned setting. The works in[578,579] provide schemes for secure vertically partitioned logistic regression based on homomorphic encryption.

2. **Vertically partitioned decision trees** The works in[191,528,527,288,109] share approaches for vertically partitioned learning with decision trees, gradient boosted decision

trees and Hoeffding trees.

3. **Vertically partitioned SVM** The work in[471] shows a threshold Paillier and blockchain-based secure approach for using support vector machines on vertically partitioned data. Similarly, the work in[489] shows a simpler approach of using core vector machines for anomaly detection using vertically partitioned data.

4. **Vertical federated learning** Learning with vertically partitioned data in the context of federated learning, a popular distributed deep learning paradigm, was studied in[390,328,445,538,465,466]. Conventional solutions in this setting make use of expensive cryptographic schemes such as Homomorphic Encryption and Multi-Party Computation and thus face critical performance challenges and communication overhead. SecureNN[445] was proposed in 2018 and achieved great success in reducing the communication by over 8 times and in eliminating the requirement to use conventional cost-intensive oblivious transfer protocols.

Other lines of work try to avoid these challenges[328] following the same design principles as[349], and they propose Federated Stochastic Block Coordinate Descent (Fed-BCD). They show that applying classical Block Coordinate Descent to the FL setting can significantly reduce the communication cost. In their setting, they reduce the amount of communication by updating the model fewer times with richer local updates. This approach maximizes the information sent in each update since having hundreds of clients means each communication round is very expensive. Further, performing local updates on the clients requires sharing the labels, which is not always feasible.

Motivated by these observations,[538] proposed an approach called split learning in which a smaller fraction of the model is present on each client network and just the output of these models is shared with the server in every iteration. This results in smaller but more frequent updates, and it helps reduce communication and computational overhead for the clients.

11.3 VERTICAL SPLITNN

The overall goal of our work is to learn a shared model while preserving data privacy. To this end, we propose to train partial neural networks (NN) on each client and then aggregate all of their outputs before feeding them to the last stage of the combined model on the server side, as seen in Figure 11.1. We are inspired from SplitNN[538]. In particular, we extend SplitNN architecture to use all of the partial clients-networks on each iteration instead of using them sequentially. We employ five pooling mechanisms to aggregate the outputs of the partial networks via element-wise average, element-wise maximum, element-wise sum, element-wise multiplication, and concatenation.

Among all the aggregation mechanisms, concatenation is the simplest approach and is the closest to training a single network with all of the input features. However, this method requires having the intermediate outputs of all the networks on every iteration, so it is not robust to stragglers. Element-wise sum and average pooling are very close to each other. The main limitation of both of these aggregation methods is that all the networks need to have compatible shapes so that their outputs can be combined together. On the other hand, these methods allow the use of a secure aggregation proto-

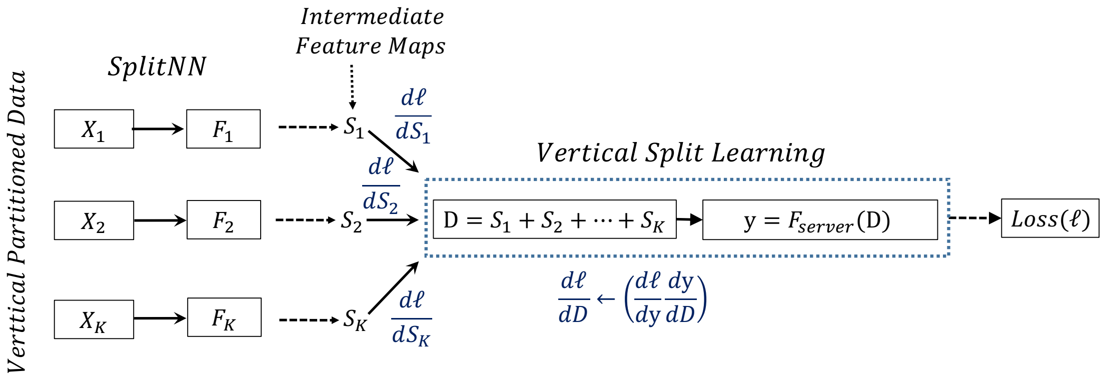


Figure 11.1: Vertical SplitNN architecture: Each client computes a fixed portion of the computation graph and passes it to the server which computes the rest and performs back-propagation and returns back the jacobians to the client which can perform their respective back-propagation.

col[66], which can enhance the privacy and security of the algorithm. Element-wise max pooling also requires all the networks to have compatible output shapes. In this case, we pick the activations with the maximum value for each neuron and discard the rest. All these setups require communication on every iteration since they are jointly optimized by back-propagating the error from the main network to the smaller ones. One can readily employ other encoding methods like Compact Bilinear Pooling[139,188], Temporal Compact Bilinear Pooling[464], NetVLAD[25] instead of the pooling mechanisms for a more robust representation learning.

Implementation: Split Learning was defined by[210]. We utilize SplitNN[210] architecture as a baseline architecture for distributed private training.

In our method, each partial clients-network encodes its data into a different space and then transmits it to train a shared deep servers-network. A deep neural network can be defined as a function F, describable as a sequence of layers $\{L_0, L_1, ...L_N\}$. For a given input X, the output of this function is given by $F(X)$, which is computed by sequential application of layers, given as:

$$F(X) \leftarrow L_N(L_{N-1}...(L_0(X)))$$

Gradients can be backpropagated over each layer to generate gradients of previous layers and to update the current layer. We will use $L_i^T(gradient)$ to denote the process of backpropagation over one layer and $F^T(gradient)$ to denote backpropagation over the entire neural network. Similar to forward propagation, backpropagation on the entire neural network is comprised of sequential backward passes, given as:

$$F^T(gradient) \leftarrow L_1^T(L_2^T...(L_N^T(gradient)))$$

The process of sequential computation and transmission followed by computation of the remaining layers is functionally identical to the application of all layers at once. Similarly, because of the chain rule in differentiation, backpropagating $F^T(gradients)$ is functionally identical to the sequential application of $F_a^T(F_b^T(gradients))$.

When extending this for multiple concurrent clients, as in the SplitNN-driven vertical partitioning case, the backpropagated error will be split, and each client network will pass the corresponding gradients. Let's take concatenation as an example. The full forward pass will be a concatenation of the forward passes coming from each client-network $\{F_1F_2\dots F_k\}$, $k \in \{1,\dots,K\}$ where K is the maximum number of clients. In the same way, the gradients on the concatenation layer will be $\{L_1L_2\dots L_k\}$. Thus, we only need to split these gradients before passing them to the upstream clients.

We evaluate our proposed method on three popular financial datasets, namely Bank

Marketing[373], Give me Some Credit[262] and Financial PhraseBank[340]. We focus on financial datasets because of the special relevance of vertically partitioned data, which is widespread in the industry. At the moment, financial institutions share plain and anonymized data with third parties for critical applications, but this is not the best solution due to multiple reasons: small amount of privacy offered by the anonymization scheme, no control over the usage, inability to audit the usage and marginal returns on the value of data is depleted with each partnership. Using our proposed split learning scheme, the need to pool all of the data together is obviated, keeping the data sources private, which enables unprecedented collaboration between the sharing parties and the non-rivalry of data[98] would potentially lead to increasing returns. Use cases in the industry range from multi-party borrowing detection, risk analysis, and fraud detection to cross-selling and customer retention.

11.4 Experiments

We present our evaluation of the proposed method on several different datasets. All of the datasets here are used for the prediction task.

Datasets and Implementation Details. We test our system experimentally on three financial datasets. The datasets are summarized in Table 11.1.

Table 11.1: Datasets. "#Samples" denotes the number of samples, "#Dim" denotes the dimensionality of the features and "#Classes" denotes the number of classes.

	bankmarketing	givemecredit	phrasebank
#Samples	45k	30k	5k
#Dim.	16	25	300
#Classes	2	2	3

Bank Marketing[373] is a dataset related to direct marketing campaigns of a Portuguese banking institution from UCI machine learning repository. We use all the 16 feature dimensions for prediction and distribute them vertically among the clients. The vertical split is done based on the source of the features, with the bank client data in one split and all the social and economic context attributes in the other.

Give Me Some Credit[262] is a dataset of financial data built for the task of predicting the likelihood of someone experiencing financial distress in the near future. Once again, there is no coherent vertical split for the data, so we chose to split the features arbitrarily into two sets.

Financial PhraseBank[340] consists of 4845 English sentences selected randomly from financial news found on the LexisNexis database. These sentences were then annotated by 16 people with backgrounds in finance and business. The annotators were asked to give labels according to how they think the information in the sentence might affect the mentioned company stock price. We use all the sentences in the dataset and apply GloVe[407] based embedding with 300 dimensions for the word embedding. After applying GloVe, we treat this embedding space as a feature space and split it arbitrarily into four vertical splits.

Evaluation Metric. We report accuracy as well as the F1 score to account for the class imbalance.

Multiple Clients. Due to the small number of features available, We use only two splits for both *Bank Marketing* and *Give me Credit* datasets. We use *Financial PhraseBank* to analyze the effect of splitting the dataset across a higher number of clients. From a practical standpoint, it is worth mentioning that unlike in horizontally distributed datasets, in vertical SplitNN, the number of clients is likely to remain small since relevant data about a single user is not usually distributed into too many sources.

11.4.1 Comparison with a centralized model

In Table 11.2, we compare the results of training a centralized model (M) with training several split models $(M1, M2, M3...)$, merging their outputs and using those as input for M. We choose max pooling as a merging technique for this comparison since it provides the best performance for the studied datasets overall.

As noted in the results, the Financial Phrasebank dataset is the only one where vertical partitioning and element-wise max pooling results in a drop in performance. This could be due to two reasons - the data we applied vertical partitioning over was obtained after the embedding, splitting the 300 GloVe features into 4 sets. The other reason could be due to the underlying semantic nature of a sentence, making it a difficult task for vertical partitioned learning from a practical standpoint. For all the other cases, the performance roughly remains the same, with some marginal improvements when using split learning.

Table 11.2: Comparison of the performance of a single model with access to the full dataset vs. a split model with four vertical partitions. We only report results for element-wise max pooling since it's the best performing merging technique.

	Single Model		Max Pooling	
Dataset	Acc	F1	Acc	F1
Bank Marketing	0.83	0.47	0.84	0.47
Give Me Credit	0.80	0.34	0.81	0.35
Financial PhraseBank	0.78	0.78	0.76	0.76

11.4.2 Comparison of merging strategies

In Table 11.3, we compare several strategies to merge the outputs of the models trained with the vertically partitioned features. We consider two pooling mechanisms as well as simple combinations such as concatenation, element-wise multiplication, and element-wise sum of the outputs.

The simplest strategy is concatenation; however, this requires all outputs from each participating client to be present during the forward pass, which could be infeasible in a real scenario since some of the clients may drop randomly or there might be synchronization issues. Therefore, any of the other strategies are preferable because of their aggregation mechanism.

Furthermore, both element-wise average pooling and simple element-wise addition over the inputs can allow us to use a secure aggregation protocol while combining the outputs of the smaller models. Thus providing an extra layer of security on top of the obfuscation provided by the models themselves and NoPeek[546].

We notice that the performance doesn't suffer huge drops with any of the methods. However, in practice, one could choose the average pooling since it allows the use of a secure aggregation protocol as well as compression techniques that can help with stronger privacy and communication overhead, respectively.

Figure 11.2 shows the loss and metrics during training for Financial PhraseBank. The centralized training (single model) takes fewer batches to converge,

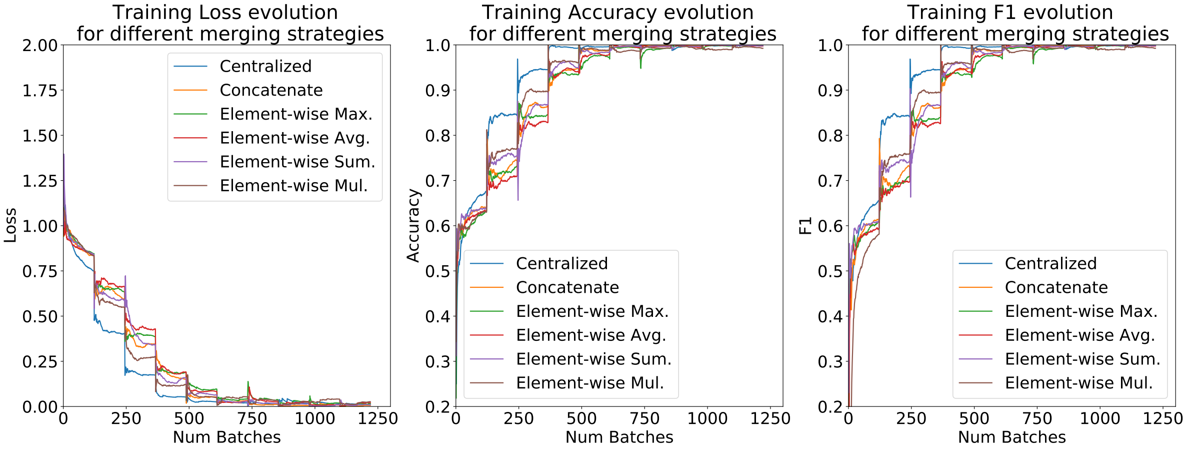


Figure 11.2: Comparison of several merging strategies for SplitNN-driven vertical partitioning with PhraseBank.

Table 11.3: Comparison of merging/pooling strategies.

	Financial PhraseBank		Bank Marketing		Give me Credit	
Merging	Acc	F1	Acc	F1	Acc	F1
Element-wise Max Pooling	0.76	0.76	0.84	0.47	0.81	0.35
Element-wise Average Pooling	0.77	0.77	0.83	0.46	0.82	0.36
Concatenation	0.76	0.76	0.82	0.46	0.83	0.37
Element-wise Multiplication	0.72	0.72	0.82	0.46	0.80	0.34
Element-wise Sum	0.77	0.76	0.83	0.46	0.77	0.32

11.4.3 Clients dropping randomly

In Table 11.4, we present the results of dropping some of the clients randomly both during training and testing.

The drop during the training means that the model is trained with the outputs of all models, but on each iteration, one or more of those outputs is missing. On the other hand, dropping during testing means that the model was trained with the outputs of all the models, but for the prediction on the test set, the output of some of the models from the client side is missing.

Table 11.4: Comparison of merging strategies when clients drop randomly. We report accuracy for the Financial PhraseBank.

	Training			Testing		
Merging	Drop 1	Drop 2	Drop 3	Drop 1	Drop 2	Drop 3
Element-wise Max Pooling	0.74	0.72	0.69	0.76	0.70	0.63
Element-wise Average Pooling	0.75	0.72	0.69	0.74	0.71	0.65
Element-wise Multiplication	0.75	0.75	0.71	0.71	0.60	0.58
Element-wise Sum	0.74	0.73	0.70	0.74	0.70	0.64

As shown in Table 11.3, in both cases, the performance suffers a significant impact as a consequence of the clients dropping. This is expected since we are missing the predictive power of several features. When we increase the number of clients that drop at each point, the performance hit is even bigger, which is consistent with our hypothesis.

Furthermore, in Figure 11.3, we can see that dropping more than two clients in a four-client setting, even affects the convergence of the model, and the loss starts to rise by the end of the training, indicating that optimization is drifting from local/global minima. This performance drop, however does not arise if a client drops just on a few iterations but is present for most of the training. This is an interesting starting point for future work since it would be interesting to analyze how to minimize the impact of stragglers with vertical SplitNN.

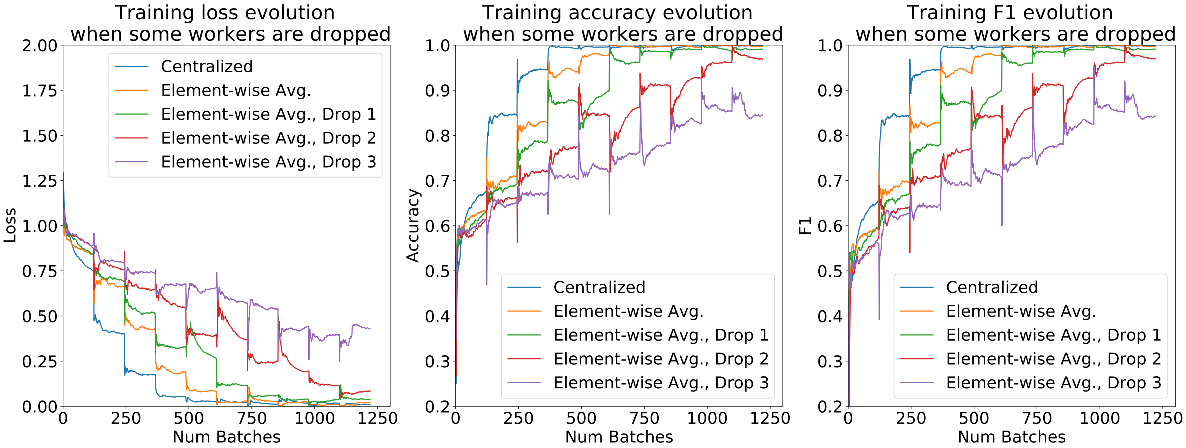


Figure 11.3: Loss and metrics for PhraseBank dataset while workers drop during training.

11.4.4 Measurement of communication and computational costs

Tests over different datasets were carried out in order to estimate the amount of communications performed in a vertical SplitNN training process. We use the roles defined in[88] to identify the type of data available for each of the participants. Role 1 only has access to features, role 3 has access to both features and labels, and role 0 is just a computation client with no data. Each test was carried out by three clients. One of them with role 1, another one with role 3 and the last one with role 0. Results are shown in Table 11.5.

Table 11.5: Communication costs in initialization, forward pass, and backward pass for each studied dataset.

Dataset	Financial PhraseBank			Bank Marketing			Give me Credit		
Role	1	3	0	1	3	0	1	3	0
Total sent per epoch (MB)	488	490	977	2,560	3,840	7,680	4,800	7,200	14,400
Total received per epoch (KB)	488	490	977	2,560	5,120	6,400	4,800	9,600	12,000

The communication cost computed for this table is based on the division of tasks according to different roles. Once the training starts for each batch, workers with roles 1 and 3 send the output of their next-to-last layer to role 0 worker, which performs its forward pass and sends the output of its next-to-last layer to worker with role 3 to compute the loss.

Similarly, once the loss is computed, the role 3 worker will send back to the role 0 worker the error at the output of the shared layer so that it can continue backpropagating it. Finally, this role 0 worker will send the error at the output of each corresponding shared layer back to its corresponding worker with role 1 or 3.

Table 11.6: Measurements of the computational costs

Dataset	Financial PhraseBank	Bank Marketing	Give me Credit
Number of parameters of the NN	3,907,059	745	457
FLOP/sample	33,667	4,041	741
us/batch (batch size=32)	26,037	911	793
MFLOPS (batch size=32)	41.377	141.945	29.902
us/batch (batch size=128)	97,871	1,114	1,107
MFLOPS (batch size=128)	44.031	464.316	85.680

The communication size in a vertical SplitNN architecture is dependent on the size of the output at the endpoints layer. The computational cost, however, is dependent on the architecture and the size of the input feature vector at each layer. For widely used architectures, everything remains the same here in comparison with traditional deep learning except for the size of the feature vector of the first layer on the central server.

Bearing this in mind, in conjunction with performance trade-offs between different merging strategies, it is extremely important to know the details and the limitations of the specific use case in order to propose the best training strategy. The neural network architecture splitting scheme between the workers or the adjustment of the hyperparameters can greatly change the speed and, therefore, the efficiency of the training process.

Thus, we find that in the training processes where the bottleneck is on the communication side, most of the training should be done by workers with roles 1 and 3 so that the outputs of their networks are already as small as possible. On the other hand, when the bottleneck is the computational cost, workers with roles 1 and 3 should have the minimum amount of layers to ensure the data is kept private, and the core of the model should be in a role 0 worker with a higher computational capacity. As shown in Table 11.6, other techniques, such as adjusting the batch size, could be highly convenient in some cases to speed up the training processes.

An interesting line of research for the future would be to study the effect on the convergence of compression techniques such as STC[451] or Random Rotation Matrix[284] as well as privacy-preserving techniques such as Secure Aggregation Protocol[66] or minimizing Distance Correlation[537], as well as their effect on the computational cost.

11.5 Conclusion

In this paper, we proposed split learning for vertically partitioned data and further addressed the specific challenges arising in this scenario. We have shown that the proposed methods to merge the outputs of the split networks result in a shared model that performs on par with the centralized model. Max-pooling is the best overall. However, we believe the small drop in performance shown with average-pooling is acceptable, considering that it allows the use of a secure aggregation protocol. We believe our approach to training models with vertically partitioned data provides a way that is better suited to its specific challenges, which are different from those arising with horizontally partitioned data.

Part III

Distributed and Private Scientific Computing

“A grain of wise subjectivity tells us more about the real world than any amount of objectivity.”

Judea Pearl

12

Regularized Eikonal PDEs for variable privacy-based geolocation release

12.1 Introduction

This chapter is based on our work in[Kabasakalolu et al.]. The work in this chapter has been initially motivated by the challenges of population-level data sharing against the backdrop of the COVID-19 pandemic. That said, we believe the solution offered widely applies to several location-enabled services and applications outside the context of pandemics. As a walkthrough example, we use the pandemic as a backdrop.

To curb disease exposure and transmission, many contact tracing apps have emerged. While most of these apps help the app users through exposure notification, they do not allow the general public communities, government entities, and city officials in any way. Any such feature is not present in many of the existing contact tracing apps because many apps use spatially invariant modalities like Bluetooth data, which does not provide any spatial context. The current GPS apps also do not provide this information due to privacy concerns concerning the infected individuals. In this paper, we propose using a system that allows individuals to release their locations in a privacy-preserving manner for the nodal authorities to build heatmaps so that governments and citizens can benefit from the aggregate data statistics without knowing about any individual's participation in this heatmap. Benefits of such heatmaps include 1) monitoring disease spread, 2) intervention planning and intervention outcome analysis, 3) epidemiological analysis, and 4) informing citizens.

12.2 LIPSCHITZ PRIVACY & DIFFERENTIAL PRIVACY

Definition 15. *(Lipschitz Privacy). Consider the normed space* $(\mathcal{U}, \|\cdot\|)$ *of private data, a privacy level* $\epsilon > 0$*, and a set* $\mathcal{Y}$ *of possible responses. Then, the mechanism* $\mathcal{Q} : \mathcal{U} \to \Delta(\mathcal{Y})$ *is* ϵ*-Lipschitz private if* $\ln \mathbb{P}(\mathcal{Q}(u) \in \mathcal{S})$ *is* ϵ*-Lipschitz in* u *for all* $\mathcal{S} \subseteq \mathcal{Y}$*.*

Assuming the mechanism Q possesses a probability density function $g(u, y) =$

$\mathbb{P}(Qu = y)$, where $g(u, y)$ is almost everywhere differentiable in u, the Lipschitz condition (3) translates to a point-wise bound on the derivative across the private input u as follows: $g(\cdot, y)$ is continuous for all $y \in \mathcal{Y}$ and,

$$\|\nabla g(u, y)\|_* \leq \epsilon g$$

, for a.e. $u \in \mathcal{U}$ and all $y \in \mathcal{Y}$, where $\|\cdot\|_*$ is the dual norm of $\|\cdot\|$. A. A Metric as Adjacency Relation The adjacency relation $\mathcal{A}$ in differential privacy is replaced by the metric $\|\cdot\|$ of the space $\mathcal{U}$ of private data. The composite adjacency relation (2) can be captured using ℓ_1 and ℓ_2-norms. Specifically, assume that the private data $u = [u_1, \ldots, u_n]$ is an aggregation of n individuals' highdimensional data $u_i \in \mathbb{R}^m$. Then, adjacency relation (2) can be relaxed to:

$$(u, u') \in \mathcal{A} \Leftrightarrow \sum_{i=1}^{n} \|u_i - u'_i\|_2 \leq \lambda$$

According to the Lipschitz-privacy framework and assuming the existence and differentiability of the density of the mechanism, adjacency relation above, the Lipschitz privacy definition translates into a bound on the derivative of the mechanism:

$$\|\nabla_{u_i} \ln g(u, y)\|_2 \leq \epsilon, \forall i \in \{1, \ldots, n\}.$$

Adjacency relation can be viewed as an ℓ_2-sensitivity constraint that ensures the privacy of high-dimensional data.

12.2.1 Equivalence with differential privacy

Proposition 1. *For any $\lambda > 0$. Then, an ϵ-Lipschitz private mechanism Q is $\alpha\lambda$-differentially private.*

Many popular differentially private mechanisms, such as the Laplace and the exponential mechanism, are also Lipschitz-differentially private. One exception that fails to satisfy Lipschitz-privacy constraints is the staircase mechanism since the underlying noise distribution is discontinuous. Specifically, the log-probability function $\ln \mathbb{P}(Qu = y\ )$ is discontinuous and, hence, is not Lipschitz.

12.2.2 Laplace mechanism as a special case of Lipschitz differential privacy

Proposition 2. *Let $s : \mathcal{U} \times \mathcal{Y} \to \mathbb{R}$ be L-Lipschitz in $\mathcal{U}$. Then, the mechanism Q with density*

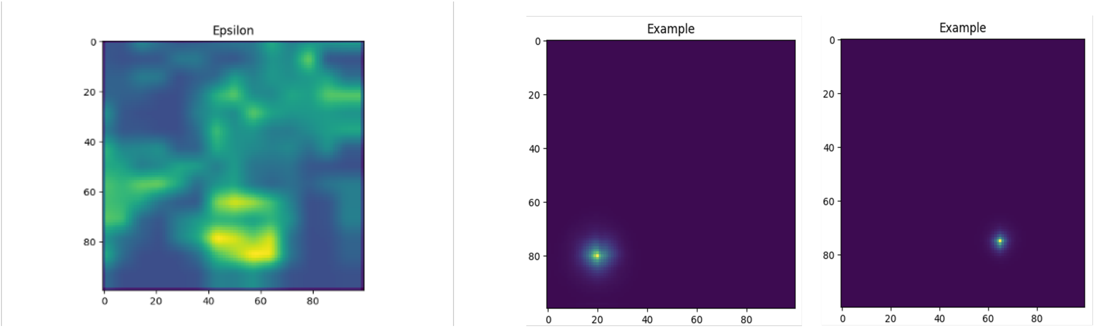
$$\mathbb{P}(Qu = y \mid u) \propto e^{\epsilon s(u,y)}$$

is ϵL-Lipschitz differentially private.

In the special case where $\mathcal{U} = \mathcal{Y} = \mathbb{R}^n$ and $s(u, y) = -\|u - y\|_p$, we recover the Laplace mechanism. Furthermore, Lipschitz privacy inherits the property of resiliency to postprocessing. Identically to differential privacy, any further, possibly randomized, postprocessing of the output carries the same privacy guarantees.

12.2.3 Example check

$$\mathbb{P}(\mathcal{Q}(u) = y) = g(u, y) = w(y)e^{-f_y(u)}$$



(a) Privacy preference map

(b) Reporting of two locations is based on a sampling from a learnt distribution with a relatively greater dispersion if the preferred privacy level is high (lower epsilon).

Figure 12.1: Illustration of Eikonal PDE-based private location reporting that ensures Lipschitz privacy (and equivalently a specific level of differential privacy).

has, by assumption, a proper probability density; $g(u, y) \geq 0$ and $\sum_{y \in \mathcal{Y}} g(u, y) = 1$. Moreover, we compute the following derivative in the weak sense

$$\|\nabla_u \ln \mathbb{P}(\mathcal{Q}(u) = y)\|_2 = \|\nabla_u (\ln w(y) - f_y(u))\| = \epsilon(u)$$

Therefore, mechanism $\mathcal{Q}$ satisfies the definition and, thus, is ϵ-locally Lipschitz private.

12.3 Regularized Eikonal Equation

The location-dependent privacy mechanism proposed in[287] assumes that the query $q : \Omega \to \mathcal{Y}$, and the privacy map $\epsilon : \Omega \to \mathbb{R}_+$, where $x \in \Omega$ represents a location.

For each possible response $y \in \mathcal{Y}$, let $\mathcal{S}_y$ be a subset of Ω, $\mathcal{S}_y = \{x \in \Omega, q(x) = y\}$. The Eikonal equation for $u(x; y)$ is given by

$$\|\nabla u(x)\|_2 = \epsilon(x), \qquad x \in \Omega, \tag{12.1}$$

subject to the internal Dirichlet condition,

$$u(x) = 0, \quad \text{for } x \in \mathcal{S}_y. \tag{12.2}$$

This means that for each response $y \in \mathcal{Y}$, an eikonal equation for $u(x; y)$ will be computed. For example, in the GPS location in the Philadelphia case, given an identity query, $q(x) = x$, if one discretizes the domain Ω by an $N \times N$ grid, then a total of N^2 solutions $u(x; y)$ needs to be computed since there are N^2 possible different responses in terms of the location y. And $u(x; y)$ represents the travel time from $y \in \Omega$ to $x \in \Omega$, where the velocity field is given by $1/\epsilon$.

It is stated in the paper that the internal condition is a design choice that stems from the need for the response y should be close to x, which originates from the particular location-based mechanism considered in the GPS location example with an identity query, $q(x) = x$ (see[287]). But it's unclear if this is a reasonable choice for a more generic query.

A normalization step is then carried out,

$$\sum_{y \in \mathcal{Y}} e^{-u(x;y)} w(y) = 1, \quad x \in \Omega. \tag{12.3}$$

If $w(y) \geq 0$, then this leads to a mechanism $\mathcal{Q}$,

$$\mathbb{P}(\mathcal{Q}(x) = y) = w(y) e^{-u(x;y)}. \tag{12.4}$$

A few issues in this framework: 1) the PDE requires smooth enough and large enough privacy ϵ at the edge of the map. 2) the normalization step does not always yield a positive solution for $w(y)$.

12.3.1 Extreme cases:

- If a trivial query $q(x) = y^*$ is used for any $x \in \Omega$, where y^* is the only response in $\mathcal{Y}, \mathcal{Y} = \{y^*\}$, then we have $u(x; y^*) = 0$ based on the the condition equation 12.2, which will lead to $w(y^*) = 1$ and cause information leakage $\mathbb{P}(\mathcal{Q}(x) = y^*) = 1$. This observation indicates that some diffusion at the response level is needed to generalize this mechanism to more generic queries.

- For a binary query with $\mathcal{Y} = \{y_0, y_1\}$, where $q(x) = y_1$ for $x \in \Omega_+$ and $q(x) = y_0$ for $x \in \Omega_-$, and $\Omega = \Omega_+ \cup \Omega_-$. According to the mechanism designed in[287], we have two PDE systems

$$\|\nabla u(x; y_0)\|_2 = \epsilon(x), \quad x \in \Omega_+, \qquad u(x; y_0)|_{x \in \Omega_-} = 0, \tag{12.5}$$

$$\|\nabla u(x; y_1)\|_2 = \epsilon(x), \quad x \in \Omega_-, \qquad u(x; y_1)|_{x \in \Omega_+} = 0. \tag{12.6}$$

The normalization step equation 12.3 yields

$$w(y_0)e^{-u(x;y_0)} + w(y_1)e^{-u(x;y_1)} = 1, \qquad x \in \Omega, \tag{12.7}$$

which is equivalent to

$$w(y_0)e^{-u(x;y_0)} + w(y_1) = 1, \qquad x \in \Omega_+, \tag{12.8}$$

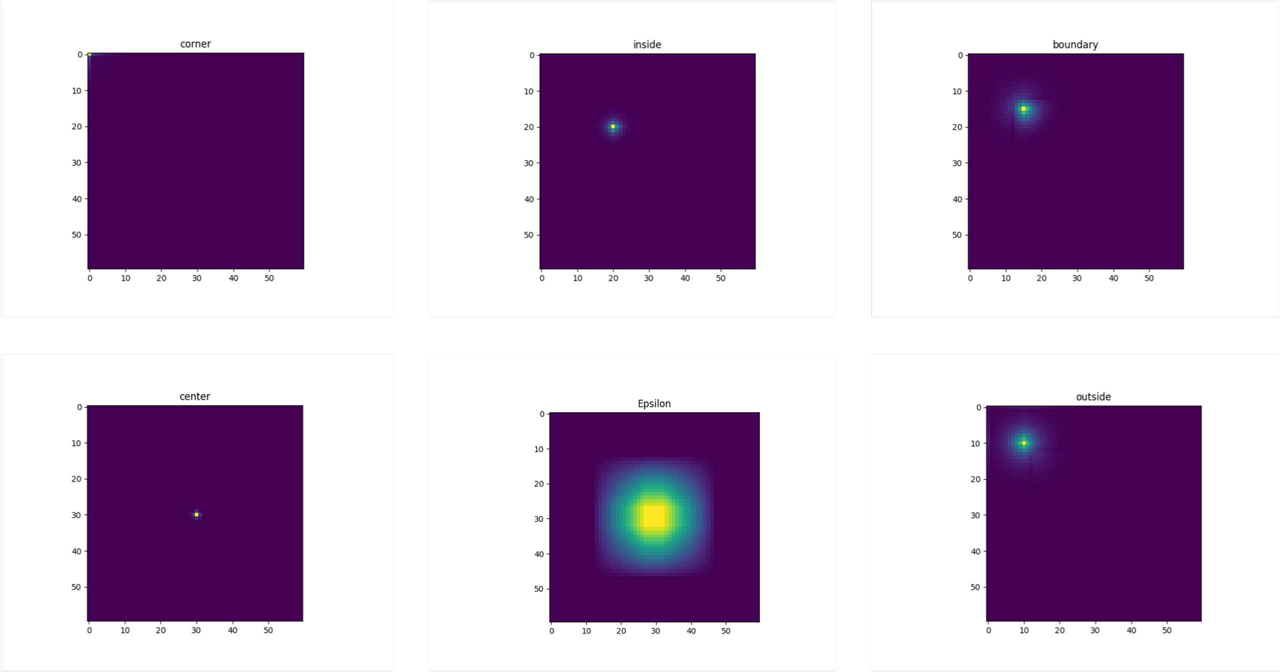


Figure 12.2: Some more instances of learnt distributions

$$w(y_0) + w(y_1)e^{-u(x;y_1)} = 1, \qquad x \in \Omega_-. \tag{12.9}$$

This is an over-determined system for the weights $w(y_0), w(y_1)$ unless we have a zero privacy map $\epsilon = 0$.

