UNIVERSITY OF CALIFORNIA SAN DIEGO

The Primacy of Applied Privacy

A dissertation submitted in partial satisfaction of the
requirements for the degree Doctor of Philosophy

in

Computer Science

by

Casey Meehan

Committee in charge:

Professor Kamalika Chaudhuri, Chair
Professor Taylor Berg-Kirkpatrick
Professor Sanjoy Dasgupta
Professor Alon Orlitsky

2023

The Dissertation of Casey Meehan is approved, and it is acceptable in quality and form for publication on microfilm and electronically.

University of California San Diego

2023

DEDICATION

The fact that I have made something I can write a dedication for is owed all to my parents. I cannot imagine following my heart these past few years without their unrelenting support and encouragement.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

ACKNOWLEDGEMENTS

Above all, I thank my advisor, Kamalika, for her mentorship over the last five years. Kamalika will do anything in her power to help her students find what success means to them, and build towards it. She puts her students' futures above all else. Part of that is mentoring her students to become thoughtful researchers. In hindsight, I see that we could have pursued much lower risk projects that made incremental gains on the hottest research directions. Instead, Kamalika guided me towards unusual and challenging problems that required me to examine my fundamentals and reconsider broad questions about the aims of data privacy. In doing so, I not only understand what solutions are effective in data privacy, but why we as a research community use them and — in some cases — why we have ignored them.

I cannot overemphasize the gratitude I have to my collaborators as well. First, to my collaborators at FAIR. I thank Chuan Guo for showing me how to navigate incredibly challenging and open-ended ML problems. His Socratic approach to mentoring helped me explore my own instincts and take ownership of the project without letting me slip down the wrong path. Florian Bordes taught me the ins and outs of contemporary vision modeling. Pascal Vincent's insights on how to design deep learning experiments were truly formative; I sincerely appreciate his effort and time. I owe an enormous debt of gratitude to Ashwin Machanavajjhala at Tumult for mentoring me on how how to reason about different privacy definitions and guarantees in applied settings.

Finally, to all of my friends. To the lab: I could not have landed in a better group. To be surrounded by such wonderful and bright companions is a win — for them to be your colleagues as well is a gift. To all my dear pre-PhD friends (you know who you are, and you really don't have read this dissertation): you are the loves of my life!

Chapter 1, in full, has been submitted for publication of the material as it may appear in Neural Information Processing Systems, 2023, Casey Meehan, Florian Bordes, Pascal Vincent, Kamalika Chaudhuri, Chuan Guo. *Do SSL models have Déjà Vu? A Case of Unintended Memorization in Self-Supervised Learning*. The dissertation author shares equal contribution with Florian Bordes.

Chapter 2, in full, is a reprint of the material as it appears in International Conference on Artificial Intelligence and Statistics, 2020. Casey Meehan, Sanjoy Dasgupta, Kamalika Chaudhuri. *A Non-parametric Test to Detect Data-Copying in Generative Models*. The dissertation author is the primary investigator and author of this paper.

Chapter 3, in full, is a reprint of the material as it appears in Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), 2022. Casey Meehan, Khalil Mrini, Kamalika Chaudhuri. *Sentence-level Privacy for Document Embeddings*. The dissertation author is the primary investigator and author of this paper.

Chapter 4, in full, is a reprint of the material as it appears in International Conference on Learning Representations. 2022. Casey Meehan, Amrita Roy-Chowdhury, Kamalika Chaudhuri, Somesh Jha. *Privacy Implications of Shuffling*. The dissertation author is the primary investigator and author of this paper.

Chapter 5, in full, is a reprint of the material as it appears in International Conference on Artificial Intelligence and Statistics, 2021. Casey Meehan, Kamalika Chaudhuri. *Location Trace Privacy Under Conditional Priors*. The dissertation author is the primary investigator and author of this paper.

VITA

2015 Bachelor of Science, Brown University

2018 Master of Science, Harvard University

2023 Doctor of Philosophy, University of California, San Diego

ABSTRACT OF THE DISSERTATION

The Primacy of Applied Privacy

by

Casey Meehan

Doctor of Philosophy in Computer Science

University of California San Diego, 2023

Professor Kamalika Chaudhuri, Chair

As data collection for machine learning (ML) tasks has become more pervasive, it has also become more heterogeneous: we share our writing, images, voices, and location online every day. Naturally, the associated privacy risks are just as complex and variable. My research advances practical data privacy through two avenues: 1) drafting provable privacy definitions and mechanisms for safely sharing data in different ML domains, and 2) empirically quantifying how ML models memorize their sensitive training data and thereby risk disclosing it. This dissertation details the various data domains/tasks considered, and the corresponding privacy methods proposed.

Introduction

In machine learning (ML) we learn broad trends and patterns from vast sums of data. What type of data is collected and how it is used introduces different kinds of privacy risks. Medical data allows us to link characteristics like genetic markers to adverse health conditions like cancer. However it suffers from unique correlation risks, since one's information can reliably be correlated from their familys'. Document embeddings—useful vector representations of text documents—are tremendously useful for inferring general document characteristics like topic, but can expose detailed personal information at the sentence level. Generative vision models can sample novel images, but also tend to copy and expose their training images in clever and inexact ways.

The privacy risks and utility requirements of each of these settings and applications warrant different approaches to privacy-preserving ML. This dissertation proposes a variety of solutions to situations like those above. In doing so, I hope to illuminate the advantage of taking an application-specific approach to both measuring privacy risks and engineering private algorithms. The following five chapters are roughly organized into two parts: the first two chapters cover *empirical* privacy methods, and the remaining three cover *formal* privacy methods. The former includes statistical tests to quantify privacy risks of large ML models. The latter proposes provably private algorithms to satisfy different privacy definitions chosen for different ML tasks.

Empirical methods. The first two chapters explore empirically measuring privacy risks in the domain of vision modeling. In both chapters, we analyze to what extent large vision models *memorize* their training images, and thereby risk exposing them. In contrast with the

following three chapters, we are not proposing a provably privacy-preserving algorithm. Instead we are designing methodical empirical tests to quantify memorization.

Provably private algorithms. The final three chapters propose algorithms that allow us to share our data in a provably private way. We explore privacy preserving algorithms in the text, location, and interpersonal-correlated domains (*e.g.* social networks or genetically linked medical data). For each of these, we study different ML tasks and privacy risks to motivate different privacy definitions and provably private algorithms.

Taken together, this document offers a mindset towards practicable privacy methods. By directly considering the data domain and the task at hand, it is possible to efficiently measure privacy risks and propose provably private algorithms while still completing the learning task at hand .

Chapter 1

Do SSL Models Have Déjà Vu? A Case of Unintended Memorization in Self-supervised Learning

1.1 Introduction

Self-supervised learning (SSL) [43, 44, 181, 14, 36, 90] aims to learn general representations of content-rich data without explicit labels by solving a *pretext task*. In many recent works, such pretext tasks rely on joint-embedding architectures whereby randomized image augmentations are applied to create multiple views of a training sample, and the model is trained to produce similar representations for those views. When using cropping as random image augmentation, the model learns to associate objects or parts (including the background scenery) that co-occur in an image. However, doing so also arguably exposes the training data to higher privacy risk as objects in training images can be explicitly memorized by the SSL model. For example, if the training data contains the photos of individuals, the SSL model may learn to associate the face of a person with their activity or physical location in the photo. This may allow an adversary to extract such information from the trained model for targeted individuals.

In this work, we aim to evaluate to what extent SSL models memorize the association of specific objects in training images or the association of objects and their specific backgrounds, and whether this memorization signal can be used to reconstruct the model's training samples.

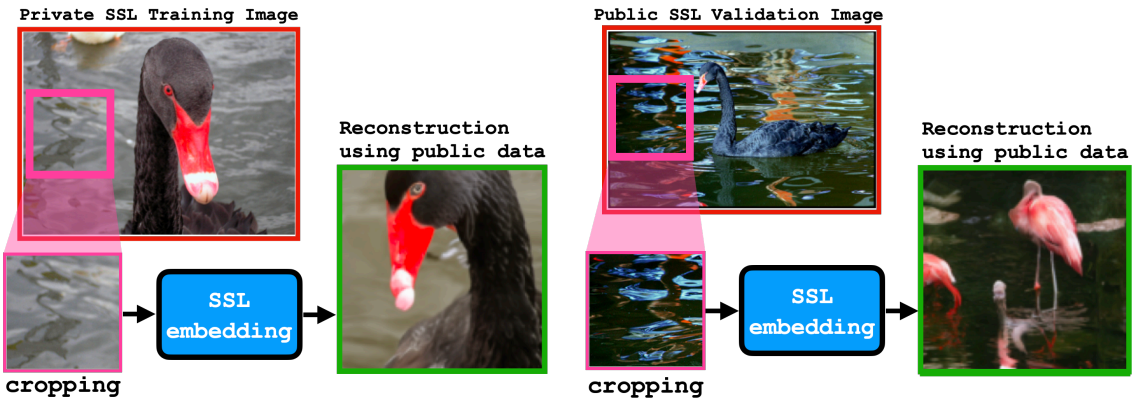


Figure 1.1. Left: Reconstruction of an SSL training image from a crop containing only the background. The SSL model memorizes the association of this *specific* patch of water (pink square) to this *specific* foreground object (a black swan) in its embedding, which we decode to visualize the full training image. **Right:** The reconstruction technique fails on a public test image that the SSL model has not seen before.

Our results demonstrate that SSL models memorize such associations beyond simple correlation. For instance, in Figure 1.1 (**left**), we use the SSL representation of a *training image crop containing only water* and this enables us to reconstruct the object in the foreground with remarkable specificity—in this case a black swan. By contrast, in Figure 1.1 (**right**), when using the *crop from the background of a test set image* that the SSL model *has not seen before*, its representation only contains enough information to infer, through correlation, that the foreground object was likely some kind of waterbird — but not the specific one in the image.

Figure 1.1 shows that SSL models suffer from the unintended memorization of images in their training data—a phenomenon we refer to as *déjà vu memorization* [1] Beyond visualizing *déjà vu* memorization through data reconstruction, we also design a series of experiments to quantify the degree of memorization for different SSL algorithms, model architectures, training set size, *etc.* We observe that *déjà vu* memorization is exacerbated by the atypically large number of training epochs often recommended in SSL training, as well as certain hyperparameters in the SSL training objective. Perhaps surprisingly, we show that *déjà vu* memorization occurs even when the training set is large—as large as half of ImageNet [50]—and can continually

[1]The French loanword *déjà vu* means 'already-seen', just as an image is seen and memorized in training.

worsen even when standard techniques for evaluating learned representation quality (such as linear probing) do not suggest increased overfitting. Our work serves as the first systematic study of unintended memorization in SSL models and motivates future work on understanding and preventing this behavior. Specifically, we:

- Elucidate how SSL representations memorize aspects of individual training images, what we call *déjà vu* memorization;
- Design a novel training data reconstruction pipeline for non-generative vision models. This is in contrast to many prominent reconstruction algorithms like [34, 32], which rely on the model itself to generate its own memorized samples and is not possible for SSL models or classifiers;
- Propose metrics to quantify the degree of *déjà vu* memorization committed by an SSL model. This allows us to observe how *déjà vu* changes with training epochs, dataset size, training criteria, model architecture and more.

1.2 Preliminaries and Related Work

Self-supervised learning (SSL) is a machine learning paradigm that leverages unlabeled data to learn representations. Many SSL algorithms rely on *joint-embedding* architectures (*e.g.*, SimCLR [43], Barlow Twins [181], VICReg [14] and Dino [37]), which are trained to associate different augmented views of a given image. For example, in SimCLR, given a set of images $\mathscr{A} = \{A_1, \ldots, A_n\}$ and a randomized augmentation function aug, the model is trained to maximize the cosine similarity of draws of $\mathrm{SSL}(\mathrm{aug}(A_i))$ with each other and minimize their similarity with $\mathrm{SSL}(\mathrm{aug}(A_j))$ for $i \neq j$. The augmentation function aug typically consists of operations such as cropping, horizontal flipping, and color transformations to create different views that preserve an image's semantic properties.

SSL representations.

Once an SSL model is trained, its learned representation can be transferred to different downstream tasks. This is often done by extracting the representation of an image from the *backbone model*[2] and either training a linear probe on top of this representation or finetuning the backbone model with a task-specific head [24]. It has been shown that SSL representations encode richer visual details about input images than supervised models do [25]. However, from a privacy perspective, this may be a cause for concern as the model also has more potential to overfit and memorize precise details about the training data compared to supervised learning. We show concretely that this privacy risk can indeed be realized by defining and measuring *déjà vu* memorization.

Privacy risks in ML.

When a model is overfit on privacy-sensitive data, it memorizes specific information about its training examples, allowing an adversary with access to the model to learn private information [176, 71]. Privacy attacks in ML range from the simplest and best-studied *membership inference attacks* [143, 141, 139] to *attribute inference* [76, 127, 95] and *data reconstruction* [34, 13, 88] attacks. In the former, the adversary only infers whether an individual participated in the training set. Our study of *déjà vu* memorization is most similar to the latter: we leverage SSL representations of the training image background to infer and reconstruct the foreground object. Our approach reflects similar work in the NLP domain [33, 34]: when prompted with a context string present in the training data, a large language model is shown to generate the remainder of string at test time, revealing sensitive text like home addresses. This method was recently extended to extract memorized images from Stable Diffusion [32]. We exploit memorization in a similar manner: given partial information about a training sample, the model is prompted to reveal the rest of the sample. In our case, however, since the SSL model is not generative, extraction is significantly harder and requires careful design.

[2]SSL methods often use a trick called *guillotine regularization* [24], which decomposes the model into two parts: a *backbone model* and a *projector* consisting of a few fully-connected layers. Such trick is needed to handle the misalignment between the pretext SSL task and the downstream task.

1.3 Defining *Déjà Vu* Memorization

What is *déjà vu* memorization?

At a high level, the objective of SSL is to learn general representations of objects that occur in nature. This is often accomplished by associating different parts of an image with one another in the learned embedding. Returning to our example in Figure 1.1, given an image whose background contains a patch of water, the model may learn that the foreground object is a water animal such as duck, pelican, otter, *etc.*, by observing different images that contain water from the training set. We refer to this type of learning as *correlation*: the association of objects that tend to co-occur in images from the training data distribution.