12.3.2 Viscosity-based regularization

Consider a regularized Eikonal equation instead. This is motivated by the recent work by Churbanov and Vabishchevich[118]. There is a slight abuse of notation here - (x, y) represents the coordinate in a 2-D domain. In a bounded domain $\Omega \subset \mathbb{R}^2$ with Lipschitz continuous boundary $\partial\Omega$, the boundary value problem for the eikonal equation for $u(x,y)$ is given by

$$a^2(x,y)\left[\left(\frac{\partial u}{\partial x}\right)^2+\left(\frac{\partial u}{\partial y}\right)^2\right]=1, \qquad \text{for } (x,y)\in\Omega, \tag{12.10}$$

$$u(x,y)=0 \quad \text{for } (x,y)\in\partial\Omega, \tag{12.11}$$

where $a(x,y)=1/\epsilon(x,y)$ in our context. Introducing the transformation

$$v_\alpha(x,y)=\exp\left(-\frac{u_\alpha(x,y)}{\alpha}\right), \qquad \text{where } \alpha>0, \tag{12.12}$$

one can show that

$$\alpha^2\mathcal{L}(v_\alpha)=v_\alpha\left(-\alpha\mathcal{L}(u_\alpha)+\mathcal{E}(u_\alpha)\right), \tag{12.13}$$

where the operators

$$\mathcal{L}(u)=\frac{\partial}{\partial x}\left(a^2\frac{\partial u}{\partial x}\right)+\frac{\partial}{\partial y}\left(a^2\frac{\partial u}{\partial y}\right), \qquad \mathcal{E}(u)=a^2(x,y)\left[\left(\frac{\partial u}{\partial x}\right)^2+\left(\frac{\partial u}{\partial y}\right)^2\right]. \tag{12.14}$$

Therefore, if u_α satisfies the regularized Eikonal equation

$$\mathcal{E}(u_\alpha)=1+\alpha\mathcal{L}(u_\alpha), \qquad (x,y)\in\Omega, \tag{12.15}$$

subject to the boundary condition

$$u_\alpha=0 \qquad (x,y)\in\partial\Omega \tag{12.16}$$

then based on the relation in equation 12.13, the nonlinear problem equation 12.19 can be solved by simply considering the linear PDE

$$\alpha^2 \mathcal{L}(v_\alpha) - v_\alpha = 0, \qquad (x, y) \in \Omega \tag{12.17}$$

subject to the boundary condition

$$v_\alpha = 1 \qquad (x, y) \in \partial\Omega. \tag{12.18}$$

The regularized solution $u_\alpha(x, y) \to u(x, y)$ as $\alpha \to 0$ (vanishing viscosity approximation). By varying α, we may explore the trade-off between added diffusion and the robustness of the algorithm, MSE metrics, etc.

12.3.3 Privacy guarantees for viscosity regularized solutions

The solution to equation 12.23, subject to the corresponding boundary condition, ultimately results in a solution to

$$\mathcal{E}(u_\alpha) = 1 + \alpha \mathcal{L}(u_\alpha), \qquad (x, y) \in \Omega, \tag{12.19}$$

subject to the boundary condition

$$u_\alpha = 0 \qquad (x, y) \in \partial\Omega \tag{12.20}$$

This is an Eikonal PDE which results in a

$$1 + \alpha\mathcal{L}(u_\alpha) - \text{Lipschitz privacy}$$

Given a max-separation condition of $\sum_{i=1}^{n} \|(x,y)_i - (x,y)'_i\|_2 \leq \lambda$, the Lipschitz privacy level is equivalent to

$$\lambda[1 + \alpha\mathcal{L}(u_\alpha)] - \text{differential privacy}$$

12.4 FEM FORMULATION

The following standard problems can be posed within a FEM formulation that utilizes a variational formulation along with Green's theorem, followed by the construction of a linear system in terms of a chosen basis. These problems include the

1. **Neumann problem** given by

$$\begin{aligned} -\Delta u &= f, \qquad \text{in } \Omega \\ n \cdot \nabla u &= g_N, \qquad\qquad \text{on } \partial\Omega \end{aligned}$$

   where f and g_N are given functions.

2. **Dirichlet problem** that considers the following model problem with inhomoge-

neous boundary conditions: Find u such that

$$\begin{aligned} -\Delta u &= f, \quad \text{in } \Omega \\ u &= g_D, \quad \text{on } \partial\Omega \end{aligned}$$

where f and g_D are given functions.

3. **Poisson equation problem** find u such that

$$\begin{aligned} -\Delta u &= f, \text{ in } \Omega \\ u &= 0, \text{ on } \partial\Omega \end{aligned}$$

where $\Delta = \partial^2/\partial x_1^2 + \partial^2/\partial x_2^2$ is the Laplace operator, and f is a given function in, say, $L^2(\Omega)$

4. **Eigenfunction problem** given by

$$-\Delta u = \lambda u, \qquad \text{in } \Omega \tag{12.21}$$

$$n \cdot \nabla u = 0, \qquad \text{on } \partial\Omega \tag{12.22}$$

12.5 Viscosity regularized solution

The viscosity regularized PDE

$$\alpha^2 \mathcal{L}(v_\alpha) - v_\alpha = 0, \qquad (x, y) \in \Omega \tag{12.23}$$

subject to the boundary condition seems to be a combination of the poisson and eigenfunction problems above. An open problem that remains is to further analyze the solution's performance under the data's privacy constraints at the interface between Ω_+ and Ω_-.

"There are mathematicians who do mathematics simply because they like it — it's a work of art. Some people work very hard on problems that have no relation to anything else."

Lucien Le Cam

13

Private matrix inverses

13.1 Introduction

We describe a method for resource-efficient computation in distributed linear algebra via differentially private shares of data called splinters, which are linear combinations of the sensitive input matrix with several random matrices. This enables resource-constrained client devices to receive the inverse of the matrix as a service from the

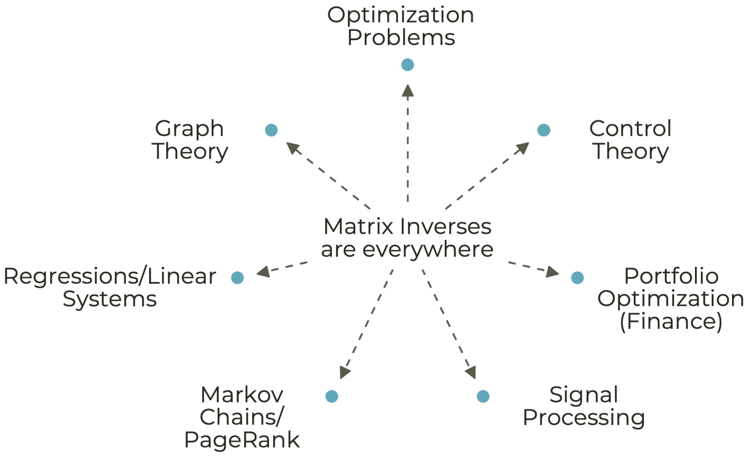

Optimization Problems
Graph Theory
Control Theory
Matrix Inverses are everywhere
Regressions/Linear Systems
Portfolio Optimization (Finance)
Markov Chains/ PageRank
Signal Processing

Figure 13.1: Matrix inverses are used quite universally in scientific computing

server without having to completely perform the needed inversion on-device. The server performs computations over these splinters, and the corresponding results are sent back to the client. The client has a required recipe of specific operations to perform over these intermediate results, referred to as unsplintering, in order to obtain the required final result. The scheme integrates well with any state-of-the-art non-distributed matrix inversion scheme of choice, such as[19], that the server could use in the context of splintering.

13.2 Related work

1. **Coded Computing** was introduced based on coding theory to perform distributed computations with benefits such as robustness to straggler clients, byzantine robustness to malicious clients, and information-theoretic privacy. Coded computing schemes for distributed matrix multiplication, in particular, include[592,228,591] for straggler mitigation,[226] for robustness from malicious actors that return cor-

rupted results,[590] for resiliency from stragglers, byzantine robustness from malicious actors and information theoretic privacy at the same time. Some of the coded computing methods are surveyed in[318]. Coded computing based on MDS codes for matrix inverse is provided in[97], and a scheme based on GASP codes is provided in[163]. While some of the above works provide privacy guarantees from an information-theoretic notion, our work focuses on differential privacy guarantees for resource-efficient distributed matrix inverse. Another significant difference is that while the above methods used encodings based on coding theory, our shares are just noisy (and private) linear combinations, even though the final operation we perform, such as the matrix inverse, is a non-linear operation.

2. **Cryptographically secure and differentially private methods** Homomorphic encryption-based protocols for delegating several linear algebra computations were presented in[370] along with secure verification guarantees on the obtained solution in $\mathcal{O}(n^2 \log n)$ time. Differentially private mechanisms for some linear algebra computations in the streaming setting with space complexity guarantees were given in[523]. In contrast, mechanisms for differentially private matrix and tensor factorizations were given in[241]. Our work instead focuses on the standard (non-streaming) setting for differentially private matrix inverse in the distributed setting.

13.3 System interactions

We now detail the first-order idea of splintering in the non-private setting. For a d dimensional input query matrix $\mathbf{X}$, the client device creates d shares corresponding to

$\mathbf{X}$ as $\{\mathbf{Z_1}, \mathbf{Z_2} \ldots \mathbf{Z_d}\}$ so that

$$\mathbf{X} = Splint(\mathbf{Z_1}, \mathbf{Z_2} \ldots \mathbf{Z_d}), \forall i \in 1..d$$

The most basic splint function that allows for such a representation is a linear combination using coefficients α_i as

$$\mathbf{X} = \sum_{i}^{d} \alpha_i \mathbf{Z_i}, \forall i \in 1..d$$

The $\alpha_i's$ are not shared with any other entity, be it another client or a server. The splinters $\mathbf{z_i}$ are shared with the server. The server performs a set of application-dependent operations on the splinters $\mathbf{Z}_i, \forall i \in 1 \ldots d$ and sends results $\{\beta_i\}$ back to the client on either all or a subset of the d shares. The client performs a local computation called $UnSplint$ using original shares $\mathbf{Z_i}$; its corresponding α_i's that are known only to the client and received $\beta_i's$ obtained from the server. This unsplintering operation reveals the true result l of the intended application to the client.

$$l = UnSplint(\alpha_i, \mathbf{Z_i}, \beta_i), \forall i \in 1..d$$

Note that although $\mathbf{x}$ is represented via a linear combination, the computation of $\{\beta_i\}$ and $UnSplint$ is not necessarily linear.

Efficient setting Note that there is a computational benefit for the client only if the $UnSplint$ operation and the generation of splinters can be performed more efficiently by the client in comparison to entirely performing the required service (say, matrix inverse) on its own premise.

Private setting Differentially privacy is quite a popular mathematical notion of privacy[160] for releasing outputs of queries. The splintering mechanism in its entirety is differentially private if it is ensured that any communication made from the client in terms of the shares is differentially private with respect to the matrix that needs to be inverted as well as with respect to the client's sensitive coefficients that it holds. These coefficients are useful for the client to perform the $unSplint$ operation. It is worth noting that without the privacy constraint, the client can trivially send over all of its matrices to the server to completely save itself from performing any computations. Therefore, it is worth noting that splintering is useful when operated in a setting that is both efficient and private. That said, a privacy-compute-communication trade-off comes into play in practice to efficiently perform splintering with differentially private guarantees.

13.4 Examples under various settings

We now provide some examples of splintering and unsplintering operations under various settings of a.) non-private and non-efficient, b.) non-private and efficient, and c.) private and efficient.

13.4.1 Splintering for sigmoid (non-private and non-efficient setting)

Theorem 13.4.1. For a scalar input $x \in \mathbb{R}$ expressed using scalar real-valued splinters $\{z_1, z_2, \ldots, z_k\}$ as $x = \sum_{i=1}^{k} \alpha_i z_i$, the unsplintering operation to compute the sigmoid function $s(x)$ using $s(z_1), s(z_2), \ldots, s(z_k)$ is given by

$$s(x) = \frac{1}{1 + \prod_{i=1}^{k} \left(\frac{1-s(z_i)}{s(z_i)}\right)^{-\alpha_i}}$$

Proof. The sigmoid function is given by

$$s(x) = \frac{1}{1 + e^{-x}} = \frac{e^x}{e^x + 1}$$

This can be rearranged as

$$\frac{1 - s(x)}{s(x)} = e^{-x}$$

By following our proposed approach of using splinters we substitute $x = \sum_{i=1}^{k} \alpha_i z_i$ in the above form of sigmoid to get $s(x) = \frac{1}{1+e^{-\sum_{i=1}^{k} \alpha_i z_i}} = \frac{1}{1+\prod_{i=1}^{k} e^{-\alpha_i z_i}}$

Now, substituting this into the rearranged form of the sigmoid above, we get $s(x) = \frac{1}{1+\prod_{i=1}^{k}\left(\frac{1-s(z_i)}{s(z_i)}\right)^{-\alpha_i}}$ Therefore, the final scheme only requires computing $s(z_i)'s$ at the server while the client can figure out $s(x)$ by using α_i's that are known only to the client and not shared with the server.

□

As the setting is non-private, the success of several attacks cannot be ruled out in this setting. We now provide one other example in this setting with regard to computing the softmax function.

13.4.2 Splintering for softmax (non-private and non-efficient setting)

Theorem 13.4.2. For a real-valued input vector $\mathbf{x}$ expressed using splinters $\{\mathbf{z_1}, \mathbf{z_2}, \ldots, \mathbf{z_k}\}$ as $\mathbf{x} = \sum_{i=1}^{k} \alpha_i \mathbf{z_i}$, the unsplintering operation to compute $s(\mathbf{x})$ where $e^{\mathbf{z}_1} = a_1, e^{\mathbf{z}_1} =$

a_2 and $e^{\mathbf{z}_3} = a_3$ and $w_1 = \frac{a_1}{a_1+a_2+a_3}$, $w_2 = \frac{a_2}{a_1+a_2+a_3}$ and $w_3 = \frac{a_3}{a_1+a_2+a_3}$ as follows

$$\text{softmax}(\alpha_1 \mathbf{z}_1 + \alpha_2 \mathbf{z}_2)_i = \frac{\left(e^{softmax^{-1}(\vec{w_1};\mathbf{z}_{11})_i}\right)^{\alpha_1} \left(e^{softmax^{-1}(\vec{w_2};\mathbf{z}_{11})_i}\right)^{\alpha_2}}{\sum_{j=1}^{d} \left(e^{softmax^{-1}(\vec{w_1};\mathbf{z}_{11})_j}\right)^{\alpha_1} \left(e^{softmax^{-1}(\vec{w_2};\mathbf{z}_{11})_j}\right)^{\alpha_2}} \tag{13.1}$$

Proof.

$$\begin{aligned} \text{softmax}(\alpha_1 \mathbf{z}_1 + \alpha_2 \mathbf{z}_2)_i &= \frac{e^{\alpha_1 \mathbf{z}_{1i} + \alpha_2 \mathbf{z}_{2i}}}{\sum_{j=1}^{d} e^{\alpha_1 \mathbf{z}_{1j} + \alpha_2 \mathbf{z}_{2j}}} && (13.2) \\ &= \frac{\left(e^{\alpha_1 \mathbf{z}_{1i}}\right)\left(e^{\alpha_2 \mathbf{z}_{2i}}\right)}{\sum_{j=1}^{d} \left(e^{\alpha_1 \mathbf{z}_{1j}}\right)\left(e^{\alpha_2 \mathbf{z}_{2j}}\right)} && (13.3) \end{aligned}$$

Assume one component each of $\vec{\mathbf{z}_1}$,$\vec{\mathbf{z}_2}$ are known and let $\vec{w_i} = \text{softmax}(\vec{\mathbf{z}_1})$. Then $\mathbf{z}_i = \text{softmax}^{-1}(w_i; \mathbf{z}_{11})$. Therefore we plugin this inverse in 13.2 to rewrite softmax as

$$\text{softmax}(\alpha_1 \mathbf{z}_1 + \alpha_2 \mathbf{z}_2)_i = \frac{\left(e^{softmax^{-1}(\vec{w_1};\mathbf{z}_{11})_i}\right)^{\alpha_1} \left(e^{softmax^{-1}(\vec{w_2};\mathbf{z}_{11})_i}\right)^{\alpha_2}}{\sum_{j=1}^{d} \left(e^{softmax^{-1}(\vec{w_1};\mathbf{z}_{11})_j}\right)^{\alpha_1} \left(e^{softmax^{-1}(\vec{w_2};\mathbf{z}_{11})_j}\right)^{\alpha_2}} \tag{13.4}$$

Analytical inverse of softmax when one component is known: Let $e^{\mathbf{z}_1} = a_1, e^{\mathbf{z}_2} = a_2$ and $e^{\mathbf{z}_3} = a_3$. Then $w_1 = \frac{a_1}{a_1+a_2+a_3}$, $w_2 = \frac{a_2}{a_1+a_2+a_3}$ and $w_3 = \frac{a_3}{a_1+a_2+a_3}$ and therefore

$$\frac{w_1}{e^{\mathbf{z}_1}} = \frac{w_2}{e^{\mathbf{z}_2}} = \frac{w_3}{e^{\mathbf{z}_3}} = \lambda$$

This implies that

$$\mathbf{z}_2 = log\left(\frac{w_2}{w_1}\right) e^{\mathbf{z}_1}$$

□

13.5 Splintering for matrix inverse (non-private and efficient setting)

We now propose a splintering scheme for the important operation of matrix inversion in this section, but in the non-private yet efficient setting. We consider a setting where the client has a large sensitive matrix $\mathrm{M}_{n\times n}$ and would like to use the service of a computationally powerful server to privately obtain the inverse M^{-1}.

Theorem 13.5.1. The unsplintering operation to compute the matrix inverse M using two splinters $\mathbf{Z_1}, \mathbf{Z_2}$ with secret coefficients α_1, α_2 while only having to compute the inverse of the splinters is given by

$$(\alpha_1\mathbf{Z_1}{+}\mathbf{UZ_2V})^{-1} = \mathbf{1/\alpha_1 Z_1}^{\mathbf{-1}} - \mathbf{1/\alpha_1 Z_1}^{\mathbf{-1}}\mathrm{U}(\mathbf{Z_2}^{\mathbf{-1}} + \mathbf{V1/\alpha_1 Z_1}^{\mathbf{-1}}\mathrm{U})^{\mathbf{-1}}\mathbf{1/\alpha_1 VZ_1}^{\mathbf{-1}}$$

where $\mathbf{UZ_2V}$ is of the form $\underbrace{\begin{bmatrix}\alpha_3 & & \\ & \alpha_3 & \\ & & \alpha_3\end{bmatrix}}_{\mathbf{U}}\mathbf{Z_2}\underbrace{\begin{bmatrix}\alpha_4 & & \\ & \alpha_4 & \\ & & \alpha_4\end{bmatrix}}_{\mathbf{V}}$

.

Proof. In order to obtain the inverse of a private matrix $\mathrm{M}_{n\times n}$, we split it into the form using $\mathbf{A}_{n\times n}, \mathbf{U}_{n\times p}, \mathbf{V}_{p\times n}$ and $\mathbf{Z_2}$ of dimension $p \times p$ as

$$\mathbf{M^{-1}} = (\mathbf{A} + \mathbf{UZ_2V})^{-1}$$

where $\mathbf{A}$ is written in terms of a splinter matrix $\mathbf{Z}_1$ of dimension $n\times n$ as $A = \alpha_1\mathbf{Z_1}$ and

$\mathbf{UZ_2V}$ is of the form $\underbrace{\begin{bmatrix} \alpha_3 & & \\ & \alpha_3 & \\ & & \alpha_3 \end{bmatrix}}_{\mathbf{U}} \mathbf{Z_2} \underbrace{\begin{bmatrix} \alpha_4 & & \\ & \alpha_4 & \\ & & \alpha_4 \end{bmatrix}}_{\mathbf{V}}$ where $\mathbf{Z}_2$ is the other splinter.

Now, by the popular matrix inversion lemma (ShermanMorrisonWoodbury formula)

$$(\mathbf{A} + \mathbf{UZ_2V})^{-1} = \mathbf{A^{-1}} - \mathbf{A^{-1}U}(\mathbf{Z_2}^{-1} + \mathbf{VA^{-1}U})^{-\mathbf{1}}\mathbf{VA^{-1}}$$

Therefore the proposed scheme now is to send $\mathbf{Z_1}, \mathbf{Z_2}$ to the server which sends back $\mathbf{Z_1^{-1}}, \mathbf{Z_2^{-1}}$ to the client that holds $\mathbf{M}$ along with $\alpha_1, \alpha_2, \alpha_3$. The client then obtains the final solution $\mathbf{M}^{-1}$ by computing

$$(\alpha_1\mathbf{Z_1}{+}\mathbf{UZ_2V})^{-1} = \mathbf{1}/\alpha_\mathbf{1}\mathbf{Z_1}^{-\mathbf{1}} - \mathbf{1}/\alpha_\mathbf{1}\mathbf{Z_1}^{-\mathbf{1}}\mathbf{U}(\mathbf{Z_2}^{-1} + \mathbf{V1}/\alpha_\mathbf{1}\mathbf{Z_1}^{-\mathbf{1}}\mathbf{U})^{-\mathbf{1}}\mathbf{1}/\alpha_\mathbf{1}\mathbf{VZ_1}^{-\mathbf{1}}$$

Computational savings: In cases where $n >> p$, the matrix $(\mathbf{Z_2}^{-1} + \mathbf{VA^{-1}U})^{-\mathbf{1}}$ which is of dimension $k \times k$ is much easier to invert than the original private data matrix $\mathbf{M}_{n\times n}$ thereby offloading the heavier computation onto the server while preserving privacy and requiring a much smaller computation on the client. □

This can further be generalized to 3 splinters as follows where

$$\mathbf{U} = \begin{bmatrix} \mathbf{U_1} & \mathbf{U_2} \end{bmatrix}, \mathbf{V} = \begin{bmatrix} \mathbf{V_1} & \mathbf{V_2} \end{bmatrix}, \mathbf{Z} = \begin{bmatrix} \mathbf{Z_1} & \mathbf{0} \\ \mathbf{0} & \mathbf{Z_2} \end{bmatrix}$$

so that

$$\mathbf{A} + \mathbf{U_1Z_1V_1^T} + \mathbf{U_2Z_2V_2^T} = \mathbf{A} + \mathbf{UZV^T}$$

Then, just apply the Woodbury matrix identity as above to complete the proof. It is well-known in linear algebra that if one can invert a nonsingular n matrix in $T(n)$ time, then one can multiply $n \times n$ matrices in $O(T(3n))$ time. To see this, let A and B be matrices and consider the following $3n \times 3n$ matrix:

$$\mathbf{D} = \begin{bmatrix} \mathbf{I} & \mathbf{A} & \mathbf{0} \\ \mathbf{0} & \mathbf{I} & \mathbf{B} \\ \mathbf{0} & \mathbf{0} & \mathbf{I} \end{bmatrix}$$

1 where $\mathbf{I}$ is the n-by- n identity matrix. One can verify by direct calculation that

$$\mathbf{D}^{-1} = \begin{bmatrix} \mathbf{I} & -\mathbf{A} & \mathbf{AB} \\ \mathbf{0} & \mathbf{I} & -\mathbf{B} \\ \mathbf{0} & \mathbf{0} & \mathbf{I} \end{bmatrix}$$

Inverting $\mathbf{D}$ takes $O(T(3n))$ time and we can find $\mathbf{AB}$ by inverting $\mathbf{D}$. Note that $\mathbf{D}$ is always invertible since its determinant is 1. Therefore, a splintering method for one operation benefits the other.

13.6 Splintering for matrix inverse with differential privacy (private and efficient setting)

In this section, we share a mechanism for matrix inversion in a private and efficient setting. Before that, we introduce some relevant preliminaries and terminology required in our use case.

13.6.1 Main idea for privatization

The main idea for privatization is that mixing up sensitive datasets with a linear combination followed by a sufficient level of additive noise leads to a mixed dataset that is differentially private. Moreover, the variance with which $\mathbf{N}$ is generated to obtain ϵ-differential privacy happens to reduce with an increase in the number of mixture components (which in our case are splinters). The noise can be calibrated using[Borgnia et al.] or[307].

13.6.2 Level of noise after linearly combining the splinters

The following is the relation between the privacy level ϵ that is maintained and required noise variance to generate $\mathbf{N}$ along with our own annotation made in parentheses to draw an exact analogy for our use-case.

Theorem 13.6.1. [307] (Privacy guarantee): Fix the mixture degree ℓ (the number of splinters in our case), the noise level σ_X and the number of mixtures T (these are the number of samples, which is 1 in our case). For any $\delta > 0$, $\mathrm{DPMix}(\ell)$ is (ε, δ)-DP such that

$$\varepsilon = \min_{\alpha \in \{2,3,\ldots\}} T\varepsilon'_\alpha + \frac{\log(1/\delta)}{\alpha - 1}$$

where

$$\varepsilon'_\alpha = \frac{1}{\alpha-1}\log\left(1+\left(\frac{\ell}{n}\right)^2\binom{\alpha}{2}\min\left(4\left(e^{\frac{\Delta^2}{\ell^2}}-1\right),2e^{\frac{\Delta^2}{\ell^2}}\right)+4G(\alpha)\right)$$

$$G(\alpha) := \sum_{j=3}^{\alpha}\left(\frac{\ell}{n}\right)^j\binom{\alpha}{j}\sqrt{B(2\lfloor j/2\rfloor)\cdot B(2\lceil j/2\rceil)}$$

$$B(\ell) := \sum_{i=0}^{\ell}(-1)^i\binom{\ell}{i}e^{\frac{i(i-1)}{2\ell^2}\Delta^2}, \Delta^2 := \left(\frac{d_X}{\sigma_X^2}\right)$$

A less-tighter privacy-noising relation is given using Laplace distribution-based noise as given below based on[Borgnia et al.].

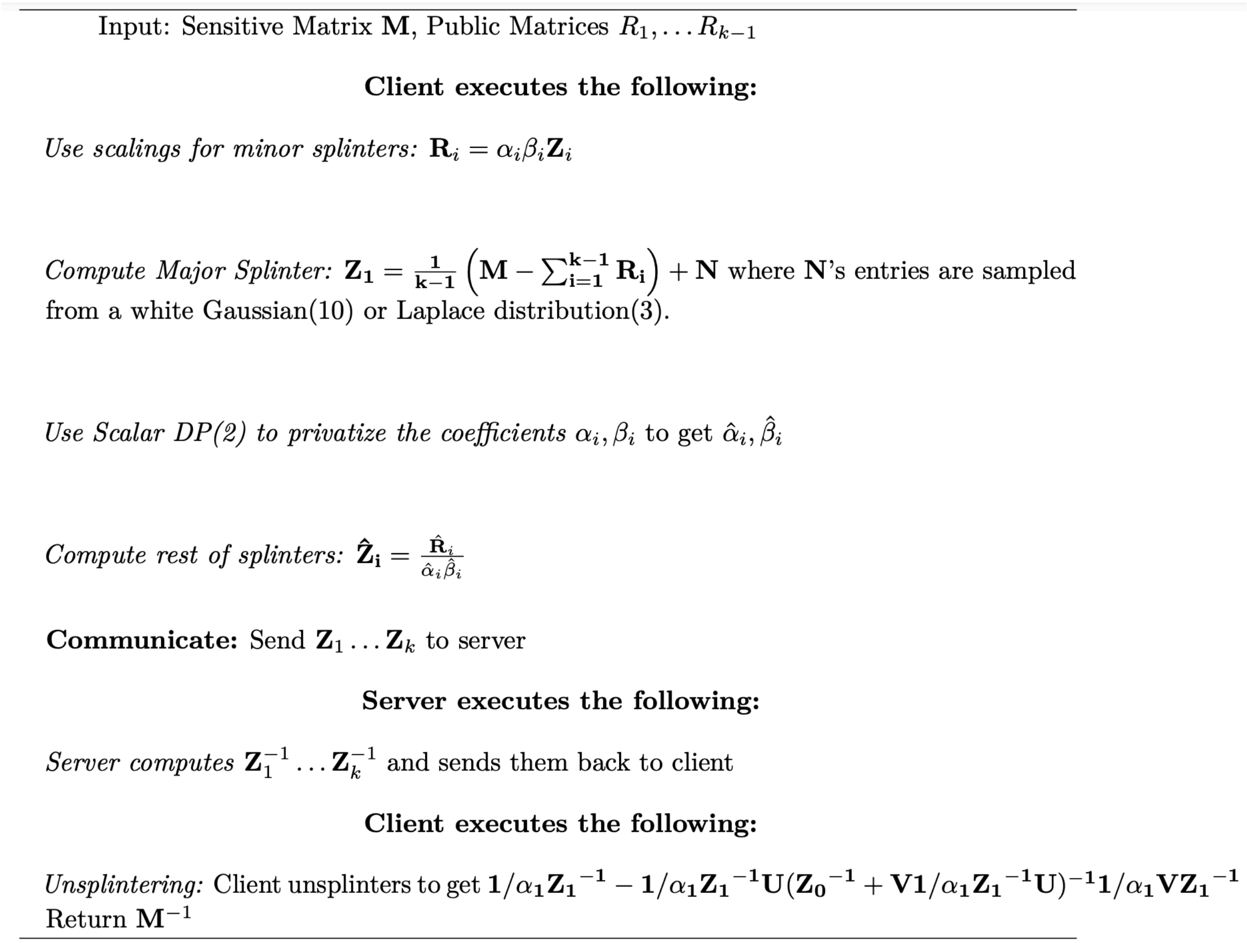
$$\epsilon = T\max\{A,B\} \leq \frac{T}{k\sigma}$$

where

$$A = \log\left(1-\frac{k}{n}+e^{\frac{1}{k\sigma}}\frac{k}{n}\right), B = \log\frac{n}{n-k+ke^{-\frac{1}{k\sigma}}}$$

13.7 Generation of Splinters

We now walk through the details of generating splinters in our proposed mechanism. As $k-1$ out of k splinters used are data-independent and sampled from a different chosen distribution each. The client first generates $k-1$ minor splinters (data-independent samples) as $\mathbf{R_i} \sim \mathcal{N}(0,\mathbf{\Sigma_i}), \forall i \in \{1\ldots d-1\}$. Only one splinter is dependent on data



Input: Sensitive Matrix $\mathbf{M}$, Public Matrices $R_1, \dots R_{k-1}$

Client executes the following:

Use scalings for minor splinters: $\mathbf{R}_i = \alpha_i \beta_i \mathbf{Z}_i$

Compute Major Splinter: $\mathbf{Z_1} = \frac{\mathbf{1}}{\mathbf{k-1}} \left(\mathbf{M} - \sum_{\mathbf{i=1}}^{\mathbf{k-1}} \mathbf{R_i} \right) + \mathbf{N}$ where $\mathbf{N}$'s entries are sampled from a white Gaussian(10) or Laplace distribution(3).

Use Scalar DP(2) to privatize the coefficients α_i, β_i to get $\hat{\alpha}_i, \hat{\beta}_i$

Compute rest of splinters: $\hat{\mathbf{Z}}_\mathbf{i} = \frac{\hat{\mathbf{R}}_i}{\hat{\alpha}_i \hat{\beta}_i}$

Communicate: Send $\mathbf{Z}_1 \dots \mathbf{Z}_k$ to server

Server executes the following:

Server computes $\mathbf{Z}_1^{-1} \dots \mathbf{Z}_k^{-1}$ and sends them back to client

Client executes the following:

Unsplintering: Client unsplinters to get $\mathbf{1}/\alpha_\mathbf{1}\mathbf{Z_1}^{-\mathbf{1}} - \mathbf{1}/\alpha_\mathbf{1}\mathbf{Z_1}^{-\mathbf{1}}\mathbf{U}(\mathbf{Z_0}^{-\mathbf{1}} + \mathbf{V1}/\alpha_\mathbf{1}\mathbf{Z_1}^{-\mathbf{1}}\mathbf{U})^{-\mathbf{1}}\mathbf{1}/\alpha_\mathbf{1}\mathbf{V}\mathbf{Z_1}^{-\mathbf{1}}$
Return $\mathbf{M}^{-1}$

Figure 13.2: Proposed splintering scheme with differential privacy

```
Privatize the magnitude with absolute error: ScalarDP Require: Magnitude $r$, privacy parameter $\varepsilon > 0, k \in \mathbb{N}$, bound $r_{\max}$ $r \leftarrow \min\{r, r_{\max}\}$
    Start by Sampling $J \in \{0, 1, \cdots, k\}$
such that
    $J = \begin{cases} \lfloor kr/r_{\max} \rfloor & \text{w.p. } (\lceil kr/r_{\max} \rceil - kr/r_{\max}) \\ \lceil kr/r_{\max} \rceil & \text{otherwise.} \end{cases}$
Use randomized response to obtain
    $\widehat{J} \mid (J = i) = \begin{cases} i & \text{w.p. } \frac{e^{\varepsilon}}{e^{\varepsilon}+k} \\ \text{uniform in } \{0, \ldots, k\} \backslash i & \text{w.p. } \frac{k}{e^{\varepsilon}+k} \end{cases}$
Debias $\widehat{J}$, by setting
    $Z = a(\widehat{J} - b) \text{ for } a = \left(\frac{e^{\varepsilon}+k}{e^{\varepsilon}-1}\right)\frac{r_{\max}}{k} \text{ and } b = \frac{k(k+1)}{2\,(e^{\varepsilon}+k)}$
Return $Z$
```

Figure 13.3: Private scalar release mechanism that we utilize from[56]

X. It is generated as

$$\mathbf{Z_1} = \frac{1}{\alpha_1}(\mathbf{X} - \sum_{i \neq 1} \mathbf{R_i})$$

where α_1 is the corresponding privatized version of coefficients for the data-dependent splinter. The rest of the coefficients are secret, only known to the client, and never shared with the server.

Rescaling step Once the data-dependent coefficient and splinter have been generated, the rest of the data-independent splinters are scaled by their corresponding privatized secret coefficients as $\mathbf{Z_i} = \frac{1}{\hat{\alpha}_i}\mathbf{Z_i}$. The secret coefficients are privatized using the scalar DP mechanism of[56], which is an optimal mechanism. The optimality properties of scalar DP[56] are given below.

Note: The notation below that is internal to the algorithm for scalar DP should not be overloaded with any of the same symbols used in the rest of the paper for a different context.

Theorem 13.7.1. [56] Let $\varepsilon > 0, k \in \mathbb{N}$, and $0 \leq r_{\max} < \infty$. Then the mechanism ScalarDP $(\cdot, \varepsilon; k, r_{\max})$ is ε-differentially private and for $Z = ScalarDP\left(r, \varepsilon; k, r_{\max}\right)$, if $0 \leq r \leq r_{\max}$, then $\mathbb{E}[Z] = r$ and

$$\mathbb{E}\left[(Z-r)^2\right] \leq \frac{k+1}{e^{\varepsilon}-1}\left[r^2 + \frac{r_{\max}^2}{4k^2} + \frac{(2k+1)\left(e^{\varepsilon}+k\right) r_{\max}^2}{6k\left(e^{\varepsilon}-1\right)}\right] + \frac{r_{\max}^2}{4k^2}$$

By choosing k appropriately, we immediately see that we can achieve optimal mean-squared error as ε grows.

Theorem 13.7.2. [56] Let $k = \left\lceil e^{\varepsilon/3} \right\rceil$. Then for $Z = \mathrm{ScalarDP}\left(r, \varepsilon; k, r_{\max}\right)$,

$$\sup_{r \in [0, r_{\max}]} \mathbb{E}\left[(Z-r)^2 \mid r\right] \leq C \cdot r_{\max}^2 e^{-2\varepsilon/3}$$

for a universal (numerical) constant C independent of $r_{\max}$ and ε.

All the secret coefficients α_i have are chosen from a p-bit base-2 floating point system allowed by the computer architecture, where $p \in \{16, 32, 64\}$. Therefore, to reconstruct a data matrix $\mathbf{X}$ from scaled $\mathbf{Z_i}$'s, one would need access to the secret coefficients. Every communication from the client to the server in this scheme involves a different set of splinters sub-sampled from the union of decoy and minor splinters. That said, our scheme is a one-shot scheme per matrix inversion, thereby $T = 1$. Moreover, we assume that our matrices have a ℓ_1 norm that is ≤ 1. Note that for a fixed noise level, the privacy guarantee increases with an increase in ℓ (number of splinters). Therefore, if one chooses a larger value of ℓ, then a smaller amount of noise is sufficient to achieve the target privacy level.

13.8 Distributed and private splintering for matrix inverses without differential privacy

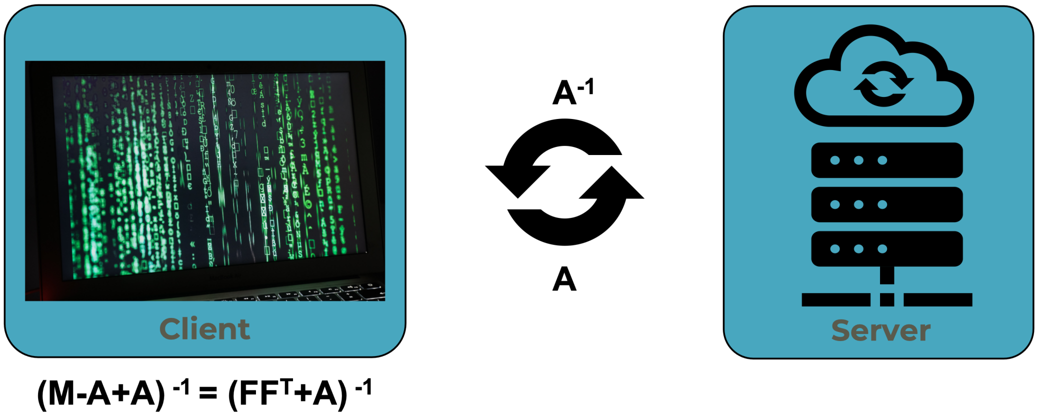


Figure 13.4: Schematic of systems interactions

Splitting a matrix $\mathbf{M}$ as follows $(\mathbf{M}-\mathbf{A}+\mathbf{A})^{-1} = \left(\mathbf{F}\mathbf{F}^\top + \mathbf{A}\right)^{-1}$ where a low-rank approximation $\mathbf{F}$ of $\mathbf{M} - \mathbf{A}$ is obtained efficiently on the client using Random Pivoted Cholesky[108] while the server computes and sends back the inverse of $\left(\mathbf{A} + \mathbf{F}\mathbf{F}^T\right)^{-1} = \mathbf{A}^{-1} - \mathbf{A}^{-1}\mathbf{F}\left(\mathbf{I} + \mathbf{F}^T\mathbf{A}^{-1}\mathbf{F}\right)^{-1}\mathbf{F}^T\mathbf{A}^{-1}$ to the client would allow the client to instead invert a smaller matrix of dimension given by the rank of $\mathbf{F}$ in order to obtain the approximate matrix inverse of $\mathbf{M}$. Over here, the communication $\mathbf{A}$ is completely sensitive data-independent, thereby ensuring privacy. That said, we empirically observe that the utility is good when the entries in data independent matrix $\mathbf{A}$ are independently sampled from $\sim Unif[-\lambda, 0]$ where λ is small and $\mathbf{M}$ is positive definite with but as less diagonally dominant as practically possible. This is illustrated in Figures 13.4 and

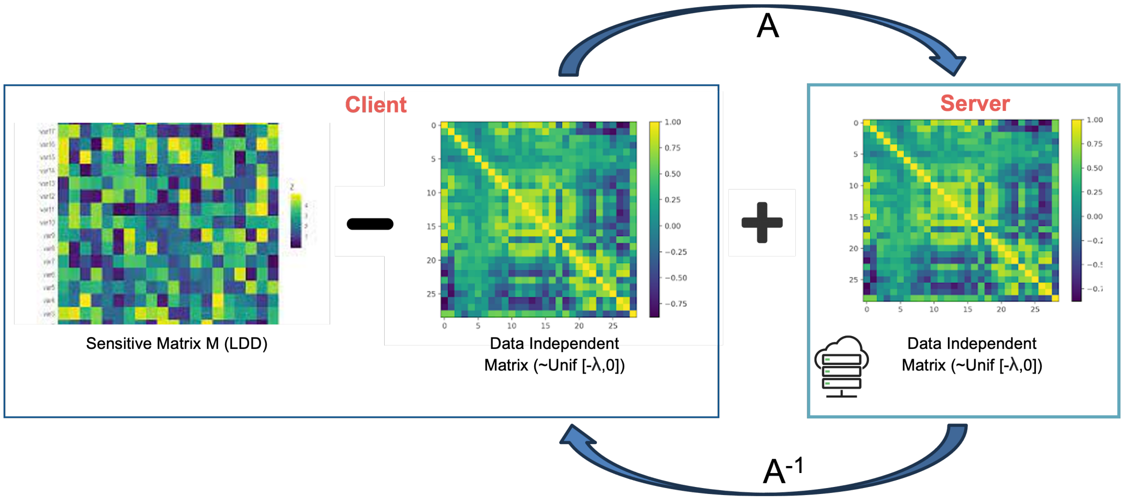

A
Client
Server
Sensitive Matrix M (LDD)
Data Independent Matrix (~Unif [-λ,0])
Data Independent Matrix (~Unif [-λ,0])
A-1

Figure 13.5: Schematic illustration 2

13.5.

If there is a problem you can't solve, then there is an easier problem you can solve: find it.

George Pólya

14

Parallel maxi-min combinatorial optimization of distance covariance

14.1 Introduction

This chapter is based on our work in[540]. The rich structure of some set function classes allows for the development of efficient algorithms for combinatorial optimization problems. To be formal, a set system $(F, \mathcal{Z})$ is a collection F of subsets of a ground set $\mathcal{Z}$. For example, F could be subsets of the power set of $\mathcal{Z}$ or could be subsets that satisfy the structure of a greedoid[285], semi-lattice[91], independence systems[124] or an antimatroid[140,272,16] and so forth.

Popular set function classes such as submodular functions[333,165,385,183,174,289,248] have resulted in a wide array of powerful algorithms for several tasks across different fields.

Under lack of submodularity, relaxations that characterize approximate submodularity,[58,64,229,112,135] have been introduced to develop combinatorial algorithms with approximation guarantees. Other set function classes beyond submodularity include those of subadditive functions, quasi-submodular functions and the lesser-known class of induced quasi-concave set functions relevant to this paper.

This paper introduces a parallel algorithm for optimizing quasi-concave set functions with global optimality guarantees as opposed to submodular optimization that provides approximate solutions. Algorithms for optimizing general quasi-concave set functions do not exist. In contrast, a specific sub-class of quasi-concave set functions that can be written in terms of monotone linkage functions can be optimized to obtain globally optimal solutions. For example, we show that certain monotone linkage functions of distance covariance induce a corresponding quasi-concave set function. We use our algorithm to find an optimally diverse set of features based on distance covariance.

14.1.1 Preliminaries

We now list the definition of *quasi-concave set functions* and state the *induced quasi-concave set function optimization problem*, which are central to the focus of this paper.

14.2 Quasi-concave set functions

Definition 16 (**Quasi-Concave Set Function**[378,298,594,539]). *A function* $F : \mathcal{F} \mapsto \mathbb{R}$ *defined on a set system* $(\mathbf{X}, \mathcal{F})$ *is quasi-concave if for each* $\mathbf{S}, \mathbf{T} \in \mathcal{F}$*,*

$$F(\mathbf{S} \cap \mathbf{T}) \geq \min\{F(\mathbf{S}), F(\mathbf{T})\} \tag{14.1}$$

Connection: We would like to note its notational similarity to its continuous counterpart of strictly quasi-concave functions, which are those real-valued functions defined on any convex subset of real-valued vector spaces such that $f(\lambda x + (1-\lambda)y) \geq \min\{f(x), f(y)\}$ for all $x \neq y$ and $\lambda \in (0,1)$.

We denote the set $2^{\mathbf{X}} \setminus \{\phi, \mathbf{X}\}$ by $\mathcal{P}^{-X}$ and we use i indexed subsets like S_i to indicate a singleton (unit cardinality) element of $\mathbf{S}$ labeled by i.

Definition 17 (**Monotone Linkage Function**[378]). *A function* $\pi(X_i, \mathbf{Z})$ *defined on* $\mathbf{Z} \in \mathcal{P}^{-X}, X_i \in \mathbf{X} \setminus \mathbf{Z}$ *is called a monotone linkage function if*

$$\pi(X_i, \mathbf{S}) \geq \pi(X_i, \mathbf{T}), \mathbf{S} \subseteq \mathbf{T} \in \mathcal{F}, \forall X_i \in \mathbf{X} \setminus T \tag{14.2}$$

We would like to note for the clarity of the reader that X_i is an element while $\mathbf{S}, \mathbf{T}$ are sets. Therefore, to make this distinction clear, we denote sets in bold-faced font and elements otherwise.

Monotone linkage functions have been introduced and used for clustering in[273,274]. A recent work[457] uses these functions to find maximum margin separations in finite closure systems.

Induced quasi-concave set function optimization This is stated as the problem of maximizing a quasi-concave set function $M_\pi(\mathbf{T})$ over the modified power set $\mathcal{P}^{-X}$:

$$\arg\max_{\mathbf{T}\subset\mathcal{P}^{-\mathbf{X}}} M_\pi(\mathbf{T}) = \arg\max_{\mathbf{T}\subset\mathcal{P}^{-\mathbf{X}}} \min_{X_i\in\mathbf{X}\setminus\mathbf{T}} \pi(X_i,\mathbf{T}) \tag{14.3}$$

where $\pi(X_i, \mathbf{Z})$ is a monotone linkage function.

14.3 Contributions

1. We provide a parallel algorithm to find all the subsets that globally optimize the induced quasi-concave set function optimization problem in (14.3).

Type	Induced Quasi-concave set function (Parallel: Ours)	Induced Quasi-concave set function	Quasi-concave set function (General purpose)	Unconstrained Submodular	Robust submodular	Unconstrained Quasi submodular	Quasi semistrictly submodular M-/L-convex	$SSQM^{\neq}$ under M-convex domain
Complexity	On n processors, $\mathcal{O}(n^2g)+\mathcal{O}(\log\log n)$. For n^2, n^3 processors, check Table 2.	$\mathcal{O}(n^3g)+\mathcal{O}(n)$	Unknown	NP-Hard	$\mathcal{O}(nk)$	$\mathcal{O}(n^2)$	$\mathcal{O}(n^2\log L)+\mathcal{O}(n^2)$	$\mathcal{O}(n^4(\log L)^2)$
Solution	Globally optimal	Globally optimal	Unknown	Unknown	Approximate	Approximate	Approximate	Approximate

Table 14.1: We show the computational complexity of our parallel algorithm and contrast it with that of its non-parallel version (cubic complexity), settings of submodular optimization and its relaxations. n is the size of the ground set, k is the cardinality of the returned set = $\max\{|x(v)-y(v)||x,y\in dom\ f, v\in V\}$ where $f: Z^V \mapsto \mathbb{R}\cup\{+\infty\}$ and g is the complexity to compute the monotone linkage function.

2. The proposed parallel algorithm has a time complexity over n processors of $\mathcal{O}(n^2g)+\mathcal{O}(\log\log n)$ where n is the cardinality of the ground set and g is the complexity to compute the monotone linkage function that induces a corresponding quasi-concave set function via a duality. The complexity reduces to $\mathcal{O}(gn\log(n))$ on n^2 processors and to $\mathcal{O}(gn)$ on n^3 processors. The parallel approach reduces the currently existing cubic computational complexity of the non-parallel version, which is $\mathcal{O}(n^3g)+\mathcal{O}(n)$.

3. As an example, we show that some functions of distance covariance (a measure of statistical dependence) are quasi-concave set functions. This lets us optimize them to obtain globally optimal maxi-min solutions for the most diverse subset of features.

14.3.1 Quasi-concave set function optimization under various set systems

A greedy-type algorithm for finding maximizers of induced quasi-concave set functions was constructed in[378,298,594]. Inspired by this work, extensions of these algorithms were developed for the setting of multipartite graphs in[534]. Similarly, quasi-concave set functions of distance covariance were derived in[539] and their optimization resulted in a solution for a diverse feature selection problem with guarantees. Furthermore, quasi-concave set functions were extended to various set systems, including antimatroids[309] and meet-semilattices in[275].

# of processors	**Time Complexity**
n (Ours)	$\mathcal{O}(n^2 g)$
n^2 (Ours)	$\mathcal{O}(gn \log n)$
n^3 (Ours)	$\mathcal{O}(gn)$
Non-parallel	$\mathcal{O}(n^3 g) + \mathcal{O}(n)$

Table 14.2: In this table, we show the complexity of our proposed parallel algorithm with respect to increasing number of processors $n, n^2 \& n^3$. Here, n is also chosen to be around the order of size of the ground set. We show that the running times can be drastically reduced from the cubic complexities in the non-parallel version.

14.4 Related work: Comparing quasi-concave set functions with submodularity

Given the seminal impact of submodular optimization, we would like to compare the definitions of quasi-concave set functions with submodular functions and their relaxations. We state some connections inline that we find accordingly.

1. **Submodular optimization**[183] Let **V** be a ground set with cardinality $|\mathbf{V}| = n$, and let $f : 2^{\mathbf{V}} \to \mathbb{R}_{\geq 0}$ be a set function defined on **V**. The function f is said to be submodular if for any sets $\mathbf{X} \subseteq \mathbf{Y} \subseteq \mathbf{V}$ and any element $e \in V \backslash Y$, it holds that the discrete derivative

   $$f(\mathbf{X} \cup \{e\}) - f(\mathbf{X}) \geq f(\mathbf{Y} \cup \{e\}) - f(\mathbf{Y})$$

   is non-increasing in **X**. That is, the incremental gain of adding an element to a subset is $\geq$ (is not smaller) the incremental gain of adding it to a superset. An equivalent definition is that for every **S**,**T** $\subseteq$ **V** we have that

   $$f(\mathbf{S}) + f(\mathbf{T}) \geq f(\mathbf{S} \cup \mathbf{T}) + f(\mathbf{S} \cap \mathbf{T}) \tag{14.4}$$

   The problem of maximizing a normalized monotone submodular function subject to a cardinality constraint has been studied extensively. A celebrated result of (Nemhauser et al., 1978) shows that a simple greedy algorithm that starts with an empty set and then iteratively adds elements with highest marginal gains provides a $(1 - 1/e)$-approximation.

Connection: Upon defining a linkage function to be equal to a discrete derivative of a submodular function as

$$\pi(e, \mathbf{X}) = f(\mathbf{X} \cup \{e\}) - f(\mathbf{X})$$

it can be seen that the derivative of a submodular function is a monotone linkage function. However, not every monotone linkage function is a derivative of some submodular function[376,377]. Combining equations (3) and (4), we can say that the functions that are both submodular and quasi-concave set functions would satisfy $f(\mathbf{S}) + f(\mathbf{T}) >= f(\mathbf{S} \cup \mathbf{T}) + f(\mathbf{S} \cap \mathbf{T}) >= f(\mathbf{S} \cup \mathbf{T}) + \min\{f(\mathbf{S}), f(\mathbf{T})\}$.

2. **Robust submodular optimization** Robust versions of submodular optimization problem were introduced in[290,367,63,271,247,39,416]. An earlier variant is of the form introduced in[290] as

$$\max_{\mathbf{S} \subseteq \mathbf{V}, |\mathbf{S}| \leq k} \min_{\mathbf{Z} \subseteq \mathbf{S}, |\mathbf{Z}| \leq \tau} f(\mathbf{S} \backslash \mathbf{Z})$$

The τ refers to a robustness parameter, representing the size of the subset $\mathbf{Z}$ that is removed from the selected set $\mathbf{S}$. The goal is to find a set $\mathbf{S}$ that is robust upon the worst possible removal of τ elements, i.e., after the removal, the objective value should remain as large as possible. For $\tau = 0$, the problem reduces to standard submodular optimization. The greedy algorithm, which is near-optimal for standard submodular optimization, can perform arbitrarily badly for the robust version of the problem.

Connection: Note that our statement of induced quasi-concave set function optimization problem naturally has a robustness component similar to the max-min

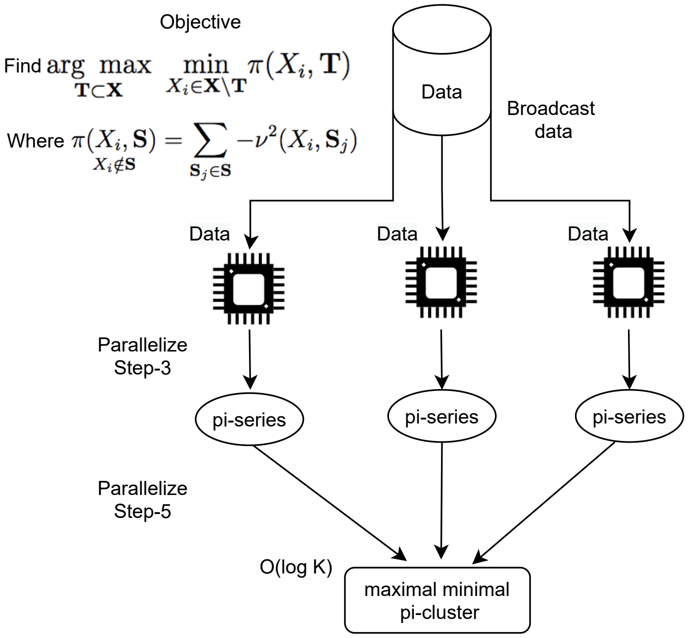

Objective
Find $\arg\max_{\mathbf{T}\subset\mathbf{X}} \min_{X_i\in\mathbf{X}\setminus\mathbf{T}} \pi(X_i,\mathbf{T})$
Where $\pi(X_i,\mathbf{S})_{X_i\notin\mathbf{S}} = \sum_{\mathbf{S}_j\in\mathbf{S}} -\nu^2(X_i,\mathbf{S}_j)$
Data
Broadcast data
Data
Data
Data
Parallelize Step-3
pi-series
pi-series
pi-series
Parallelize Step-5
O(log K)
maximal minimal pi-cluster

Figure 14.1: The proposed parallel algorithm consists of generating a π-series at each parallel entity over a copy of the data. The π-series at each entity starts with a different X_i. Each entity then generates a π-cluster corresponding to its generated $\pi - series$. The final step involves picking the best $\pi - cluster$. This is the only step that is not done in parallel.

constraints used in the literature on robust submodular optimization.

3. **Quasi submodular and semi-strictly submodular functions**[353] A set function $F : 2^N \mapsto \mathbb{R}$ is quasi-submodular function if $\forall$**X, Y** $\subseteq$ **N**, *both* of the following conditions are satisfied

$$F(\mathbf{X}\cap\mathbf{Y}) \geq F(\mathbf{X}) \Rightarrow F(\mathbf{Y}) \geq F(\mathbf{X}\cup\mathbf{Y})$$
$$F(\mathbf{X}\cap\mathbf{Y}) > F(\mathbf{X}) \Rightarrow F(\mathbf{Y}) > F(\mathbf{X}\cup\mathbf{Y})$$

On a similar note, a rich family of semistrictly submodular, discrete Quasi L-convex and discrete M-convex functions were introduced in[379,380].

14.5 Algorithm and proof of optimality

We now introduce the required definitions and corresponding theory to derive the algorithm. This includes definitions for π-series and π-clusters

Definition 18 (π**-series).** *We refer to a series* $s_\pi = (X_{i_1}, \ldots, X_{i_N})$ *as a* π*-series if*

$$\pi(X_{i_{k+1}}, \overline{\mathbf{S}}_{\mathbf{k}}) = \underset{\mathbf{X_i} \in \mathbf{X} \setminus \overline{\mathbf{S}}_{\mathbf{k}}}{\mathit{min}} \pi(\mathrm{X_i}, \overline{\mathbf{S}}_{\mathbf{k}}) \tag{14.5}$$

for any starting set $\overline{\mathbf{S}}_{\mathbf{k}} = \{\mathbf{X_{i_1}}, \ldots, \mathbf{X_{i_k}}\}, \mathbf{k} = \mathbf{1}, \ldots, \mathbf{N} - \mathbf{1}$*.*

Therefore, it is a way of greedily populating a series that can start with any first element $\mathbf{X_{i_1}}$ being the current series. Still, the subsequent element to be added to the series must be the element that minimizes the element to current series function of $\pi(\mathbf{X_{i_{k+1}}}, \overline{\mathbf{S}}_{\mathbf{k}})$ where $\mathbf{X_{i_{k+1}}}$ is the next element added and $\overline{\mathbf{S}}_{\mathbf{k}}$ is the current series.

Definition 19 (π**-cluster).** *A subset* $\mathrm{S} \in \mathcal{P}^{-\mathbf{X}}$ *will be referred to as a* π*-cluster if there exists a* π*-series,* $s_\pi = (X_{i_1}, \ldots, X_{i_N})$*, such that* S *is a maximizer of* $M_\pi(\overline{\mathbf{S}}_{\mathbf{k}})$ *over all starting sets* $\overline{\mathbf{S}}_{\mathbf{k}}$ *of* s_π*.*

Theorem 14.5.1. [273] If for a π-series $s_\pi = (X_{i_1}, X_{i_2}, \ldots, X_{i_N})$, a subset $\mathbf{S} \subset \mathbf{X}$ contains X_{i_1}, and if $X_{i_{k+1}}$ is the first element in s_π not contained in S (for some $k \in \{1, \ldots, N-1\}$, then $M_\pi(\overline{\mathbf{S}}_{\mathbf{k}}) \geq M_\pi(\mathbf{S})$

where $\overline{\mathbf{S}}_{\mathbf{k}} = (X_{i_1}, \ldots, X_{i_k})$. In particular, if S is an inclusion-minimal maximizer of M_π (with regard to $\mathcal{P}^{-\mathbf{X}}$), then $\mathrm{S} = \overline{\mathbf{S}}_{\mathbf{k}}$, that is, S is a π-cluster.

From[273] we have

Proposition 4. *If* $\mathbf{S_1}, \mathbf{S_2} \subset \mathbf{X}$ *are overlapping maximizers of a quasi-concave set function* $M_\pi(\mathbf{S})$ *over* $\mathcal{P}^{-\mathbf{X}}$*, then* $\mathbf{S_1} \cap \mathbf{S_2}$ *is also a maximizer of* $M_\pi(\mathbf{S})$*.*

This means that the minimal maximizers of a quasi-convex set function are not overlapping. Moreover, any nonminimal maximizer can be uniquely partitioned into a set of minimal ones.

Theorem 14.5.2. Each maximizer of a quasi-concave set function on $\mathcal{P}^{-\mathbf{X}}$ is a union of its inclusion-minimal maximizers.

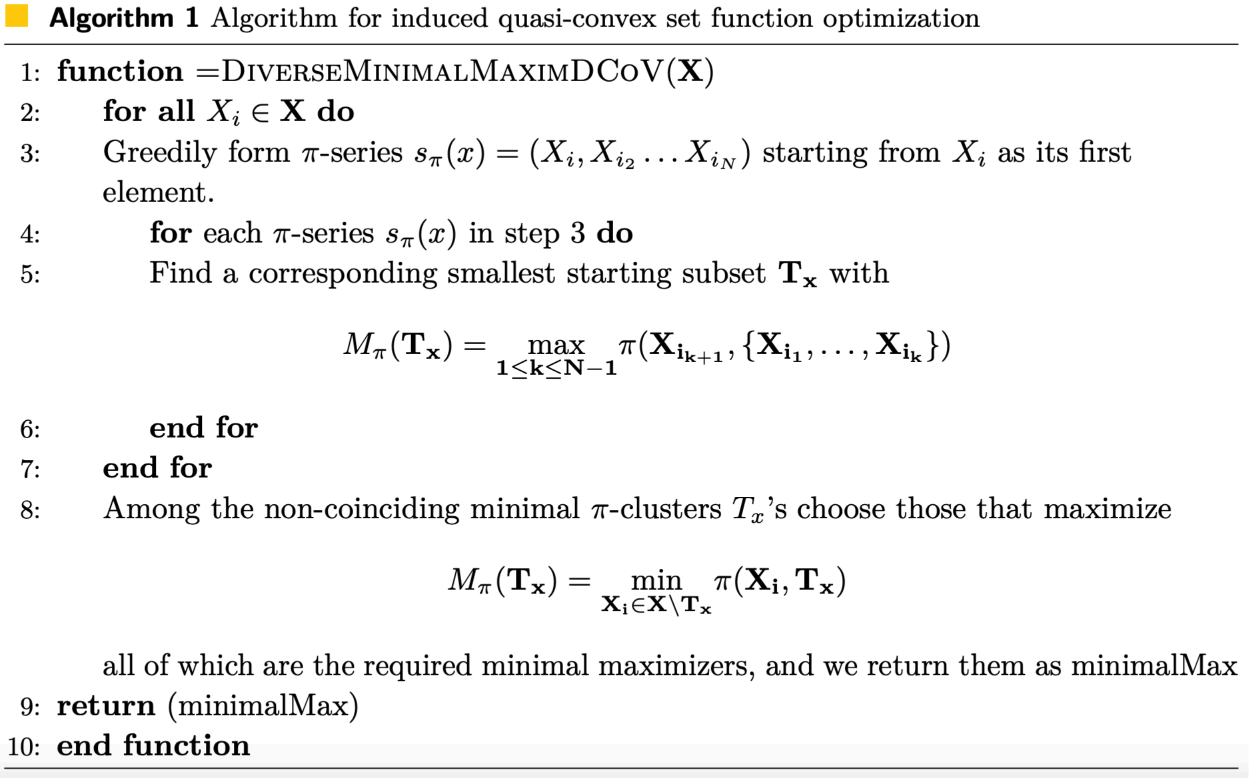
Proof. Indeed, if $\mathbf{S}^*$ is a maximizer of $M_\pi(\mathbf{S})$ over $\mathcal{P}^{-\mathbf{X}}$, then, according to Theorem 14.5.1, for any $X_i \in \mathbf{S}^*$, there exists a minimal maximizer included in $\mathbf{S}^*$ and containing X_i. □

Theorem 14.5.3. The algorithm above finds all the minimal maximizers over $\mathcal{P}^{-\mathbf{X}}$.

Proof. From Theorem 14.5.2, it follows that each element of minimalMax is a maximizer of $M_\pi(\mathbf{S})$ over $\mathcal{P}^{-\mathbf{X}}$. Assume that there is a minimal maximizer $\mathbf{S}$ that does not belong to minimalMax, and let $X_i \in \mathbf{S}$. Then, according to Theorem 14.5.1, there exist π-series starting from X_i and minimal π-cluster $T_x \subseteq \mathbf{S}$ containing X_i with $M_\pi(\mathbf{T_x}) \geq \mathbf{M}_\pi(\mathbf{S})$. Since $\mathbf{S}$ does not belong to minimalMax, and, according to Steps 5 and 8 of the algorithm, T_x or some subset of T_x belongs to minimalMax, there is a minimal maximizer strictly included in $\mathbf{S}$ which contradicts the minimality of $\mathbf{S}$. □

14.6 Computational complexity

When we have n processors, then we can build each π-series (in step-3 of the algorithm) in $\mathcal{O}(n^2 g)$ on one processor (including step 5), and because we build them in parallel,



Algorithm 1 Algorithm for induced quasi-convex set function optimization

```
1:  function =DiverseMinimalMaximDCoV($\mathbf{X}$)
2:      for all $X_i \in \mathbf{X}$ do
3:      Greedily form $\pi$-series $s_\pi(x) = (X_i, X_{i_2} \dots X_{i_N})$ starting from $X_i$ as its first
        element.
4:          for each $\pi$-series $s_\pi(x)$ in step 3 do
5:          Find a corresponding smallest starting subset $\mathbf{T_x}$ with
                $$M_\pi(\mathbf{T_x}) = \max_{\mathbf{1 \le k \le N-1}} \pi(\mathbf{X_{i_{k+1}}}, \{\mathbf{X_{i_1}}, \dots, \mathbf{X_{i_k}}\})$$
6:          end for
7:      end for
8:      Among the non-coinciding minimal $\pi$-clusters $T_x$'s choose those that maximize
                $$M_\pi(\mathbf{T_x}) = \min_{\mathbf{X_i \in X \setminus T_x}} \pi(\mathbf{X_i}, \mathbf{T_x})$$
        all of which are the required minimal maximizers, and we return them as minimalMax
9:  return (minimalMax)
10: end function
```

steps 3-5 take $\mathcal{O}(n^2g)$ time. Finding the maximum in step 8 takes $\mathcal{O}(\log\log n)$ time on n processors, under the CRCW (concurrent-read-concurrent-write) mode[231,230,529,293]. If we have n^2 processors, n processors are used to build each π-series. To add one element to a series, we have to find $\min$ between n elements, that takes $\mathcal{O}(\log\log n)$ on n processors, so to build each pi-series takes $g * (\log 1 + \log 2 + \ldots + \log n) = \mathcal{O}(gn\log n)$, and to finish it we have to find $\max$ with n^2 processors which takes $\mathcal{O}(1)$ time. This gives us $\mathcal{O}(gnloglogn)$ complexity. If we have n^3 processors, then we can use n^2 processors to build each π-series. To add one element to a series, we have to find $\min$ between n elements, which takes $\mathcal{O}(1)$ on n^2 processors. So to build each π-series takes $\mathcal{O}(gn)$ time, and to finish we have to find $\max$ with n^3 processors, that takes $\mathcal{O}(1)$ time. These are summarized in Tables 1 and 2.

14.7 Maxi-min Diverse Variable Selection

As an illustrating example, we aim to find all the subsets that maximize the function $M_\pi(\mathbf{T})$, which results in the solutions which are diverse features in the context of statistics/machine learning as follows

$$\arg\max_{\mathbf{T}\subset\mathbf{X}} M_\pi(\mathbf{T}) = \arg\max_{\mathbf{T}\subset\mathbf{X}} \min_{X_i\in\mathbf{X}\setminus\mathbf{T}} \pi(X_i,\mathbf{T}) \tag{14.6}$$

For specificity, we use distance covariance upon normalization of the data as a measure of statistical dependence to model the diversity via $\pi(\mathbf{X_i},\mathbf{S})$ as defined in Lemma 8.1.

14.8 Relevant Background on Distance Covariance and Distance Correlation

In this section, we introduce some preliminaries about distance correlation and distance covariance and illustrate a connection between these functions and quasi-concave set function optimization. Distance Correlation[500] is a measure of nonlinear statistical dependencies between random vectors of arbitrary dimensions. We describe below distance covariance $\nu^2(\mathbf{x}, \mathbf{y})$ between random variables $\mathbf{x} \in \mathbb{R}^d$ and $\mathbf{y} \in \mathbb{R}^m$ with finite first moments is a non-negative number as

$$\nu^2(\mathbf{x}, \mathbf{y}) = \int_{\mathbb{R}^{d+m}} |f_{\mathbf{x},\mathbf{y}}(t, s) - f_{\mathbf{x}}(t) f_{\mathbf{y}}(s)|^2 w(t, s) dtds \tag{14.7}$$

where $w(t, s)$ is a weight function as defined in[500], $f_{\mathbf{x}}, f_{\mathbf{y}}$ are characteristic functions of $\mathbf{x}, \mathbf{y}$ and $f_{\mathbf{x},\mathbf{y}}$ is the joint characteristic function.

The distance covariance is zero if and only if random variables $\mathbf{x}$ and $\mathbf{y}$ are independent. Using the above definition of distance covariance, we have the following expression for Distance Correlation[500]:

The squared Distance Correlation between random variables $\mathbf{x} \in \mathbb{R}^d$ and $\mathbf{y} \in \mathbb{R}^m$ with finite first moments is a nonnegative number is defined as

$$\rho^2(\mathbf{x}, \mathbf{y}) = \begin{cases} \frac{\nu^2(\mathbf{x},\mathbf{y})}{\sqrt{\nu^2(\mathbf{x},\mathbf{x})\nu^2(\mathbf{y},\mathbf{y})}}, & \nu^2(\mathbf{x}, \mathbf{x})\nu^2(\mathbf{y}, \mathbf{y}) > 0. \\ 0, & \nu^2(\mathbf{x}, \mathbf{x})\nu^2(\mathbf{y}, \mathbf{y}) = 0. \end{cases} \tag{14.8}$$

The Distance Correlation defined above has the following interesting properties.

1. $\rho^2(\mathbf{x}, \mathbf{y})$ is applicable for arbitrary dimensions d and m of $\mathbf{x}$ and $\mathbf{y}$ respectively.

2. $\rho^2(\mathbf{x}, \mathbf{y}) = 0$ if and only if $\mathbf{x}$ and $\mathbf{y}$ are independent.