A natural question to ask is *"Can the reconstruction of the black swan in Figure 1.1 be reasoned as correlation?"* The intuitive answer may be no, since the reconstructed image is qualitatively very similar to the original image. However, this reasoning implicitly assumes that for a random image from the training data distribution containing a patch of water, the foreground object is unlikely to be a black swan. Mathematically, if we denote by $\mathscr{P}$ the training data distribution and A the image, then

$$p_{\text{corr}} := \mathbb{P}_{A\sim\mathscr{P}}(\text{object}(A) = \texttt{black swan} \mid \text{crop}(A) = \texttt{water})$$

is the probability of inferring that the foreground object is a black swan through *correlation*. This probability may be naturally high due to biases in the distribution $\mathscr{P}$, *e.g.*, if $\mathscr{P}$ contains no other water animal except for black swans. In fact, such correlations are often exploited to learn a model for image inpainting with great success [178, 155].

Despite this, we argue that reconstruction of the black swan in Figure 1.1 is *not* due to correlation, but rather due to *unintended memorization*: the association of objects unique to a single training image. As we will show in the following sections, the example in Figure 1.1 is not a rare success case and can be replicated across many training samples. More importantly,

failure to reconstruct the foreground object in Figure 1.1 (**right**) on test images hints at inferring through correlation is unlikely to succeed—a fact that we verify quantitatively in Section 1.4.1. Motivated by this discussion, we give a verbal definition of *déjà vu* memorization below, and design a testing methodology to quantify *déjà vu* memorization in Section 1.3.1.

Definition: A model exhibits *déjà vu memorization* when it retains information so specific to an individual training image, that it enables recovery of aspects particular to that image given a part that does not contain them. The recovered aspect must be beyond what can be inferred using only correlations in the data distribution.

We intentionally kept the above definition broad enough to encompass different types of information that can be inferred about the training image, including but not restricted to object category, shape, color and position. For example, if one can infer that the foreground object is red given the background patch with accuracy significantly beyond correlation, we consider this an instance of *déjà vu* memorization as well. We mainly focus on object category to quantify *déjà vu* memorization in Section 1.4 since the ground truth label can be easily obtained. We consider other types of information more qualitatively in the visual reconstruction experiments in Section 1.5.

Privacy implications of *déjà vu* memorization.

Déjà vu memorization can be a cause for concern when the training data contains privacy-sensitive information. As a motivating example, consider an SSL model trained on photos of individuals. If the model exhibits *déjà vu* memorization then, given the face of an individual, it may be possible to infer where the individual was or even visually reconstruct their location in the training image. Such information leakage raises privacy concerns, especially if there was no prior agreement that the trained model may reveal such information to third parties. This hypothetical scenario serves as a motivation that *déjà vu* memorization should be carefully examined to avoid unintended disclosure of private information in practical applications.

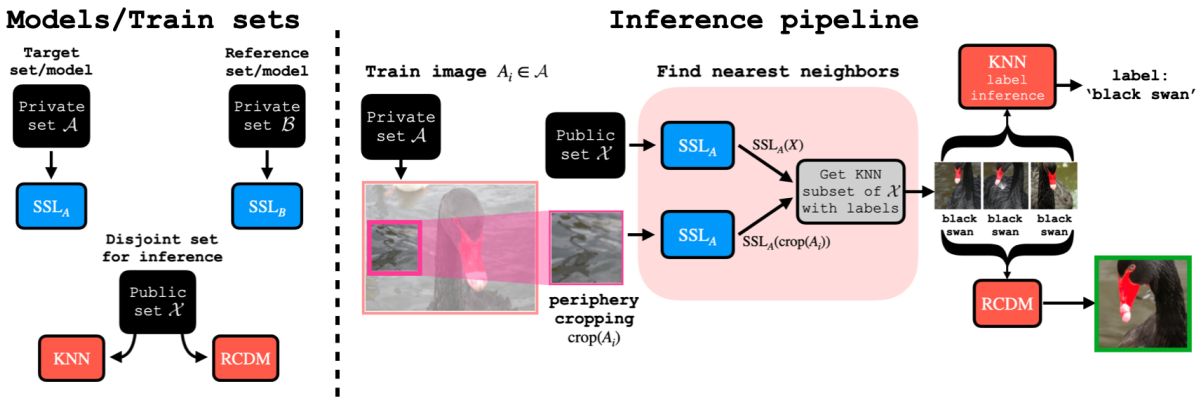


Figure 1.2. Overview of testing methodology. **Left:** Data is split into *target set* $\mathscr{A}$, *reference set* $\mathscr{B}$ and *public set* $\mathscr{X}$ that are pairwise disjoint. $\mathscr{A}$ and $\mathscr{B}$ are used to train two SSL models SSL_A and SSL_B in the same manner. $\mathscr{X}$ is used for KNN decoding or for training an RCDM to reconstruct the input at test time. **Right:** Given a training image $A_i \in \mathscr{A}$, we use SSL_A to embed $\text{crop}(A_i)$ containing only the background, as well as the entire set $\mathscr{X}$ and find the k-nearest neighbors of $\text{crop}(A_i)$ in $\mathscr{X}$ in the embedding space. These KNN samples can be used directly to infer the foreground object (*i.e.*, class label) in A_i using a KNN classifier, or their embeddings can be averaged as input to the trained RCDM to visually reconstruct the image A_i. For instance, the RCDM reconstruction results in Figure 1.1 (left) when given $\text{SSL}_A(\text{crop}(A_i))$ and results in Figure 1.1 (right) when given $\text{SSL}_A(\text{crop}(B_i))$ for an image $B_i \in \mathscr{B}$.

Distinguishing memorization from correlation. When measuring *déjà vu* memorization, it is crucial to differentiate what the model associates through *memorization* and what it associates through *correlation*. Our testing methodology is based on the following intuitive definition.

Definition: If an SSL model associates two parts in a training image, we say that it is due to *correlation* if other SSL models trained on a similar dataset from $\mathscr{P}$ without this image would likely make the same association. Otherwise, we say that it is due to *memorization*.

Notably, such intuition forms the basis for differential privacy (DP; [58, 61])—the most widely accepted notion of privacy in ML.

1.3.1 Testing Methodology for Measuring *Déjà Vu* Memorization

In this section, we use the above intuition to measure the extent of *déjà vu* memorization in SSL. Figure 1.2 gives an overview of our testing methodology.

Dataset splitting.

We focus on testing *déjà vu* memorization for SSL models trained on the ImageNet-1K dataset [50]. Our test first splits the ImageNet training set into three independent and disjoint subsets $\mathscr{A}$, $\mathscr{B}$ and $\mathscr{X}$. The dataset $\mathscr{A}$ is called the *target set* and $\mathscr{B}$ is called the *reference set*. The two datasets are used to train two separate SSL models, SSL_A and SSL_B, called the *target model* and the *reference model*. Finally, the dataset set $\mathscr{X}$ is used as an auxiliary public dataset to extract information from SSL_A and SSL_B. Our dataset splitting serves the purpose of distinguishing memorization from correlation in the following manner. Given a sample $A_i \in \mathscr{A}$, if our test returns the same result on SSL_A and SSL_B then it is likely due to correlation because A_i is not a training sample for SSL_B. Otherwise, because $\mathscr{A}$ and $\mathscr{B}$ are drawn from the same underlying distribution, our test must have inferred some information unique to A_i due to memorization. Thus, by comparing the difference in the test results for SSL_A and SSL_B, we can measure the degree of *déjà vu* memorization[3].

Extracting foreground and background crops.

Our testing methodology aims at measuring what can be inferred about the foreground object in an ImageNet sample given a background crop. This is made possible because ImageNet provides bounding box annotations for a subset of its training images—around 150K out of 1.3M samples. We split these annotated images equally between $\mathscr{A}$ and $\mathscr{B}$. Given an annotated image A_i, we treat everything inside the bounding box as the foreground object associated with the image label, denoted $\text{object}(A_i)$. We take the largest possible crop that does not intersect with any bounding box as the background crop (or *periphery crop*), denoted $\text{crop}(A_i)$[4]

[3] See Appendix A.0.2 for details on how the dataset splits are generated.

[4] We also present another heuristic in Appendix A.0.5 which takes a corner crop as the background crop, allowing our test to be run without bounding box annotations.

KNN-based test design.

Joint-embedding SSL approaches encourage the embeddings of random crops of a training image $A_i \in \mathscr{A}$ to be similar. Intuitively, if the model exhibits *déjà vu* memorization, it is reasonable to expect that the embedding of crop(A_i) is similar to that of object(A_i) since both crops are from the same training image. In other words, $\text{SSL}_A(\text{crop}(A_i))$ encodes information about object(A_i) that cannot be inferred through correlation. However, decoding such information is challenging as these approaches do not learn a decoder associated with the encoder SSL_A.

Here, we leverage the public set $\mathscr{X}$ to decode the information contained in crop(A_i) about object(A_i). More specifically, we map images in $\mathscr{X}$ to their embeddings using SSL_A and extract the k-nearest-neighbor (KNN) subset of $\text{SSL}_A(\text{crop}(A_i))$ in $\mathscr{X}$. We can then decode the information contained in crop(A_i) in one of two ways:

- *Label inference:* Since $\mathscr{X}$ is a subset of ImageNet, each embedding in the KNN subset is associated with a class label. If crop(A_i) encodes information about the foreground object, its embedding will be close to samples in $\mathscr{X}$ that have the same class label (*i.e.*, foreground object category). We can then use a KNN classifier to infer the foreground object in A_i given crop(A_i).

- *Visual reconstruction:* Following **(author?)** [25], we train an RCDM—a conditional generative model—on $\mathscr{X}$ to decode SSL_A embeddings into images. The RCDM reconstruction can recover qualitative aspects of an image remarkably well, such as recovering object color or spatial orientation using its SSL embedding. Given the KNN subset, we average their SSL embeddings and use the trained RCDM model to visually reconstruct A_i.

In Section 1.4, we focus on quantitatively measuring *déjà vu* memorization with label inference, and then use the RCDM reconstruction to visualize *déjà vu* memorization in Section 1.5.

1.4 Quantifying *Déjà Vu* Memorization

We apply our testing methodology to quantify a specific form of *déjà vu* memorization: inferring the foreground object (class label) given a crop of the background.

Extracting model embeddings.

We test *déjà vu* memorization on a variety of popular SSL algorithms, with a focus on VICReg [14]. These algorithms produce two embeddings given an input image: a *backbone* embedding and a *projector* embedding that is derived by applying a small fully-connected network on top of the backbone embedding. Unless otherwise noted, all SSL embeddings refer to the projector embedding. To understand whether *déjà vu* memorization is particular to SSL, we also evaluate embeddings produced by a supervised model CLF_A trained on $\mathscr{A}$. We apply the same set of image augmentations as those used in SSL and train CLF_A using the cross-entropy loss to predict ground truth labels.

Identifying the most memorized samples.

Prior works have shown that certain training samples can be identified as more prone to memorization than others [71, 161, 174]. Similarly, we provide a heuristic to identify the most memorized samples in our label inference test using confidence of the KNN prediction. Given a periphery crop, $\text{crop}(A_i)$, let $\text{KNN}_A\big(\text{crop}(A_i)\big) \subseteq \mathscr{X}$ denote its k-nearest neighbors in the embedding space of SSL_A. From this KNN subset we can obtain: **(1)** $\text{KNN}_A^{\text{prob}}\big(\text{crop}(A_i)\big)$, the vector of class probabilities (normalized counts) induced by the KNN subset, and **(2)** $\text{KNN}_A^{\text{conf}}\big(\text{crop}(A_i)\big)$, the negative entropy of the probability vector $\text{KNN}_A^{\text{prob}}\big(\text{crop}(A_i)\big)$, as confidence of the KNN prediction. When entropy is low, the neighbors agree on the class of A_i and hence confidence is high. We can sort the confidence score $\text{KNN}_A^{\text{conf}}\big(\text{crop}(A_i)\big)$ across samples A_i in decreasing order to identify the most confidently predicted samples, which likely correspond to the most memorized samples when $A_i \in \mathscr{A}$.

1.4.1 Population-level Memorization

Our first measure of *déjà vu* memorization is population-level label inference accuracy: *What is the average label inference accuracy over a subset of SSL training images given their periphery crops?* To understand how much of this accuracy is due to SSL_A's *déjà vu* memorization, we compare with a correlation baseline using the reference model: KNN_B's label inference accuracy on images $A_i \in \mathscr{A}$. In principle, this inference accuracy should be significantly above chance level ($1/1000$ for ImageNet) because the periphery crop may be highly indicative of the foreground object through correlation, *e.g.*, if the periphery crop is a basketball player then the foreground object is likely a basketball.

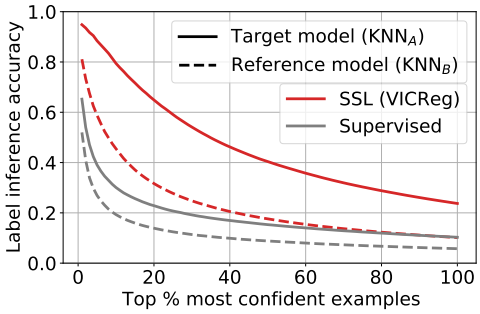


Figure 1.3. Accuracy of label inference using the target model (trained on $\mathscr{A}$) vs. the reference model (trained on $\mathscr{B}$) on the top % most confident examples $A_i \in \mathscr{A}$ using only $\text{crop}(A_i)$. For VICReg, there is a large accuracy gap between the two models, indicating a significant degree of *déjà vu* memorization.

Figure 1.3 compares the accuracy of KNN_A to that of KNN_B when inferring the labels of images in $A_i \in \mathscr{A}$[5] using $\text{crop}(A_i)$. Results are shown for VICReg and the supervised model; trends for other models are shown in Appendix A.0.3. For both VICReg and supervised models, inferring the class of $\text{crop}(A_i)$ using KNN_B (dashed line) through correlation achieves a reasonable accuracy that is significantly above chance level. However, for VICReg, the inference accuracy using KNN_A (solid red line) is significantly higher, and the accuracy gap between KNN_A and KNN_B indicates the degree of *déjà vu* memorization. We highlight two observations:

- The accuracy gap of VICReg is significantly larger than that of the supervised model. This is especially notable when accounting for the fact that the supervised model is trained to associate randomly augmented crops of images with their ground truth labels. In contrast, VICReg has no label access during training but the embedding of a periphery crop can still encode the image label.

[5]The sets $\mathscr{A}$ and $\mathscr{B}$ are exchangeable, and in practice we repeat this test on images from $\mathscr{B}$ using SSL_B as the target model and SSL_A as the reference model, and average the two sets of results.

- For VICReg, inference accuracy on the 1% most confident examples is nearly 95%, which shows that our simple confidence heuristic can effectively identify the most memorized samples. This result suggests that an adversary can use this heuristic to identify vulnerable training samples to launch a more focused privacy attack.