3. $\rho^2(\mathbf{x}, \mathbf{y})$ satisfies the relation $0 \leq \rho^2(\mathbf{x}, \mathbf{y}) \leq 1$.

14.8.1 Sample Distance Covariance and Sample Distance Correlation

We provide the definition of sample version of distance covariance given samples $\{(\mathbf{x}_k, \mathbf{y}_k)|k = 1, 2, \ldots, n\}$ sampled i.i.d. from joint distribution of random vectors $\mathbf{x} \in \mathbb{R}^d$ and $\mathbf{y} \in \mathbb{R}^m$. To do so, we define two squared Euclidean distance matrices $\mathbf{E_X}$ and $\mathbf{E_Y}$, where each entry $[\mathbf{E_X}]_{k,l} = \|\mathbf{x}_k - \mathbf{x}_l\|$ and $[\mathbf{E_Y}]_{k,l} = \|\mathbf{y}_k - \mathbf{y}_l\|$ with $k, l \in \{1, 2, \ldots, n\}$. These squared distance matrices are double-centered by making their row and column sums zero and is denoted as $\widehat{\mathbf{E}}_\mathbf{X}, \widehat{\mathbf{Q}}_\mathbf{X}$, respectively. So given a double-centering matrix $\mathbf{J} = \mathbf{I} - \frac{1}{n}\mathbf{1}\mathbf{1}^T$, we have $\widehat{\mathbf{E}}_\mathbf{X} = \mathbf{J}\mathbf{E_X}\mathbf{J}$ and $\widehat{\mathbf{E}}_\mathbf{Y} = \mathbf{J}\mathbf{E_Y}\mathbf{J}$. The sample distance covariance and sample distance correlation can now be defined as follows.

Definition 20. ***Sample Distance Covariance***[500]***:*** *Given i.i.d samples $\mathcal{X} \times \mathcal{Y} = \{(\mathbf{x}_k, \mathbf{y}_k)|k = 1, 2, 3, \ldots, n\}$ and corresponding double centered Euclidean distance matrices $\widehat{\mathbf{E}}_\mathbf{X}$ and $\widehat{\mathbf{E}}_\mathbf{Y}$, the squared sample distance correlation is defined as,*

$$\hat{\nu}^2(\mathbf{X}, \mathbf{Y}) = \frac{1}{n^2} \sum_{k,l=1}^{n} [\widehat{\mathbf{E}}_\mathbf{X}]_{k,l} [\widehat{\mathbf{E}}_\mathbf{Y}]_{k,l},$$

Using this, sample distance correlation is given by

$$\hat{\rho}^2(\mathbf{X}, \mathbf{Y}) = \begin{cases} \frac{\hat{\nu}^2(\mathbf{X}, \mathbf{Y})}{\sqrt{\hat{\nu}^2(\mathbf{X}, \mathbf{X})\hat{\nu}^2(\mathbf{Y}, \mathbf{Y})}}, & \hat{\nu}^2(\mathbf{X}, \mathbf{X})\hat{\nu}^2(\mathbf{Y}, \mathbf{Y}) > 0. \\ 0, & \hat{\nu}^2(\mathbf{X}, \mathbf{X})\hat{\nu}^2(\mathbf{Y}, \mathbf{Y}) = 0. \end{cases}$$

Monotonicity of distance covariance under lack of independence: If $\mathbf{X}, \mathbf{Z} \in \mathbb{R}^p$ and $\mathbf{Y} \in \mathbb{R}^q$ and if $\mathbf{Z} \perp\!\!\!\perp (\mathbf{X}, \mathbf{Y})$ then

$$\nu^2(\mathbf{X} + \mathbf{Z}, \mathbf{Y}) \leq \nu^2(\mathbf{X}, \mathbf{Y}) \tag{14.9}$$

Note that $\perp\!\!\!\perp$ indicates 'statistically independent' in the statistical literature.

14.8.2 Motivating applications for modelling diversity with quasi-concave set function optimization

A minor sampling of applications that benefit from the results in this paper does parallel traditional applications seen in submodular optimization literature. A few directions are listed below.

1. Maximally/minimally correlated marginal selection for private data synthesis[604].
2. Modeling diversity in active learning[559], determinantal point processes[516].
3. Diverse sample selection, feature selection and data summarization in machine learning and statistics.[417,134]

14.8.3 A monotone linkage function of distance covariance

Lemma 14.8.1. *The function $\pi(X_i, \mathbf{S})$ of distance covariance defined on $X_i \notin \mathbf{S}$ as*

$$\pi(X_i, \mathbf{S})_{X_i \notin \mathbf{S}} = \sum_{\mathbf{S}_j \in \mathbf{S}} -\nu^2(X_i, \mathbf{S}_j) \tag{14.10}$$

is a monotone linkage function.

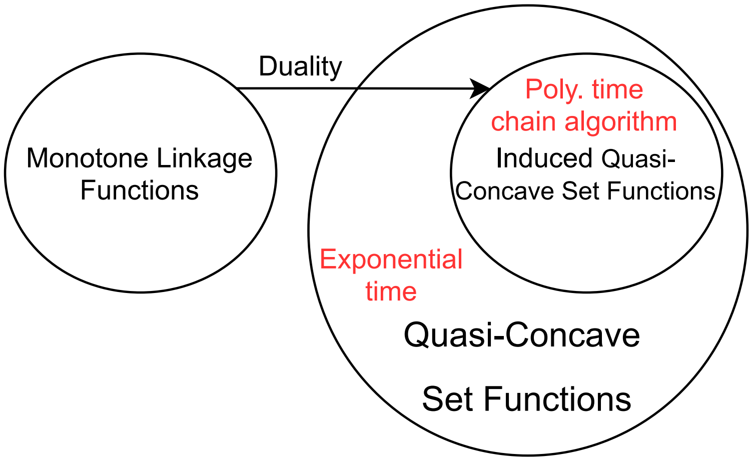

Duality
Monotone Linkage Functions
Poly. time chain algorithm
Induced Quasi-Concave Set Functions
Exponential time
Quasi-Concave Set Functions

Figure 14.2: This illustration refers to the duality between monotone linkage functions and quasi-concave set functions. Optimization algorithms for general quasi-concave set functions do not exist, while those induced via monotone linkage functions can be optimized in polynomial time.

Proof. For $\mathbf{S} \subseteq \mathbf{T}$ we have

$$\pi(X_i, \mathbf{T})_{X_i \notin \mathbf{T}} = \sum_{\mathbf{S}_j \in \mathbf{S}} -\nu_i^2(X_i, \mathbf{S}_j) - \sum_{\mathbf{T}_j \in \mathbf{T} \setminus \mathbf{S}} \nu_i^2(X_i, \mathbf{T}_j) \tag{14.11}$$

$$\leq \pi(X_i, \mathbf{S})_{X_i \notin \mathbf{T}} = \sum_{\mathbf{S}_j \in \mathbf{S}} -\nu_i^2(X_i, \mathbf{S}_j) \tag{14.12}$$

We would also like to note that as $\nu(\cdot)$ is a non-negative function the above inequality does hold true. □

By Assertion 1 from[273], we conclude that the function $M_\pi(\mathbf{T}) = \min_{X_i \in \mathbf{X} \setminus \mathbf{T}} \pi(X_i, \mathbf{T})$ is a quasi-concave set function.

Theorem 14.8.1 (Quasi-Concave Distance Covariance Set Function Theorem). If we have $\mathbf{S} \cap \mathbf{T} \neq \varnothing$ and $\forall \mathbf{S}, \mathbf{T}, \mathbf{Y}$ if $\nu^2(\mathbf{S}, \mathbf{T}) > 0 \wedge \nu^2(\mathbf{S}, \mathbf{Y}) > 0 \wedge \nu^2(\mathbf{T}, \mathbf{Y}) >$

0 then, we have

$$-\nu^2(\mathbf{S} \cap \mathbf{T}, \mathbf{Y}) \geq min(-\nu^2(\mathbf{S}, \mathbf{Y}), -\nu^2(\mathbf{T}, \mathbf{Y})) \tag{14.13}$$

Proof. [539]

If $\mathbf{S} \cap \mathbf{T} = \mathbf{S}$ then since $\mathbf{S} \subseteq \mathbf{T}$

the Kosorok's distance covariance inequality simplifies to give

$$-\nu^2(\mathbf{S}, \mathbf{Y}) \geq -\nu^2(\mathbf{T}, \mathbf{Y}) \tag{14.14}$$

Therefore, we have

$$-\nu^2(\mathbf{S} \cap \mathbf{T}, \mathbf{Y}) \geq min(-\nu^2(\mathbf{S}, \mathbf{Y}), -\nu^2(\mathbf{T}, \mathbf{Y}))$$

Similarly, if $\mathbf{S} \cap \mathbf{T} = \mathbf{T}$, then since $\mathbf{T} \subseteq \mathbf{S}$

$$-\nu^2(\mathbf{T}, \mathbf{Y}) \geq -\nu^2(\mathbf{S}, \mathbf{Y}) \tag{14.15}$$

and therefore,

$$-\nu^2(\mathbf{S} \cap \mathbf{T}, \mathbf{Y}) \geq min(-\nu^2(\mathbf{S}, \mathbf{Y}), -\nu^2(\mathbf{T}, \mathbf{Y})) \tag{14.16}$$

In the cases of $\mathbf{S} \cap \mathbf{T} \subset \mathbf{S}$ and $\mathbf{S} \cap \mathbf{T} \subset \mathbf{T}$ the Kosorok's distance covariance inequality gives

$$-\nu^2(\mathbf{S} \cap \mathbf{T}, \mathbf{Y}) > -\nu^2(\mathbf{S}, \mathbf{Y}) \tag{14.17}$$

and

$$-\nu^2(\mathbf{S} \cap \mathbf{T}, \mathbf{Y}) > -\nu^2(\mathbf{T}, \mathbf{Y}) \tag{14.18}$$

Thus,

$$-\nu^2(\mathbf{S} \cap \mathbf{T}, \mathbf{Y}) \geq min(-\nu^2(\mathbf{S}, \mathbf{Y}), -\nu^2(\mathbf{T}, \mathbf{Y})) \tag{14.19}$$

□

Note: In the case where considering $(\mathbf{X} \cup \mathbf{Z})$ is of interest, we could use the above theorem by incorporating degenerated random vectors as follows: Suppose $\mathbf{X} \in \mathbb{R}^{p1}$ and $\mathbf{Z} \in \mathbb{R}^{p2}$, then we augment $\mathbf{X}$ and $\mathbf{Z}$ to be $\tilde{\mathbf{X}} = (\mathbf{X}, \mathbf{0}_{p2})$ and $\tilde{\mathbf{Z}} = (\mathbf{0}_{p1}, \mathbf{Z})$ respectively. $\tilde{\mathbf{X}}$ and $\tilde{\mathbf{Z}}$ are therefore of the same dimension and $\tilde{\mathbf{X}} + \tilde{\mathbf{Z}} = (\mathbf{X}, \mathbf{Z})$. Therefore the $\mathbf{X} \cup \mathbf{Z}$ operation in the context of computing $\hat{\nu}(\mathbf{X} \cup \mathbf{Z}, \mathbf{Y})$ with matrices $\mathbf{X}, \mathbf{Z}, \mathbf{Y}$ is equivalent to appending the columns of $\mathbf{X}$ with the columns of $\mathbf{Z}$ followed by computing the sample-distance covariance between the resulting matrix and $\mathbf{Y}$.

14.9 NON-SUBMODULARITY OF DISTANCE COVARIANCE AND CONDITIONAL DISTANCE COVARIANCE

So far, we have shown a path to optimize quasi-concave set functions of distance covariance and compared their complexities to those of submodular optimization and other combinatorial optimization methods. We now show that distance covariance and its conditional variant are both subadditive and not necessarily submodular. This further motivates the optimization regime discussed in this paper. Imagine forming a ground set $\mathbf{T} = \mathbf{X} \cup \mathbf{Y}$ where since $\mathbf{X}$ and $\mathbf{Y}$ are given and disjoint. Therefore for any subset $\mathbf{U} \subseteq \mathbf{T}$, we can easily extract the corresponding subsets $\mathbf{C}$ of $\mathbf{X}$ and $\mathbf{R}$ of $\mathbf{Y}$ by just setting $\mathbf{C} = \mathbf{U} \cap \mathbf{X}$ and $\mathbf{R} = \mathbf{U} \cap \mathbf{Y}$. Now, given $\mathbf{X}, \mathbf{Y}, \mathbf{Z}$, we can define a function g on this large ground set as $g(\mathbf{U}) = -CDCOV(\mathbf{U} \cap \mathbf{X}, \mathbf{U} \cap \mathbf{Y}|\mathbf{Z})$. We now look at two subsets (say, $\mathbf{U_1}$ and $\mathbf{U_2}$) of $\mathbf{T}$. We then set $\mathbf{X_1} = \mathbf{U_1} \cap \mathbf{X}$ and $\mathbf{Y_1} = \mathbf{U_1} \cap \mathbf{Y}$, and similarly $\mathbf{X_2} = \mathbf{U_2} \cap \mathbf{X}$, $\mathbf{Y_2} = \mathbf{U_2} \cap \mathbf{Y}$. That is, $\mathbf{X_1}$ and $\mathbf{Y_1}$ are the $\mathbf{X}$ and $\mathbf{Y}$ parts of $\mathbf{U_1}$ and similarly for $\mathbf{U_2}$. Then, we find that:

$$
\begin{aligned}
g(U_1 \cup U_2) &= -CDCOV(X_1 \cup X_2, Y_1 \cup Y_2|Z) && (14.20)\\
&\leq -CDCOV(X_1, Y_1|Z) - CDCOV(X_2, Y_2|Z) && (14.21)\\
&= g(U_1) + g(U_2) && (14.22)
\end{aligned}
$$

Since CDCOV (conditional distance covariance) is non-negative, we have the subadditivity above, as we have that g is non-positive. The above result holds for distance covariance as well without loss of generality.

$$\mathbf{g(U_1) + g(U_2) \geq g(U_1 \cup U_2) \geq g(U_1 \cup U_2) + g(U_1 \cap U_2)}$$

$$\mathbf{CDCOV(U) + CDCOV(V) \geq CDCOV(U \cup V)}$$

only when U and V are disjoint.

$$\mathbf{CDCOV(U) + CDCOV(V \setminus U) \geq CDCOV(U \cup V)}$$

The subadditivity means that

$$-\mathbf{CDCOV}(\mathbf{X1} \cup \mathbf{X2} \cup \mathbf{X3}, \mathbf{Y1} \cup \mathbf{Y2} \cup \mathbf{Y3}|\mathbf{Z})$$

can be bounded above by

$$-\mathbf{CDCOV}(\mathbf{X1}, \mathbf{Y1}|\mathbf{Z}) - \mathbf{CDCOV}(\mathbf{X2} \cup \mathbf{X3}, \mathbf{Y2} \cup \mathbf{Y3}|\mathbf{Z})$$

or:

$$-\mathbf{CDCOV}(\mathbf{X1} \cup \mathbf{X2}, \mathbf{Y1} \cup \mathbf{Y2}|\mathbf{Z}) - \mathbf{CDCOV}(\mathbf{X3}, \mathbf{Y3}|\mathbf{Z})$$

or:

$$-\mathbf{CDCOV}(\mathbf{X1} \cup \mathbf{X3}, \mathbf{Y1} \cup \mathbf{Y3}|\mathbf{Z}) - \mathbf{CDCOV}(\mathbf{X2}, \mathbf{Y2}|\mathbf{Z})$$

This is enough for subadditivity, but for $-CDCOV$ to be bisubmodular (or submodular), we also need inequalities bounding it above by (for example)

$$-\mathbf{CDCOV}(\mathbf{X1} \cup \mathbf{X2}, \mathbf{Y1} \cup \mathbf{Y2}|\mathbf{Z}) - \mathbf{CDCOV}(\mathbf{X2} \cup \mathbf{X3}, \mathbf{Y2} \cup \mathbf{Y3}|\mathbf{Z}) + \mathbf{CDOV}(\mathbf{X2}, \mathbf{Y2}|\mathbf{Z})$$

14.10 Conclusion

We showed that Algorithm 1 gives globally exact solutions to the induced quasi-concave set function optimization and is highly parallelizable. This opens doors to a wide variety of real-world applications that we would like to pursue as part of future work.

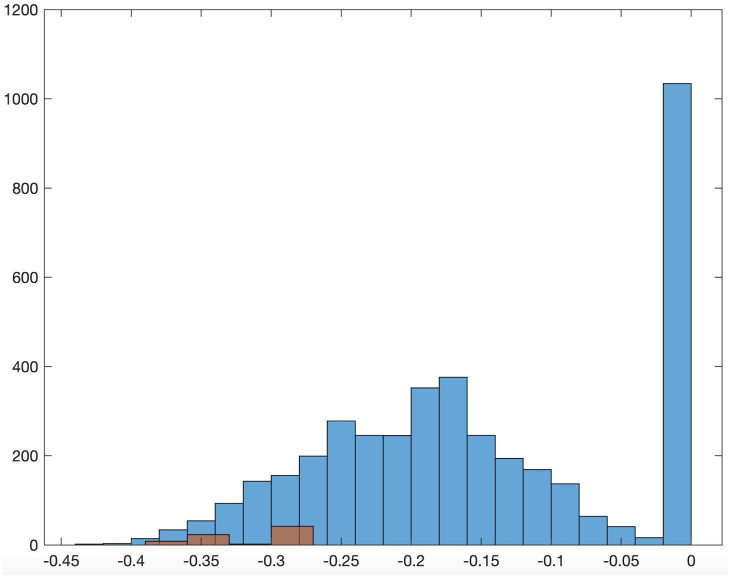


Figure 14.3: Empirically motivated conjecture

14.11 Open questions

That said, although we show that conditional distance covariance is not necessarily submodular, we conjecture that it could be approximately submodular to a reasonable degree based on the fact that submodular optimization libraries happened to pick solutions that are closer to the optimal on small datasets. We performed this experiment by enumerating the entire power set (with objective values in blue below) and compared the objectives obtained by the picks of non-deterministic fast submodular optimization (in orange). As we see, the orange picks were quite close to the optimal pick, and hence our experimentally motivated conjecture.

15

A comparative review of secure computation libraries for homomorphic encryption

15.1 Introduction

This chapter is based on our survey in[450]. Homomorphic Encryption is a method of secure computation on encrypted data (ciphertext) such that the result of the computation is also a ciphertext. Once this resultant ciphertext is decrypted, the decrypted result should match the output of operations on the corresponding unencrypted (plaintext) data.

For example, a hospital with a significant amount of private and sensitive patient information can homomorphically encrypt the data and send it to a third party for analysis. The third-party can perform calculations on encrypted data and send the results (also encrypted) back to the hospital. The hospital can then view the results by decrypting the data using a private key.

Several schemes of homomorphic encryption are categorized based on the number of operations allowed on the encrypted data. For a cryptosystem to be **Fully Homomorphic** (FHE), it should support any number of arbitrary computations. Brakerski-

Gentry-Vaikuntanathan (BGV)[76] and CGGI[114,113] are examples of Fully Homomorphic schemes. In practice, Fully Homomorphic schemes have tremendous overhead and are computationally very expensive. **Somewhat Homomorphic Encryption** (SWHE) schemes are practically more feasible but allow only certain operations on encrypted data and limit the number of computations as the Ciphertext size increases with each step due to noise. Some examples of Somewhat Homomorphic Encryption schemes are[Fan & Vercauteren,Yao,447,Boneh et al.,Ishai & Paskin]. **Partially Homomorphic Encryption** (PHE) schemes such as[52,51,Rivest et al.,199,166,382,Okamoto & Shigenori Uchiyama,398,133,270,185] allow only one type of operation any number of times - either addition or multiplication on encrypted data as compared to Somewhat Homomorphic schemes that support both. Generation of such schemes continues to be an active area of research, and the development of standards for homomorphic encryption has recently taken off as described in[Chase et al.,Albrecht et al.]. Below is an example to introduce the high-level concept of homomorphic encryption:

1. Let m be the plaintext message

2. Let a shared public key be a random odd integer p

3. Choose a random large q, small r, $(|r| \leq p/2)$

4. Ciphertext $c = pq + 2r + m$ (Ciphertext c is close to multiple of p)

5. Perform homomorphic addition/multiplication as required

6. Decrypt: $m = (c \mod p) \mod 2$

In this case, the corresponding homomorphic operations of addition and multiplication are given below:

Homomorphic Addition

$$c_1 = q_1 * p + 2 * r_1 + m_1$$

$$c_2 = q_2 * p + 2 * r_2 + m_2$$

$$c_1 + c_2 = (q_1 + q_2) * p + 2 * (r_1 + r_2) + (m_1 + m_2)$$

Homomorphic Multiplication

$$c_1 = q_1 * p + 2 * r_1 + m_1$$

$$c_2 = q_2 * p + 2 * r_2 + m_2$$

$$c_1 * c_2 = ((c_1 * q_2) + q_1 * c_2 * q_1 * q_2) * p + 2(2 * r_1 * r_2 + r_1 * m_2 + m_1 * r_2) + m_1 * m_2$$

If more complicated functions that require operations other than addition and multiplication need to be homomorphically encrypted, an alternative would be to generate a polynomial approximation (using the Taylor series, for example) and then apply homomorphic encryption on the resulting polynomial instead.

Homomorphic encryption libraries are based on different schemes and hence feature different behaviour. Microsoft's SEAL(V2.3.1)[Laine] is based on BFV[Fan & Vercauteren], HElib is based on BGV[76], and TFHE is based on CGGI[114,113].

15.2 Features of Homomorphic Encryption Libraries

In this section, we introduce important features of homomorphic encryption libraries. Features such as asymmetry, negative computations, noise budget, recrypt, ciphertext packing, bootstrapping[76,114] and relinearization are discussed in subsections 2.1 and 2.2. In sub-section 2.3, operations (atomic) allowed by various libraries are discussed and supported languages are also mentioned for all the libraries.

15.2.1 Basic features

Asymmetry

All homomorphic encryption libraries in this study have been implemented in an asymmetric manner where they use a pair of keys for the encryption and decryption of data. Specifically, keys used in asymmetric cryptography include a public key to encrypt the plaintext data that may be shared widely and a private key to decrypt the encrypted result. This differs from symmetric cryptographic systems that use a single key to encrypt the plaintext and decrypt the ciphertext.

Serialization

Specific homomorphic encryption libraries such as SEAL, HElib and TFHE provide custom APIs to serialize (and deserialize) keys and ciphertexts for local storage and retrieval. Libraries that don,t provide this feature require the developers to create their own implementation of serialization (which could be challenging with the complex data type) unless the ciphertext can be represented in a primitive type such as String or BigInteger using the same library.

NEGATIVE COMPUTATIONS

Negative computations correspond to subtracting operand 1 from operand 2 (where operand 2 > operand 1). This means the result of this computation should be a negative number. Microsoft SEAL(V2.3.1) uses BFV and Cheon-Kim-Kim-Song (CKKS) for encryption. In this scheme, the integers or real numbers correspond to polynomials in a specifically chosen ring[178]. Different kinds of encoders, such as Integer, Scalar, Fractional, and PolyCRTBuilder, can be used to convert the integers/reals in the input data into the corresponding coefficients in the polynomial space. In SEAL, if the ciphertext is encoded using Integer Encoder or Fractional Encoder, then the negative computations are supported. On the other hand, if the ciphertext is composed and encrypted using a PolyCRTBuilder then the resultant ciphertext after homomorphic subtraction will not be negative. This is due to the limitations in the Chinese remainder theorem when dealing with absolute values.

ENCRYPTION PARAMETERS, CIPHERTEXT SIZE AND MEMORY REQUIREMENTS

Implementing homomorphic encryption through any library requires initialising certain encryption parameters, such as polynomial modulus, coefficient modulus, plain modulus, noise standard deviation, a random generator, etc. Choice of these parameters can significantly affect the size of the ciphertext, RAM required, noise budget (refer to section 2.2.1), speed, performance and security[Laine] of the encryption. The size of the ciphertext is usually large with the complex Ciphertext data type in libraries like SEAL and HElib, and operations such as matrix rotation that use Galois keys (in SEAL) could result in substantial RAM requirements.

Basic Features	**SEAL**	**HElib**	**TFHE**	**Paillier**	**ELGamal**	**RSA**
Asymmetric	Yes	Yes	Yes	Yes	Yes	Yes
Serialization and Deserialization of keys and ciphertexts	Yes	Yes	Yes	No	No	No
Negative computations support	Yes	No	No	No	No	No
Ciphertext size (less than 1MB for 1 input)	No	No	Yes	Yes	Yes	Yes
Can run on less than 2GB RAM	No	Yes	Yes	Yes	Yes	Yes

Table 15.1: Comparison of Homomorphic Encryption libraries based on basic features

15.2.2 Advanced features

Noise budget

A noise term is generally appended to a ciphertext in the encryption operation to guarantee the security of the cryptosystem. This term could be an integer (if the scheme is based on integers) or a polynomial (if the scheme is based on polynomials) with coefficients in {1,0,1}. The term's size depends on each system's security and correctness properties (for instance, a polynomial is typically considered small if all its coefficients are small). Homomorphic operations increase the noise, and beyond a threshold, the resultant ciphertext becomes too corrupt to be decrypted. The noise budget (invariant) is the total amount of noise that can be added until the decryption fails. Addition and subtraction have a minimal impact on noise compared to multiplication, and Partially Homomorphic Encryption schemes are unaffected by noise.

Recryption

Recryption is a technique to re-generate the noise budget of a ciphertext that was depleted by arbitrary computations. Recryption boosts bounded-depth homomorphism to unbounded-depth homomorphism. This implies that the noisy ciphertext can be converted into a noise-free ciphertext (of the same plaintext) without the secret key[193,Brakerski et al.].
Libraries that do not have recryption functionality implemented provide no means of converting a noisy ciphertext to a noise-free ciphertext. They, therefore, limit the number of arbitrary computations on a ciphertext.

Ciphertext Packing

In some homomorphic encryption libraries such as SEAL and HElib, a list of plain values can be packed into a single ciphertext vector by a technique called ciphertext packing using the Chinese Remainder Theorem (CRT)[Brakerski et al.]. Homomorphic operations are performed on these vectors, component-wise, in a SIMD (Single Instruction Multiple Data) fashion. Ciphertext packing achieves a nearly optimal homomorphic evaluation (up to polylogarithmic factors). Homomorphic operations act element-wise between encrypted matrices, allowing the user to obtain several orders of magnitude speed-ups in naively vectorizable computations.

BOOTSTRAPPING

In specific homomorphic encryption schemes, arithmetic operations on ciphertext can be performed using basic gates (AND, OR, NOT, etc), but arbitrary operations could reduce the available noise budget. Bootstrapping[76,114] is a technique to remove noise by passing a ciphertext and encrypted private key into a circuit that represents the decryption algorithm of an FHE scheme. This results in a new ciphertext corresponding to the original but with no noise. In the TFHE library, after every gate-by-gate operation, bootstrapping is applied to the resultant ciphertext, and hence any number of arbitrary operations can be performed.

RELINEARIZATION

Two input ciphertexts of sizes m and n, respectively, result in a ciphertext of the size $m + n - 1$ after multiplication. Consumption of the noise budget is also much higher during multiplication, especially when the input ciphertext sizes are enormous. Relinearization reduces the size of the resultant ciphertext after a multiplication operation to the initial size. A ciphertext of size $k + 1$ when relinearized produces a ciphertext of size k. After repeated steps, this can result in a ciphertext of size 2 that can be decrypted using a smaller degree decryption function to yield the same result[Laine]. Thus, relinearization of the resultant ciphertext after multiplication can significantly improve the performance on the subsequent operations, although relinearization by itself has both a computational cost and a noise budget cost.

MULTITHREADING

In homomorphic encryption libraries, multithreading corresponds to APIs exposed by the libraries being thread safe. Thread-safe APIs help avoid deadlock and ease effective inter thread communication. Most of the tools in SEAL, such as Encryptor, Decryptor, PolyCRTBuilder, and Evaluator, are thread-safe by default. HElib can be multithreaded by setting NTL_THREADS=on, -DFHE_THREADs, -DFHE_DCRT_THREADS flags before making the project. Multithreading is not supported in Partial Homomorphic Encryption libraries discussed in the paper.

Advanced Features	**SEAL**	**HElib**	**TFHE**	**Paillier**	**ELGamal**	**RSA**
Noise affected after each computation	Yes	Yes	Yes	No	No	No
Recryption	No	Yes	Yes	N/A	N/A	N/A
Ciphertext p̊acking	Yes	Yes	No	No	No	No
Relinearization	Yes	Yes	No	N/A	N/A	N/A
Multithreading	Yes	Yes	No	No	No	No

Table 15.2: Comparison of Homomorphic Encryption Libraries based on advanced features

15.2.3 OPERATIONS

CIPHERTEXT COMPARISON

Two ciphertexts can be compared for equality: greater than, greater than, equal to, less than or less than or equal to. TFHE allows the evaluation of an arbitrary boolean circuit composed of binary gates over encrypted data. A custom comparator circuit can be used to perform comparisons using TFHE. In SEAL and HElib, a Binary-Encoder must generate a ciphertext comprising only 0s and 1s. Two such ciphertexts can then be compared in a bit-wise manner. This process is time-consuming and less secure. A computer with limited resources can decrypt a ciphertext by randomly comparing it with a known one. Due to this security threat, HE libraries do not readily expose a comparison API.

DIVISION

BGV or BFV schemes do not allow the division of ciphertexts due to the randomness and complexity of the ciphertext. Its possible to approximate division by using all kinds of expansions. In fully homomorphic encryption, the division of ciphertext A and ciphertext B is performed by computing the inverse of ciphertext B (decrypt, inverse and encrypt) and multiplying the inverse ciphertext by ciphertext A (multiplicative inverse). Another technique is recursive subtraction. Recursive subtraction can work only if $A\%B$ equals zero.

Boolean Operations

Some homomorphic encryption libraries that are based on Secure Multilayer Perceptron[Bellafqira et al.] and Doubly Permuted Homomorphic Encryption[587] allow evaluating an arbitrary boolean circuit composed of binary gates over encrypted data.

Matrix Operations

SEAL exposes an API to perform matrix rotation and element-wise addition, multiplication and subtraction. In PHE libraries, a ciphertext matrix must be created by performing element-wise encryption on a $n \times n$ plaintext matrix. Then, custom logic must be implemented to perform row and column rotation.

Exponentiation

Exponentiation of ciphertexts is usually A raised to the power of $B(A^B)$ where A and B are ciphertexts. Current FHE libraries only provide an implementation to raise a ciphertext base with a plain text exponent. This is accomplished through repetitive multiplication of the ciphertext. Eg: 3^4 is $3 * 3 * 3 * 3 = 81$. The same can be accomplished on PHE schemes. In additive PHE scheme, 3^3 can be calculated as $3^3 = 27|3 + 3 + 3 = 9|9 + 9 + 9 = 27$.

Add Plain, Subtract Plain, Multiple Plain

Homomorphic operations are usually carried out between two cipher texts. If one of the operands could be a plaintext, it could significantly improve the performance. The size of the resultant ciphertext remains the same as the input ciphertext, and the relinearlization step could be skipped. SEAL provides functions to perform addition, subtraction and multiplication of a ciphertext with a plaintext. The plain operations are implemented in SEAL as Evaluator::add_plain, Evaluator::sub_plain and Evaluator::multiply_plain.

Operations	**SEAL**	**HElib**	**TFHE**	**Paillier**	**ELGamal**	**RSA**
Addition, S̊ubtraction	Yes	Yes	Yes	Yes	No	No
Multiplication	Yes	Yes	Yes	No	Yes	Yes
Comparison	No	No	No	No	No	No
Division	No	No	No	No	No	No
Boolean o̊perations	No	No	Yes	No	No	No
Bitwise operations	Yes	Yes	Yes	No	No	No
Matrix operations	Yes	Yes	No	No	No	No
Exponentiation	Yes	Yes	No	No	No	No
Square	Yes	Yes	Yes	No	Yes	Yes
Negation	Yes	Yes	No	No	No	No
Add Plain, Subtract Plain, Multiply Plain	Yes	No	No	No	No	No

Table 15.3: Different operations supported by Homomorphic Encryption libraries

Languages	SEAL	HElib	TFHE	Paillier	ELGamal	RSA
C++	Yes	Yes	No	Yes	Yes	Yes
Python	Yes	Yes	No	Yes	Yes	Yes
Java	No	No	No	Yes	Yes	Yes
C	No	No	Yes	No	No	No

Table 15.4: Homomorphic Library implementations across programming languages

15.3 Applications

The need to create models or derive predictions from confidential distributed datasets is a commonly surfacing theme in many industries. For example, medical information might be distributed across multiple clinics.[26] outlines various potential real-world applications of homomorphic Encryption. Some of the emerging applications are:

15.3.1 Healthcare

In healthcare, maintaining patient information privacy is critical; therefore, their private data is often protected by law. However, sharing and computing information that is distributed across systems is essential for diverse use cases such as coordinated patient care, fraud billing and reimbursements. It is, therefore difficult to strike a balance between risk and utility. For example, in 2018, there were 11 significant HIPAA enforcement actions with an average fine of $1.9 million[Group & by Year and]. Homomorphic Encryption can help balance the risk vis-a-vis utility by enabling the analysis of billing records across patient data to uncover potential cases of fraud reimbursement or billing without violating the patients privacy.

15.3.2 Financial Services

Clients and businesses in the financial services work with confidential information. Consequently, data, the models and functions computed on them are often considered proprietary and confidential. Data in financial services functions may even be a continuous stream reflecting the most up-to-date information necessary for decision-making and is often a result of exclusive research or data feeds available to a particular client and is often very expensive. Homomorphic Encryption provides the appropriate way to evaluate and run these data and functions privately. For instance, a client can upload an encrypted version of the function to the cloud, and the streaming data on which the functions/models run could be encrypted using the customer's public key and uploaded to the cloud.

15.3.3 Smart Grid

Consider a smart grid consisting of multiple microgrids, such as solar panel generators used by individuals. Each node in such a grid generates useful data like electrical generation and usage, temperatures of physical equipment, energy flows, etc. In the case of a generator, if the nodes belong to a smart grid, then measurements include current energy usage, smart lights, sensors in use, etc.

When the municipality or any other government entity wants an aggregate measure or an alert about the data, they can use homomorphic encryption to compute data from nodes. They can do this without violating the terms of business contracts that prohibit them from disclosing confidential information such as usage of energy in a particular mall or the location of the surveillance cameras in a household. In this way, they can develop trust and improve the credibility of the smart grids with the public. Homomorphic encryption plays a vital role in achieving this.

15.3.4 Genomics

Private data generated from sequencing the human genome for complex diseases or epidemiology can be a powerful tool in developing a cure or a therapy/treatment for the disease. DNA and RNA sequences can be generated rapidly; consequently, many such sequences are now available in laboratories and medical institutes. However, significant challenges exist in sharing this data.[280]. Individual DNA sequences are as unique as fingerprints - they can be tracked down to an individual. They can determine say, for e.g. if they are susceptible to Alzheimer's disease or heart attack. Existing rules for protecting genomics data have created a lot of limitations for researchers. Homomorphic Encryption can enable researchers to speed up sharing information while safeguarding the privacy of the individuals and thus significantly speed up discovery.

15.4 Conclusion

In this paper, we survey and compare libraries across various dimensions for homomorphic encryption. These techniques enable us to perform computations on encrypted data against having to decrypt data to perform computations. In this way, it allows for collaborative computing between multiple parties via encrypted ciphertexts. Although the field is rapidly progressing on the theoretical front, significant recent progress has made it practical from an application/practical standpoint. These factors are crucial for the rapid adoption and further development of this field.

Applications of homomorphic encryption primarily involve distributed applications in diverse sectors such as healthcare, smart grids or genomics. In these applications, ciphertexts, public keys, and other low-level information need to be shared between data providers, encrypted computing hosts, and the desired recipients of the results of the computation.

There are many scenarios, such as the one mentioned in healthcare detection or genomics research, where these applications are almost impossible to develop due to technical or legal reasons. In cases where the technology is available, one still has to cross the expensive and time-consuming barrier of legal processes, driven by the need to maintain strict privacy. However, we can hope that practical homomorphic encryption will lead to a dramatic rise in applications in cloud and edge computation where privacy is critical. We intend to share our learnings, motivate our colleagues, and help the research and technology community progress.

15.5 Other Homomorphic Encryption Libraries

Schneider publishes an exhaustive list of Homomorphic Encryption Libraries:

HEAAN - Scheme with native support for fixed point approximate arithmetic

FHEW - Homomorphic Encryption library based on Fast Fourier Transform

- Haskell library for ring-based[178] lattice cryptography that supports FHE

NFLlib - NTT-based Fast Lattice library

PALISADE - Lattice encryption library

Pyfhel - PYthon For HElib

libshe - Symmetric SWHE library based on DGHV scheme

cuHE - GPU-accelerated HE library for NVIDIA CUDA-Enabled GPUs

cuYASHE - Based on levelled FHE scheme YASHE for GPGPUs

python-paillier - PHE based on Paillier scheme

krypto - C++ implementation of multivariate quadratic FHE

petlib - Python library that implements several Privacy Enhancing Technologies

16

Future work

1. **Going beyond federated learning and split learning** Traditional backpropagation for the training of deep learning models is highly sequential by design. This is a major constraint in current-day distributed machine learning paradigms such as federated learning in terms of computing/communication resource efficiency, improved capacity to handle straggler clients, and synchronization bottlenecks.

Future work would focus on alternate paradigms for distributed machine learning along with applications in decentralized finance, decentralized digital health, decentralized customer discovery or retention (churn prediction) and distributed regulatory compliance monitoring over cohorts of large companies that hold private financial data.

2. **Distributed and private causal inference** There is a scope for future work on hypothesis testing (specifically independence testing and conditional independence testing) to perform causal inference in the setting when the data is privacy sensitive and distributed across siloes. This would have applications in decentralized data science such as decentralized finance, monitoring for dark-web-related activity, decentralized digital health and causal discovery from data siloes.

3. **Data markets** This section is based on our article in[420]. Data is increasingly concentrated in large firms. For startups and small organizations to compete, data availability can hinder any efforts to build better machine learning algorithms. Algorithmic capability indeed increases with the availability and quality of data. One way to tackle this is the marketplace approach. By creating conditions such that data, the raw material for artificial intelligence, can be bought and sold with security, privacy and consent safeguarded, specialized niches will be created, and firms will be able to tackle a smaller subset of the problem. A precondition for this large-scale collaboration is the existence of liquid markets at various steps of the value chain.

16.1 AI could benefit from Liquid Markets:

This section expands more on data markets to motivate future work. Data is increasingly concentrated in large firms. For startups and small organizations, competing is increasingly difficult as the lack of data availability can hamper any efforts to build better machine learning algorithms. Algorithmic capability indeed increases with the availability and quality of data. One way to tackle this is the marketplace approach. By creating conditions such that data, the raw material for AI, can be bought and sold with security, privacy and consent safeguarded, specialized niches will be made, and firms will be able to tackle a smaller subset of the problem. A precondition for this large-scale collaboration is the existence of liquid markets at various steps of the value chain.

Example: Consider a diagnostic healthcare company aiming to acquire labelled X-ray images from various hospitals for developing state-of-the-art diagnostics. The key problem in such a setting is: "How can the value of datasets from each hospital be estimated to decide their price?" The data for some hospitals can belong to unique health traits and demographics. They can be very valuable for the diagnostic use-case of the company, while data from some other hospitals may be of a relatively much lower value.

A fundamental problem, therefore, is that obtaining large amounts of diverse yet valuable data costs a lot of resources. There are also diminishing returns at some point when additional data does not improve the algorithm's capabilities if other data is not acquired intelligently in a cost-effective manner.

Absolute, relative or conditional data purchase: Another required facet to setting up a data market is to build a capability to perform the valuation of data in absolute terms (i.e. just by itself), relatively (i.e. in comparison to multiple datasets) or in conditional terms (i.e. valuating new data given currently existing data).

Intrinsic or extrinsic data valuation: Any of these data valuation use-cases can be performed via intrinsic factors of evaluation such as based on the quality of information within the dataset or via extrinsic factors of evaluation such as based on demand-supply, market economics, game theoretic mechanisms and speculative market forces or via a combination of both.

Goal dependent or independent data trading: An additional slicing to this problem includes goal-specific or goal-independent data valuation depending on whether there is a specific, well-defined goal for the data purchase or if it is exploratory by design for a goal that is currently undefined; but would be drafted later on.

Horizontal or vertical data acquisition: In addition to all these situations of data valuation, yet another categorization is based on whether the data acquisition is being done vertically (in terms of acquiring attributes/columns) or horizontally (acquiring records/rows) as in[195,253]. This terminology of 'vertical partitioning' and 'horizontal partitioning' extends from the databases and distributed systems research communities.

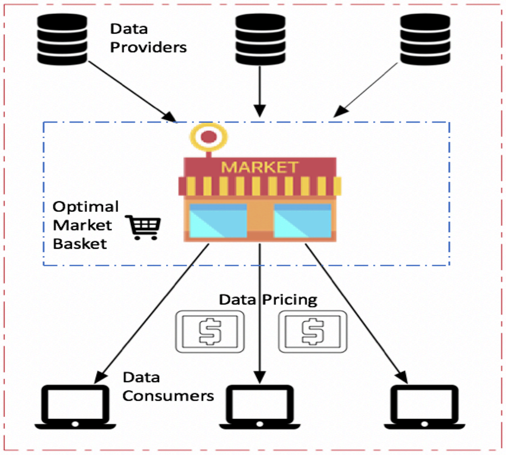
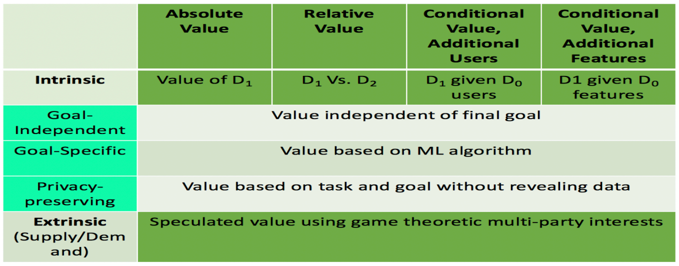


Figure 16.1: Data market showing data providers, data customers, notions of market basket and data pricing

Privacy aware data valuation: Ideally, such a data valuation needs to be achieved by looking at as few records per data source as possible or via privacy-aware AI. Pooling of all data at a centralized location defeats the central purpose, and the data sharing constraints of privacy, security, safety, fairness and resource efficiency need to be considered regarding a data valuation solution for data markets.

Relevance and diversity of data acquisition: An optimal data purchase under these constraints must cater to high utility and low redundancy (high diversity) of data regarding the incremental benefit obtained. There is often a tradeoff of utility vs. diversity of data that needs to be considered in realistic settings. This concept has been the guiding principle for techniques like sure independence screening (SIS) and conditional sure independence screening, currently actively being studied in the field of statistics and min redundancy max relevance (mRMR) introduced in the field of data mining during the precursory periods of current day AI and machine learning. A robust data valuation acts as a good input for data pricing and for building an optimal market basket of data for every data consumer.

The intent of sharing these possibilities is to motivate further discussion and research. We summarize some of these points about data valuation in the context of data markets, as shown below:



	Absolute Value	**Relative Value**	**Conditional Value, Additional Users**	**Conditional Value, Additional Features**
Intrinsic	Value of D_1	D_1 Vs. D_2	D_1 given D_0 users	D1 given D_0 features
Goal-Independent	Value independent of final goal			
Goal-Specific	Value based on ML algorithm			
Privacy-preserving	Value based on task and goal without revealing data			
Extrinsic (Supply/Demand)	Speculated value using game theoretic multi-party interests			

Figure 16.2: Landscape of data valuation problems for data markets

16.3 Data is very complex to price

The value of an incremental data unit is also conditionally dependent on data already possessed by the prospective data buyer entity that is valuating it. This is because one wants to obtain relevant yet diverse data from what is already available in-house. In addition, data can be acquired for performing a similar or a more diverse task compared to the current use cases being applied to data already available in-house. Also, there are so many archetypes that it is difficult to find a proxy variable (like weight or number in the case of other goods) that can be used to define the data. Since seamless discovery and a small spread in price are essential for a marketplace to function well, it has been challenging thus far to create a functioning data marketplace. A thorough data pricing strategy needs to adhere to the following guiding principles.

Data Pricing Guidelines

1. Liquidity: models freshness of that in terms of value vs diminished/increased value over time
2. Traceability: can be only 'sold' once or sold non-exclusively
3. Consent: maintains the owner's privacy, tracks consent over time, and reduces friction with smart contracts or data concierges.
4. Neutrality: accessible to all buyers to prevent unfair trading practices. Otherwise, it would encourage some players (large or small) to unfairly price out the rest of the prospective buyers during the trading.
5. Recourse: Allows for calling back, provides right to be forgotten, allows for some course correction, broadly remains self-sustaining.

16.4 Data sharing challenges that data markets need to address

Although acquiring the right amount of quality data is ideal, data sharing is heavily impeded by friction caused by a lack of trust, data sharing regulations such as HIPAA/GDPR, lack of ease and lack of incentive. We further expand on these factors that cause data friction.

1. **Lack of incentives:**
   (a) Large organizations need incentive mechanisms to share data with small players. For example, an incentive for data sharing between large central-

ized hospitals and local clinic testing centres could be to foster the better provision of health.

   (b) Big technology players have taken the lead and are rapidly collecting and hoarding data while monopolizing the data resources and preventing small players from entering into data acquisition. This stifles innovation.

   (c) Individuals need incentives to share their data as they generate and own a tremendous amount daily. However, this leads to the burden of consent management, which is too complex to manage granularly across different modalities, time horizons, and trust levels in data buyers.

   (d) Governments and non-profits are often prohibited from selling data for monetary gains.

2. **Lack of ease of sharing data:** Due to a lack of automated processes, digitization, access to data pre-processing pipelines, compatible data schemas, standardization across data sources and other forms of siloing of socially beneficial data, seamless data sharing is restricted. To summarize, these factors include:

   (a) Lack of digitization and lack of use cases

   (b) Lack of data standardization across multiple sources

   (c) Collection of data currently will likely cost more than the market price of data

   (d) Socially beneficial good data is locked away (e.g. with government, non-profits, hospitals, remote sensing data)

3. **Lack of trust:** Data sharing can also be impeded by market forces, the need for maintaining trade secrets, a competitive economy that impedes trust, and fear of losing control and accountability over future data usage for adversarial purposes. To summarize, these factors include cases when:

    (a) Data owner does not trust what the buyer will do with data in a computing environment

    (b) Data indirectly contains trade secrets of the data owner

    (c) Fear of adversarial future usage of shared data

4. **Regulations:** Data sharing is regulated for privacy, security, fairness and safety, and therefore, any data transactions for performing basic data analysis or for any advanced AI/ML use cases have to be aware of these constraints and be able to safely circumvent these friction points while also maintaining compliance with the law. To summarize:

    (a) In sectors such as health, finance and cybersecurity tightly governed by local, federal and international data-sharing regulations such as HIPAA, GDPR, COX, PCI, and SHIELD, we need a new strategy for safe data sharing.

    (b) Policies for inter and intra-organizational data sharing must be adhered to.

    (c) The origination of data may have country-specific regulations on usage. Therefore, international rules concerning the data provider and the data consumer must be adhered to, both concerning the data provider and consumer.

(d) There are policies where data cannot physically leave the premises of the data owners.

16.5 Governance and encouragement for a data market ecosystem

In addition, from the governance perspective, the following would be key to supporting the setup and sustaining of a good ecosystem for data markets.

(a) Need to support technological solution vs market solution vs policy-driven solutions.

(b) Data governance policies by studying existing legal/regulatory framework changes.

(c) Standardization of data sharing

(d) Setting up national ’nodes’ of servers for data exchange (like stock exchanges)

(e) ’Clean Data’ credits like ’clear air’ carbon credits

(f) Treat data as a labour (it’s from activity that creates value)

(g) Ethics and bias: self-certification as well as audits

References

[1] Abadi, M., Chu, A., Goodfellow, I., McMahan, H. B., Mironov, I., Talwar, K., & Zhang, L. (2016a). Deep learning with differential privacy. In *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security*, CCS 16 (pp. 308318). New York, NY, USA: Association for Computing Machinery.

[2] Abadi, M., Chu, A., Goodfellow, I., McMahan, H. B., Mironov, I., Talwar, K., & Zhang, L. (2016b). Deep learning with differential privacy. In *Proceedings of the 2016 ACM SIGSAC conference on computer and communications security* (pp. 308–318).

[3] Abadi, M., Chu, A., Goodfellow, I., McMahan, H. B., Mironov, I., Talwar, K., & Zhang, L. (2016c). Deep learning with differential privacy. *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security*.

[4] Abowd, J., Ashmead, R., Simson, G., Kifer, D., Leclerc, P., Machanavajjhala, A., & Sexton, W. (2019). *Census TopDown: Differentially private data, incremental schemas, and consistency with public knowledge*. Technical report, Technical Report. US Census Bureau.

[5] Abowd, J. M. (2018). The us census bureau adopts differential privacy. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining* (pp. 2867–2867).

[6] Acharya, J., Canonne, C. L., Freitag, C., Sun, Z., & Tyagi, H. (2021). Inference under information constraints iii: Local privacy constraints. *IEEE Journal on Selected Areas in Information Theory*, 2(1), 253–267.

[7] Acharya, J., Sun, Z., & Zhang, H. (2018). Differentially private testing of identity and closeness of discrete distributions. In *NeurIPS* (pp. 6879–6891).

[8] Ács, G., Melis, L., Castelluccia, C., & Cristofaro, E. D. (2017). Differentially private mixture of generative neural networks. *CoRR*, abs/1709.04514.

[9] Agarwal, N., Suresh, A. T., Yu, F., Kumar, S., & Mcmahan, H. B. (2018). cpsgd: Communication-efficient and differentially-private distributed sgd. *arXiv preprint arXiv:1805.10559*.

[10] Aggarwal, C. C. (2005). On k-anonymity and the curse of dimensionality. In *Proceedings of the 31st International Conference on Very Large Data Bases*, VLDB 05 (pp. 901909).: VLDB Endowment.

[11] Agrawal, N., Shamsabadi, A. S., Kusner, M. J., & Gascón, A. (2019). QUOTIENT: two-party secure neural network training and prediction. *CoRR*, abs/1907.03372.

[12] Ahmed, N., Michelin, R. A., Xue, W., Ruj, S., Malaney, R., Kanhere, S. S., Seneviratne, A., Hu, W., Janicke, H., & Jha, S. K. (2020). A survey of covid-19 contact tracing apps. *IEEE Access*, 8, 134577–134601.

[13] Akhtar, N. & Mian, A. (2018). Threat of adversarial attacks on deep learning in computer vision: A survey. *IEEE Access*, 6, 14410–14430.

[Albrecht et al.] Albrecht, M., Chase, M., Chen, H., Ding, J., Goldwasser, S., Gorbunov, S., Halevi, S., Hoffstein, J., Lauter, K., Lokam, S., Micciancio, D., Moody, D., Morrison, T., Sahai, A., & Vaikuntanathan, V. Homomorphic encryption standard. 2018.

[15] Alfke, D., Potts, D., Stoll, M., & Volkmer, T. (2018). Nfft meets krylov methods: Fast matrix-vector products for the graph laplacian of fully connected networks. *Frontiers in Applied Mathematics and Statistics*, 4, 61.

[16] Algaba, E., Bilbao, J. M., Van den Brink, R., & Jiménez-Losada, A. (2004). Cooperative games on antimatroids. *Discrete Mathematics*, 282(1-3), 1–15.

[17] Aliakbarpour, M., Diakonikolas, I., Kane, D., & Rubinfeld, R. (2019). Private testing of distributions via sample permutations. In *Thirty-third Conference on Neural Information Processing Systems (NeurIPS 2019)*.

[18] Aliakbarpour, M., Diakonikolas, I., & Rubinfeld, R. (2018). Differentially private identity and equivalence testing of discrete distributions. In *ICML*, volume 80 of *Proceedings of Machine Learning Research* (pp. 169–178).: PMLR.

[19] Alman, J. & Williams, V. V. (2021). A refined laser method and faster matrix multiplication. In *Proceedings of the 2021 ACM-SIAM Symposium on Discrete Algorithms (SODA)* (pp. 522–539).: SIAM.

[20] Alon, N., Matias, Y., & Szegedy, M. (1999). The space complexity of approximating the frequency moments. *Journal of Computer and system sciences*, 58(1), 137–147.

[21] Altuwaiyan, T., Hadian, M., & Liang, X. (2018). Epic: efficient privacy-preserving contact tracing for infection detection. In *2018 IEEE International Conference on Communications (ICC)* (pp. 1–6).: IEEE.

[22] Amin, K., Dick, T., Kulesza, A., Munoz, A., & Vassilvitskii, S. (2019). Differentially private covariance estimation. *Advances in Neural Information Processing Systems*, 32.

[23] Andoni, A., Do Ba, K., Indyk, P., & Woodruff, D. (2009). Efficient sketches for earth-mover distance, with applications. In *2009 50th Annual IEEE Symposium on Foundations of Computer Science* (pp. 324–330).: IEEE.

[24] Andrés, M. E., Bordenabe, N. E., Chatzikokolakis, K., & Palamidessi, C. (2013). Geo-indistinguishability: Differential privacy for location-based systems. In *Proceedings of the 2013 ACM SIGSAC conference on Computer & communications security* (pp. 901–914).

[25] Arandjelović, R., Gronat, P., Torii, A., Pajdla, T., & Sivic, J. (2016). NetVLAD: CNN architecture for weakly supervised place recognition. In *CVPR*.

[26] Archer, D., Chen, L., Cheon, J. H., Gilad-Bachrach, R., Hallman, R. A., Huang, Z., Jiang, X., Kumaresan, R., Malin, B. A., Sofia, H., Song, Y., & Wang, S. (2018). Applications of homomorphic encryption. Technical report.

[27] Armaan Bhojwani, Praneeth Vepakomma, R. R. (2010 (accessed December 7, 2014)). Python differential privacy. https://github.com/acheam0/python_dp?fbclid=IwAR3S1EbQhkXb2zWvrgoZQHS_hP3aWBrrVxZpggDsFqrrotA4jA5KxU0xFVg.

[28] Armerding, T. (2018). The 18 biggest data breaches of the 21st century. online accessed February 2020 https://www.csoonline.com/article/2130877/the-biggest-data-breaches-of-the-21st-century.html.

[29] Arora, R. & Upadhyay, J. (2019). Differentially private graph sparsification and applications. *Advances in neural information processing systems*.

[30] Arpit, D., Jastrzundefinedbski, S., Ballas, N., Krueger, D., Bengio, E., Kanwal, M. S., Maharaj, T., Fischer, A., Courville, A., Bengio, Y., & et al. (2017). A closer look at memorization in deep networks. In *Proceedings of the 34th International Conference on Machine Learning - Volume 70*, ICML17 (pp. 233242).: JMLR.org.

[31] Arrow, K. J. (1950). A difficulty in the concept of social welfare. *Journal of political economy*, 58(4), 328–346.

[32] Arrow, K. J. (1951). *Social choice and individual values*. Yale university press.

[33] Arsigny, V., Fillard, P., Pennec, X., & Ayache, N. (2006). Log-euclidean metrics for fast and simple calculus on diffusion tensors. *Magnetic Resonance in Medicine: An Official Journal of the International Society for Magnetic Resonance in Medicine*, 56(2), 411–421.

[34] Arsigny, V., Fillard, P., Pennec, X., & Ayache, N. (2007). Geometric means in a novel vector space structure on symmetric positive-definite matrices. *SIAM journal on matrix analysis and applications*, 29(1), 328–347.

[35] Asghar, H. J., Ding, M., Rakotoarivelo, T., Mrabet, S., & Kaafar, D. (2021). Differentially private release of datasets using gaussian copula. *Journal of Privacy and Confidentiality*, 10(2).

[36] Aslett, L. J. M., Esperança, P. M., & Holmes, C. C. (2015). Encrypted statistical machine learning: new privacy preserving methods.

[37] Ateniese, G., Felici, G., Mancini, L. V., Spognardi, A., Villani, A., & Vitali, D. (2013). Hacking smart machines with smarter ones: How to extract meaningful data from machine learning classifiers. *CoRR*, abs/1306.4447.

[38] Austin, L. M. (2005). Is consent the foundation of fair information practices? canadas experience under pipeda. *SSRN Electronic Journal*.

[39] Avdiukhin, D., Mitrović, S., Yaroslavtsev, G., & Zhou, S. (2019). Adversarially robust submodular maximization under knapsack constraints. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining* (pp. 148–156).

[40] Bagdasaryan, E., Poursaeed, O., & Shmatikov, V. (2019). Differential privacy has disparate impact on model accuracy. In *Advances in Neural Information Processing Systems* (pp. 15479–15488).

[Balla et al.] Balla, J., Vepakomma, P., & Raskar, R. Sniper gmms: Structured gaussian mixtures poison ml on large n small p data with high efficacy.

[42] Balle, B. & Wang, Y.-X. (2018). Improving the gaussian mechanism for differential privacy: Analytical calibration and optimal denoising. In *International Conference on Machine Learning* (pp. 394–403).: PMLR.

[Basser et al.] Basser, P. J., Mattiello, J., & Le Bihan, D. MR diffusion tensor spectroscopy and imaging. *Biophysical Journal*, 66(1), 259–67.

[44] Beaulieu-Jones, B. K., Yuan, W., Finlayson, S. G., & Wu, Z. S. (2018). Privacy-preserving distributed deep learning for clinical data. *CoRR*, abs/1812.01484.

[45] Belghazi, M. I., Baratin, A., Rajeswar, S., Ozair, S., Bengio, Y., Courville, A., & Hjelm, R. D. (2018). Mine: mutual information neural estimation. *arXiv preprint arXiv:1801.04062*.

[46] Belkin, M. & Niyogi, P. (2003). Laplacian eigenmaps for dimensionality reduction and data representation. *Neural computation*, 15(6), 1373–1396.

[47] Belkin, M. & Niyogi, P. (2005). Towards a theoretical foundation for laplacian-based manifold methods. In *International Conference on Computational Learning Theory* (pp. 486–500).: Springer.

[48] Belkin, M. & Niyogi, P. (2007). Convergence of laplacian eigenmaps. *Advances in Neural Information Processing Systems*, 19, 129.

[Bellafqira et al.] Bellafqira, R., Coatrieux, G., Genin, E., & Cozic, M. Technical report, preprint, 2018, title = Secure Multilayer Perceptron Based On Homomorphic Encryption, arXiv,.

[50] Ben-Or, M. & Linial, N. (1985). Collective coin flipping, robust voting schemes and minima of banzhaf values. *Proceedings of the 26th Annual IEEE Symposium on Foundations of Computer Science*, (pp. 408416).

[51] Benaloh, J. In *Proceedings of the Workshop on Selected Areas of Cryptography* (pp. 120–128).: lafourcade//PAPERS/FLA11.pdf, title = Dense probabilistic encryption, year = 1994.

[52] Benaloh, J. D. C. *Thesis, Yale University, Department of Computer Science, , title = Verifiable secret-ballot elections, volume = 1987, url = https://www.microsoft.com/en-us/research/wp-content/uploads/1987/01/thesis.pdf.*

[53] Berke, A., Bakker, M., Vepakomma, P., Raskar, R., Larson, K., & Pentland, A. (2020). Assessing disease exposure risk with location histories and protecting privacy: A cryptographic approach in response to a global pandemic. *arXiv preprint arXiv:2003.14412*.

[54] Bertran, M., Martinez, N., Papadaki, A., Qiu, Q., Rodrigues, M., Reeves, G., & Sapiro, G. (2019). Adversarially learned representations for information obfuscation and inference. In *International Conference on Machine Learning* (pp. 614–623).: PMLR.

[55] Beskorovajnov, W., Dörre, F., Hartung, G., Koch, A., Müller-Quade, J., & Strufe, T. (2020). Contra corona: Contact tracing against the coronavirus by bridging the centralized-decentralized divide for stronger privacy. *IACR Cryptol. ePrint Arch.*, 2020, 505.

[56] Bhowmick, A., Duchi, J., Freudiger, J., Kapoor, G., & Rogers, R. (2018a). Protection against reconstruction and its applications in private federated learning. *arXiv preprint arXiv:1812.00984*.

[57] Bhowmick, A., Duchi, J. C., Freudiger, J., Kapoor, G., & Rogers, R. (2018b). Protection against reconstruction and its applications in private federated learning. *ArXiv*, abs/1812.00984.

[58] Bian, A. A., Buhmann, J. M., Krause, A., & Tschiatschek, S. (2017). Guarantees for greedy maximization of non-submodular functions with applications. In *International conference on machine learning* (pp. 498–507).: PMLR.

[59] Biggio, B., Nelson, B., & Laskov, P. (2012). Poisoning attacks against support vector machines. In *Proceedings of the 29th International Coference on International Conference on Machine Learning*, ICML12 (pp. 14671474). Madison, WI, USA: Omnipress.

[60] Biswas, S., Dong, Y., Kamath, G., & Ullman, J. (2020). Coinpress: Practical private mean and covariance estimation. *Advances in Neural Information Processing Systems*, 33, 14475–14485.

[61] Blocki, J., Blum, A., Datta, A., & Sheffet, O. (2012). The johnson-lindenstrauss transform itself preserves differential privacy. In *2012 IEEE 53rd Annual Symposium on Foundations of Computer Science* (pp. 410–419).: IEEE.

[62] Boddeti, V. N. (2018). Secure face matching using fully homomorphic encryption. *ArXiv*, abs/1805.00577.

[63] Bogunovic, I., Mitrović, S., Scarlett, J., & Cevher, V. (2017). Robust submodular maximization: A non-uniform partitioning approach. In *International Conference on Machine Learning* (pp. 508–516).: PMLR.

[64] Bogunovic, I., Zhao, J., & Cevher, V. (2018). Robust maximization of non-submodular objectives. In *International Conference on Artificial Intelligence and Statistics* (pp. 890–899).: PMLR.

[65] Bolton, R. J. & Hand, D. J. (2002). Statistical fraud detection: A review. *Statist. Sci.*, 17(3), 235–255.

[66] Bonawitz, K., Ivanov, V., Kreuter, B., Marcedone, A., McMahan, H. B., Patel, S., Ramage, D., Segal, A., & Seth, K. (2016). Practical secure aggregation for federated learning on user-held data. *arXiv preprint arXiv:1611.04482*.

[67] Bonawitz, K., Ivanov, V., Kreuter, B., Marcedone, A., McMahan, H. B., Patel, S., Ramage, D., Segal, A., & Seth, K. (2017). Practical secure aggregation for privacy-preserving machine learning. In *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security*, CCS 17 (pp. 11751191). New York, NY, USA: Association for Computing Machinery.

[Boneh et al.] Boneh, D., Goh, E.-J., & Nissim, K. In *Springer* (pp. 325–341). dabo/papers/2dnf.pdf, title = Evaluating 2-DNF Formulas on Ciphertexts, Theory of Cryptography, year = 2006, url = https://crypto.stanford.edu/,.

[Borgnia et al.] Borgnia, E., Geiping, J., Cherepanova, V., Fowl, L. H., Gupta, A., Ghiasi, A., Huang, F., Goldblum, M., & Goldstein, T. Dp-instahide: Data augmentations provably enhance guarantees against dataset manipulations.

[70] Borgwardt, K. M., Gretton, A., Rasch, M. J., Kriegel, H.-P., Schölkopf, B., & Smola, A. J. (2006). Integrating structured biological data by kernel maximum mean discrepancy. *Bioinformatics*, 22(14), e49–e57.

[71] Bortolato, B., Ivanovska, M., Rot, P., Križaj, J., Terhörst, P., Damer, N., Peer, P., & Štruc, V. (2020). Learning privacy-enhancing face representations through feature disentanglement. In *2020 15th IEEE International Conference on Automatic Face and Gesture Recognition (FG 2020)* (pp. 495–502).: IEEE.

[72] Bost, R., Popa, R. A., Tu, S., & Goldwasser, S. (2014). Machine learning classification over encrypted data. *IACR Cryptology ePrint Archive*, 2014, 331.

[73] Boumal, N. (2020). An introduction to optimization on smooth manifolds. *Available online, May*, 3.

[74] Bourse, F., Minelli, M., Minihold, M., & Paillier, P. (2017). Fast homomorphic evaluation of deep discretized neural networks. In *CRYPTO*.

[Brakerski et al.] Brakerski, Z., Gentry, C., & Halevi, S. Packed ciphertexts in lwe-based homomorphic encryption. *Cryptology ePrint Archive Report*, 2012.

[76] Brakerski, Z., Gentry, C., & Vaikuntanathan, V. (2015). *Fully Homomorphic Encryption without Bootstrapping*. IACR Cryptology ePrint Archive.

[77] Braverman, V. & Ostrovsky, R. (2013). Generalizing the layering method of indyk and woodruff: Recursive sketches for frequency-based vectors on streams. In *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques* (pp. 58–70). Springer.

[78] Broder, A. & Mitzenmacher, M. (2004). Network applications of bloom filters: A survey. *Internet mathematics*, 1(4), 485–509.

[79] Broder, A. Z. (2000). Identifying and filtering near-duplicate documents. In *Annual Symposium on Combinatorial Pattern Matching* (pp. 1–10).: Springer.

[80] Bu, Z., Dong, J., Long, Q., & Su, W. (2019). Deep learning with gaussian differential privacy. *ArXiv*, abs/1911.11607.

[81] Bun, M. & Steinke, T. (2016). Concentrated differential privacy: Simplifications, extensions, and lower bounds. In *Theory of Cryptography: 14th International Conference, TCC 2016-B, Beijing, China, October 31-November 3, 2016, Proceedings, Part I* (pp. 635–658).: Springer.

[82] Cai, Y., Takala, V., & Pietikainen, M. (2010). Matching groups of people by covariance descriptor. In *2010 20th International Conference on Pattern Recognition* (pp. 2744–2747).: IEEE.

[83] Canella, C., Genkin, D., Giner, L., Gruss, D., Lipp, M., Minkin, M., Moghimi, D., Piessens, F., Schwarz, M., Sunar, B., Van Bulck, J., & Yarom, Y. (2019). Fallout: Leaking data on meltdown-resistant cpus. In *Proceedings of the ACM SIGSAC Conference on Computer and Communications Security (CCS)*: ACM.

[84] Canonne, C. (2021). What is delta, and what difference does it make? DifferentialPrivacy.org. https://differentialprivacy.org/flavoursofdelta/.

[85] Canonne, C. L., Kamath, G., & Steinke, T. (2020). The discrete gaussian for differential privacy. *Advances in Neural Information Processing Systems*, 33, 15676–15688.

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

[87] Caseiro, R., Henriques, J. F., Martins, P., & Batista, J. (2012). Semi-intrinsic mean shift on riemannian manifolds. In *European conference on computer vision* (pp. 342–355).: Springer.

[88] Ceballos, I., Mugica, E., Roman, A., Singh, A., Vepakomma, P., & Raskar, R. (2020a). Towards split learning at scale: System design. In *Workshop on MLOps Systems*.

[89] Ceballos, I., Sharma, V., Mugica, E., Singh, A., Roman, A., Vepakomma, P., & Raskar, R. (2020b). Splitnn-driven vertical partitioning. *arXiv preprint arXiv:2008.04137*.

[90] Chabanne, H., de Wargny, A., Milgram, J., Morel, C., & Prouff, E. (2017). Privacy-preserving classification on deep neural network. *IACR Cryptology ePrint Archive*, 2017, 35.

[91] Chajda, I., Halaš, R., & Kühr, J. (2007). *Semilattice structures*, volume 30. Heldermann Lemgo.

[92] Chakraborty, A., Alam, M., Dey, V., Chattopadhyay, A., & Mukhopadhyay, D. (2018a). Adversarial attacks and defences: A survey. *CoRR*, abs/1810.00069.

[93] Chakraborty, A., Alam, M., Dey, V., Chattopadhyay, A., & Mukhopadhyay, D. (2018b). Adversarial attacks and defences: A survey. *arXiv preprint arXiv:1810.00069*.

[94] Chamikara, M. A. P., Bertok, P., Khalil, I., Liu, D., & Camtepe, S. (2020). Privacy preserving face recognition utilizing differential privacy. *Computers & Security*, 97, 101951.

[95] Chan, J., Gollakota, S., Horvitz, E., Jaeger, J., Kakade, S., Kohno, T., Langford, J., Larson, J., Singanamalla, S., Sunshine, J., et al. (2020). Pact: Privacy sensitive protocols and mechanisms for mobile contact tracing. *arXiv preprint arXiv:2004.03544*.