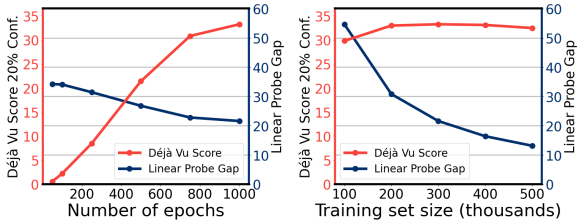
The *déjà vu* score.

The curves of Figure 1.3 show memorization across confidence values for a single training scenario. To study how memorization changes with different hyperparamters, we extract a single value from these curves: the *déjà vu score* at confidence level p. In Figure 1.3, this is the gap between the solid red (or gray) and dashed red (or gray) where confidence (x-axis) equal $p\%$. In other words, given the periphery crops of set $\mathscr{A}$, KNN_A and KNN_B separately select and label their top $p\%$ most confident examples, and we report the difference in their accuracy. The *déjà vu* score captures both the degree of memorization by the accuracy gap and the *ability to identify memorized examples* by the confidence level. If the score is 10% for $p = 33\%$, KNN_A has 10% higher accuracy on its most confident third of $\mathscr{A}$ than KNN_B does on its most confident third. In the following, we set $p = 20\%$, approximately the largest gap for VICReg (red lines) in Figure 1.3.

Comparison with the linear probe train-test gap.

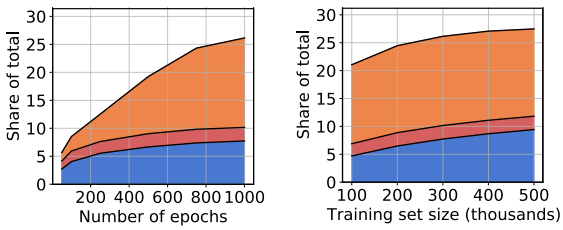
A standard method for measuring SSL performance is to train a linear classifier—what we call a 'linear probe'—on its embeddings and compute its performance on a held out test set. From a learning theory standpoint, one might expect the linear probe's train-test accuracy gap to be indicative of memorization: the more a model overfits, the larger is the difference between train set and test set accuracy. However, as seen in Figure 1.4, the linear probe gap (dark blue) fails to reveal memorization captured by the *déjà vu* score (red) [6].

[6] See section 1.6 for further discussion of the *déjà vu* score trends of Figure 1.4.



(a) *déjà vu* vs. epochs **(b)** *déjà vu* vs. train set size

Figure 1.4. Effect of training epochs and train set size with VICReg on *déjà vu* score (red) in comparison with linear probe accuracy train-test gap (dark blue). **Left:** *déjà vu* score increases with training epochs, indicating growing memorization while the linear probe baseline decreases significantly. **Right:** *déjà vu* score stays roughly constant with training set size suggesting that memorization may be problematic even for large datasets.



(a) *déjà vu* vs. epochs **(b)** *déjà vu* vs. train set size

Figure 1.5. Partition of samples $A_i \in \mathscr{A}$ into the four categories: unassociated (not shown), memorized, misrepresented and correlated for VICReg. The memorized samples—those whose labels are predicted by KNN_A but not by KNN_B—occupy a significantly larger share of the training set than the misrepresented samples—those predicted by KNN_B but not KNN_A by chance.

1.4.2 Sample-level Memorization

The *déjà vu* score shows, *on average*, how much better an adversary can select and classify images when using the target model trained on them. This average score does not tell us how many individual images have their label successfully recovered by KNN_A but not by KNN_B. In other words, how many images are exposed by virtue of *being in training set* $\mathscr{A}$: a risk notion foundational to differential privacy. To better quantify what fraction of the dataset is at risk, we perform a sample-level analysis by fixing a sample $A_i \in \mathscr{A}$ and observing the label inference result of KNN_A vs. KNN_B. To this end, we partition samples $A_i \in \mathscr{A}$ based on the result of label inference into four distinct categories: **Unassociated** - label inferred with neither KNN; **Memorized** - label inferred only with KNN_A; **Misrepresented** - label inferred only with KNN_B; **Correlated** - label inferred with both KNNs. Intuitively, unassociated samples are ones where the embedding of $\text{crop}(A_i)$ does not encode information about the label. Correlated samples are ones where the label can be inferred from $\text{crop}(A_i)$ using correlation, *e.g.*, inferring the foreground object is basketball given a crop showing a basketball player. Ideally, the misrepresented set should be empty but contains a small portion of examples due to chance. *Déjà vu* memorization occurs for memorized samples where the embedding of SSL_B does not encode the label but the

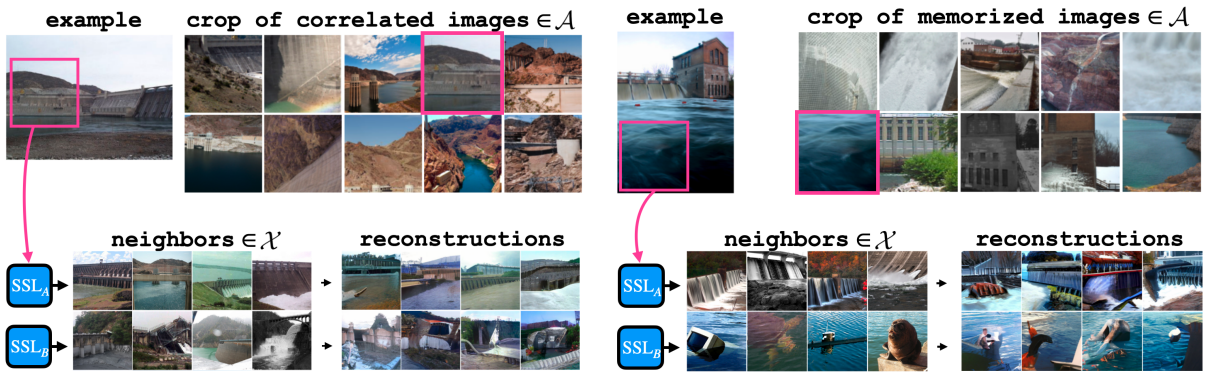
embedding of SSL_A does. To measure the pervasiveness of *déjà vu* memorization, we compare the size of the memorized and misrepresented sets. Figure 1.5 shows how the four categories of examples change with number of training epochs and training set size. The unassociated set is not shown since the total share adds up to one. The misrepresented set remains under 5% and roughly unchanged across all settings, consistent with our explanation that it is due to chance. In comparison, VICReg's memorized set surpasses 15% at 1000 epochs. Considering that up to 5% of these memorized examples could also be due to chance, we conclude that **at least 10% of VICReg's training set is *déjà vu* memorized.**

1.5 Visualizing *Déjà Vu* Memorization

Beyond enabling label inference using a periphery crop, we show that *déjà vu* memorization allows the SSL model to encode other forms of information about a training image. Namely, we train an RCDM [25] on the public dataset $\mathscr{X}$ and use it to visually reconstruct training images given their periphery crop. We aim to answer the following two questions: **(1)** Can we visualize the distinction between correlation and *déjà vu* memorization? **(2)** What foreground object details can be extracted from the SSL model beyond class label?

Reconstruction pipeline.

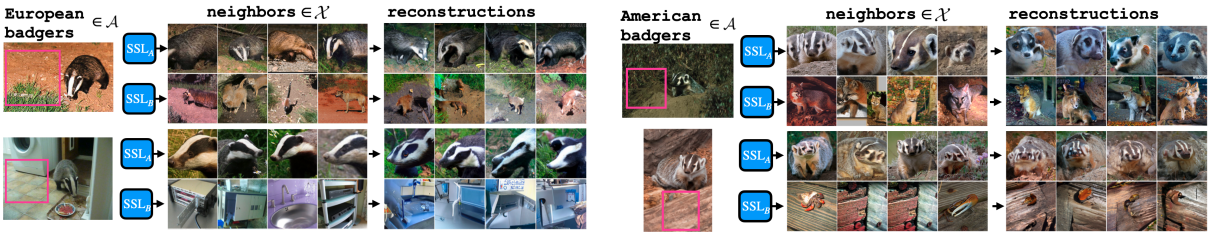
RCDM is a conditional generative model that is trained on the *backbone embedding* of images $X_i \in \mathscr{X}$ to generate an image that resembles X_i. All training images are first face-blurred for privacy purposes. **(author?)** [25] showed that the backbone embedding of SSL models contains more low-level information about the image, making them better suited for conditioning the RCDM. At test time, following the pipeline in Figure 1.2, we first use the projector embedding to find the KNN subset for the periphery crop, $\text{crop}(A_i)$, and then average their backbone embeddings as input to the RCDM model. Ideally, when the public set contains enough representative images, the average representation of the KNN subset encodes objects present in A_i, and the RCDM model decodes this representation to visualize these objects.



(a) A correlated dam example

(b) A memorized dam example

Figure 1.6. Correlated and Memorized examples from the *dam* class. Both SSL_A and SSL_B are SimCLR models. **Left:** The periphery crop (pink square) contains a concrete structure that is often present in images of dams. Consequently, the trained RCDM can reconstruct the foreground object using representations from both SSL_A and SSL_B through this correlation. **Right:** The periphery crop only contains a patch of water. The embedding produced by SSL_B only contains enough information to infer that the foreground object is related to water, as reflected by its KNN set and RCDM reconstruction. In contrast, the embedding produced by SSL_A memorizes the association of this patch of water with dam and the RCDM can visualize the embedding to produce images of dams.



(a) Memorized European badgers

(b) Memorized American badgers

Figure 1.7. Visualization of *déjà vu* memorization beyond class label. Both SSL_A and SSL_B are VICReg models. The four images shown belong to the memorized set of SSL_A from the *badger* class. RCDM reconstruction using embeddings from SSL_A can reveal not only the correct class label, but also the specific badger species: *European* (left) and *American* (right). Such information does not appear to be memorized by the reference model SSL_B.

Visualizing Correlation vs. Memorization.

Figure 1.6 shows examples of dams from the correlated set (left) and the memorized set (right) as defined in Section 1.4.2, along with the associated KNN set and RCDM reconstruction. Both SSL_A and SSL_B are SimCLR models. In Figure 1.6a, the periphery crop is represented by the pink square, which contains concrete structure attached to the dam's main structure.

As a result, both SSL_A and SSL_B produce embeddings of $\text{crop}(A_i)$ whose KNN set in $\mathscr{X}$ consist of dams, *i.e.*, there is a correlation between the concrete structure in $\text{crop}(A_i)$ and the foreground dam. The RCDM reconstructions also consist of dams or structures that closely resemble dams. In Figure 1.6b, the periphery crop only contains a patch of water, which does not strongly correlate with dams in the ImageNet distribution. Evidently, the reference model SSL_B embeds $\text{crop}(A_i)$ close to that of other objects commonly found in water, such as sea turtle and submarine. In contrast, the KNN set according to SSL_A all contain dams despite the vast number of alternative possibilities within the ImageNet classes, and the RCDM reconstruction outputs dams as well which highlight memorization in SSL_A between this specific patch of water and the dam.

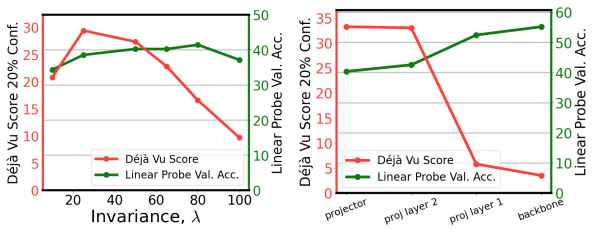
Visualizing Memorization Beyond Class Label.

We now use our reconstruction algorithm to show that *déjà vu* memorization can be exploited to reveal detailed information beyond class label. Figure 1.7 shows four examples of badgers from the memorized set. In all four images, the periphery crop (pink square) does not contain any indication that the foreground object is a badger. Despite this, the KNN set and the RCDM reconstruction using SSL_A consistently produce images of badgers, while the same does not hold for SSL_B. More interestingly, reconstructions using SSL_A in Figure 1.7a all contain *European* badgers, while reconstructions in Figure 1.7b all contain *American* badgers, accurately reflecting the species of badger present in the respective training images. Since ImageNet-1K does *not* differentiate between these two species of badgers, our reconstructions show that SSL models can memorize information that is highly specific to a training sample beyond its class label[7].

1.6 Mitigation of *déjà vu* memorization

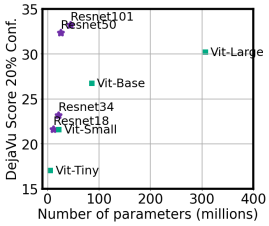
We cannot yet make claims on why *déjà vu* occurs so strongly for some SSL training settings and not for others. To gain some intuition for future work, we present additional

[7] See Appendix A.0.4 for additional visualization experiments.



(a) Loss hyper-parameter **(b)** Guillotine regularization

Figure 1.8. Effect of two kinds of hyper-parameters on VICReg memorization. **Left:** *déjà vu* score (red) versus the *invariance* loss parameter, λ, used in the VICReg criterion (100k dataset). Larger λ significantly reduces *déjà vu* , with minimal effect on linear probe validation performance (green). $\lambda = 25$ (near maximum *déjà vu*) is recommended in the original paper **Right:** *déjà vu* score versus projector layer—guillotine regularization [24]—from projector to backbone. Removing the projector can significantly reduce *déjà vu* . Appendix A.0.3 shows that the backbone still can memorize, however; we demonstrate reconstructions using the SimCLR backbone.



(a) *déjà vu* vs. capacity

Criteria	DV	Acc P/B
Supervised	8.9	55.3/61.1
Byol[85]	8.0	54.3/59.4
SimCLR[43]	10.0	44.2/54.1
Dino[37]	14.5	26.3/55.7
Barlow T.[181]	30.5	33.7/54.4
VICReg[14]	**33.2**	40.3/55.2

(b) *déjà vu* (DV) vs. Criterion

Figure 1.9. Effect of model architecture and criterion on *déjà vu* memorization. **Left:** *déjà vu* score with VICReg for resnet (purple) and vision transformer (green) architectures versus number of model parameters. As expected, memorization grows with larger model capacity. This trend is more pronounced for convolutional (resnet) than transformer (ViT) architectures. **Right:** Comparison of *déjà vu* score 20% conf. and ImageNet linear probe validation accuracy (P: using projector embeddings, B: using backbone embeddings) for various SSL criteria.

observations that shed light on which parameters have the most salient impact on *déjà vu* memorization.

Déjà vu memorization worsens by increasing number of training epochs.

Figure 1.4a shows how *déjà vu* memorization changes with number of training epochs for VICReg. The training set size is fixed to 300K samples. From 250 to 1000 epochs, the *déjà vu* score (red curve) grows *threefold*: from under 10% to over 30%. Remarkably, this trend in memorization is *not* reflected by the linear probe gap (dark blue), which only changes by a few percent beyond 250 epochs.

Training set size has minimal effect on *déjà vu* memorization.

Figure 1.4b shows how *déjà vu* memorization responds to the model's training set size. The number of training epochs is fixed to 1000. Interestingly, training set size appears to have almost *no* influence on the *déjà vu* score (red line), indicating that memorization is equally

prevalent with a 100K dataset and a 500K dataset. This result suggests that *déjà vu* memorization may be detectable even for large datasets. Meanwhile, the standard linear probe train-test accuracy gap *declines* by more than half as the dataset size grows, failing to represent the memorization quantified by our test.

Training loss hyper-parameter has a strong effect.

Loss hyper-parameters, like VICReg's invariance coefficient (Figure 1.8a) or SimCLR's temperature parameter (Appendix Figure A.8a) significantly impact *déjà vu* with minimal impact on the linear probe validation accuracy.

Some SSL criteria promote stronger *déjà vu* memorization.

Table 1.9b demonstrates that the degree of memorization varies widely for different training criteria. VICReg and Barlow Twins have the highest *déjà vu* scores while SimCLR and Byol have the lowest. With the exception of Byol, all SSL models have more *déjà vu* memorization than the supervised model. Interestingly, different criteria can lead to similar linear probe validation accuracy and very different degrees of *déjà vu* as seen with SimCLR and Barlow Twins. Note that low degrees of *déjà vu* can still risk training image reconstruction, as exemplified by the SimCLR reconstructions in Figures 1.6 and **??**.

Larger models have increased *déjà vu* memorization.

Figure 1.9a validates the common intuition that lower capacity architectures (Resnet18/34) result in less memorization than their high capacity counterparts (Resnet50/101). We see the same trend for vision transformers as well.

Guillotine regularization can help reduce *déjà vu* memorization.

Previous experiments were done using the projector embedding. In Figure 1.8b, we present how Guillotine regularization[24] (removing final layers in a trained SSL model) impacts *déjà vu* with VICReg[8]. Using the backbone embedding instead of the projector embedding seems to be the most straightforward way to mitigate *déjà vu* memorization. However, as demonstrated

[8]Further experiments are available in Appendix A.0.3.

in Appendix A.0.4, backbone representation with low *déjà vu* score can still be leveraged to reconstruct some of the training images.

1.7 Conclusion

We defined and analyzed *déjà vu* memorization, a notion of unintended memorization of partial information in image data. As shown in Sections 1.4 and 1.5, SSL models can largely exhibit *déjà vu* memorization on their training data, and this memorization signal can be extracted to infer or visualize image-specific information. Since SSL models are becoming increasingly widespread as foundation models for image data, negative consequences of *déjà vu* memorization can have profound downstream impact and thus deserves further attention. Future work should focus on understanding how *déjà vu* emerges in the training of SSL models and why methods like Byol are much more robust to *déjà vu* than VICReg and Barlow Twins. In addition, trying to characterize which data points are the most at risk of *déjà vu* could be crucial to get a better understanding on this phenomenon.

Chapter 2

A Non-Parametric Test to Detect Data-Copying in Generative Models

2.1 Introduction

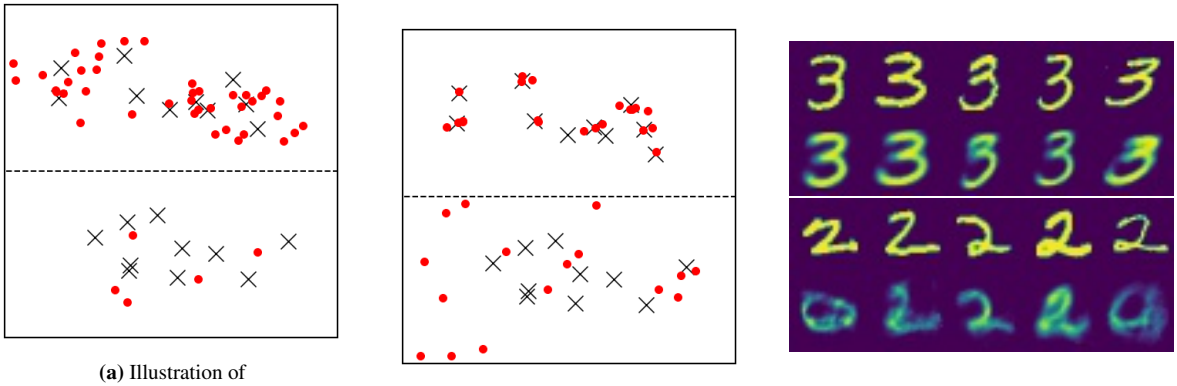
Overfitting is a basic stumbling block of any learning process. While it has been studied in great detail in the context of supervised learning, it has received much less attention in the unsupervised setting, despite being just as much of a problem.

To start with a simple example, consider a classical kernel density estimator (KDE), which given data $x_1, \ldots, x_n \in \mathbb{R}^d$, constructs a distribution over $\mathbb{R}^d$ by placing a Gaussian of width $\sigma > 0$ at each of these points, yielding the density

$$q_\sigma(x) = \frac{1}{(2\pi)^{d/2}\sigma^d n} \sum_{i=1}^{n} \exp\left(-\frac{\|x - x_i\|^2}{2\sigma^2}\right). \tag{2.1}$$

The only parameter is the scalar σ. Setting it too small makes $q(x)$ too concentrated around the given points: a clear case of overfitting (see Appendix **Figure B.1**). This cannot be avoided by choosing the σ that maximizes the log likelihood on the training data, since in the limit $\sigma \to 0$, this likelihood goes to ∞.

The classical solution is to find a parameter σ that has a low *generalization gap* – that is, a low gap between the training log-likelihood and the log-likelihood on a held-out validation set. This method however often does not apply to the more complex generative models that



(a) Illustration of over-/under-representation Training sample: $\times$, Generated sample: $\bullet$

(b) Illustration of data-copying/underfitting Training sample: $\times$, Generated sample: $\bullet$

(c) VAE copying/underfitting on MNIST top: $Z_U = -8.54$, bottom: $Z_U = +3.30$

Figure 2.1. Comparison of data-copying with over/under representation. Each image depicts a single instance space partitioned into two regions. Illustration **(a)** depicts an over-represented region (top) and under-represented region (bottom). This is the kind of overfitting evaluated by methods like FID score and Precision and Recall. Illustration **(b)** depicts a data-copied region (top) and underfit region (bottom). This is the type of overfitting focused on in this work. Figure **(c)** shows VAE-generated and training samples from a data-copied (top) and underfit (bottom) region of the MNIST instance space. In each 10-image strip, the bottom row provides random generated samples from the region and the top row shows their training nearest neighbors. Samples in the bottom region are on average further to their training nearest neighbor than held-out test samples in the region, and samples in the top region are closer, and thus 'copying' (computed in embedded space, see Experiments section).

have emerged over the past decade or so, such as Variational Auto Encoders (VAEs) [107] and Generative Adversarial Networks (GANs) [83]. These models easily involve millions of parameters, and hence overfitting is a serious concern. Yet, a major challenge in evaluating overfitting is that these models do not offer exact, tractable likelihoods. VAEs can tractably provide a log-likelihood lower bound, while GANs have no accompanying density estimate at all. Thus any method that can assess these generative models must be based only on the samples produced.

A body of prior work has provided tests for evaluating generative models based on samples drawn from them [142, 140, 164, 92]; however, the vast majority of these tests focus on 'mode dropping' and 'mode collapse': the tendency for a generative model to either merge or delete high-density modes of the true distribution. A generative model that simply reproduces the training set or minor variations thereof will pass most of these tests.

In contrast, this work formalizes and investigates a type of overfitting that we call 'data-copying': the propensity of a generative model to recreate minute variations of a subset of training examples it has seen, rather than represent the true diversity of the data distribution. An example is shown in **Figure 2.1b**; in the top region of the instance space, the generative model data-copies, or creates samples that are very close to the training samples; meanwhile, in the bottom region, it underfits. To detect this, we introduce a test that relies on three independent samples: the original training sample used to produce the generative model; a separate (held-out) test sample from the underlying distribution; and a synthetic sample drawn from the generator.

Our key insight is that an overfit generative model would produce samples that are too close to the training samples – closer on average than an independently drawn test sample from the same distribution. Thus, if a suitable distance function is available, then we can test for data-copying by testing whether the distances to the closest point in the training sample are on average smaller for the generated sample than for the test sample.

A further complication is that modern generative models tend to behave differently in different regions of space; a configuration as in **Figure 2.1b** for example could cause a global test

to fail. To address this, we use ideas from the design of non-parametric methods. We divide the instance space into cells, conduct our test separately in each cell, and then combine the results to get a sense of the average degree of data-copying.

Finally, we explore our test experimentally on a variety of illustrative data sets and generative models. Our results demonstrate that given enough samples, our test can successfully detect data-copying in a broad range of settings.

2.1.1 Related work

There has been a large body of prior work on the evaluation of generative models [142, 121, 136, 140, 169, 164] . Most are geared to detect some form of mode-collapse or mode-dropping: the tendency to either merge or delete high-density regions of the training data. Consequently, they fail to detect even the simplest case of extreme data-copying – where a generative model memorizes and exactly reproduces a bootstrap sample from the training set. We discuss below a few such canonical tests.

To-date there is a wealth of techniques for evaluating whether a model mode-drops or -collapses. Tests like the popular Inception Score (IS), Frechét Inception Distance (FID) [92], Precision and Recall test [140], and extensions thereof [108, 40] all work by embedding samples using the features of a discriminative network such as 'InceptionV3' and checking whether the training and generated samples are similar *in aggregate*. The hypothesis-testing binning method proposed by [136] also compares aggregate training and generated samples, but without the embedding step. The parametric Kernel MMD method proposed by [148] uses a carefully selected kernel to estimate the distribution of both the generated and training samples and reports the maximum mean discrepancy between the two. All these tests, however, reward a generative model that only produces slight variations of the training set, and do not successfully detect even the most egregious forms of data-copying.

A test that can detect some forms of data-copying is the *Two-Sample Nearest Neighbor*, a non-parametric test proposed by [121]. Their method groups a training and generated sample

of equal cardinality together, with training points labeled '1' and generated points labeled '0', and then reports the Leave-One-Out (LOO) Nearest-Neighbor (NN) accuracy of predicting '1's and '0's. Two values are then reported as discussed by [169] – the leave-one-out accuracy of the training points, and the leave-one-out accuracy of the generated points. An ideal generative model should produce an accuracy of 0.5 for each. More often, a mode-collapsing generative model will leave the training accuracy low and generated accuracy high, while a generative model that exactly reproduces the entire training set should produce zero accuracy for both. Unlike this method, our test not only detects exact data-copying, which is unlikely, but estimates whether a given model generates samples closer to the training set than it should, as determined by a held-out test set.

The concept of data-copying has also been explored by [169] (where it is called 'memorization') for a variety of generative models and several of the above two-sample evaluation tests. Their results indicate that out of a variety of popular tests, only the two-sample nearest neighbor test is able to capture instances of extreme data-copying.

[27] explores three-sample testing, but for comparing the performance of different models, not for detecting overfitting. [67] uses the three-sample test proposed by [27] for detecting data-copying; unlike ours, their test is global in nature.

Finally, other works concurrent with ours have explored parametric approaches to rooting out data-copying. A recent work by [87] suggests that, given a large enough sample from the model, Neural Network Divergences are sensitive to data-copying. In a slightly different vein, a recent work by [162] investigates whether latent-parameter models memorize training data by learning the reverse mapping from image to latent code. The present work departs from those by offering a probabilistically motivated non-parametric test that is entirely model agnostic.

2.2 Preliminaries

We begin by introducing some notation and formalizing the definitions of overfitting. Let $\mathscr{X}$ denote an instance space in which data points lie, and P an unknown underlying distribution on this space. A training set T is drawn from P and is used to build a generative model Q. We then wish to assess whether Q is the result of overfitting: that is, whether Q produces samples that are too close to the training data. To help ascertain this, we are able to draw two additional samples:

- A fresh sample of n points from P; call this P_n.
- A sample of m points from Q; call this Q_m.

As illustrated in **Figures 2.1a, 2.1b**, a generative model can overfit locally in a region $\mathscr{C} \subseteq \mathscr{X}$. To characterize this, for any distribution D on $\mathscr{X}$, we use $D|_{\mathscr{C}}$ denote its restriction to the region $\mathscr{C}$, that is,

$$D|_{\mathscr{C}}(\mathscr{A}) = \frac{D(\mathscr{A} \cap \mathscr{C})}{D(\mathscr{C})} \quad \text{for any } \mathscr{A} \subseteq \mathscr{X}.$$

2.2.1 Definitions of Overfitting

We now formalize the notion of data-copying, and illustrate its distinction from other types of overfitting.

Intuitively, data-copying refers to situations where Q is "too close" to the training set T; that is, closer to T than the target distribution P happens to be. We make this quantitative by choosing a distance function $d : \mathscr{X} \to \mathbb{R}$ from points in $\mathscr{X}$ to the training set, for instance, $d(x) = \min_{t \in T} \|x - t\|^2$, if $\mathscr{X}$ is a subset of Euclidean space.

Ideally, we desire that Q's expected distance to the training set is the same as that of P's, namely $\mathbb{E}_{X \sim P}[d(X)] = \mathbb{E}_{Y \sim Q}[d(Y)]$. We may rewrite this as follows: given any distribution D over $\mathscr{X}$, define $L(D)$ to be the one-dimensional distribution of $d(X)$ for $X \sim D$. We consider

data-copying to have occurred if random draws from $L(P)$ are systematically larger than from $L(Q)$. The above equalized expected distance condition can be rewritten as

$$\mathbb{E}_{Y\sim Q}[d(Y)] - \mathbb{E}_{X\sim P}[d(X)] = \mathbb{E}_{\substack{A\sim L(P)\\ B\sim L(Q)}}[B-A] = 0 \tag{2.2}$$

However, we are less interested in how *large* the difference is, and more in how *often* B is larger than A. Let

$$\Delta_T(P,Q) = \Pr\left(B > A \;\middle|\; B\sim L(Q), A\sim L(P)\right)$$

where $0 \leq \Delta_T(P,Q) \leq 1$ represents how 'far' Q is from training sample T as compared to true distribution P. A more interpretable yet equally meaningful condition is which guarantees (2.2) if densities $L(P)$ and $L(Q)$ have the same shape, but could plausibly be mean-shifted.