[96] Chanyaswad, T., Liu, C., & Mittal, P. (2019). Ron-gauss: Enhancing utility in non-interactive private data release. *Proceedings on Privacy Enhancing Technologies*, 2019(1), 26–46.

[97] Charalambides, N., Pilanci, M., & Hero, A. O. (2022). Secure linear mds coded matrix inversion. In *2022 58th Annual Allerton Conference on Communication, Control, and Computing (Allerton)* (pp. 1–8).: IEEE.

[98] Charles, I., J. & Christopher, T. (2018). Nonrivalry and the economics of data. *2018 Meeting Papers*, (477).

[Chase et al.] Chase, M., Chen, H., Ding, J., Goldwasser, S., Gorbunov, S., Hoffstein, J., Lauter, K., Lokam, S., Moody, D., Morrison, T., Sahai, A., & Vaikuntanathan, V. Security of homomorphic encryption. 2018.

[100] Chase, M., Gilad-Bachrach, R., Laine, K., Lauter, K. E., & Rindal, P. (2017). Private collaborative neural network learning. *IACR Cryptology ePrint Archive*, 2017, 762.

[101] Chatzikokolakis, K., Palamidessi, C., & Stronati, M. (2015). Location privacy via geo-indistinguishability. *ACM SIGLOG News*, 2(3), 46–69.

[102] Chaudhuri, A. & Hu, W. (2019). A fast algorithm for computing distance correlation. *Computational Statistics & Data Analysis*.

[103] Chaudhuri, K. & Monteleoni, C. (2009). Privacy-preserving logistic regression. In D. Koller, D. Schuurmans, Y. Bengio, & L. Bottou (Eds.), *Advances in Neural Information Processing Systems 21* (pp. 289–296). Curran Associates, Inc.

[104] Chaudhuri, K., Monteleoni, C., & Sarwate, A. D. (2011). Differentially private empirical risk minimization. *J. Mach. Learn. Res.*, 12(null), 10691109.

[105] Chaudhuri, K., Sarwate, A. D., & Sinha, K. (2013). A near-optimal algorithm for differentially-private principal components. *Journal of Machine Learning Research*, 14.

[106] Chen, B.-R. & Hu, Y.-C. (2020). Blindsignedid: Mitigate denial-of-service attacks on digital contact tracing. *arXiv preprint arXiv:2008.09351*.

[107] Chen, W., Liu, Y., Wang, W., Bakker, E., Georgiou, T., Fieguth, P., Liu, L., & Lew, M. S. (2021). Deep image retrieval: A survey. *arXiv preprint arXiv:2101.11282*.

[108] Chen, Y., Epperly, E. N., Tropp, J. A., & Webber, R. J. (2022). Randomly pivoted cholesky: Practical approximation of a kernel matrix with few entry evaluations.

[109] Cheng, K., Fan, T., Jin, Y., Liu, Y., Chen, T., & Yang, Q. (2019). Secureboost: A lossless federated learning framework. *arXiv preprint arXiv:1901.08755*.

[110] Cheng, L., Liu, F., & Yao, D. D. (2017). Enterprise data breach: causes, challenges, prevention, and future directions. *WIREs Data Mining and Knowledge Discovery*, 7(5), e1211.

[111] Chevallier, S., Kalunga, E. K., Barthélemy, Q., & Monacelli, E. (2021). Review of riemannian distances and divergences, applied to ssvep-based bci. *Neuroinformatics*, 19(1), 93–106.

[112] Chierichetti, F., Dasgupta, A., & Kumar, R. (2020). On additive approximate submodularity. *arXiv e-prints*, (pp. arXiv–2010).

[113] Chillotti, I., Gama, N., Georgieva, M., & Izabachène, M. Faster fully homomorphic encryption: Bootstrapping[76] in less than 0.1 seconds. *Cryptology ePrint Archive Report*, 2016.

[114] Chillotti, I., Gama, N., Georgieva, M., & Izabachène, M. Improving tfhe: Faster packed homomorphic operations and efficient circuit bootstrapping. *Cryptology ePrint Archive Report*, 2017.

[115] Cho, H., Ippolito, D., & Yu, Y. W. (2020). Contact tracing mobile apps for covid-19: Privacy considerations and related trade-offs. *arXiv preprint arXiv:2003.11511*.

[116] Choromanska, A., Choromanski, K., Jagannathan, G., & Monteleoni, C. (2016). Differentially-private learning of low dimensional manifolds. *Theoretical Computer Science*, 620, 91–104.

[117] Chowdhury, A. R., Wang, C., He, X., Machanavajjhala, A., & Jha, S. (2019). Cryptepsilon: Crypto-assisted differential privacy on untrusted servers. *arXiv preprint arXiv:1902.07756*.

[118] Churbanov, A. G. & Vabishchevich, P. N. (2019). Numerical solution of boundary value problems for the eikonal equation in an anisotropic medium. *Journal of Computational and Applied Mathematics*, 362, 55–67.

[119] Cirujeda, P. & Binefa, X. (2014). 4dcov: A nested covariance descriptor of spatio-temporal features for gesture recognition in depth sequences. In *2014 2nd International Conference on 3D Vision*, volume 1 (pp. 657–664).: IEEE.

[120] Cirujeda, P., Cid, Y. D., Müller, H., Rubin, D., Aguilera, T. A., Loo, B. W., Diehn, M., Binefa, X., & Depeursinge, A. (2016). A 3-d riesz-covariance texture model for prediction of nodule recurrence in lung ct. *IEEE transactions on medical imaging*, 35(12), 2620–2630.

[121] Clanuwat, T., Bober-Irizar, M., Kitamoto, A., Lamb, A., Yamamoto, K., & Ha, D. (2018). Deep learning for classical japanese literature. *arXiv preprint arXiv:1812.01718*.

[122] Cohen, A. & Nissim, K. (2020). Towards formalizing the gdprs notion of singling out. *Proceedings of the National Academy of Sciences*, 117(15), 8344–8352.

[123] Coifman, R. R. & Lafon, S. (2006). Diffusion maps. *Applied and computational harmonic analysis*, 21(1), 5–30.

[124] Conforti, M. & Laurent, M. (1989). On the geometric structure of independence systems. *Mathematical programming*, 45(1), 255–277.

[125] Cormode, G. & Garofalakis, M. (2007). Sketching probabilistic data streams. In *Proceedings of the 2007 ACM SIGMOD international conference on Management of data* (pp. 281–292).

[126] Cormode, G., Jha, S., Kulkarni, T., Li, N., Srivastava, D., & Wang, T. (2018a). Privacy at scale: Local differential privacy in practice. In *Proceedings of the 2018 International Conference on Management of Data* (pp. 1655–1658).

[127] Cormode, G., Kulkarni, T., & Srivastava, D. (2018b). Marginal release under local differential privacy. In *Proceedings of the 2018 International Conference on Management of Data* (pp. 131–146).

[128] Cormode, G. & Muthukrishnan, S. (2005). An improved data stream summary: the count-min sketch and its applications. *Journal of Algorithms*, 55(1), 58–75.

[129] Costan, V. & Devadas, S. (2016). Intel sgx explained. *IACR Cryptology ePrint Archive*, 2016, 86.

[130] Cranko, Z., Menon, A., Nock, R., Ong, C. S., Shi, Z., & Walder, C. (2019). Monge blunts bayes: Hardness results for adversarial training. In *International Conference on Machine Learning* (pp. 1406–1415).: PMLR.

[131] Cristianini, N., Shawe-Taylor, J., Elisseeff, A., & Kandola, J. (2001). On kernel-target alignment. *Advances in neural information processing systems*, 14.

[132] Cummings, R., Gupta, V., Kimpara, D., & Morgenstern, J. (2019). On the compatibility of privacy and fairness. In *Adjunct Publication of the 27th Conference on User Modeling, Adaptation and Personalization* (pp. 309–315).

[133] Damgård, I. & Mads Jurik, A. G. (2001). a simplification and some applications of paillier's probabilistic public-key system, public key cryptography, international workshop on public key cryptography, springer. *pp*, (pp. 119–136).

[134] Das, A., Dasgupta, A., & Kumar, R. (2012). Selecting diverse features via spectral regularization. *Advances in neural information processing systems*, 25, 1583–1591.

[135] Das, A. & Kempe, D. (2018). Approximate submodularity and its applications: Subset selection, sparse approximation and dictionary selection. *The Journal of Machine Learning Research*, 19(1), 74–107.

[136] Deng, J., Dong, W., Socher, R., Li, L.-J., Li, K., & Fei-Fei, L. (2009). Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition* (pp. 248–255).: Ieee.

[137] Diakonikolas, I., Hardt, M., & Schmidt, L. (2015). Differentially private learning of structured discrete distributions. In *NIPS* (pp. 2566–2574).

[138] Diaz, M., Wang, H., Calmon, F. P., & Sankar, L. (2018). On the robustness of information-theoretic privacy measures and mechanisms. *CoRR*, abs/1811.06057.

[139] Diba, A., Sharma, V., & Van Gool, L. (2017). Deep temporal linear encoding networks. In *CVPR*.

[140] Dietrich, B. L. (1989). Matroids and antimatroidsa survey. *Discrete Mathematics*, 78(3), 223–237.

[141] Do Carmo, M. P. & Flaherty Francis, J. (1992). *Riemannian geometry*, volume 6. Springer.

[142] Dong, J., Roth, A., & Su, W. (2019a). Gaussian differential privacy. *ArXiv*, abs/1905.02383.

[143] Dong, J., Roth, A., & Su, W. J. (2019b). Gaussian differential privacy. *arXiv preprint arXiv:1905.02383*.

[144] Donoho, D. L. & Grimes, C. (2003). Hessian eigenmaps: Locally linear embedding techniques for high-dimensional data. *Proceedings of the National Academy of Sciences*, 100(10), 5591–5596.

[145] Dosovitskiy, A. & Brox, T. (2016). Inverting visual representations with convolutional networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 4829–4837).

[146] Dowlin, N., Gilad-Bachrach, R., Laine, K., Lauter, K., Naehrig, M., & Wernsing, J. (2016a). Cryptonets: Applying neural networks to encrypted data with high throughput and accuracy. In *Proceedings of the 33rd International Conference on International Conference on Machine Learning - Volume 48*, ICML'16 (pp. 201–210).: JMLR.org.

[147] Dowlin, N., Gilad-Bachrach, R., Laine, K., Lauter, K., Naehrig, M., & Wernsing, J. (2016b). *CryptoNets: Applying Neural Networks to Encrypted Data with High Throughput and Accuracy*. Technical Report MSR-TR-2016-3.

[148] Dragomir, S. S. & Gluscevic, V. (2000). Some inequalities for the kullback-leibler and x^2- distances in information theory and applications. *RGMIA research report collection*, 3(2), 199–210.

[149] Dubey, S. R. (2020). A decade survey of content based image retrieval using deep learning. *arXiv preprint arXiv:2012.00641*.

[150] Dudley, R. M. (2010). Universal donsker classes and metric entropy. In *Selected Works of RM Dudley* (pp. 345–365). Springer.

[151] Durrieu, J.-L., Thiran, J.-P., & Kelly, F. (2012). Lower and upper bounds for approximation of the kullback-leibler divergence between gaussian mixture models. In *2012 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)* (pp. 4833–4836).: Ieee.

[152] Dwork, C. (2006a). Differential privacy. In *International Colloquium on Automata, Languages, and Programming* (pp. 1–12).: Springer.

[153] Dwork, C. (2006b). Differential privacy. In M. Bugliesi, B. Preneel, V. Sassone, & I. Wegener (Eds.), *Automata, Languages and Programming* (pp. 1–12). Berlin, Heidelberg: Springer Berlin Heidelberg.

[154] Dwork, C. (2008). Differential privacy: A survey of results. In *International conference on theory and applications of models of computation* (pp. 1–19).: Springer.

[155] Dwork, C., Kenthapadi, K., McSherry, F., Mironov, I., & Naor, M. (2006a). Our data, ourselves: Privacy via distributed noise generation. In *EUROCRYPT* (pp. 486–503).: Springer Berlin Heidelberg.

[156] Dwork, C. & Lei, J. (2009). Differential privacy and robust statistics. In *Proceedings of the forty-first annual ACM symposium on Theory of computing* (pp. 371–380).

[157] Dwork, C., McSherry, F., Nissim, K., & Smith, A. (2006b). Calibrating noise to sensitivity in private data analysis. In *Proceedings of the Third Conference on Theory of Cryptography*, TCC'06 (pp. 265–284). Berlin, Heidelberg: Springer-Verlag.

[158] Dwork, C., McSherry, F., Nissim, K., & Smith, A. (2006c). Calibrating noise to sensitivity in private data analysis. In *Theory of cryptography conference* (pp. 265–284).: Springer.

[159] Dwork, C., McSherry, F., Nissim, K., & Smith, A. (2006d). Calibrating noise to sensitivity in private data analysis. In S. Halevi & T. Rabin (Eds.), *Theory of Cryptography* (pp. 265–284). Berlin, Heidelberg: Springer Berlin Heidelberg.

[160] Dwork, C., Roth, A., et al. (2014). The algorithmic foundations of differential privacy. *Found. Trends Theor. Comput. Sci.*, 9(3-4), 211–407.

[161] Dwork, C. & Smith, A. (2010). Differential privacy for statistics: What we know and what we want to learn. *Journal of Privacy and Confidentiality*, 1(2).

[162] Dwork, C., Smith, A., Steinke, T., & Ullman, J. (2017). Exposed! a survey of attacks on private data. *Annual Review of Statistics and Its Application*, 4, 61–84.

[163] DOliveira, R. G., El Rouayheb, S., & Karpuk, D. (2020). Gasp codes for secure distributed matrix multiplication. *IEEE Transactions on Information Theory*, 66(7), 4038–4050.

[164] Edelmann, D., Richards, D., & Vogel, D. (2020). The distance standard deviation. *Annals of Statistics*.

[165] Edmonds, J. (2003). Submodular functions, matroids, and certain polyhedra. In *Combinatorial OptimizationEureka, You Shrink!* (pp. 11–26). Springer.

[166] ElGamal, T. & and, A. P. K. C. (1985). and a signature scheme based on discrete logarithms, ieee transactions in information theory. 31, 469–472.

[167] Elmehdwi, Y., Samanthula, B. K., & Jiang, W. (2014). Secure k-nearest neighbor query over encrypted data in outsourced environments. In *2014 IEEE 30th International Conference on Data Engineering* (pp. 664–675).: IEEE.

[168] Elthakeb, A. T., Pilligundla, P., Mireshghallah, F., Cloninger, A., & Esmaeilzadeh, H. (2020). Divide and conquer: Leveraging intermediate feature representations for quantized training of neural networks. In *International Conference on Machine Learning* (pp. 2880–2891).: PMLR.

[169] Erlingsson, Ú., Pihur, V., & Korolova, A. (2014). Rappor: Randomized aggregatable privacy-preserving ordinal response. In *Proceedings of the 2014 ACM SIGSAC conference on computer and communications security* (pp. 1054–1067).

[170] Evren, I. S., Vepakomma, P., & Raskar, R. (2022). The privacy-welfare tradeoff: Effects of differential privacy on influence & welfare in social choice. *arXiv preprint arXiv:2201.10115*.

[171] Evren, S. & Vepakomma, P. (2022). Effects of privacy-inducing noise on welfare and influence of referendum systems. *arXiv e-prints*, (pp. arXiv–2201).

[172] Fan, J. & Lv, J. (2018). Sure independence screening. *Wiley StatsRef: Statistics Reference Online*.

[Fan & Vercauteren] Fan, J. & Vercauteren, F. Somewhat practical fully homomorphic encryption. *Cryptology ePrint Archive Report*, 2012.

[174] Feige, U., Mirrokni, V. S., & Vondrák, J. (2011). Maximizing non-monotone submodular functions. *SIAM Journal on Computing*, 40(4), 1133–1153.

[175] Fitzsimons, J. K., Mantri, A., Pisarczyk, R., Rainforth, T., & Zhao, Z. (2020). A note on blind contact tracing at scale with applications to the covid-19 pandemic. *arXiv preprint arXiv:2004.05116*.

[176] Fletcher, P. T., Lu, C., Pizer, S. M., & Joshi, S. (2004). Principal geodesic analysis for the study of nonlinear statistics of shape. *IEEE transactions on medical imaging*, 23(8), 995–1005.

[177] Fletcher, T. (2011). Geodesic regression on riemannian manifolds. In *Proceedings of the Third International Workshop on Mathematical Foundations of Computational Anatomy-Geometrical and Statistical Methods for Modelling Biological Shape Variability* (pp. 75–86).

[178] Fraleigh, J. B. & First, A. (2002). Course in abstract algebra. 7.

[179] Fréchet, M. (1948). Les éléments aléatoires de nature quelconque dans un espace distancié. In *Annales de l'institut Henri Poincaré*, volume 10 (pp. 215–310).

[180] Fredrikson, M., Jha, S., & 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 (pp. 13221333). New York, NY, USA: Association for Computing Machinery.

[181] Fredrikson, M., Lantz, E., Jha, S., Lin, S., Page, D., & Ristenpart, T. (2014). Privacy in pharmacogenetics: An end-to-end case study of personalized warfarin dosing. In *Proceedings of the 23rd USENIX Conference on Security Symposium*, SEC14 (pp. 1732). USA: USENIX Association.

[182] Frome, A., Cheung, G., Abdulkader, A., Zennaro, M., Wu, B., Bissacco, A., Adam, H., Neven, H., & Vincent, L. (2009). Large-scale privacy protection in google street view. *2009 IEEE 12th International Conference on Computer Vision*, (pp. 2373–2380).

[183] Fujishige, S. (2005). *Submodular functions and optimization*. Elsevier.

[184] Gaboardi, M., Lim, H., Rogers, R., & Vadhan, S. (2016). Differentially private chi-squared hypothesis testing: Goodness of fit and independence testing. In *International conference on machine learning* (pp. 2111–2120).: PMLR.

[185] Galbraith, S. D. (2002). Elliptic curve paillier schemes. *Journal of Cryptology*, 15(2), 129–138.

[186] Ganju, K., Wang, Q., Yang, W., Gunter, C. A., & Borisov, N. (2018). Property inference attacks on fully connected neural networks using permutation invariant representations. In *Proceedings of the 2018 ACM SIGSAC Conference on Computer and Communications Security*, CCS 18 (pp. 619633). New York, NY, USA: Association for Computing Machinery.

[187] Gao, L., Fan, Y., Lv, J., & Shao, Q.-M. (2021). Asymptotic distributions of high-dimensional distance correlation inference. *The Annals of Statistics*, 49(4), 1999–2020.

[188] Gao, Y., Beijbom, O., Zhang, N., & Darrell, T. (2016). Compact bilinear pooling. In *CVPR*.

[189] Garfinkel, S., Abowd, J. M., & Martindale, C. (2018). Understanding database reconstruction attacks on public data: These attacks on statistical databases are no longer a theoretical danger. *Queue*, 16(5), 28–53.

[190] Garman, M. B. & Kamien, M. I. (1968). The paradox of voting: Probability calculations. *Behavioral Science*, 13(4), 306–316.

[191] Gascón, A., Schoppmann, P., Balle, B., Raykova, M., Doerner, J., Zahur, S., & Evans, D. (2016). Secure linear regression on vertically partitioned datasets. *IACR Cryptol. ePrint Arch.*, 2016, 892.

[192] Gentry, C. (2009). Fully homomorphic encryption using ideal lattices. In *In Proc. STOC* (pp. 169–178).

[193] Gentry, C., Sahai, A., & Waters, B. (2013). *Homomorphic Encryption from Learning with Errors: Conceptually Simpler*. Attribute Based, Advances in Cryptology CRYPTO, Springer: Asymptotically Faster.

[194] Geyer, R. C., Klein, T., & Nabi, M. (2017). Differentially private federated learning: A client level perspective. *CoRR*, abs/1712.07557.

[195] Ghorbani, A. & Zou, J. (2019). Data shapley: Equitable valuation of data for machine learning. In *International conference on machine learning* (pp. 2242–2251).: PMLR.

[196] Gibbard, A. (1973). Manipulation of voting schemes: A general result. *Econometrica*, 41(4), 587–601.

[197] Giné, E., Koltchinskii, V., et al. (2006). Empirical graph laplacian approximation of laplace–beltrami operators: Large sample results. In *High dimensional probability* (pp. 238–259). Institute of Mathematical Statistics.

[198] Goddard, M. (2017). The eu general data protection regulation (gdpr): European regulation that has a global impact. *International Journal of Market Research*, 59(6), 703705.

[199] Goldwasser, S. & Micali, S. (1982). Probabilistic encryption & how to play mental poker keeping secret all partial information. In *Proceedings of the Fourteenth Annual ACM Symposium on Theory of Computing* (pp. 365–377).: ACM.

[200] Goldwasser, S. & Micali, S. (1984). Probabilistic encryption. *Journal of computer and system sciences*, 28(2), 270–299.

[201] Gondara, L. & Wang, K. (2020). Differentially private small dataset release using random projections. In *Conference on Uncertainty in Artificial Intelligence* (pp. 639–648).: PMLR.

[202] Graepel, T., Lauter, K., & Naehrig, M. (2013). Ml confidential: Machine learning on encrypted data. In T. Kwon, M.-K. Lee, & D. Kwon (Eds.), *Information Security and Cryptology – ICISC 2012* (pp. 1–21). Berlin, Heidelberg: Springer Berlin Heidelberg.

[203] Greiner, A. L., Angelo, K. M., McCollum, A. M., Mirkovic, K., Arthur, R., & Angulo, F. J. (2015). Addressing contact tracing challengescritical to halting ebola virus disease transmission. *International Journal of Infectious Diseases*, 41, 53–55.

[204] Gretton, A., Bousquet, O., Smola, A., & Schölkopf, B. (2005a). Measuring statistical dependence with hilbert-schmidt norms. In *International conference on algorithmic learning theory* (pp. 63–77).: Springer.

[205] Gretton, A., Herbrich, R., Smola, A., Bousquet, O., Schölkopf, B., et al. (2005b). Kernel methods for measuring independence.

[206] Groce, A., Rindal, P., & Rosulek, M. (2019). Cheaper private set intersection via differentially private leakage. *Proceedings on Privacy Enhancing Technologies*, 2019(3), 6–25.

[Group & by Year and] Group, C. & by Year and, H. F. L. *, volume = 2018, url = https://compliancy-group.com/hipaa-fines-directory-year/.*

[208] Guilbaud, G.-T. (2012). Les théories de l'intérêt général et le problème logique de l'agrégation. *Revue économique*, 63(4), 659–720.

[209] Gupta, O. & Raskar, R. (2018a). Distributed learning of deep neural network over multiple agents. *J. Netw. Comput. Appl.*, 116, 1–8.

[210] Gupta, O. & Raskar, R. (2018b). Distributed learning of deep neural network over multiple agents. *arXiv preprint arXiv:1810.06060*.

[211] Hajri, H., Ilea, I., Said, S., Bombrun, L., & Berthoumieu, Y. (2016). Riemannian laplace distribution on the space of symmetric positive definite matrices. *Entropy*, 18(3), 98.

[212] Hamm, J. (2017). Minimax filter: Learning to preserve privacy from inference attacks. *Journal of Machine Learning Research*, 18(129), 1–31.

[213] Hamm, J., Cao, P., & Belkin, M. (2016). Learning privately from multiparty data. *CoRR*, abs/1602.03552.

[214] Han, Q. & Shen, Y. (2021). Generalized kernel distance covariance in high dimensions: non-null clts and power universality. *arXiv preprint arXiv:2106.07725*.

[215] Harchaoui, Z., Bach, F., Cappe, O., & Moulines, E. (2013). Kernel-based methods for hypothesis testing: A unified view. *IEEE Signal Processing Magazine*, 30(4), 87–97.

[216] Hatke, G. F., Montanari, M., Appadwedula, S., Wentz, M., Meklenburg, J., Ivers, L., Watson, J., & Fiore, P. (2020). Using bluetooth low energy (ble) signal strength estimation to facilitate contact tracing for covid-19. *arXiv preprint arXiv:2006.15711*.

[217] Hay, M., Elagina, L., & Miklau, G. (2017). *Differentially Private Rank Aggregation*, (pp. 669–677).

[218] Hayes, J., Melis, L., Danezis, G., & Cristofaro, E. D. (2017). LOGAN: evaluating privacy leakage of generative models using generative adversarial networks. *CoRR*, abs/1705.07663.

[219] He, K., Zhang, X., Ren, S., & Sun, J. (2015a). Deep residual learning for image recognition. *CoRR*, abs/1512.03385.

[220] He, K., Zhang, X., Ren, S., & Sun, J. (2015b). Deep residual learning for image recognition. *arXiv preprint arXiv:1512.03385*.

[221] He, Z., Zhang, T., & Lee, R. B. (2019). Model inversion attacks against collaborative inference. In *Proceedings of the 35th Annual Computer Security Applications Conference*, ACSAC 19 (pp. 148162). New York, NY, USA: Association for Computing Machinery.

[222] Helgason, S. (1979). *Differential geometry, Lie groups, and symmetric spaces*. Academic press.

[223] Heller, R., Heller, Y., Kaufman, S., Brill, B., & Gorfine, M. (2016). Consistent distribution-free k-sample and independence tests for univariate random variables. *The Journal of Machine Learning Research*, 17(1), 978–1031.

[224] Hesamifard, E., Takabi, H., & Ghasemi, M. (2017). Cryptodl: Deep neural networks over encrypted data. *CoRR*, abs/1711.05189.

[225] Hillinger, C. (2005). The case for utilitarian voting. *Homo Oeconomicus*, 22(3), 295–321.

[226] Hofmeister, C., Bitar, R., Xhemrishi, M., & Wachter-Zeh, A. (2022). Secure private and adaptive matrix multiplication beyond the singleton bound. *IEEE Journal on Selected Areas in Information Theory*, 3(2), 275–285.

[227] Homer, N., Szelinger, S., Redman, M., Duggan, D., Tembe, W., Muehling, J., Pearson, J. V., Stephan, D. A., Nelson, S. F., & Craig, D. W. (2008). Resolving individuals contributing trace amounts of dna to highly complex mixtures using high-density snp genotyping microarrays. *PLoS genetics*, 4(8).

[228] Hong, S., Yang, H., Yoon, Y., Cho, T., & Lee, J. (2021). Chebyshev polynomial codes: Task entanglement-based coding for distributed matrix multiplication. In *International Conference on Machine Learning* (pp. 4319–4327).: PMLR.

[229] Horel, T. & Singer, Y. (2016). Maximization of approximately submodular functions. In *NIPS*, volume 16 (pp. 3045–3053).

[230] Horiguchi, S. & Miranker, W. L. (1989). A parallel algorithm for finding the maximum value. *Parallel computing*, 10(1), 101–108.

[231] Horowitz, E. & Sahni, S. (1978). Fundamentals of computer algorithms.

[232] Hsu, H., Asoodeh, S., & Calmon, F. d. P. (2019a). Obfuscation via information density estimation. *arXiv preprint arXiv:1910.08109*.

[233] Hsu, H. L., Asoodeh, S., & du Pin Calmon, F. (2019b). Obfuscation via information density estimation. *ArXiv*, abs/1910.08109.

[234] Huang, C. & Huo, X. (2017). A statistically and numerically efficient independence test based on random projections and distance covariance. *arXiv preprint arXiv:1701.06054*.

[235] Huang, C., Kairouz, P., Chen, X., Sankar, L., & Rajagopal, R. (2017). Context-aware generative adversarial privacy. *CoRR*, abs/1710.09549.

[236] Huckemann, S., Hotz, T., & Munk, A. (2010). Intrinsic shape analysis: Geodesic pca for riemannian manifolds modulo isometric lie group actions. *Statistica Sinica*, (pp. 1–58).

[237] Hunter, D. R. & Lange, K. (2004). A tutorial on mm algorithms. *The American Statistician*, 58(1), 30–37.

[238] Huo, X. & Székely, G. J. (2016). Fast computing for distance covariance. *Technometrics*, 58(4), 435–447.

[239] Huo, Z., Gu, B., & Huang, H. (2018). Training neural networks using features replay. *arXiv preprint arXiv:1807.04511*.

[240] Hussein, M. E., Torki, M., Gowayyed, M. A., & El-Saban, M. (2013). Human action recognition using a temporal hierarchy of covariance descriptors on 3d joint locations. In *Twenty-third international joint conference on artificial intelligence*.

[241] Imtiaz, H. & Sarwate, A. D. (2018). Distributed differentially private algorithms for matrix and tensor factorization. *IEEE journal of selected topics in signal processing*, 12(6), 1449–1464.

[inc.] inc., A. Arm trustzone technology. online accessed February 2020 https://developer.arm.com/ip-products/security-ip/trustzone.

[243] Indyk, P. (2007). Sketching, streaming and sublinear-space algorithms. *Graduate course notes, available at*, 33, 617.

[244] Ioffe, S. (2010). Improved consistent sampling, weighted minhash and l1 sketching. In *2010 IEEE International Conference on Data Mining* (pp. 246–255).: IEEE.

[Ishai & Paskin] Ishai, Y. & Paskin, A. Evaluating branching programs on encrypted data, theory of cryptography, springer. *pp*, 2007, 575–594.

[246] Iyengar, R., Near, J. P., Song, D., Thakkar, O., Thakurta, A., & Wang, L. (2019). Towards practical differentially private convex optimization. In *2019 IEEE Symposium on Security and Privacy (SP)* (pp. 299–316).: IEEE.

[247] Iyer, R. (2019). A unified framework of robust submodular optimization. *arXiv preprint arXiv:1906.06393*.

[248] Iyer, R. & Bilmes, J. (2013). Submodular optimization with submodular cover and submodular knapsack constraints. *arXiv preprint arXiv:1311.2106*.

[249] Izzo, Z., Smart, M. A., Chaudhuri, K., & Zou, J. (2020). Approximate data deletion from machine learning models: Algorithms and evaluations. *ArXiv*, abs/2002.10077.

[250] Jagielski, M., Ullman, J., & Oprea, A. (2020). Auditing differentially private machine learning: How private is private sgd? *ArXiv*, abs/2006.07709.

[251] Jayaraman, B. & Evans, D. (2019). Evaluating differentially private machine learning in practice. In *28th {USENIX} Security Symposium ({USENIX} Security 19)* (pp. 1895–1912).

[252] Jayasumana, S., Hartley, R., Salzmann, M., Li, H., & Harandi, M. (2015). Kernel methods on riemannian manifolds with gaussian rbf kernels. *IEEE transactions on pattern analysis and machine intelligence*, 37(12), 2464–2477.

[253] Jia, R., Dao, D., Wang, B., Hubis, F. A., Hynes, N., Gürel, N. M., Li, B., Zhang, C., Song, D., & Spanos, C. J. (2019). Towards efficient data valuation based on the shapley value. In *The 22nd International Conference on Artificial Intelligence and Statistics* (pp. 1167–1176).: PMLR.

[254] Jin, J., McMurtry, E., Rubinstein, B. I., & Ohrimenko, O. (2022). Are we there yet? timing and floating-point attacks on differential privacy systems. In *2022 IEEE Symposium on Security and Privacy (SP)* (pp. 473–488).: IEEE.

[255] Jitkrittum, W., Szabó, Z., & Gretton, A. (2016). An adaptive test of independence with analytic kernel embeddings. *arXiv preprint arXiv:1610.04782*.

[256] Jitkrittum, W., Xu, W., Szabó, Z., Fukumizu, K., & Gretton, A. (2017). A linear-time kernel goodness-of-fit test. *Advances in Neural Information Processing Systems*, 30.

[257] Jones, P. W., Maggioni, M., & Schul, R. (2008). Manifold parametrizations by eigenfunctions of the laplacian and heat kernels. *Proceedings of the National Academy of Sciences*, 105(6), 1803–1808.

[258] Joseph, M., Mao, J., Neel, S., & Roth, A. (2019). The role of interactivity in local differential privacy. In *2019 IEEE 60th Annual Symposium on Foundations of Computer Science (FOCS)* (pp. 94–105).: IEEE.

[259] Jun, H., Ko, B., Kim, Y., Kim, I., & Kim, J. (2019). Combination of multiple global descriptors for image retrieval. *arXiv preprint arXiv:1903.10663*.

[260] Juvekar, C., Vaikuntanathan, V., & Chandrakasan, A. (2018). Gazelle: A low latency framework for secure neural network inference. In *Proceedings of the 27th USENIX Conference on Security Symposium*, SEC18 (pp. 16511668). USA: USENIX Association.

[Kabasakalolu et al.] Kabasakalolu, E., Vepakomma, P., & Ji, H. Regularized eikonal pdes for variable privacy-based geolocation release.

[262] Kaggle (2011). *Give me some credit dataset*.

[263] Kahn, J., Kalai, G., & Linial, N. (1988). The influence of variables on boolean functions. In *[Proceedings 1988] 29th Annual Symposium on Foundations of Computer Science* (pp. 68–80).

[264] Kairouz, P., Liao, J., Huang, C., & Sankar, L. (2019a). Censored and fair universal representations using generative adversarial models.

[265] Kairouz, P., McMahan, H. B., Avent, B., Bellet, A., Bennis, M., Bhagoji, A. N., Bonawitz, K., Charles, Z., Cormode, G., Cummings, R., D'Oliveira, R. G. L., Rouayheb, S. E., Evans, D., Gardner, J., Garrett, Z. A., Gascón, A., Ghazi, B., Gibbons, P. B., Gruteser, M., Harchaoui, Z., He, C., He, L., Huo, Z., Hutchinson, B., Hsu, J., Jaggi, M., Javidi, T., Joshi, G., Khodak, M., Konecný, J., Korolova, A., Koushanfar, F., Koyejo, O., Lepoint, T., Liu, Y., Mittal, P., Mohri, M., Nock,

R., Özgür, A., Pagh, R., Raykova, M., Qi, H., Ramage, D., Raskar, R., Song, D. X., Song, W., Stich, S. U., Sun, Z., Suresh, A. T., Tramèr, F., Vepakomma, P., Wang, J., Xiong, L., Xu, Z., Yang, Q., Yu, F. X., Yu, H., & Zhao, S. (2019b). Advances and open problems in federated learning. *ArXiv*, abs/1912.04977.

[266] Kairouz, P., McMahan, H. B., Avent, B., Bellet, A., Bennis, M., Bhagoji, A. N., Bonawitz, K., Charles, Z., Cormode, G., Cummings, R., et al. (2019c). Advances and open problems in federated learning. *arXiv preprint arXiv:1912.04977*.

[267] Kairouz, P., Oh, S., & Viswanath, P. (2014). Extremal mechanisms for local differential privacy. In *Advances in neural information processing systems* (pp. 2879–2887).

[268] Kairouz, P., Oh, S., & Viswanath, P. (2015). The composition theorem for differential privacy. In *International conference on machine learning* (pp. 1376–1385).: PMLR.

[269] Kaissis, G. A., Makowski, M. R., Rückert, D., & Braren, R. F. (2020). Secure, privacy-preserving and federated machine learning in medical imaging. *Nature Machine Intelligence*, (pp. 1–7).

[270] Kawachi, A., Tanaka, K., & Xagawa, K. (2007). Multi-bit cryptosystems based on lattice problems, public key cryptography–pkc, springer. *pp*, (pp. 315–329).

[271] Kazemi, E., Zadimoghaddam, M., & Karbasi, A. (2018). Scalable deletion-robust submodular maximization: Data summarization with privacy and fairness constraints. In *International conference on machine learning* (pp. 2544–2553).: PMLR.

[272] Kempner, Y. & Levit, V. E. (2003). Correspondence between two antimatroid algorithmic characterizations. *The Electronic Journal of Combinatorics, 10, 2003*.

[273] Kempner, Y., Mirkin, B., & Muchnik, I. (1997). Monotone linkage clustering and quasi-concave set functions. *Applied Mathematics Letters*, 10(4), 19–24.

[274] Kempner, Y. & Muchnik, I. (2003). Clustering on antimatroids and convex geometries. *WSEAS Transactions on Mathematics*, 2(1), 54–59.

[275] Kempner, Y. & Muchnik, I. (2008). Quasi-concave functions on meet-semilattices. *Discrete applied mathematics*, 156(4), 492–499.

[276] Kenthapadi, K., Korolova, A., Mironov, I., & Mishra, N. (2012). Privacy via the johnson-lindenstrauss transform. *arXiv preprint arXiv:1204.2606*.

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

[278] Këpuska, V. & Bohouta, G. (2018). Next-generation of virtual personal assistants (microsoft cortana, apple siri, amazon alexa and google home). *2018 IEEE 8th Annual Computing and Communication Workshop and Conference (CCWC)*, (pp. 99–103).

[279] Khan, A. M., Sirinukunwattana, K., & Rajpoot, N. (2015). A global covariance descriptor for nuclear atypia scoring in breast histopathology images. *IEEE journal of biomedical and health informatics*, 19(5), 1637–1647.

[280] Kim, M. & Lauter, K. E. (2015). Private genome analysis through homomorphic encryption. *BMC medical informatics and decision making*, 15.

[281] Kinney, J. B. & Atwal, G. S. (2014). Equitability, mutual information, and the maximal information coefficient. *Proceedings of the National Academy of Sciences*, 111(9), 3354–3359.

[282] Kocher, P., Horn, J., Fogh, A., , Genkin, D., Gruss, D., Haas, W., Hamburg, M., Lipp, M., Mangard, S., Prescher, T., Schwarz, M., & Yarom, Y. (2019). Spectre attacks: Exploiting speculative execution. In *40th IEEE Symposium on Security and Privacy (S&P'19)*.

[283] Konečnỳ, J., McMahan, H. B., Yu, F. X., Richtárik, P., Suresh, A. T., & Bacon, D. (2016). Federated learning: Strategies for improving communication efficiency. *arXiv preprint arXiv:1610.05492*.

[284] Konený, J., McMahan, H. B., Yu, F. X., Suresh, A. T., Bacon, D., & Richtárick, P. (2017). Federated learning: Strategies for improving communication efficiency. *arXiv preprint arXiv:1610.05492*.

[285] Korte, B., Lovász, L., & Schrader, R. (2012). *Greedoids*, volume 4. Springer Science & Business Media.

[286] Koufogiannis, F., Han, S., & Pappas, G. J. (2015). Gradual release of sensitive data under differential privacy. *arXiv preprint arXiv:1504.00429*.

[287] Koufogiannis, F. & Pappas, G. J. (2016). Location-dependent privacy. In *2016 IEEE 55th Conference on Decision and Control (CDC)* (pp. 7586–7591).: IEEE.

[288] Kourtellis, N., Morales, G. D. F., Bifet, A., & Murdopo, A. (2016). Vht: Vertical hoeffding tree. In *2016 ieee international conference on big data (big data)* (pp. 915–922).: IEEE.

[289] Krause, A. & Golovin, D. (2014). Submodular function maximization. *Tractability*, 3, 71–104.

[290] Krause, A., McMahan, H. B., Guestrin, C., & Gupta, A. (2008). Robust submodular observation selection. *Journal of Machine Learning Research*, 9(12).

[291] Krause, J., Stark, M., Deng, J., & Fei-Fei, L. (2013). 3d object representations for fine-grained categorization. In *4th International IEEE Workshop on 3D Representation and Recognition (3dRR-13)* Sydney, Australia.

[292] Križaj, J., Štruc, V., & Dobrišek, S. (2013). Combining 3d face representations using region covariance descriptors and statistical models. In *2013 10th IEEE international conference and workshops on automatic face and gesture recognition (FG)* (pp. 1–7).: IEEE.

[293] Krizanc, D. (1999). A survey of randomness and parallism in comparison problems. In *Advances in Randomized Parallel Computing* (pp. 25–39). Springer.

[294] Krizhevsky, A., Hinton, G., et al. (2009). Learning multiple layers of features from tiny images.

[295] Kumar, N., Rathee, M., Chandran, N., Gupta, D., Rastogi, A., & Sharma, R. (2020). Cryptflow: Secure tensorflow inference. *2020 IEEE Symposium on Security and Privacy (SP)*, (pp. 336–353).

[296] Kusner, M. J., Sun, Y., Sridharan, K., & Weinberger, K. Q. (2015). Inferring the causal direction privately. *stat*, 1050, 17.

[297] Kusner, M. J., Sun, Y., Sridharan, K., & Weinberger, K. Q. (2016). Private causal inference. In *Artificial Intelligence and Statistics* (pp. 1308–1317).: PMLR.

[298] Kuznecov, E., Muchnik, I., & Shvartzer, L. (1985). Monotonic systems and their properties.

[299] Kärkkäinen, K. & Joo, J. (2019). Fairface: Face attribute dataset for balanced race, gender, and age.

[Laine] Laine, K. *Microsoft Research , title = Simple Encrypted Arithmetic Library 2.3.1, volume = 2017, url = https://www.microsoft.com/en-us/research/uploads/prod/2017/11/sealmanual-2-3-1.pdf.*

[301] Laine, S., Karras, T., Aila, T., Herva, A., Saito, S., Yu, R., Li, H., & Lehtinen, J. (2017). Production-level facial performance capture using deep convolutional neural networks. In *Proceedings of the ACM SIGGRAPH / Eurographics Symposium on Computer Animation*, SCA 17 New York, NY, USA: Association for Computing Machinery.

[302] Lange, K. (2016). *MM optimization algorithms*. SIAM.

[303] Langley, P. (2000). Crafting papers on machine learning. In P. Langley (Ed.), *Proceedings of the 17th International Conference on Machine Learning (ICML 2000)* (pp. 1207–1216). Stanford, CA: Morgan Kaufmann.

[304] Laskin, M., Metz, L., Nabarrao, S., Saroufim, M., Noune, B., Luschi, C., Sohl-Dickstein, J., & Abbeel, P. (2020). Parallel training of deep networks with local updates. *arXiv preprint arXiv:2012.03837*.

[305] Lee, D. T. (2015). Efficient, private, and eps-strategyproof elicitation of tournament voting rules. In *Twenty-Fourth International Joint Conference on Artificial Intelligence*.

[306] Lee, J. M. (2006). *Riemannian manifolds: an introduction to curvature*, volume 176. Springer Science & Business Media.

[307] Lee, K., Kim, H., Lee, K., Suh, C., & Ramchandran, K. (2019). Synthesizing differentially private datasets using random mixing. In *2019 IEEE International Symposium on Information Theory (ISIT)* (pp. 542–546).: IEEE.

[308] Lei, X., Liu, A. X., Li, R., & Tu, G.-H. (2019). Seceqp: A secure and efficient scheme for sknn query problem over encrypted geodata on cloud. In *2019 IEEE 35th International Conference on Data Engineering (ICDE)* (pp. 662–673).: IEEE.

[309] Levit, V. E. & Kempner, Y. (2004). Quasi-concave functions on antimatroids. *arXiv preprint math/0408365*.

[310] Li, A., Duan, Y., Yang, H., Chen, Y., & Yang, J. (2020a). Tiprdc: task-independent privacy-respecting data crowdsourcing framework for deep learning

with anonymized intermediate representations. In *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining* (pp. 824–832).

[311] Li, A., Guo, J., Yang, H., & Chen, Y. (2019a). Deepobfuscator: Adversarial training framework for privacy-preserving image classification. *ArXiv*, abs/1909.04126.

[312] Li, A., Guo, J., Yang, H., & Chen, Y. (2019b). Deepobfuscator: Adversarial training framework for privacy-preserving image classification. *arXiv preprint arXiv:1909.04126*.

[313] Li, C., Liu, H., Chen, C., Pu, Y., Chen, L., Henao, R., & Carin, L. (2017). Alice: Towards understanding adversarial learning for joint distribution matching. In *Advances in Neural Information Processing Systems* (pp. 5495–5503).

[314] Li, H., Xiong, L., & Jiang, X. (2014). Differentially private synthesization of multi-dimensional data using copula functions. In *Advances in database technology: proceedings. International conference on extending database technology*, volume 2014 (pp. 475).: NIH Public Access.

[315] Li, M., Poovendran, R., & Narayanan, S. (2005). Protecting patient privacy against unauthorized release of medical images in a group communication environment. *Computerized Medical Imaging and Graphics*, 29(5), 367–383.

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

[317] Li, R., Zhong, W., & Zhu, L. (2012). Feature screening via distance correlation learning. *Journal of the American Statistical Association*, 107(499), 1129–1139.

[318] Li, S., Avestimehr, S., et al. (2020b). Coded computing: Mitigating fundamental bottlenecks in large-scale distributed computing and machine learning. *Foundations and Trends® in Communications and Information Theory*, 17(1), 1–148.

[319] Li, Z. & Zhang, Y. (2020). Label-leaks: Membership inference attack with label. *arXiv preprint arXiv:2007.15528*.

[320] Lin, M., Ji, R., Wang, Y., Zhang, Y., Zhang, B., Tian, Y., & Shao, L. (2020). Hrank: Filter pruning using high-rank feature map. In *Proceedings of the*

IEEE/CVF Conference on Computer Vision and Pattern Recognition (pp. 1529–1538).

[321] Lin, X., Sur, I., Nastase, S. A., Divakaran, A., Hasson, U., & Amer, M. R. (2019). Data-efficient mutual information neural estimator. *arXiv preprint arXiv:1905.03319*.

[322] Lipp, M., Schwarz, M., Gruss, D., Prescher, T., Haas, W., Fogh, A., Horn, J., Mangard, S., Kocher, P., Genkin, D., Yarom, Y., & Hamburg, M. (2018). Meltdown: Reading kernel memory from user space. In *27th USENIX Security Symposium (USENIX Security 18)*.

[323] Liu, C., Chakraborty, S., & Mittal, P. (2016). Dependence makes you vulnberable: Differential privacy under dependent tuples. In *NDSS*, volume 16 (pp. 21–24).

[324] Liu, J., Juuti, M., Lu, Y., & Asokan, N. (2017). Oblivious neural network predictions via minionn transformations. In *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security*, CCS 17 (pp. 619631). New York, NY, USA: Association for Computing Machinery.

[325] Liu, J., Xiong, L., & Luo, J. (2013). Semantic security: Privacy definitions revisited. *Trans. Data Privacy*, 6(3), 185–198.

[326] Liu, S., Du, J., Shrivastava, A., & Zhong, L. (2019). Privacy adversarial network: representation learning for mobile data privacy. *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies*, 3(4), 1–18.

[327] Liu, X., Trieu, N., Kornaropoulos, E. M., & Song, D. (2020a). Beetrace: A unified platform for secure contact tracing that breaks data silos. *arXiv preprint arXiv:2007.02285*.

[328] Liu, Y., Kang, Y., Zhang, X., Li, L., Cheng, Y., Chen, T., Hong, M., & Yang, Q. (2020b). A communication efficient collaborative learning framework for distributed features. *arXiv preprint arXiv:1912.11187*.

[329] Liu, Z., Luo, P., Wang, X., & Tang, X. (2018). Large-scale celebfaces attributes (celeba) dataset. *Retrieved August*, 15, 2018.

[330] Lloyd, S. & Weedbrook, C. (2018). Quantum generative adversarial learning. *Physical review letters*, 121(4), 040502.

[331] Long, Y., Bindschaedler, V., & Gunter, C. A. (2017). Towards measuring membership privacy. *ArXiv*, abs/1712.09136.

[332] Lotan, E., Tschider, C., Sodickson, D. K., Caplan, A. L., Bruno, M., Zhang, B., & Lui, Y. W. (2020). Medical imaging and privacy in the era of artificial intelligence: myth, fallacy, and the future. *Journal of the American College of Radiology*, 17(9), 1159–1162.

[333] Lovász, L. (1983). Submodular functions and convexity. In *Mathematical programming the state of the art* (pp. 235–257). Springer.

[334] Lowd, D. & Meek, C. (2005). Adversarial learning. In *Proceedings of the eleventh ACM SIGKDD international conference on Knowledge discovery in data mining* (pp. 641–647).

[335] Lu, S., Chen, X., & Wang, H. (2021). Conditional distance correlation sure independence screening for ultra-high dimensional survival data. *Communications in Statistics-Theory and Methods*, 50(8), 1936–1953.

[336] Ma, B., Su, Y., & Jurie, F. (2014). Covariance descriptor based on bio-inspired features for person re-identification and face verification. *Image and Vision Computing*, 32(6-7), 379–390.

[337] Machanavajjhala, A., Kifer, D., Gehrke, J., & Venkitasubramaniam, M. (2007). L-diversity: Privacy beyond k-anonymity. *ACM Trans. Knowl. Discov. Data*, 1(1), 3es.

[338] Makri, E., Rotaru, D., Smart, N. P., & Vercauteren, F. (2019). Epic: Efficient private image classification (or: Learning from the masters). In *CT-RSA*.

[339] Malekzadeh, M., Clegg, R. G., Cavallaro, A., & Haddadi, H. (2019). Mobile sensor data anonymization. In *Proceedings of the International Conference on Internet of Things Design and Implementation*, IoTDI 19 (pp. 4958). New York, NY, USA: Association for Computing Machinery.

[340] Malo, P., Sinha, A., Takala, P., Korhonen, P. J., & Wallenius, J. (2013). Good debt or bad debt: Detecting semantic orientations in economic texts. *CoRR*, abs/1307.5336.

[341] Mandal, D., Procaccia, A. D., Shah, N., & Woodruff, D. (2019). Efficient and thrifty voting by any means necessary. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, & R. Garnett (Eds.), *Advances in Neural Information Processing Systems*, volume 32: Curran Associates, Inc.

[342] Mandal, D., Shah, N., & Woodruff, D. P. (2020). Optimal communication-distortion tradeoff in voting. In *Proceedings of the 21st ACM Conference on Economics and Computation*, EC '20 (pp. 795813). New York, NY, USA: Association for Computing Machinery.

[343] Mangat, N. S. (1994). An improved randomized response strategy. *Journal of the Royal Statistical Society: Series B (Methodological)*, 56(1), 93–95.

[344] Martin, T., Karopoulos, G., Hernández-Ramos, J. L., Kambourakis, G., & Fovino, I. N. (2020). Demystifying covid-19 digital contact tracing: A survey on frameworks and mobile apps. *arXiv preprint arXiv:2007.11687*.

[345] Matsui, Y., Yamaguchi, T., & Wang, Z. (2020). Cvpr2020 tutorial on image retrieval in the wild. https://matsui528.github.io/cvpr2020_tutorial_retrieval/.

[346] Matsukawa, T., Okabe, T., Suzuki, E., & Sato, Y. (2016). Hierarchical gaussian descriptor for person re-identification. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 1363–1372).

[347] May, K. O. (1952). A set of independent necessary and sufficient conditions for simple majority decisions. *Econometrica*, 20(4), 680–684.

[348] McMahan, H. B., Moore, E., Ramage, D., Hampson, S., et al. (2016). Communication-efficient learning of deep networks from decentralized data. *arXiv preprint arXiv:1602.05629*.

[349] McMahan, H. B., Moore, E., Ramage, D., Hampson, S., & y Arcas, B. A. (2017a). Communication-efficient learning of deep networks from decentralized data. *arXiv preprint arXiv:1602.05629*.

[350] McMahan, H. B., Ramage, D., Talwar, K., & Zhang, L. (2017b). Learning differentially private language models without losing accuracy. *CoRR*, abs/1710.06963.

[351] McSherry, F. & Talwar, K. (2007). Mechanism design via differential privacy. In *48th Annual IEEE Symposium on Foundations of Computer Science (FOCS'07)* (pp. 94–103).: IEEE.

[352] Meehan, C. R., Chaudhuri, K., & Dasgupta, S. (2020). A non-parametric test to detect data-copying in generative models. *ArXiv*, abs/2004.05675.

[353] Mei, J., Zhao, K., & Lu, B.-L. (2015). On unconstrained quasi-submodular function optimization. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 29.

[354] Mercuri, R. T. (2004). The hipaa-potamus in health care data security. *Communications of the ACM*, 47(7), 2528.

[355] Miolane, N. (2016). *Geometric statistics for computational anatomy*. PhD thesis, Université Côte d'Azur.

[356] Miolane, N., Caorsi, M., Lupo, U., Guerard, M., Guigui, N., Mathe, J., Cabanes, Y., Reise, W., Davies, T., Leitão, A., et al. (2021). Iclr 2021 challenge for computational geometry & topology: Design and results. *arXiv preprint arXiv:2108.09810*.

[357] Miolane, N., Guigui, N., Le Brigant, A., Mathe, J., Hou, B., Thanwerdas, Y., Heyder, S., Peltre, O., Koep, N., Zaatiti, H., et al. (2020). Geomstats: a python package for riemannian geometry in machine learning. *Journal of Machine Learning Research*, 21(223), 1–9.

[358] Mireshghallah, F., Taram, M., Jalali, A., Elthakeb, A. T., Tullsen, D. M., & Esmaeilzadeh, H. (2020a). A principled approach to learning stochastic representations for privacy in deep neural inference. *ArXiv*, abs/2003.12154.

[359] Mireshghallah, F., Taram, M., Jalali, A., Tullsen, D., & Esmaeilzadeh, H. (2020b). Shredder: Learning noise distributions to protect inference privacy. In *Proceedings of the Twenty-Fifth International Conference on Architectural Support for Programming Languages and Operating Systems*, ASPLOS 20 New York, NY, USA: Association for Computing Machinery.

[360] Mireshghallah, F., Taram, M., Ramrakhyani, P., Tullsen, D., & Esmaeilzadeh, H. (2019). Shredder: Learning noise distributions to protect inference privacy.

[361] Mireshghallah, F., Taram, M., Vepakomma, P., Singh, A., Raskar, R., & Esmaeilzadeh, H. (2020c). Privacy in deep learning: A survey. *arXiv preprint arXiv:2004.12254*.

[362] Mirjalili, V., Raschka, S., & Ross, A. (2019a). Flowsan: Privacy-enhancing semi-adversarial networks to confound arbitrary face-based gender classifiers. *IEEE Access*, 7, 99735–99745.

[363] Mirjalili, V., Raschka, S., & Ross, A. (2019b). Flowsan: Privacy-enhancing semi-adversarial networks to confound arbitrary face-based gender classifiers. *arXiv preprint arXiv:1905.01388*.

[364] Mirjalili, V., Raschka, S., & Ross, A. (2020). Privacynet: semi-adversarial networks for multi-attribute face privacy. *IEEE Transactions on Image Processing*, 29, 9400–9412.

[365] Mironov, I. (2012). On significance of the least significant bits for differential privacy. In *Proceedings of the 2012 ACM conference on Computer and communications security* (pp. 650–661).

[366] Mironov, I. (2017). Rényi differential privacy. In *2017 IEEE 30th Computer Security Foundations Symposium (CSF)* (pp. 263–275).: IEEE.

[367] Mirzasoleiman, B., Karbasi, A., & Krause, A. (2017). Deletion-robust submodular maximization: Data summarization with the right to be forgotten. In *International Conference on Machine Learning* (pp. 2449–2458).: PMLR.

[368] Mishra, P., Lehmkuhl, R. T., Srinivasan, A., Zheng, W., & Popa, R. A. (2020). Delphi: A cryptographic inference service for neural networks. *IACR Cryptology ePrint Archive*, 2020, 50.

[369] Mo, F., Shamsabadi, A. S., Katevas, K., Demetriou, S., Leontiadis, I., Cavallaro, A., & Haddadi, H. (2020). Darknetz: Towards model privacy at the edge using trusted execution environments. *ArXiv*, abs/2004.05703.

[370] Mohassel, P. (2011). Efficient and secure delegation of linear algebra. *Cryptology ePrint Archive*.

[371] Mohassel, P. & Rindal, P. (2018). Aby3: A mixed protocol framework for machine learning. In *Proceedings of the 2018 ACM SIGSAC Conference on Computer and Communications Security*, CCS 18 (pp. 3552). New York, NY, USA: Association for Computing Machinery.

[372] Mohassel, P. & Zhang, Y. (2017). Secureml: A system for scalable privacy-preserving machine learning. In *2017 IEEE Symposium on Security and Privacy (SP)* (pp. 19–38).

[373] Moro, S., Cortez, P., & Rita, P. (2014). A data-driven approach to predict the success of bank telemarketing. *Decision Support Systems*, 62.

[374] Mossel, E., O'Donnell, R., & Oleszkiewicz, K. (2010). Noise stability of functions with low influences: Invariance and optimality. *Annals of Mathematics*, 171(1), 295341.

[375] Moustafa, N., Choo, K.-K. R., Radwan, I., & Camtepe, S. (2019). Outlier dirichlet mixture mechanism: Adversarial statistical learning for anomaly detection in the fog. *IEEE Transactions on Information Forensics and Security*, 14(8), 1975–1987.

[376] Muchnik, I. & Shvartser, L. (1987a). Submodular set functions and monotone systems in aggregation, i. *Automation and Remote Control 1987*, (5).

[377] Muchnik, I. & Shvartser, L. (1987b). Submodular set functions and monotone systems in aggregation, ii. *Automation and Remote Control 1987*, (5).

[378] Mullat, I. (1976). Extremal subsystems of monotonic systems. 1. *Automation and Remote Control*, 37(5), 758–766.

[379] Murota, K. (1998). Discrete convex analysis. *Mathematical Programming*, 83(1), 313–371.

[380] Murota, K. (2009). Recent developments in discrete convex analysis. In *Research trends in combinatorial optimization* (pp. 219–260). Springer.

[381] Myers, A., Utpala, S., Talbar, S., Sanborn, S., Shewmake, C., Donnat, C., Mathe, J., Sonthalia, R., Cui, X., Szwagier, T., et al. (2022). Iclr 2022 challenge for computational geometry & topology: Design and results. In *Topological, Algebraic and Geometric Learning Workshops 2022* (pp. 269–276).: PMLR.

[382] Naccache, D. & Jacques Stern, A. (1998). New public key cryptosystem based on higher residues. In *Proceedings of the 5th ACM Conference on Computer and Communications Security* (pp. 59–66).: ACM.

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

[384] Narra, K. G., Lin, Z., Wang, Y., Balasubramaniam, K., & Annavaram, M. (2019). Privacy-preserving inference in machine learning services using trusted execution environments. *ArXiv*, abs/1912.03485.

[385] Nemhauser, G. L., Wolsey, L. A., & Fisher, M. L. (1978). An analysis of approximations for maximizing submodular set functionsi. *Mathematical programming*, 14(1), 265–294.

[386] Neustaedter, C., Greenberg, S., & Boyle, M. (2006). Blur filtration fails to preserve privacy for home-based video conferencing. *ACM Transactions on Computer-Human Interaction (TOCHI)*, 13(1), 1–36.

[387] Newcomb, A. (2018). Facebook data harvesting scandal widens to 87 million people. online accessed February 2020 https://www.nbcnews.com/tech/tech-news/facebook-data-harvesting-scandal-widens-87-million-people-n862771.

[388] Nissim, K., Raskhodnikova, S., & Smith, A. (2007). Smooth sensitivity and sampling in private data analysis. In *Proceedings of the thirty-ninth annual ACM symposium on Theory of computing* (pp. 75–84).

[389] Nissim, K. & Wood, A. (2018). Is privacy privacy? *Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences*, 376(2128), 20170358.

[390] Nock, R., Hardy, S., Henecka, W., Ivey-Law, H., Patrini, G., Smith, G., & Thorne, B. (2018). Entity resolution and federated learning get a federated resolution. *arXiv preprint arXiv:1803.04035*.

[391] O'Donnell, R. (2004). Hardness amplification within np. *Journal of Computer and System Sciences*, 69(1), 68–94. Special Issue on Computational Complexity 2002.

[392] O'Donnell, R. (2014). *Analysis of boolean functions*. Cambridge University Press.

[393] Oh, S. J., Benenson, R., Fritz, M., & Schiele, B. (2016). Faceless person recognition: Privacy implications in social media. In *European Conference on Computer Vision* (pp. 19–35).: Springer.

[Okamoto & Shigenori Uchiyama] Okamoto, T. & Shigenori Uchiyama, A. *New public-key cryptosystem as secure as factoring, In Advances in Cryptology—EUROCRYPT '98*. Springer.

[395] Osia, S. A., Shamsabadi, A. S., Sajadmanesh, S., Taheri, A., Katevas, K., Rabiee, H. R., Lane, N. D., & Haddadi, H. (2020). A hybrid deep learning architecture for privacy-preserving mobile analytics. *IEEE Internet of Things Journal*, 7(5), 4505–4518.

[396] Osia, S. A., Taheri, A., Shamsabadi, A. S., Katevas, K., Haddadi, H., & Rabiee, H. R. (2018). Deep private-feature extraction.

[397] Ossia, S. A., Taheri, A., Shamsabadi, A. S., Katevas, K., Haddadi, H., & Rabiee, H. R. (2018). Deep private-feature extraction. *IEEE Transactions on Knowledge and Data Engineering*, 32, 54–66.

[398] Paillier, P. (1999). *Public-key Cryptosystems Based on Composite Degree Residuosity Classes, Advances in cryptology—EUROCRYPT '99*. pp. 223–238: Springer.

[399] Pang, Y., Yuan, Y., & Li, X. (2008). Gabor-based region covariance matrices for face recognition. *IEEE Transactions on circuits and systems for video technology*, 18(7), 989–993.

[400] Papernot, N., Abadi, M., Úlfar Erlingsson, Goodfellow, I., & Talwar, K. (2016). Semi-supervised knowledge transfer for deep learning from private training data.

[401] Papernot, N., Song, S., Mironov, I., Raghunathan, A., Talwar, K., & Úlfar Erlingsson (2018). Scalable private learning with pate.

[402] Park, T., Shao, X., & Yao, S. (2015). Partial martingale difference correlation. *Electronic Journal of Statistics*, 9(1), 1492–1517.

[403] Pasquini, D., Ateniese, G., & Bernaschi, M. (2020). Unleashing the tiger: Inference attacks on split learning. *arXiv preprint arXiv:2012.02670*.

[404] Pennec, X. (2006). Intrinsic statistics on riemannian manifolds: Basic tools for geometric measurements. *Journal of Mathematical Imaging and Vision*, 25(1), 127–154.

[405] Pennec, X., Fillard, P., & Ayache, N. (2006). A riemannian framework for tensor computing. *International Journal of computer vision*, 66(1), 41–66.

[406] Pennec, X., Sommer, S., & Fletcher, T. (2019). *Riemannian geometric statistics in medical image analysis*. Academic Press.

[407] Pennington, J., Socher, R., & Manning, C. D. (2014). Glove: Global vectors for word representation. In *Empirical Methods in Natural Language Processing (EMNLP)* (pp. 1532–1543).

[408] Penrose, L. (1946). The elementary statistics of majority voting. *Journal of the Royal Statistical Society*, (pp. 109(1):5357).

[409] Phan, N., Wang, Y., Wu, X., & 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*, AAAI16 (pp. 13091316).: AAAI Press.

[410] Phan, N., Wu, X., & Dou, D. (2017a). Preserving differential privacy in convolutional deep belief networks. *Mach. Learn.*, 106(910), 16811704.

[411] Phan, N., Wu, X., Hu, H., & Dou, D. (2017b). Adaptive laplace mechanism: Differential privacy preservation in deep learning. *CoRR*, abs/1709.05750.

[412] Phong, L. T., Aono, Y., Hayashi, T., Wang, L., & Moriai, S. (2018). Privacy-preserving deep learning via additively homomorphic encryption. *IEEE Transactions on Information Forensics and Security*, 13(5), 1333–1345.

[413] Pilanci, M. & Wainwright, M. J. (2015). Randomized sketches of convex programs with sharp guarantees. *IEEE Transactions on Information Theory*, 61(9), 5096–5115.

[414] Pinceti, A., Kosut, O., & Sankar, L. (2019). Data-driven generation of synthetic load datasets preserving spatio-temporal features. In *2019 IEEE Power Energy Society General Meeting (PESGM)* (pp. 1–5).

[415] Pitropakis, N., Panaousis, E., Giannetsos, T., Anastasiadis, E., & Loukas, G. (2019). A taxonomy and survey of attacks against machine learning. *Computer Science Review*, 34, 100199.

[416] Powers, T., Bilmes, J., Wisdom, S., Krout, D. W., & Atlas, L. (2016). Constrained robust submodular optimization. In *NIPS OPT2016 workshop*.

[417] Prasad, A., Jegelka, S., & Batra, D. (2014). Submodular meets structured: Finding diverse subsets in exponentially-large structured item sets. *arXiv preprint arXiv:1411.1752*.

[418] Rachuri, R. & Suresh, A. (2019). Trident: Efficient 4pc framework for privacy preserving machine learning. *IACR Cryptol. ePrint Arch.*, 2019, 1315.

[419] Raskar, R., Schunemann, I., Barbar, R., Vilcans, K., Gray, J., Vepakomma, P., Kapa, S., Nuzzo, A., Gupta, R., Berke, A., et al. (2020). Apps gone rogue: Maintaining personal privacy in an epidemic. *arXiv preprint arXiv:2003.08567*.

[420] Raskar, R., Vepakomma, P., Swedish, T., & Sharan, A. (2019). Data markets to support ai for all: Pricing, valuation and governance. *arXiv preprint arXiv:1905.06462*.

[421] Rassouli, B. & Gündüz, D. (2019). Optimal utility-privacy trade-off with total variation distance as a privacy measure. *IEEE Transactions on Information Forensics and Security*, 15, 594–603.

[422] Raval, N., Machanavajjhala, A., & Pan, J. (2019). Olympus: sensor privacy through utility aware obfuscation. *Proceedings on Privacy Enhancing Technologies*, 2019(1), 5–25.

[423] Reagen, B., Choi, W., Ko, Y., Lee, V., Wei, G.-Y., Lee, H.-H. S., & Brooks, D. (2020). Cheetah: Optimizing and accelerating homomorphic encryption for private inference.

[424] Reichert, L., Brack, S., & Scheuermann, B. (2020). A survey of automatic contact tracing approaches. *Cryptol. ePrint Arch., IACR, Bellevue, WA, USA, Tech. Rep*, 672, 2020.

[425] Reimherr, M., Bharath, K., & Soto, C. (2021). Differential privacy over riemannian manifolds. *Advances in Neural Information Processing Systems*, 34.

[426] Research, F. (2020). Protecting privacy in facebook mobility data during the covid-19 response. In *Blog*.

[427] Riazi, M. S., Samragh, M., Chen, H., Laine, K., Lauter, K., & Koushanfar, F. (2019). Xonn: Xnor-based oblivious deep neural network inference. In *Proceedings of the 28th USENIX Conference on Security Symposium*, SEC19 (pp. 15011518). USA: USENIX Association.

[428] Riazi, M. S., Weinert, C., Tkachenko, O., Songhori, E. M., Schneider, T., & Koushanfar, F. (2018). Chameleon: A hybrid secure computation framework for machine learning applications. *CoRR*, abs/1801.03239.

[Rivest et al.] Rivest, R. L., Shamir, A., & Len Adleman, A. Method for obtaining digital signatures and public-key cryptosystems. *Communications of the ACM*, 21(2), 120–126.

[430] Rivest, R. L., Weitzner, D., Ivers, L., Soibelman, I., & Zissman, M. (2020). Pact: Private automated contact tracing.

[431] Rizzo, M. L. & Székely, G. J. (2016). Energy distance. *wiley interdisciplinary reviews: Computational statistics*, 8(1), 27–38.

[432] Robert, C. P., Casella, G., & Casella, G. (1999). *Monte Carlo statistical methods*, volume 2. Springer.

[433] Rouhani, B. D., Riazi, M. S., & Koushanfar, F. (2017). Deepsecure: Scalable provably-secure deep learning. *CoRR*, abs/1705.08963.

[434] Rousseau, J. J. (1762). Du contrat social. *Marc Michel Rey*.

[435] Roy, P. C. & Boddeti, V. N. (2019a). Mitigating information leakage in image representations: A maximum entropy approach. *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, (pp. 2581–2589).

[436] Roy, P. C. & Boddeti, V. N. (2019b). Mitigating information leakage in image representations: A maximum entropy approach. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*.

[437] Sablayrolles, A., Douze, M., Ollivier, Y., Schmid, C., & Jégou, H. (2019). White-box vs black-box: Bayes optimal strategies for membership inference.

[438] Sadeghi, B., Yu, R., & Boddeti, V. (2019). On the global optima of kernelized adversarial representation learning. In *Proceedings of the IEEE International Conference on Computer Vision* (pp. 7971–7979).

[439] Saha, D. K., Calhoun, V. D., Du, Y., Fu, Z., Panta, S. R., Kwon, S., Sarwate, A., & Plis, S. M. (2021). Privacy-preserving quality control of neuroimaging datasets in federated environment. *bioRxiv*, (pp. 826974).