If $\Delta_T(P,Q) \ll \frac{1}{2}$, Q is data-copying training set T, since samples from Q are systematically closer to T than are samples from P. However, even if $\Delta_T(P,Q) \geq \frac{1}{2}$, Q may still be data-copying. As exhibited in **Figures 2.1b** and **2.1c**, a model Q may data-copy in one region and underfit in others. In this case, Q may be further from T than is P *globally*, but much closer to T *locally*. As such, we consider Q to be data-copying if it is overfit in a subset $\mathscr{C} \subseteq \mathscr{X}$:

Definition 2.2.1 (Data-Copying)**.** A generative model Q is *data-copying* training set T if, in some region $\mathscr{C} \subseteq \mathscr{X}$, it is systematically closer to T by distance metric $d : \mathscr{X} \to \mathbb{R}$ than are samples from P . Specifically, if

$$\Delta_T(P|_{\mathscr{C}}, Q|_{\mathscr{C}}) < \frac{1}{2}$$

Observe that data-copying is orthogonal to the type of overfitting addressed by many previous works [92, 140], which we call 'over-representation'. There, Q overemphasizes some region of the instance space $\mathscr{C} \subseteq \mathscr{X}$, often a region of high density in the training set T. For the

sake of completeness, we provide a formal definition below.

Definition 2.2.2 (Over-Representation)**.** A generative model Q is *over-representing* P in some region $\mathscr{C} \subseteq \mathscr{X}$, if the probability of drawing $Y \sim Q$ is much greater than it is of drawing $X \sim P$. Specifically, if

$$Q(\mathscr{C}) - P(\mathscr{C}) \gg 0$$

Observe that it is possible to over-represent without data-copying and vice versa. For example, if P is an equally weighted mixture of two Gaussians, and Q perfectly models one of them, then Q is over-representing without data-copying. On the other hand, if Q outputs a bootstrap sample of the training set T, then it is data-copying without over-representing. The focus of the rest of this work is on data-copying.

2.3 A Test For Data-Copying

Having provided a formal definition, we next propose a hypothesis test to detect data-copying.

2.3.1 A Global Test

We introduce our data-copying test in the global setting, when $\mathscr{C} = \mathscr{X}$. Our null hypothesis H_0 suggests that Q may equal P:

$$H_0 : \Delta_T(P,Q) \;=\; \frac{1}{2} \tag{2.3}$$

There are well-established non-parametric tests for this hypothesis, such as the Mann-Whitney U test [125]. Let $A_i \sim L(P_n), B_j \sim L(Q_m)$ be samples of $L(P), L(Q)$ given by P_n, Q_m and their distances $d(X)$ to training set T. The U statistic estimates the probability in Equation 2.3 by measuring the number of all mn pairwise comparisons in which $B_j > A_i$. An efficient and simple

method to gather and interpret this test is as follows:

1. Sort the $n+m$ values $L(P_n) \cup L(Q_m)$ such that each instance $l_i \in L(P_n), l_j \in L(Q_m)$ has rank $R(l_i), R(l_j)$, starting from rank 1, and ending with rank $n+m$. $L(P_n), L(Q_m)$ have no tied ranks with probability 1 assuming their distributions are continuous.

2. Calculate the rank-sum for $L(Q_m)$ denoted R_{Q_m}, and its U score denoted U_{Q_m}:

$$R_{Q_m} = \sum_{l_j \in L(Q_m)} R(l_j), \quad U_{Q_m} = R_{Q_m} - \frac{m(m+1)}{2}$$

   Consequently, $U_{Q_m} = \sum_{ij} \mathbb{1}_{B_j > A_i}$.

3. Under H_0, U_{Q_m} is approximately normally distributed with > 20 samples in both $L(Q_m)$ and $L(P_n)$, allowing for the following z-scored statistic

$$Z_U\big(L(P_n), L(Q_m); T\big) = \frac{U_{Q_m} - \mu_U}{\sigma_U},$$
$$\mu_U = \frac{mn}{2}, \quad \sigma_U = \sqrt{\frac{mn(m+n+1)}{12}}$$

Z_U provides us a data-copying statistic with normalized expectation and variance under H_0. $Z_U \ll 0$ implies data-copying, $Z_U \gg 0$ implies underfitting. $Z_U < -5$ implies that if H_0 holds, Z_U is as likely as sampling a value < -5 from a standard normal.

Observe that this test is completely model agnostic and uses no estimate of likelihood. It only requires a meaningful distance metric, which is becoming common practice in the evaluation of mode-collapse and -dropping [92, 140] as well.

2.3.2 Handling Heterogeneity

As described in Section 2.2.1, the above global test can be fooled by generators Q which are very close to the training data in some regions of the instance space (overfitting) but very far from the training data in others (poor modeling).

We handle this by introducing a *local* version of our test. Let Π denote any partition of the instance space $\mathscr{X}$, which can be constructed in any manner. In our experiments, for instance, we run the k-means algorithm on T, so that $|\Pi| = k$. As the number of training and test samples grows, we may increase k and thus the instance-space resolution of our test. Letting $L_\pi(D) = L(D|_\pi)$ be the distribution of distances-to-training-set within cell $\pi \in \Pi$, we probe each cell of the partition Π individually.

Data Copying.

To offer a summary statistic for data copying, we collect the z-scored Mann-Whitney U statistic, Z_U, described in Section 2.3.1 in each cell π. Let $P_n(\pi) = |\{x : x \in P_n, x \in \pi\}|/n$ denote the fraction of P_n points lying in cell π, and similarly for $Q_m(\pi)$. The Z_U test for cell π and training set T will then be denoted as $Z_U\big(L_\pi(P_n), L_\pi(Q_m); T\big)$, where $L_\pi(P_n) = \{d(x) : x \in P_n, x \in \pi\}$ and similarly for $L_\pi(Q_m)$. See **Figure 2.1c** for examples of these in-cell scores. For stability, we only measure data-copying for those cells significantly represented by Q, as determined by a threshold τ. Let Π_τ be the set of all cells in the partition Π for which $Q_m(\pi) \geq \tau$. Then, our summary statistic for data copying averages across all cells represented by Q:

$$C_T(P_n, Q_m) \coloneqq \frac{\sum_{\pi \in \Pi_\tau} P_n(\pi) Z_U\big(L_\pi(P_n), L_\pi(Q_m); T\big)}{\sum_{\pi \in \Pi_\tau} P_n(\pi)}$$

Over-Representation.

The above test will not catch a model that heavily over- or under-represents cells. For completeness, we next provide a simple representation test that is essentially used by [136], now with an independent test set instead of the training set.

With $n, m \geq 20$ in cell π, we may treat $Q_m(\pi), P_n(\pi)$ as Gaussian random variables. We then check the null hypothesis $H_0 : 0 = P(\pi) - Q(\pi)$. Assuming this null hypothesis, a simple

z-test is:

$$Z_\pi = \frac{Q_m(\pi) - P_n(\pi)}{\sqrt{\widehat{p}(1-\widehat{p})\left(\frac{1}{n}+\frac{1}{m}\right)}}$$

where $\widehat{p} = \frac{nP_n(\pi)+mQ_m(\pi)}{n+m}$. We then report two values for a significance level $s = 0.05$: the number of significantly different cells ('bins') with $Z_\pi > s$ (NDB over-representing), and the number with $Z_\pi < -s$ (NDB under-representing).

Together, these summary statistics — C_T, NDB-over, NDB-under — detect the ways in which Q broadly represents P without directly copying the training set T.

2.3.3 Performance Guarantees

We next provide some simple guarantees on the performance of the global test statistic $U(Q_m)$. Guarantees for the average test is more complicated, and is left as a direction for future work.

We begin by showing that when the null hypothesis H_0 does not hold, U_{Q_m} has some desirable properties – $\frac{1}{mn}U_{Q_m}$ is a consistent estimator of the quantity of interest, $\Delta_T(P,Q)$:

Theorem 1. *For true distribution P, model distribution Q, and distance metric $d : \mathscr{X} \to \mathbb{R}$, the estimator $\frac{1}{mn}U_{Q_m} \to_P \Delta(P,Q)$ according to the concentration inequality*

$$\Pr\Big(\,\Big|\frac{1}{mn}U_{Q_m} - \Delta(P,Q)\Big| \geq t\Big) \leq \exp\left(-\frac{2t^2mn}{m+n}\right)$$

Furthermore, when the model distribution Q actually matches the true distribution P, under modest assumptions we can expect $\frac{1}{mn}U_{Q_m}$ to be near $\frac{1}{2}$:

Theorem 2. *If $Q = P$, and the corresponding distance distribution $L(Q) = L(P)$ is non-atomic, then*

$$\mathbb{E}\Big[\frac{1}{mn}U_{Q_m}\Big] = \frac{1}{2} \quad \textit{and} \quad \mathbb{E}[Z_U] = 0$$

Proofs are provided in Appendices B.0.1 and B.0.2.

Additionally, we show that for a Gaussian Kernel Density Estimator, the parameter σ that satisfies the condition in Equation 2.2 is the σ corresponding to a maximum likelihood Gaussian KDE model. Recall that a KDE model is described by

$$q_\sigma(x) = \frac{1}{(2\pi)^{k/2}|T|\sigma^k} \sum_{t\in T} \exp\left(-\frac{\|x-t\|^2}{2\sigma^2}\right), \tag{2.4}$$

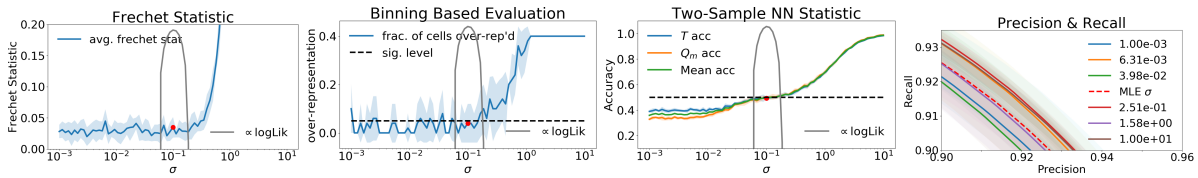
where the posterior probability that a random draw $x \sim q_\sigma(x)$ comes from the Gaussian component centered at training point t is

$$Q_\sigma(t|x) = \frac{\exp(-\|x-t\|^2/(2\sigma^2))}{\sum_{t'\in T}\exp(-\|x-t'\|^2/(2\sigma^2))}$$

Lemma 3. *For the kernel density estimator (2.4), the maximum-likehood choice of σ, namely the maximizer of $\mathbb{E}_{X\sim P}[\log q_\sigma(X)]$, satisfies*

$$\mathbb{E}_{X\sim P}\left[\sum_{t\in T} Q_\sigma(t|X)\|X-t\|^2\right] = \mathbb{E}_{Y\sim Q_\sigma}\left[\sum_{t\in T} Q_\sigma(t|Y)\|Y-t\|^2\right]$$

See Appendix B.0.3 for proof. Unless σ is large, we know that for any given $x \in \mathscr{X}$, $\sum_{t\in T} Q_\sigma(t|x)\|x-t\|^2 \approx d(x) = \min_{t\in T}\|x-t\|^2$. So, enforcing that $\mathbb{E}_{X\sim P}[d(X)] = \mathbb{E}_{Y\sim Q}[d(Y)]$, and more loosely that $\mathbb{E}_{\substack{A\sim L(P)\\ B\sim L(Q)}}[\mathbb{1}_{B>A}] = \frac{1}{2}$ provides an excellent non-parametric approach to selecting a Gaussian KDE, and ought to be enforced for any Q attempting to emulate P; after all, Theorem 2 points out that effectively any model with $Q = P$ also yields this condition.



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

Figure 2.2. Response of four baseline test methods to data-copying of a Gaussian KDE on 'moons' dataset. Only the two-sample NN test **(c)** is able to detect data-copying KDE models as σ moves below σ_{MLE} (depicted as a red dot). The gray trace is proportional to the KDE's log-likelihood measured on a held-out validation set.

2.4 Experiments

Having clarified what we mean by data-copying in theory, we turn our attention to data copying by generative models in practice. We leave representation test results for the appendix, since this behavior has been well studied in previous works. Specifically, we aim to answer the two following questions:

1. Are the existing tests that measure generative model overfitting able to capture data-copying?

2. As popular generative models range from over- to underfitting, does our test indicate data-copying, and if so, to what degree?

Training, Generated and Test Sets.

In all of the following experiments, we select a training dataset T with test split P_n, and a generative model Q producing a sample Q_m. We perform k-means on T to determine partition Π, with the objective having a reasonable population of both T and P_n in each $\pi \in \Pi$. We set threshold τ, such that we are guaranteed to have at least 20 samples in each cell in order to validate the gaussian assumption of Z_π, Z_U.

2.4.1 Detecting data-copying

First, we investigate which of the existing generative model tests can detect explicit data-copying.

Baselines and Dataset

Here, we probe the four of the methods described in our Related Work section, to see how they react to data-copying: two-sample NN [121], FID [92], Binning-Based Evaluation [136], and Precision & Recall [140], which are described in detail in Appendix B.0.4. We run this test on the two-dimensional 'moons' dataset, as it affords us limitless training and test samples and requires no feature embedding (see Appendix B.0.4 for an example). Note that, without an embedding, FID is simply the Frechét distance between two MLE normal distributions fit to T and Q_m. We use the same size generated and training sample for all methods, when $m < |T|$ (especially for large datasets and computationally burdensome samplers) we are forced to use an m-size training subsample $\widetilde{T}$ for running the two-sample NN test due to its constraint that $m = |T|$.

The canonical method of measuring the generalization gap (difference between training and test set likelihoods under the model) is not one of our primary baselines due to the fact that it cannot scale to contemporary models with intractable likelihoods (e.g. GANs / VAEs). It is, however, included for reference in Figure 2.3. While this method naturally exposes data-copying, it is generally insensitive to underfitting: $\Delta(P,Q) > \frac{1}{2}$.

We make Q a Gaussian KDE since, as described in Section 2.3.3, it allows us to force explicit data-copying by setting σ very low. As $\sigma \to 0$, Q becomes a bootstrap sampler of the original training set. If a given test method can detect the level of data-copying by Q on T, it will provide a different response to a heavily over-fit KDE Q ($\sigma \ll \sigma_{\text{MLE}}$), a well-fit KDE Q ($\sigma \approx \sigma_{\text{MLE}}$), and an underfit KDE Q ($\sigma \gg \sigma_{\text{MLE}}$).