[440] Saha, D. K., Calhoun, V. D., Panta, S. R., & Plis, S. M. (2017). See without looking: joint visualization of sensitive multi-site datasets. In *IJCAI* (pp. 2672–2678).

[441] Saha, D. K., Calhoun, V. D., Yuhui, D., Zening, F., Panta, S. R., & Plis, S. M. (2020). dsne: a visualization approach for use with decentralized data. *BioRxiv*, (pp. 826974).

[442] Salakhutdinov, R. R., Roweis, S. T., & Ghahramani, Z. (2012). On the convergence of bound optimization algorithms. *arXiv preprint arXiv:1212.2490*.

[443] Salem, A., Bhattacharyya, A., Backes, M., Fritz, M., & Zhang, Y. (2019). Updates-leak: Data set inference and reconstruction attacks in online learning. *CoRR*, abs/1904.01067.

[444] Salem, A., Zhang, Y., Humbert, M., Fritz, M., & Backes, M. (2018). Ml-leaks: Model and data independent membership inference attacks and defenses on machine learning models. *ArXiv*, abs/1806.01246.

[445] Sameer, W., Divya, G., & Nishanth, C. (2018). Securenn: Efficient and private neural network training. *IACR Cryptol. ePrint Arch.*, 2018, 442.

[446] Samragh, M., Hosseini, H., Triastcyn, A., Azarian, K., Soriaga, J., & Koushanfar, F. (2021). Unsupervised information obfuscation for split inference of neural networks. *arXiv preprint arXiv:2104.11413*.

[447] Sander, T., Young, A., & Yung, M. (1999). Non-interactive cryptocomputing for nc sup1, ieee foundations of computer science, 40th annual symposium on. *pp*, (pp. 554–566).

[448] Sanjeev, A. & Kannan, R. (2001). Learning mixtures of arbitrary gaussians. In *Proceedings of the thirty-third annual ACM symposium on Theory of computing* (pp. 247–257).

[449] Sanyal, A., Kusner, M. J., Gascón, A., & Kanade, V. (2018). TAPAS: tricks to accelerate (encrypted) prediction as a service. *CoRR*, abs/1806.03461.

[450] Sathya, S. S., Vepakomma, P., Raskar, R., Ramachandra, R., & Bhattacharya, S. (2018). A review of homomorphic encryption libraries for secure computation. *arXiv preprint arXiv:1812.02428*.

[451] Sattler, F., Wiedemann, S., Müller, K.-R., & Samek, W. (2019). Robust and communication-efficient federatedlearning from non-iid data. *arXiv preprint arXiv:1903.02891*.

[452] Schiff, J., Meingast, M., Mulligan, D. K., Sastry, S. S., & Goldberg, K. (2009). Respectful cameras: Detecting visual markers in real-time to address privacy concerns. In *Protecting Privacy in Video Surveillance*.

[Schneider] Schneider, J. *Awesome - A curated list of amazing Homomorphic Encryption libraries*. software and resources.

[454] Schwartzman, A. (2016). Lognormal distributions and geometric averages of symmetric positive definite matrices. *International Statistical Review*, 84(3), 456–486.

[455] Schwarzschild, A., Goldblum, M., Gupta, A., Dickerson, J. P., & Goldstein, T. (2020). Just how toxic is data poisoning? a unified benchmark for backdoor and data poisoning attacks. *arXiv preprint arXiv:2006.12557*.

[456] Security, I. (2015). Data exfiltration study: actors, tactics, and detection (2015).

[457] Seiffarth, F., Horváth, T., & Wrobel, S. (2021). Maximum margin separations in finite closure systems. In *Machine Learning and Knowledge Discovery in Databases: European Conference, ECML PKDD 2020, Ghent, Belgium, September 14–18, 2020, Proceedings, Part I* (pp. 3–18).: Springer International Publishing.

[458] Sejdinovic, D., Sriperumbudur, B., Gretton, A., & Fukumizu, K. (2013). Equivalence of distance-based and rkhs-based statistics in hypothesis testing. *The Annals of Statistics*, (pp. 2263–2291).

[459] Senior, A., Pankanti, S., Hampapur, A., Brown, L., Ying-Li Tian, Ekin, A., Connell, J., Chiao Fe Shu, & Lu, M. (2005). Enabling video privacy through computer vision. *IEEE Security Privacy*, 3(3), 50–57.

[460] Shafahi, A., Huang, W. R., Najibi, M., Suciu, O., Studer, C., Dumitras, T., & Goldstein, T. (2018). Poison frogs! targeted clean-label poisoning attacks on neural networks. In *Advances in Neural Information Processing Systems* (pp. 6103–6113).

[461] Shamsabadi, A. S., Gascón, A., Haddadi, H., & Cavallaro, A. (2020). Privedge: From local to distributed private training and prediction. *ArXiv*, abs/2004.05574.

[462] Shang, S., Wang, T., Cuff, P., & Kulkarni, S. (2014). The application of differential privacy for rank aggregation: Privacy and accuracy.

[463] Sharaf, A., Torki, M., Hussein, M. E., & El-Saban, M. (2015). Real-time multi-scale action detection from 3d skeleton data. In *2015 IEEE Winter Conference on Applications of Computer Vision* (pp. 998–1005).: IEEE.

[464] Sharma, V., Tapaswi, M., & Stiefelhagen, R. (2020). Deep multimodal feature encoding for video ordering. In *ICCV workshop on Large Scale Holistic Video Understanding*.

[465] Sharma, V., Vepakomma, P., Swedish, T., Chang, K., Kalpathy-Cramer, J., & Raskar, R. (2019a). Expertmatcher: Automating ml model selection for users in resource constrained countries. In *NeurIPS Workshop on Robust AI in Financial Services: Data, Fairness, Explainability, Trustworthiness, and Privacy*.

[466] Sharma, V., Vepakomma, P., Swedish, T., Chang, K., Kalpathy-Cramer, J., & Raskar, R. (2019b). Expertmatcher: Automating ml model selection for users in resource constrained countries. In *NeurIPS Workshop on Machine learning for the Developing World*.

[467] Sheffet, O. (2018). Locally private hypothesis testing. In *International Conference on Machine Learning* (pp. 4605–4614).: PMLR.

[468] Shen, C., Panda, S., & Vogelstein, J. T. (2022). The chi-square test of distance correlation. *Journal of Computational and Graphical Statistics*, 31(1), 254–262.

[469] Shen, C., Priebe, C. E., & Vogelstein, J. T. (2019a). The exact equivalence of independence testing and two-sample testing. *arXiv preprint arXiv:1910.08883*.

[470] Shen, C. & Vogelstein, J. T. (2021). The exact equivalence of distance and kernel methods in hypothesis testing. *AStA Advances in Statistical Analysis*, 105(3), 385–403.

[471] Shen, M., Zhang, J., Zhu, L., Xu, K., & Tang, X. (2019b). Secure svm training over vertically-partitioned datasets using consortium blockchain for vehicular social networks. *IEEE Transactions on Vehicular Technology*.

[472] Shi, Y., Davaslioglu, K., & Sagduyu, Y. E. (2020). Over-the-air membership inference attacks as privacy threats for deep learning-based wireless signal classifiers. In *Proceedings of the 2nd ACM Workshop on Wireless Security and Machine Learning* (pp. 61–66).

[473] Shokri, R. & Shmatikov, V. (2015). Privacy-preserving deep learning. In *Proceedings of the 22nd ACM SIGSAC Conference on Computer and Communications Security*, CCS 15 (pp. 13101321). New York, NY, USA: Association for Computing Machinery.

[474] Shokri, R., Stronati, M., & Shmatikov, V. (2016). Membership inference attacks against machine learning models. *CoRR*, abs/1610.05820.

[475] Shokri, R., Stronati, M., Song, C., & Shmatikov, V. (2017). Membership inference attacks against machine learning models. In *2017 IEEE Symposium on Security and Privacy (SP)* (pp. 3–18).: IEEE.

[476] Shrivastava, A. & Li, P. (2014). In defense of minhash over simhash. In *Artificial Intelligence and Statistics* (pp. 886–894).

[477] Singh, A., Vepakomma, P., Sharma, V., & Raskar, R. (2023). Posthoc privacy guarantees for collaborative inference with modified propose-test-release. In *Thirty-seventh Conference on Neural Information Processing Systems*.

[478] Singh, P., Singh, A., Cojocaru, G., Vepakomma, P., & Raskar, R. (2020a). Pp-contacttracing: A privacy-preserving contact tracing protocol for covid-19 pandemic. *arXiv preprint arXiv:2008.06648*.

[479] Singh, S., Sikka, H., Kotti, S., & Trask, A. (2020b). Benchmarking differentially private residual networks for medical imagery. *arXiv preprint arXiv:2005.13099*.

[480] Smirnov, O. (2021). Tensorflow riemopt: a library for optimization on riemannian manifolds. *arXiv preprint arXiv:2105.13921*.

[481] Smith, A., Thakurta, A., & Upadhyay, J. (2017). Is interaction necessary for distributed private learning? In *2017 IEEE Symposium on Security and Privacy (SP)* (pp. 58–77).: IEEE.

[482] Sommer, S., Lauze, F., Hauberg, S., & Nielsen, M. (2010). Manifold valued statistics, exact principal geodesic analysis and the effect of linear approximations. In *European conference on computer vision* (pp. 43–56).: Springer.

[483] Song, C. & Shmatikov, V. (2018). The natural auditor: How to tell if someone used your words to train their model. *ArXiv*, abs/1811.00513.

[484] Song, C. & Shmatikov, V. (2019). Auditing data provenance in text-generation models. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '19 (pp. 196206). New York, NY, USA: Association for Computing Machinery.

[485] Song, L., Shokri, R., & Mittal, P. (2019). Membership inference attacks against adversarially robust deep learning models. In *2019 IEEE Security and Privacy Workshops (SPW)* (pp. 50–56).: IEEE.

[486] Sriperumbudur, B. K., Fukumizu, K., Gretton, A., Schölkopf, B., & Lanckriet, G. R. (2012). On the empirical estimation of integral probability metrics. *Electronic Journal of Statistics*, 6, 1550–1599.

[487] Steerneman, T. (1983). On the total variation and hellinger distance between signed measures; an application to product measures. *Proceedings of the American Mathematical Society*, 88(4), 684–688.

[488] Steil, J., Hagestedt, I., Huang, M. X., & Bulling, A. (2019). Privacy-aware eye tracking using differential privacy. In *Proceedings of the 11th ACM Symposium on Eye Tracking Research & Applications* (pp. 1–9).

[489] Stolpe, M., Bhaduri, K., Das, K., & Morik, K. (2013). Anomaly detection in vertically partitioned data by distributed core vector machines. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases* (pp. 321–336).: Springer.

[490] Stummer, W. & Vajda, I. (2012). On bregman distances and divergences of probability measures. *IEEE Transactions on Information Theory*, 58(3), 1277–1288.

[491] Su, J., Vargas, D. V., & Sakurai, K. (2019). One pixel attack for fooling deep neural networks. *IEEE Transactions on Evolutionary Computation*, 23(5), 828–841.

[492] Subbarao, R. & Meer, P. (2009). Nonlinear mean shift over riemannian manifolds. *International journal of computer vision*, 84(1), 1–20.

[493] Sun, L., Dou, Y., Yang, C., Wang, J., Yu, P. S., & Li, B. (2018). Adversarial attack and defense on graph data: A survey. *arXiv preprint arXiv:1812.10528*.

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

[495] Székely, G. J. & Rizzo, M. L. (2009). Brownian distance covariance. *The annals of applied statistics*, 3(4), 1236–1265.

[496] Székely, G. J. & Rizzo, M. L. (2013a). The distance correlation t-test of independence in high dimension. *Journal of Multivariate Analysis*, 117, 193–213.

[497] Székely, G. J. & Rizzo, M. L. (2013b). Energy statistics: A class of statistics based on distances. *Journal of statistical planning and inference*, 143(8), 1249–1272.

[498] Székely, G. J. & Rizzo, M. L. (2014). Partial distance correlation with methods for dissimilarities. *The Annals of Statistics*, 42(6), 2382–2412.

[499] Székely, G. J. & Rizzo, M. L. (2017). The energy of data. *Annual Review of Statistics and Its Application*, 4, 447–479.

[500] Székely, G. J., Rizzo, M. L., Bakirov, N. K., et al. (2007). Measuring and testing dependence by correlation of distances. *The annals of statistics*, 35(6), 2769–2794.

[501] Tabia, H., Laga, H., Picard, D., & Gosselin, P.-H. (2014). Covariance descriptors for 3d shape matching and retrieval. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 4185–4192).

[502] Tang, Q. (2020). Privacy-preserving contact tracing: current solutions and open questions. *arXiv preprint arXiv:2004.06818*.

[503] Taram, M., Venkat, A., & Tullsen, D. (2019a). Context-sensitive fencing: Securing speculative execution via microcode customization. In *Proceedings of the Twenty-Fourth International Conference on Architectural Support for Programming Languages and Operating Systems*, ASPLOS 19 (pp. 395410). New York, NY, USA: Association for Computing Machinery.

[504] Taram, M., Venkat, A., & Tullsen, D. M. (2019b). Packet chasing: Spying on network packets over a cache side-channel. *ArXiv*, abs/1909.04841.

[505] Team, D. P. (2017). Learning with privacy at scale. In *https://machinelearning.apple.com/*.

[506] Thanwerdas, Y. & Pennec, X. (2021). O (n)-invariant riemannian metrics on spd matrices. *arXiv preprint arXiv:2109.05768*.

[507] Thomas Fletcher, P. (2013). Geodesic regression and the theory of least squares on riemannian manifolds. *International journal of computer vision*, 105(2), 171–185.

[508] Thompson, S. A. & Warzel, C. (2019). The privacy project: Twelve million phones, one dataset, zero privacy. online accessed February 2020 https://www.nytimes.com/interactive/2019/12/19/opinion/location-tracking-cell-phone.html.

[509] Titsworth, R. (1962). Correlation properties of cyclic sequences. *PhD thesis, CalTech*.

[510] Tonde, C. J. (2016). *Supervised feature learning via dependency maximization*. PhD thesis, Rutgers University-Graduate School-New Brunswick.

[511] Torgerson, W. S. (1952). Multidimensional scaling: I. theory and method. *Psychometrika*, 17(4), 401–419.

[512] Tramèr, F., Zhang, F., Juels, A., Reiter, M. K., & Ristenpart, T. (2016). Stealing machine learning models via prediction apis. *CoRR*, abs/1609.02943.

[513] Trieu, N., Shehata, K., Saxena, P., Shokri, R., & Song, D. (2020). Epione: Lightweight contact tracing with strong privacy. *arXiv preprint arXiv:2004.13293*.

[514] Truex, S., Liu, L., Gursoy, M. E., Yu, L., & Wei, W. (2018a). Towards demystifying membership inference attacks. *ArXiv*, abs/1807.09173.

[515] Truex, S., Liu, L., Gursoy, M. E., Yu, L., & Wei, W. (2018b). Towards demystifying membership inference attacks. *arXiv preprint arXiv:1807.09173*.

[516] Tschiatschek, S., Djolonga, J., & Krause, A. (2016). Learning probabilistic submodular diversity models via noise contrastive estimation. In *Artificial Intelligence and Statistics* (pp. 770–779).: PMLR.

[517] Tuzel, O., Porikli, F., & Meer, P. (2006). Region covariance: A fast descriptor for detection and classification. In *European conference on computer vision* (pp. 589–600).: Springer.

[518] Tuzel, O., Porikli, F., & Meer, P. (2007). Human detection via classification on riemannian manifolds. In *2007 IEEE Conference on Computer Vision and Pattern Recognition* (pp. 1–8).: IEEE.

[519] Ulyanov, D., Vedaldi, A., & Lempitsky, V. S. (2017). Deep image prior. *CoRR*, abs/1711.10925.

[520] Umeyama, S. (1991). Least-squares estimation of transformation parameters between two point patterns. *IEEE Computer Architecture Letters*, 13(04), 376–380.

[521] Unuchek, R. (April 2018). Leaking ads is user data truly secure? online accessed February 2020 https://published-prd.lanyonevents.com/published/rsaus18/sessionsFiles/8161/ASEC-T08-Leaking-Ads-Is-User-Data-Truly-Secure.pdf.

[522] Upadhyay, J. (2013). Random projections, graph sparsification, and differential privacy. In *International Conference on the Theory and Application of Cryptology and Information Security* (pp. 276–295).: Springer.

[523] Upadhyay, J. (2014a). Differentially private linear algebra in the streaming model. *arXiv preprint arXiv:1409.5414*.

[524] Upadhyay, J. (2014b). Randomness efficient fast-johnson-lindenstrauss transform with applications in differential privacy and compressed sensing. *arXiv preprint arXiv:1410.2470*.

[525] Utpala, S., Han, A., Jawanpuria, P., & Mishra, B. (2022a). Rieoptax: Riemannian optimization in jax. In *OPT 2022: Optimization for Machine Learning (NeurIPS 2022 Workshop)*.

[526] Utpala, S., Vepakomma, P., & Miolane, N. (2022b). Differentially private fréchet mean on the manifold of sym-metric positive definite (spd) matrices with log-euclidean metric. *Transactions on Machine Learning Research*.

[527] Vaidya, J. & Clifton, C. (2004). Privacy preserving naive bayes classifier for vertically partitioned data. In *Proceedings of the 2004 SIAM international conference on data mining* (pp. 522–526).: SIAM.

[528] Vaidya, J., Clifton, C., Kantarcioglu, M., & Patterson, A. S. (2008). Privacy-preserving decision trees over vertically partitioned data. *ACM Transactions on Knowledge Discovery from Data (TKDD)*, 2(3), 1–27.

[529] Valiant, L. G. (1975). Parallelism in comparison problems. *SIAM Journal on Computing*, 4(3), 348–355.

[530] Vallender, S. (1974). Calculation of the wasserstein distance between probability distributions on the line. *Theory of Probability & Its Applications*, 18(4), 784–786.

[531] Van der Maaten, L. & Hinton, G. (2008). Visualizing data using t-sne. *Journal of machine learning research*, 9(11).

[532] Van Erven, T. & Harremos, P. (2014). Rényi divergence and kullback-leibler divergence. *IEEE Transactions on Information Theory*, 60(7), 3797–3820.

[533] Varodayan, D. P. & Khisti, A. (2011). Smart meter privacy using a rechargeable battery: Minimizing the rate of information leakage. *2011 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, (pp. 1932–1935).

[534] Vashist, A. K. (2006). *PhD Thesis: Multipartite graph clustering for structured datasets and automating ortholog extraction*, volume 68.

[535] Vepakomma, P., Amiri, M. M., Canonne, C. L., Raskar, R., & Pentland, A. (2022a). Private independence testing across two parties. *arXiv preprint arXiv:2207.03652*.

[536] Vepakomma, P., Balla, J., & Raskar, R. (2022b). Privatemail: Supervised manifold learning of deep features with privacy for image retrieval. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 36 (pp. 8503–8511).

[537] Vepakomma, P., Gupta, O., Dubey, A., & Raskar, R. (2019). Reducing leakage in distributed deep learning for sensitive health data. *arXiv preprint arXiv:1812.00564*.

[538] Vepakomma, P., Gupta, O., Swedish, T., & Raskar, R. (2018a). Split learning for health: Distributed deep learning without sharing raw patient data. *arXiv preprint arXiv:1812.00564*.

[539] Vepakomma, P. & Kempner, Y. (2019). Diverse data selection via combinatorial quasi-concavity of distance covariance: A polynomial time global minimax algorithm. *Discrete Applied Mathematics*, 265, 182–191.

[540] Vepakomma, P., Kempner, Y., & Raskar, R. (2021a). Parallel quasi-concave set optimization: A new frontier that scales without needing submodularity. *arXiv preprint arXiv:2108.08758*.

[541] Vepakomma, P., Ponkshe, K., & Raskar, R. (2023). Power learning for the private embedding release problem in collaborative learning. In *Preprint*.

[542] Vepakomma, P., Ponkshe, K., & Raskar, R. (2024). Power mechanisms: Private few-shot distributed learning in one (1) communication round. *e-prints*.

[543] Vepakomma, P., Pushpita, S. N., & Raskar, R. (2021b). *DAMS: Meta-estimation of private sketch data structures for differentially private COVID-19 contact tracing*. Technical report, Tech. Rep., 2021. Accessed: Aug. 25, 2021.[Online]. Available: https://www .

[544] Vepakomma, P., Pushpita, S. N., & Raskar, R. (2021c). Private measurement of nonlinear correlations between data hosted across multiple parties. *arXiv preprint arXiv:2110.09670*.

[545] Vepakomma, P., Singh, A., Zhang, E., Gupta, O., & Raskar, R. (2021d). Nopeek-infer: Preventing face reconstruction attacks in distributed inference after on-premise training. In *2021 16th IEEE International Conference on Automatic Face and Gesture Recognition (FG 2021)* (pp. 1–8).: IEEE.

[546] Vepakomma, P., Swedish, T., Raskar, R., Gupta, O., & Dubey, A. (2018b). No peek: A survey of private distributed deep learning. *arXiv preprint arXiv:1812.03288*.

[547] Vepakomma, P., Tonde, C., & Elgammal, A. (2018c). Supervised dimensionality reduction via distance correlation maximization. *Electronic Journal of Statistics*, 12(1), 960–984.

[548] Verdú, S. (2014). Total variation distance and the distribution of relative information. In *2014 Information Theory and Applications Workshop (ITA)* (pp. 1–3).: IEEE.

[549] Vural, E. & Guillemot, C. (2017). A study of the classification of low-dimensional data with supervised manifold learning. *J. Mach. Learn. Res.*, 18(1), 5741–5795.

[550] Wagh, S., Gupta, D., & Chandran, N. (2019). Securenn: 3-party secure computation for neural network training. *Proceedings on Privacy Enhancing Technologies*, 2019, 26 – 49.

[551] Wagh, S., He, X., Machanavajjhala, A., & Mittal, P. (2020). Dp-cryptography: Marrying differential privacy and cryptography in emerging applications. *arXiv preprint arXiv:2004.08887*.

[552] Wang, B. & Gong, N. Z. (2018). Stealing hyperparameters in machine learning. *2018 IEEE Symposium on Security and Privacy (SP)*, (pp. 36–52).

[553] Wang, J., Zhang, J., Bao, W., Zhu, X., Cao, B., & Yu, P. S. (2018). Not just privacy. *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*.

[554] Wang, L., Pang, Q., & Song, D. (2020). Towards practical differentially private causal graph discovery. *Advances in Neural Information Processing Systems*, 33, 5516–5526.

[555] Wang, X., Pan, W., Hu, W., Tian, Y., & Zhang, H. (2015). Conditional distance correlation. *Journal of the American Statistical Association*, 110(512), 1726–1734.

[556] Warner, S. L. (1965). Randomized response: A survey technique for eliminating evasive answer bias. *Journal of the American Statistical Association*, 60(309), 63–69.

[557] Warzel, C. (2020). Chinese hacking is alarming. so are data brokers. online accessed February 2020 https://www.nytimes.com/2020/02/10/opinion/equifax-breach-china-hacking.html.

[558] Wasserman, L. & Zhou, S. (2010). A statistical framework for differential privacy. *Journal of the American Statistical Association*, 105(489), 375–389.

[559] Wei, K., Iyer, R., & Bilmes, J. (2015). Submodularity in data subset selection and active learning. In *International Conference on Machine Learning* (pp. 1954–1963).: PMLR.

[560] Wei, K., Li, J., Ding, M., Ma, C., Yang, H. H., Farokhi, F., Jin, S., Quek, T. Q., & Poor, H. V. (2020). Federated learning with differential privacy: Algorithms and performance analysis. *IEEE Transactions on Information Forensics and Security*, 15, 3454–3469.

[561] Weisse, O., Van Bulck, J., Minkin, M., Genkin, D., Kasikci, B., Piessens, F., Silberstein, M., Strackx, R., Wenisch, T. F., & Yarom, Y. (2018). Foreshadow-NG: Breaking the virtual memory abstraction with transient out-of-order execution. *Technical report*.

[562] Welinder, P., Branson, S., Mita, T., Wah, C., Schroff, F., Belongie, S., & Perona, P. (2010). *Caltech-UCSD Birds 200*. Technical Report CNS-TR-2010-001, California Institute of Technology.

[563] Wen, Y., Song, L., Liu, B., Ding, M., & Xie, R. (2021). Identitydp: Differential private identification protection for face images. *arXiv preprint arXiv:2103.01745*.

[564] Whitten, A. & Tygar, J. D. (1999). Why johnny cant encrypt: A usability evaluation of pgp 5.0. In *Proceedings of the 8th Conference on USENIX Security Symposium - Volume 8*, SSYM99 (pp.$\tilde{1}$4). USA: USENIX Association.

[565] Williams, C. K. & Rasmussen, C. E. (2006). *Gaussian processes for machine learning*, volume 2. MIT press Cambridge, MA.

[566] Wu, T. T., Lange, K., et al. (2010). The mm alternative to em. *Statistical Science*, 25(4), 492–505.

[567] Wu, X., Fredrikson, M., Jha, S., & Naughton, J. F. (2016). A methodology for formalizing model-inversion attacks. In *2016 IEEE 29th Computer Security Foundations Symposium (CSF)* (pp. 355–370).

[568] Wu, Y., Yang, F., & Ling, H. (2018a). Privacy-protective-gan for face de-identification. *arXiv preprint arXiv:1806.08906*.

[569] Wu, Z., Wang, Z., Wang, Z., & Jin, H. (2018b). Towards privacy-preserving visual recognition via adversarial training: A pilot study. *ArXiv*, abs/1807.08379.

[570] Wu, Z., Wang, Z., Wang, Z., & Jin, H. (2018c). Towards privacy-preserving visual recognition via adversarial training: A pilot study. In *Proceedings of the European Conference on Computer Vision (ECCV)* (pp. 606–624).

[571] Xia, Z., Zhu, Y., Sun, X., Qin, Z., & Ren, K. (2015). Towards privacy-preserving content-based image retrieval in cloud computing. *IEEE Transactions on Cloud Computing*, 6(1), 276–286.

[572] Xiao, H., Rasul, K., & Vollgraf, R. (2017). Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms. *arXiv preprint arXiv:1708.07747*.

[573] Xie, L., Lin, K., Wang, S., Wang, F., & Zhou, J. (2018). Differentially private generative adversarial network. *CoRR*, abs/1802.06739.

[574] Xu, C., Ren, J., Zhang, D., Zhang, Y., Qin, Z., & Ren, K. (2019). Ganobfuscator: Mitigating information leakage under gan via differential privacy. *IEEE Transactions on Information Forensics and Security*, 14, 2358–2371.

[575] Xu, C., Ren, J., Zhang, Y., Qin, Z., & Ren, K. (2017). Dppro: Differentially private high-dimensional data release via random projection. *IEEE Transactions on Information Forensics and Security*, 12(12), 3081–3093.

[576] Yan, M., Fletcher, C. W., & Torrellas, J. (2018). Cache telepathy: Leveraging shared resource attacks to learn dnn architectures. *ArXiv*, abs/1808.04761.

[577] Yang, J., Shi, R., Wei, D., Liu, Z., Zhao, L., Ke, B., Pfister, H., & Ni, B. (2021). Medmnist v2: A large-scale lightweight benchmark for 2d and 3d biomedical image classification. *arXiv preprint arXiv:2110.14795*.

[578] Yang, K., Fan, T., Chen, T., Shi, Y., & Yang, Q. (2019a). A quasi-newton method based vertical federated learning framework for logistic regression. *arXiv preprint arXiv:1912.00513*.

[579] Yang, S., Ren, B., Zhou, X., & Liu, L. (2019b). Parallel distributed logistic regression for vertical federated learning without third-party coordinator. *arXiv preprint arXiv:1911.09824*.

[580] Yang, Z., Chang, E.-C., & Liang, Z. (2019c). Adversarial neural network inversion via auxiliary knowledge alignment. *ArXiv*, abs/1902.08552.

[Yao] Yao, A. C. Protocols for secure computations. *University of California Berkeley, California, IEEE Foundations of Computer Science*, 23.

[582] Yao, B., Li, F., & Xiao, X. (2013). Secure nearest neighbor revisited. In *2013 IEEE 29th international conference on data engineering (ICDE)* (pp. 733–744).: IEEE.

[583] Yao, S., Zhang, X., & Shao, X. (2018). Testing mutual independence in high dimension via distance covariance. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 80(3), 455–480.

[584] Yeom, S., Giacomelli, I., Fredrikson, M., & Jha, S. (2018). Privacy risk in machine learning: Analyzing the connection to overfitting. In *2018 IEEE 31st Computer Security Foundations Symposium (CSF)* (pp. 268–282).

[585] Yger, F., Berar, M., & Lotte, F. (2016). Riemannian approaches in brain-computer interfaces: a review. *IEEE Transactions on Neural Systems and Rehabilitation Engineering*, 25(10), 1753–1762.

[586] Yonetani, R., Boddeti, V. N., Kitani, K. M., & Sato, Y. (2017a). Privacy-preserving visual learning using doubly permuted homomorphic encryption. *2017 IEEE International Conference on Computer Vision (ICCV)*, (pp. 2059–2069).

[587] Yonetani, R., Boddeti, V. N., Kitani, K. M., & Sato, Y. (2017b). Privacy-preserving visual learning using doubly permuted homomorphic encryption. In *IEEE International Conference on Computer Vision* (pp. 2059–2069).

[588] Yu, K. & Salzmann, M. (2017). Second-order convolutional neural networks. *arXiv preprint arXiv:1703.06817*.

[589] Yu, L., Liu, L., Pu, C., Gursoy, M. E., & Truex, S. (2019a). Differentially private model publishing for deep learning. *CoRR*, abs/1904.02200.

[590] Yu, Q., Li, S., Raviv, N., Kalan, S. M. M., Soltanolkotabi, M., & Avestimehr, S. A. (2019b). Lagrange coded computing: Optimal design for resiliency, security, and privacy. In *The 22nd International Conference on Artificial Intelligence and Statistics* (pp. 1215–1225).: PMLR.

[591] Yu, Q., Maddah-Ali, M., & Avestimehr, S. (2017). Polynomial codes: an optimal design for high-dimensional coded matrix multiplication. *Advances in Neural Information Processing Systems*, 30.

[592] Yu, Q., Maddah-Ali, M. A., & Avestimehr, A. S. (2020). Straggler mitigation in distributed matrix multiplication: Fundamental limits and optimal coding. *IEEE Transactions on Information Theory*, 66(3), 1920–1933.

[593] Yuille, A. L. & Rangarajan, A. (2002). The concave-convex procedure (cccp). In *Advances in neural information processing systems* (pp. 1033–1040).

[594] Zaks, Y. M. & Muchnik, I. (1989). Incomplete classifications of a finite set of objects using monotone systems. *Automation and Remote Control*, 50, 553–560.

[595] Zanini, P., Congedo, M., Jutten, C., Said, S., & Berthoumieu, Y. (2017). Transfer learning: A riemannian geometry framework with applications to brain–computer interfaces. *IEEE Transactions on Biomedical Engineering*, 65(5), 1107–1116.

[596] Zeng, M., Wu, Z., Tian, C., Zhang, L., & Hu, L. (2015). Efficient person re-identification by hybrid spatiogram and covariance descriptor. In *Proceedings of the IEEE conference on computer vision and pattern recognition workshops* (pp. 48–56).

[597] Zhang, B. H., Lemoine, B., & Mitchell, M. (2018a). Mitigating unwanted biases with adversarial learning. In *Proceedings of the 2018 AAAI/ACM Conference on AI, Ethics, and Society* (pp. 335–340).: ACM.

[598] Zhang, H., Zhou, S., Zhang, K., & Guan, J. (2017). Causal discovery using regression-based conditional independence tests. In *Thirty-First AAAI Conference on Artificial Intelligence*.

[599] Zhang, K., Peters, J., Janzing, D., & Schölkopf, B. (2012). Kernel-based conditional independence test and application in causal discovery. *arXiv preprint arXiv:1202.3775*.

[600] Zhang, Q., Filippi, S., Gretton, A., & Sejdinovic, D. (2018b). Large-scale kernel methods for independence testing. *Statistics and Computing*, 28(1), 113–130.

[601] Zhang, T., He, Z., & Lee, R. B. (2018c). Privacy-preserving machine learning through data obfuscation. *ArXiv*, abs/1807.01860.

[602] Zhang, W. E., Sheng, Q. Z., Alhazmi, A., & Li, C. (2020). Adversarial attacks on deep-learning models in natural language processing: A survey. *ACM Transactions on Intelligent Systems and Technology (TIST)*, 11(3), 1–41.

[603] Zhang, Y. & Li, S. (2011). Gabor-lbp based region covariance descriptor for person re-identification. In *2011 Sixth International Conference on Image and Graphics* (pp. 368–371).: IEEE.

[604] Zhang, Z., Wang, T., Li, N., Honorio, J., Backes, M., He, S., Chen, J., & Zhang, Y. (2021). Privsyn: Differentially private data synthesis. In *30th {USENIX} Security Symposium ({USENIX} Security 21)*.

[605] Zhao, L., Zhang, Y., Wang, Q., Chen, Y., Wang, C., & Zou, Q. (2018). Privacy-preserving collaborative deep learning with irregular participants. *CoRR*, abs/1812.10113.

[606] Zheng, Y., Li, Q., Chen, Y., Xie, X., & Ma, W.-Y. (2008). Understanding mobility based on gps data. In *Proceedings of the 10th international conference on Ubiquitous computing* (pp. 312–321).

[607] Zheng, Y., Xie, X., Ma, W.-Y., et al. (2010). Geolife: A collaborative social networking service among user, location and trajectory. *IEEE Data Eng. Bull.*, 33(2), 32–39.

[608] Zheng, Y., Zhang, L., Xie, X., & Ma, W.-Y. (2009). Mining interesting locations and travel sequences from gps trajectories. In *Proceedings of the 18th international conference on World wide web* (pp. 791–800).

[609] Zhong, W. & Zhu, L. (2015). An iterative approach to distance correlation-based sure independence screening. *Journal of Statistical Computation and Simulation*, 85(11), 2331–2345.

[610] Zhou, H., Hu, L., Zhou, J., & Lange, K. (2019). Mm algorithms for variance components models. *Journal of Computational and Graphical Statistics*, 28(2), 350–361.

[611] Zhou, W., Li, H., & Tian, Q. (2017). Recent advance in content-based image retrieval: A literature survey. *arXiv preprint arXiv:1706.06064*.