Figure 2.2 depicts how each baseline method responds to KDE Q models of varying degrees of data-copying, as Q ranges from data-copying ($\sigma = 0.001$) up to heavily underfit ($\sigma = 10$). The Frechét and Binning methods report effectively the same value for all $\sigma \leq \sigma_{\text{MLE}}$, indicating inability to detect data-copying. Similarly, the PR curves for different σ values are high variance and show no meaningful order with respect to σ.

The two-sample NN test does show a mild change in response as σ decreases below

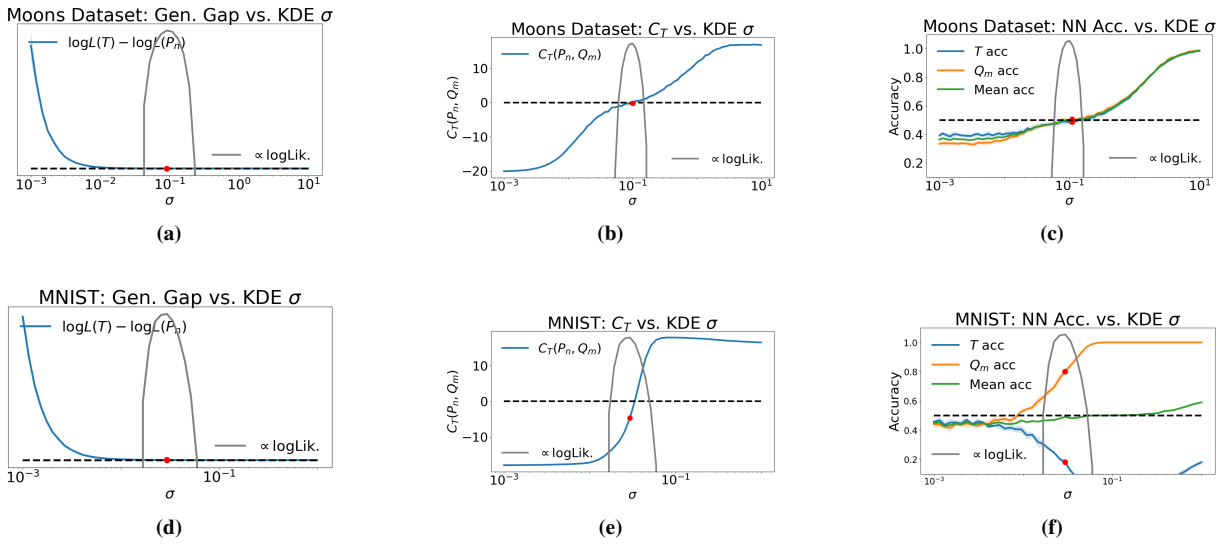


Figure 2.3. $C_T(P_n, Q_m)$ vs. NN baseline and generalization gap on moons and MNIST digits datasets. **(a,b,c)** compare the three methods on the moons dataset. **(d,e,f)** compare the three methods on MNIST. In both data settings, the C_T statistic is far more sensitive to the data-copying regime $\sigma \ll \sigma_{\text{MLE}}$ than the NN baseline. It is more sensitive to underfitting $\sigma \gg \sigma_{\text{MLE}}$ than the generalization gap test. The red dot denotes σ_{MLE}, and the gray trace is proportional to the KDE's log-likelihood measured on a held-out validation set.

σ_{MLE}. This makes sense; as points in Q_m become closer to points in T, the two-sample NN accuracy should steadily drop to zero. The reason it does not drop to zero is due to the m subsampled training points, $\widetilde{T} \subset T$, needed to perform this test. As such, each training point $t \in T$ being copied by generated point $q \in Q_m$ is unlikely to be present in $\widetilde{T}$ during the test. This phenomenon is especially pronounced in some of the following settings.

The reason most of these tests fail to detect data-copying is because most existing methods focus on another type of overfitting: mode-collapse and -dropping, wherein entire modes of P are either forgotten or averaged together. However, if a model begins to data-copy, it is definitively overfitting *without* mode-collapsing.

Next, we will demonstrate our method on a variety of datasets, models, and embeddings. We will compare our method to the two-sample NN method in each setting, as it is the only baseline that responds to explicit data-copying.

2.4.2 Measuring degree of data-copying

We now aim to answer the second question raised at the beginning of this section: does $C_T(P_n, Q_m)$ detect and quantify data-copying? We focus on two types of generative model: Gaussian KDEs, and neural models.

KDE-based tests

While KDEs do not provide a reliable likelihood in high dimension [151], they do provide an informative first benchmark of the C_T statistic. KDEs allow us to directly force data-copying, and confirm the theoretical connection between the MLE KDE and $C_T \approx 0$ described in Lemma 3.

KDEs: 'moons' dataset

Here, we repeat the experiment performed in Section 2.4.1, now including the C_T statistic for comparison. Refer to Appendix B.0.4 for experimental details, and examples of the dataset. For reference, **Figure 2.3a** depicts how the generalization gap dwindles as KDE σ increases. While this test is capable of capturing data-copying, it is insensitive to underfitting and relies on a tractable likelihood.

Figures 2.3b and **2.3c** give a side-by-side depiction of C_T and the two-sample NN test accuracies across a range of KDE σ values. Think of C_T values as z-score standard deviations. We see that the C_T statistic in **Figure 2.3b** precisely identifies the MLE model when $C_T \approx 0$, and responds sharply to σ values above and below σ_{MLE}. The baseline in **Figure 2.3c** similarly identifies the MLE Q model when training accuracy ≈ 0.5, but is higher variance and less sensitive to changes in σ, especially for over-fit $\sigma \ll \sigma_{\text{MLE}}$. We will see in the next experiment, that this test breaks down for more complex datasets when $m \ll |T|$.

KDEs: MNIST Handwritten Digits

We now extend the KDE test performed on the moons dataset to the significantly more complex MNIST handwritten digit dataset [111].

While it would be convenient to directly apply the KDE σ-sweeping tests discussed in the previous section, there are two primary barriers. The first is that KDE model relies on L_2 norms being perceptually meaningful, which is well understood not to be true in pixel space. The second problem is that of dimensionality: the 784-dimensional space of digits is far too high for a KDE to be even remotely efficient at interpolating the space.

To handle these issues, we first embed each image, $x \in \mathscr{X}$, to a perceptually meaningful 64-dimensional latent code, $z \in \mathscr{Z}$. We achieve this by training a convolutional autoencoder with a VGGnet perceptual loss produced by [183] (see Appendix B.0.4 for more detail). Surely, even in the lower 64-dimensional space, the KDE will suffer some from the curse of dimensionality. We are not promoting this method as a powerful generative model, but rather as an instructive tool for probing a test's response to data-copying in the image domain.

Again, the likelihood generalization gap is depicted in **Figure 2.3d** repeating the trend seen with the 'moons' dataset. Here, we run all tests in the compressed latent space. See Appendix B.0.4 for experimental details.

Figure 2.3e shows how $C_T(P_n, Q_m)$ reacts decisively to over- and underfitting. It falsely determines the MLE σ value as slightly over-fit. However, the region of where C_T transitions from over- to underfit (say $-13 \leq C_T \leq 13$) is relatively tight and includes the MLE σ.

Meanwhile, **Figure 2.3f** shows how — with the generated sample smaller than the training sample, $m \ll |T|$ — the two-sample NN baseline provides no meaningful estimate of data-copying. In fact, the most data-copying models with low σ achieve the best scores closest to 0.5. Again, we are forced to use the m-subsampled $\widetilde{T} \subset T$, and most instances of data copying are completely missed. C_T has no such restriction.

These results are promising, and demonstrate the reliability of this hypothesis testing approach to probing for data-copying across different data domains. In the next section, we explore how these tests perform on more sophisticated, non-KDE models.

2.4.3 Neural Model Tests

Gaussian KDE's may have nice theoretical properties, but are relatively ineffective in high-dimensional settings, precluding domains like images. As such, we also demonstrate our experiments on more practical neural models trained on higher dimensional image datasets (MNIST and ImageNet), with the goal of observing whether the C_T statistic indicates data-copying as these models range from over- to underfit.

MNIST VAE

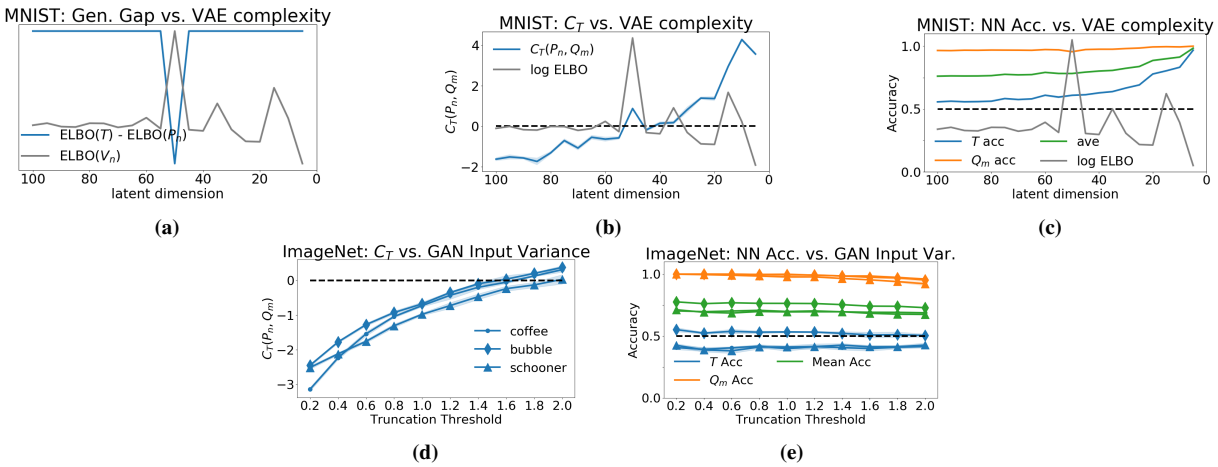
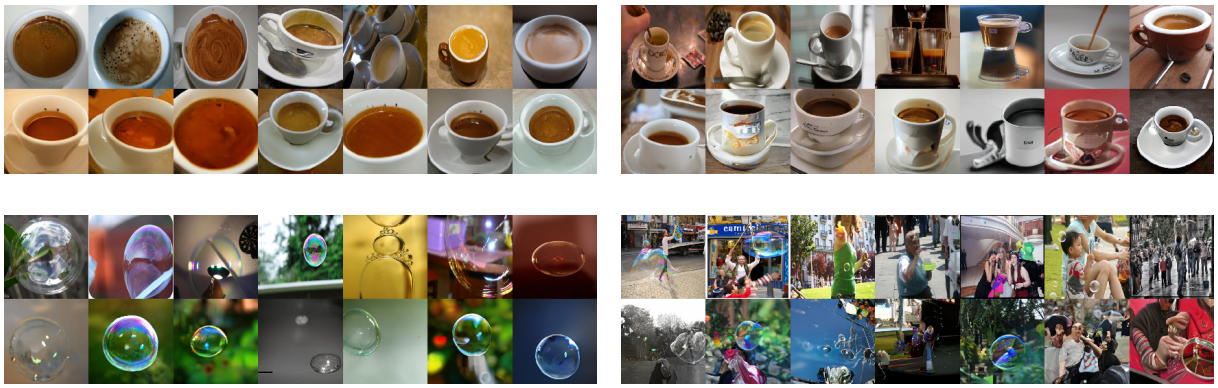


Figure 2.4. Neural model data-copying: figures **(b)** and **(d)** demonstrate the C_T statistic identifying data-copying in an MNIST VAE and ImageNet GAN as they range from heavily over-fit to underfit. **(c)** and **(e)** demonstrate the relative insensitivity of the NN baseline to this overfitting, as does figure **(a)** of the generalization (ELBO) gap method for VAEs. (Note, the markers for **(d)** apply to the traces of **(e)**)

Here, we employ our data-copying test, $C_T(P_n, Q_m)$, on a range of VAEs of varying complexity trained on the MNIST handwritten digit dataset. Experimental and theoretical findings have suggested that VAE samplers — under certain assumptions — simply produce convex combinations of training set samples [28]. In generating an out-of-distribution sample, an overly complex VAE effectively reproduces nearest-neighbor training samples. Our findings appear to corroborate this. We vary model complexity by increasing the width (neurons per layer) in a three-layer VAE (see Appendix B.0.4 for details). As an embedding, we pass all



(a) Data-copied cells; top: $Z_U = -1.46$, bottom: $Z_U = -1.00$

(b) Underfit cells; top: $Z_U = +1.40$, bottom: $Z_U = +0.71$

Figure 2.5. Data-copied and underfit cells of ImageNet12 'coffee' and 'soap bubble' instance spaces (trunc. threshold = 2). In each 14-figure strip, the top row provides a random series of training samples from the cell, and the bottom row provides a random series of generated samples from the cell. **(a)** Data-copied cells. **(a), top**: Random training and generated samples from a $Z_U = -1.46$ cell of the coffee instance space. **(a), bottom**: Random training and generated samples from a $Z_U = -1.00$ cell of the bubble instance space. **(b)** Underfit cells. **(b), top**: Random training and generated samples from a $Z_U = +1.40$ cell of the coffee instance space. **(b), bottom**: Random training and generated samples from a $Z_U = +0.71$ cell of the bubble instance space.

samples through the the convolutional autoencoder of Section 2.4.2, and collect statistics in this 64-dimensional space.

Figures 2.4b and **2.4c** compare the C_T statistic to the NN accuracy baseline . C_T behaves as it did in the previous sections: more complex models over-fit, forcing $C_T \ll 0$, and less complex models underfit forcing it $\gg 0$. We note that the range of C_T values is far less dramatic, which is to be expected since the KDEs were forced to explicitly data-copy. As likelihood is not available for VAEs, we compute each model's ELBO on a 10,000 sample held out validation set, and plot it in gray. We observe that the ELBO spikes for models with C_T near 0. **Figure 2.4a** shows the ELBO approximation of the generalization gap as the latent dimension (and number of units in each layer) is decreased. This method is entirely insensitive to over- and underfit models. This may be because the ELBO is only a lower bound, not the actual likelihood.

The NN baseline in **Figure 2.4c** is less interpretable, and fails to capture the overfitting trend as C_T does. While all three test accuracies still follow the upward-sloping trend of **Figure**

2.3c, they do not indicate where the highest validation set ELBO is. Furthermore, the NN accuracy statistics are shifted upward when compared to the results of the previous section: all NN accuracies are above 0.5 for all latent dimensions. This is problematic. A test statistic's absolute score ought to bear significance between very different data and model domains like KDEs and VAEs.

ImageNet GAN

Finally, we scale our experiments up to a more practical image domain. We gather our test statistics on a state of the art conditional GAN, '*BigGan*' [29], trained on the Imagenet 12 dataset [138]. Conditioning on an input code, this GAN will generate one of 1000 different Imagenet classes. We run our experiments separately on three classes: 'coffee', 'soap bubble', and 'schooner'. All generated, test, and training images are embedded to a 64-dimensional space by first gathering the 2048-dimensional features of an InceptionV3 network 'Pool3' layer, and then projecting them onto the 64 principal components of the training embeddings. Appendix B.0.4 has more details.

Being limited to one pre-trained model, we increase model variance ('truncation threshold') instead of decreasing model complexity. As proposed by *BigGan*'s authors, all standard normal input samples outside of this truncation threshold are resampled. The authors suggest that lower truncation thresholds, by only producing samples at the mode of the input, output higher quality samples at the cost of variety, as determined by Inception Score (IS). Similarly, the FID score finds suitable variety until truncation approaches zero.

As depicted in **Figure 2.4d**, the C_T statistic remains well below zero until the truncation threshold is nearly maximized, indicating that Q produces samples closer to the training set than real samples tend to be. While FID finds that *in aggregate* the distributions are roughly similar, a closer look suggests that Q allocates too much probability mass near the training samples.

Meanwhile, the two-sample NN baseline in **Figure 2.4e** hardly reacts to changes in truncation, even though the generated and training sets are the same size, $m = |T|$. Across all

truncation values, the training sample NN accuracy remains around 0.5, not quite implying over- or underfitting.

A useful feature of the C_T statistic is that one can examine the Z_U scores it is composed of to see which of the cells $\pi \in \Pi_\tau$ are or are not copying. **Figure 2.5** shows the samples of over- and underfit clusters for two of the three classes. For both 'coffee' and 'bubble' classes, the underfit cells are more diverse than the data-copied cells. While it might seem reasonable that these generated samples are further from nearest neighbors in more diverse clusters, keep in mind that the $Z_U > 0$ statistic indicates that they are further from training neighbors than test set samples are. For instance, the people depicted in underfit 'bubbles' cell are highly distorted.

2.5 Conclusion

In this work, we have formalized *data-copying*: an under-explored failure mode of generative model overfitting. We have provided preliminary tests for measuring data-copying and experiments indicating its presence in a broad class of generative models. In future work, we plan to establish more theoretical properties of data-copying, convergence guarantees of these tests, and experiments with different model parameters.

2.6 Acknowledgements

We thank Rich Zemel for pointing us to [169], which was the starting point of this work. Thanks to Arthur Gretton and Ruslan Salakhutdinov for pointers to prior work, and Philip Isola and Christian Szegedy for helpful advice. Finally, KC and CM would like to thank ONR under N00014-16-1-2614, UC Lab Fees under LFR 18-548554 and NSF IIS 1617157 for research support.

Chapter 3

Sentence-level Privacy for Document Embeddings

3.1 Introduction

Language models have now become ubiquitous in NLP [51, 120, 6], pushing the state of the art in a variety of tasks [147, 119, 129]. While language models capture meaning and various linguistic properties of text [94, 175], an individual's written text can include highly sensitive information. Even if such details are not needed or used, sensitive information has been found to be vulnerable and detectable to attacks [132, 5, 35]. Reconstruction attacks [167] have even successfully broken through private learning schemes that rely on encryption-type methods [93].

As of now, there is no broad agreement on what constitutes good privacy for natural language [96]. **(author?)** [93] argue that different applications and models require different privacy definitions. Several emerging works propose to apply Metric Differential Privacy [7] at the word level [73, 74, 38, 134, 179, 168] . They propose to add noise to word embeddings, such that they are indistinguishable from their nearest neighbours.

At the document level, however, the above definition has two areas for improvement. First, it may not offer the level of privacy desired. Having each word indistinguishable with similar words may not hide higher level concepts in the document, and may not be satisfactory for many users. Second, it may not be very interpretable or easy to communicate to end-users, since the privacy definition relies fundamentally on the choice of embedding model to determine

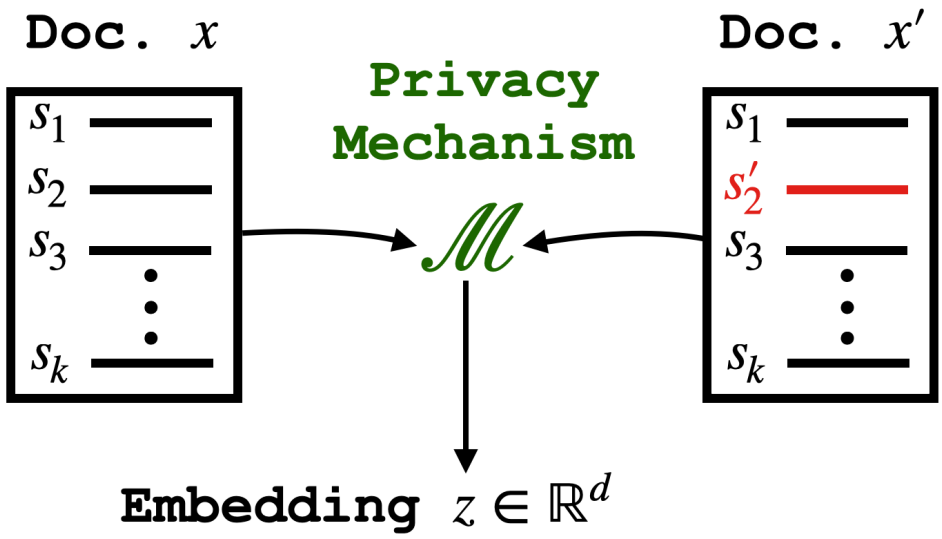


Figure 3.1. x and x' yield $z \in \mathbb{R}^d$ with similar probability.

which words are indistinguishable with a given word. This may not be clear and precise enough for end-users to grasp.

In this work, we propose a new privacy definition for documents: sentence privacy. This guarantee is both strong and interpretable: any sentence in a document must be indistinguishable with *any* other sentence. A document embedding is sentence-private if we can replace any single sentence in the document and have a similar probability of producing the same embedding. As such, the embedding only stores limited information unique to any given sentence. This definition is easy to communicate and strictly stronger than word-level definitions, as modifying a sentence can be changing one word.

Although this definition is strong, we are able to produce unsupervised, general embeddings of documents that are useful for downstream tasks like sentiment analysis and topic classification. To achieve this we propose a novel privacy mechanism, DeepCandidate, which privately samples a high-dimensional embedding from a preselected set of candidate embeddings derived from public, non-private data. DeepCandidate works by first pre-tuning a sentence encoder on public data such that semantically different document embeddings are far apart from each other. Then, we approximate each candidate's Tukey Depth within the private documents'

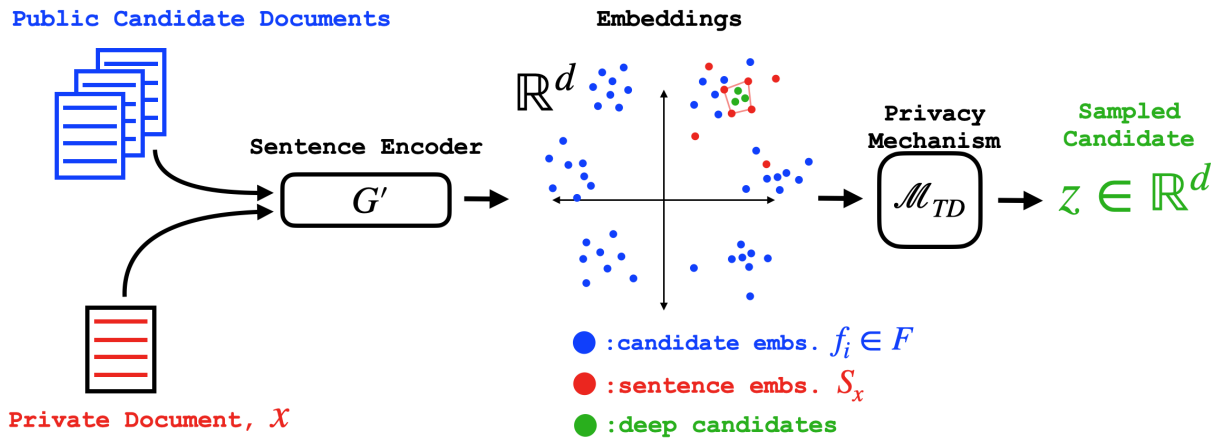


Figure 3.2. DeepCandidate generates a private embedding z of document x by selecting from a set F of public, non-private document embeddings. Sentences from x are encoded by G'. The privacy mechanism $\mathscr{M}_{\text{TD}}$, then privately samples from F, with a preference for candidates with high Tukey Depth, 'deep candidates'. G' is trained beforehand to ensure that deep candidates are likely to exist and are relevant to x.

sentence embeddings. Deeper candidates are the most likely to be sampled to represent the private document. We evaluate DeepCandidate on three illustrative datasets, and show that these unsupervised private embeddings are useful for both sentiment analysis and topic classification as compared to baselines.

In summary, this work makes the following contributions to the language privacy literature:

1. A new, strong, and interpretable privacy definition that offers complete indistinguishability to each sentence in a document.
2. A novel, unsupervised embedding technique, DeepCandidate, to generate sentence-private document embeddings.
3. An empirical assessment of DeepCandidate, demonstrating its advantage over baselines, delivering strong privacy and utility.

3.2 Background and Related Work

Setting.

We denote a 'document' as a sequence of sentences. Let $s \in \mathscr{S}$ be any finite-length sentence. Then, the space of all documents is $\mathscr{X} = \mathscr{S}^*$ and document $x \in \mathscr{X}$ is written as

$x = (s_1, s_2, \ldots, s_k)$ for any non-negative integer k of sentences. In this work, we focus on cohesive documents of sentences written together like reviews or emails, but our methods and guarantees apply to any sequence of sentences, such as a collection of messages written by an individual over some period of time.

Our task is to produce an embedding $z \in \mathbb{R}^d$ of any document $x \in \mathscr{X}$ such that any single sentence $s_i \in x$ is indistinguishable with every other sentence $s_i' \in \mathscr{S} \backslash s_i$. That is, if one were to replace any single sentence in the document $s_i \in x$ with *any other* sentence $s_i' \in \mathscr{S} \backslash s_i$, the probability of producing a given embedding z is similar. To achieve this, we propose a randomized embedding function (the embedding *mechanism*) $\mathscr{M} : \mathscr{X} \to \mathbb{R}^d$, that generates a private embedding $z = \mathscr{M}(x)$ that is useful for downstream tasks.

3.2.1 Differential Privacy

The above privacy notion is inspired by Differential Privacy (DP) [56]. It guarantees that — whether an individual participates (dataset D) or not (dataset D') — the probability of any output only chances by a constant factor.

Definition 3.2.1 (Differential Privacy)**.** Given any pair of datasets $D, D' \in \mathscr{D}$ that differ only in the information of a single individual, we say that the mechanism $\mathscr{A} : \mathscr{D} \to \mathscr{O}$, satisfies ε-DP if

$$\Pr[\mathscr{A}(D) \in O] \leq e^{\varepsilon} \Pr[\mathscr{A}(D') \in O]$$

for any event $O \subseteq \mathscr{O}$.

Note that we take probability over the randomness of the mechanism $\mathscr{A}$ only, not the data distribution. DP has several nice properties that make it easy to work with including closure under post-processing, an additive privacy budget (composition), and closure under group privacy guarantees (guarantees to a *subset* of multiple participants). See **(author?)** 63 for more details.

The *exponential mechanism* [126] allows us to make a DP selection from an arbitrary output space $\mathscr{O}$ based on private dataset D. A *utility function* over input/output pairs, u :

$\mathscr{D} \times \mathscr{O} \to \mathbb{R}$ determines which outputs are the best selection given dataset D. The log probability of choosing output $o \in \mathscr{O}$ when the input is dataset $D \in \mathscr{D}$ is then proportional to its utility $u(D,o)$. The *sensitivity* of $u(\cdot,\cdot)$ is the worst-case change in utility over pairs of neighboring datasets (D,D') that change in one entry, $\Delta u = \max_{D,D',o} |u(D,o) - u(D',o)|$.

Definition 3.2.2. The *exponential mechanism* $\mathscr{A}_{Exp} : \mathscr{D} \to \mathscr{O}$ is a randomized algorithm with output distribution

$$\Pr[\mathscr{A}_{Exp}(D) = o] \propto \exp\left(\frac{\varepsilon u(D,o)}{2\Delta u}\right) \quad .$$

3.2.2 Related Work

Natural Language Privacy.

Previous work has demonstrated that NLP models and embeddings are vulnerable to reconstruction attacks [35, 5, 132]. In response there have been various efforts to design privacy-preserving techniques and definitions across NLP tasks. A line of work focuses on how to make NLP model training satisfy DP [101, 9]. This is distinct from our work in that it satisfies central DP – where data is first aggregated non-privately and then privacy preserving algorithms (i.e. training) are run on that data. We model this work of the *local* version of DP [57], wherein each individual's data is made private before centralizing. Our definition guarantees privacy to a single document as opposed to a single individual.

A line of work more comparable to our approach makes documents locally private by generating a randomized version of a document that satisfies some formal privacy definition. As with the private embedding of our work, this generates locally private *representation* of a given document x. The overwhelming majority of these methods satisfy an instance of Metric-DP [7] at the word level [73, 74, 38, 134, 179, 168]. As discussed in the introduction, this guarantees that a document x is indistinguishable with any other document x' produced by swapping a single word in x with a similar word. Two words are 'similar' if they are close in the word embeddings space (e.g. GloVe). This guarantee is strictly weaker than our proposed definition, SentDP, which

offers indistinguishability to any two documents that differ in an entire sentence.

Privacy-preserving embeddings.

There is a large body of work on non-NLP privacy-preserving embeddings, as these embeddings have been shown to be vulnerable to attacks [144]. **(author?)** [114] attempt to generate locally private embeddings by bounding the embedding space, and we compare with this method in our experiments. **(author?)** [97] propose a method for privately publishing the average of embeddings, but their algorithm is not suited to operate on the small number of samples (sentences) a given document offers. Finally, **(author?)** [16] propose a method for privately learning halfspaces in $\mathbb{R}^d$, which is relevant to private Tukey Medians, but their method would restrict input examples (sentence embeddings) to a finite discrete set in $\mathbb{R}^d$, a restriction we cannot tolerate.

3.3 Sentence-level Privacy

We now introduce our simple, strong privacy definition, along with concepts we use to satisfy it.

3.3.1 Definition

In this work, we adopt the *local* notion of DP [57], wherein each individual's data is guaranteed privacy locally before being reported and centralized. Our mechanism $\mathscr{M}$ receives a single document from a single individual, $x \in \mathscr{X}$. We require that $\mathscr{M}$ provides indistinguishability between documents x, x' differing *in one sentence*.

Definition 3.3.1 (Sentence Privacy, SentDP)**.** Given any pair of documents $x, x' \in \mathscr{X}$ that differ only in one sentence, we say that a mechanism
$\mathscr{M} : \mathscr{X} \to \mathscr{O}$ satisfies ε-SentDP if

$$\Pr[\mathscr{M}(x) \in O] \leq e^{\varepsilon} \Pr[\mathscr{M}(x') \in O]$$

for any event $O \subseteq \mathscr{O}$.

We focus on producing an embedding of the given document x, thus the output space is $\mathscr{O} = \mathbb{R}^d$. For instance, consider the neighboring documents $x = (s_1, s_2, \ldots, s_k)$ and $x' = (s_1, s_2', \ldots, s_k)$ that differ in the second sentence, i.e. s_2, s_2' can be *any* pair of sentences in $\mathscr{S}^2$. This is a strong notion of privacy in comparison to existing definitions across NLP tasks. However, we show that we can guarantee SentDP while still providing embeddings that are useful for downstream tasks like sentiment analysis and classification. In theory, a SentDP private embedding z should be able to encode any information from the document that is not unique to a small subset of sentences. For instance, z can reliably encode the sentiment of x as long as *multiple* sentences reflect the sentiment. By the group privacy property of DP, which SentDP maintains, two documents differing in a sentences are $a\varepsilon$ indistinguishable. So, if more sentences reflect the sentiment, the more $\mathscr{M}$ can encode this into z without compromising on privacy.

3.3.2 Sentence Mean Embeddings

Our approach is to produce a private version of the average of general-purpose sentence embeddings. By the post-processing property of DP, this embedding can be used repeatedly in any fashion desired without degrading the privacy guarantee. Our method makes use of existing pre-trained sentence encoding models. We denote this general sentence encoder as $G : \mathscr{S} \to \mathbb{R}^d$. We show in our experiments that the mean of sentence embeddings,

$$\overline{g}(x) = \frac{1}{k} \sum_{s_i \in x} G(s_i) \ , \tag{3.1}$$

maintains significant information unique to the document and is useful for downstream tasks like classification and sentiment analysis.

We call $\overline{g}(x)$ the *document embedding* since it summarizes the information in document x. While there exist other definitions of document embeddings [173, 152, 18], we decide to use

averaging as it is a simple and established embedding technique [23, 89, 113].

3.3.3 Tukey Depth

Depth is a concept in robust statistics used to describe how central a point is to a distribution. We borrow the definition proposed by **(author?)** [154]:

Definition 3.3.2. Given a distribution P over $\mathbb{R}^d$, the Tukey Depth of a point $y \in \mathbb{R}^d$ is

$$\mathrm{TD}_P(y) = \inf_{w \in \mathbb{R}^d} P\{y' : w \cdot (y' - y) \geq 0\} \quad .$$

In other words, take the hyperplane orthogonal to vector w, h_w, that passes through point y. Let P_1^w be the probability under P that a point lands on one side of h_w and let P_2^w be the probability that a point lands on the other side, so $P_1^w + P_2^w = 1$. y is considered deep if $\min(P_1^w, P_2^w)$ is close to a half for *all* vectors w (and thus all h passing through y). The *Tukey Median* of distribution P, $\mathrm{T}_{\mathrm{MED}}(P)$, is the set of all points with maximal Tukey Depth,

$$\mathrm{T}_{\mathrm{MED}}(P) = \arg\max y \in \mathbb{R}^d \mathrm{TD}_P(y) \quad . \tag{3.2}$$

We only access the distribution P through a finite sample of i.i.d. points, $Y = \{y_1, y_2, \ldots, y_n\}$. The Tukey Depth w.r.t. Y is given by

$$\mathrm{TD}_Y(y) = \inf_{w \in \mathbb{R}^d} |\{y' \in Y : w \cdot (y' - y) \geq 0\}| \ ,$$

and the median, $\mathrm{T}_{\mathrm{MED}}(Y)$, maximizes the depth and is at most half the size of our sample $\left\lfloor \frac{n}{2} \right\rfloor$.

Generally, finding a point in $\mathrm{T}_{\mathrm{MED}}(Y)$ is hard; SOTA algorithms have an exponential dependency in dimension [39], which is a non-starter when working with high-dimensional embeddings. However, there are efficient approximations which we will take advantage of.

3.4 DeepCandidate

While useful and general, the document embedding $\overline{g}(x)$ does not satisfy SentDP. We now turn to describing our privacy-preserving technique, DeepCandidate, which generates general, ε-SentDP document embeddings that preserve relevant information in $\overline{g}(x)$, and are useful for downstream tasks. To understand the nontrivial nature of this problem, we first analyze why the simplest, straightfoward approaches are insufficient.

Motivation.

Preserving privacy for high dimensional objects is known to be challenging [97, 74, 186] . For instance, adding Laplace noise directly to $\overline{g}(x)$, as done to satisfy some privacy definitions [73, 7], does not guarantee SentDP for any ε. Recall that the embedding space is all of $\mathbb{R}^d$. A change in one sentence can lead to an unbounded change in $\overline{g}(x)$, since we do not put any restrictions on the general encoder G. Thus, no matter how much noise we add to $\overline{g}(x)$ we cannot satisfy SentDP.

A straightforward workaround might be to simply truncate embeddings such that they all lie in a limited set such as a sphere or hypercube as done in prior work [114, 4]. In doing so, we bound how far apart embeddings can be for any two sentences, $\|G(s_i) - G(s_i')\|_1$, thus allowing us to satisfy SentDP by adding finite variance noise. However, such schemes offer poor utility due to the high dimensional nature of useful document embeddings (we confirm this in our experiments). We must add noise with standard deviation proportional to the dimension of the embedding, thus requiring an untenable degree of noise for complex encoders like BERT which embed into $\mathbb{R}^{768}$.

Our method has three pillars: **(1)** sampling from a candidate set of public, non-private document embeddings to represent the private document, **(2)** using the Tukey median to approximate the document embedding, and **(3)** pre-training the sentence encoder, G, to produce relevant candidates with high Tukey depth for private document x.

3.4.1 Taking advantage of public data: sampling from candidates

Instead of having our mechanism select a private embedding z from the entire space of $\mathbb{R}^d$, we focus the mechanism to select from a set of m candidate embeddings, F, generated by m public, non-private documents. We assume the document x is drawn from some distribution μ over documents $\mathscr{X}$. For example, if we know x is a restaurant review, μ may be the distribution over all restaurant reviews. F is then a collection of document embeddings over m publicly accessible documents $x_i \sim \mu$,

$$F = \{f_i = \overline{g}(x_i) : x_1, \ldots, x_m \overset{\text{iid}}{\sim} \mu\} ,$$

and denote the corresponding distribution over f_i as $\overline{g}(\mu)$. By selecting candidate documents that are similar in nature to the private document x, we inject an advantageous inductive bias into our mechanism, which is critical to satisfy strong privacy while preserving information relevant to x.

3.4.2 Approximating the document embedding: The Tukey Median

We now propose a novel mechanism $\mathscr{M}_{\text{TD}}$, which approximates $\overline{g}(x)$ by sampling a candidate embedding from F. $\mathscr{M}_{\text{TD}}$ works by concentrating probability on candidates with high Tukey Depth w.r.t. the set of sentence embeddings $S_x = \{G(s_i) : s_i \in x\}$. We model sentences s_i from document x as i.i.d. draws from distribution ν_x. Then, S_x is k draws from $g(\nu_x)$, the distribution of sentences from ν_x passing through G. Deep points are a good approximation of the mean under light assumptions. If $g(\nu_x)$ belongs to the set of halfspace-symmetric distributions (including all elliptic distributions e.g. Gaussians), we know that its mean lies in the Tukey Median [187].

Formally, $\mathscr{M}_{\text{TD}}$ is an instance of the exponential mechanism (Definition 3.2.2), and is defined by its utility function. We set the utility of a candidate document embedding $f_i \in F$ to be

an approximation of its depth w.r.t. sentence embeddings S_x,

$$u(x, f_i) = \widehat{\mathrm{TD}}_{S_x}(f_i) \quad . \tag{3.3}$$

The approximation $\widehat{\mathrm{TD}}_{S_x}$, which we detail in the Appendix, is necessary for computational efficiency. If the utility of f_i is high, we call it a 'deep candidate' for sentence embeddings S_x.

The more candidates sampled (higher m), the higher the probability that at least one has high depth. Without privacy, we could report the deepest candidate, $z = \arg\max f_i \in F \widehat{\mathrm{TD}}_{S_x}(f_i)$. However, when preserving privacy with $\mathscr{M}_{\mathrm{TD}}$, increasing m has diminishing returns. To see this, fix a set of sentence embeddings S_x for document x and the i.i.d. distribution over candidate embeddings $f_i \sim \overline{g}(\mu)$. This induces a multinomial distribution over depth,

$$u_j(x) = \Pr[u(x, f_i) = j], \ \sum_{j=0}^{\lfloor \frac{k}{2} \rfloor} u_j(x) = 1 \ ,$$

where randomness is taken over draws of f_i.

For candidate set F and sentence embeddings S_x, the probability of $\mathscr{M}_{\mathrm{TD}}$'s selected candidate, z, having (approximated) depth j^* is given by

$$\Pr[u(x,z) = j^*] = \frac{a_{j^*}(x) e^{\varepsilon j^*/2}}{\sum_{j=0}^{\lfloor \frac{k}{2} \rfloor} a_j(x) e^{\varepsilon j/2}} \tag{3.4}$$

where $a_j(x)$ is the fraction of candidates in F with depth j w.r.t. the sentence embeddings of document x, S_x. For m sufficiently large, $a_j(x)$ concentrates around $u_j(x)$, so further increasing m does not increase the probability of $\mathscr{M}_{\mathrm{TD}}$ *sampling* a deep candidate.

Table 3.1. Conditions for deep candidates

ε	b	j^*
3	55	5
6	25	3
10	5	2
23	1	1

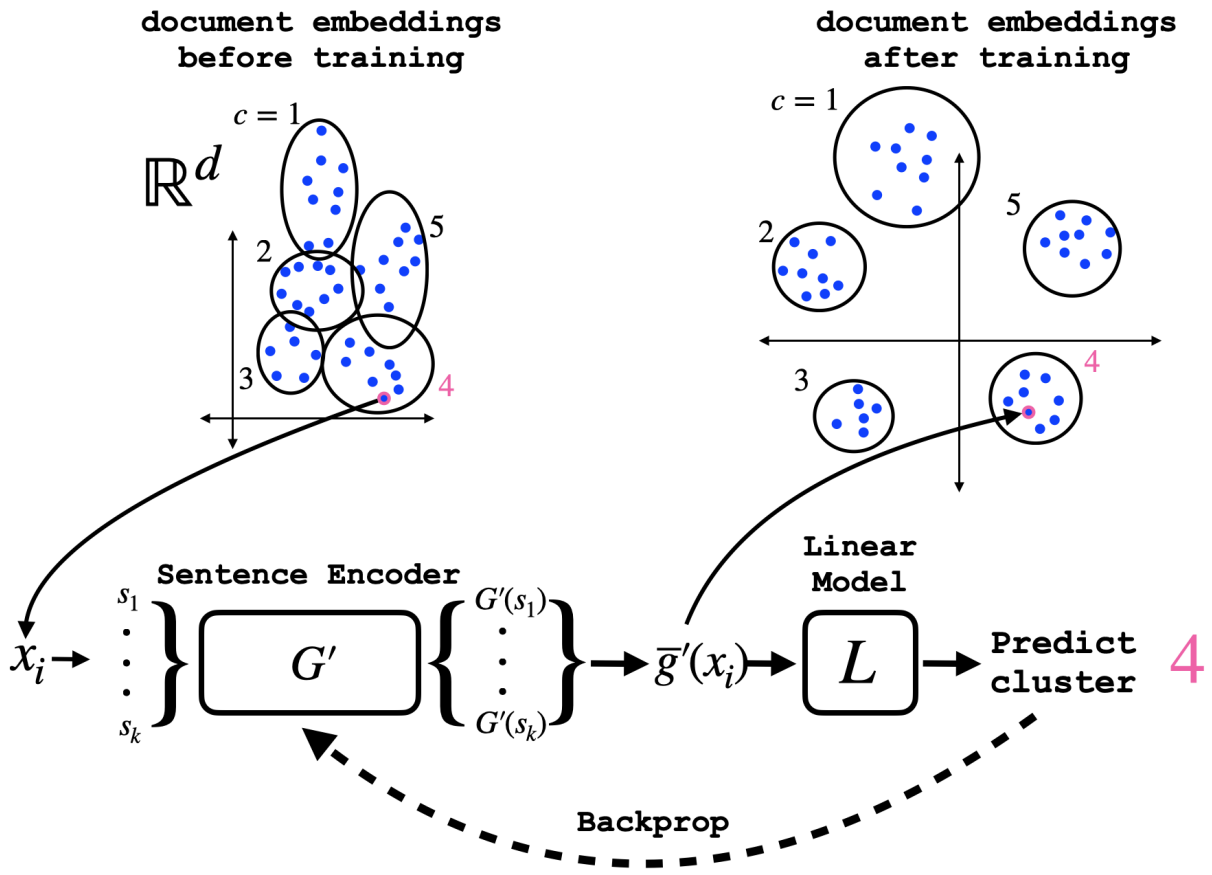


Figure 3.3. G' is trained to encourage similar documents to embed close together and different documents to embed far apart. We first compute embeddings of all (public, non-private) training set documents T with pretrained encoder G, $T_G = \{t_i = \overline{g}(x_i) : x_i \in T\}$ (blue dots). We run k-means to define n_c clusters, and label each training document embedding $t_i \in T_G$ with its cluster c. We then train H to recode sentences to S'_x such that their mean $\overline{g}'(x)$ can be used by a linear model L to predict cluster c. Our training objective is the cross-entropy loss of the linear model L in predicting c.

For numerical intuition, suppose $m = 5000$ (as in our experiments), $\geq b$ candidates have depth $\geq j^*$, and all other candidates have depth 0, $\mathscr{M}_{\text{TD}}$ will sample one of these deep candidates w.p. ≥ 0.95 under the settings in Table 3.1.

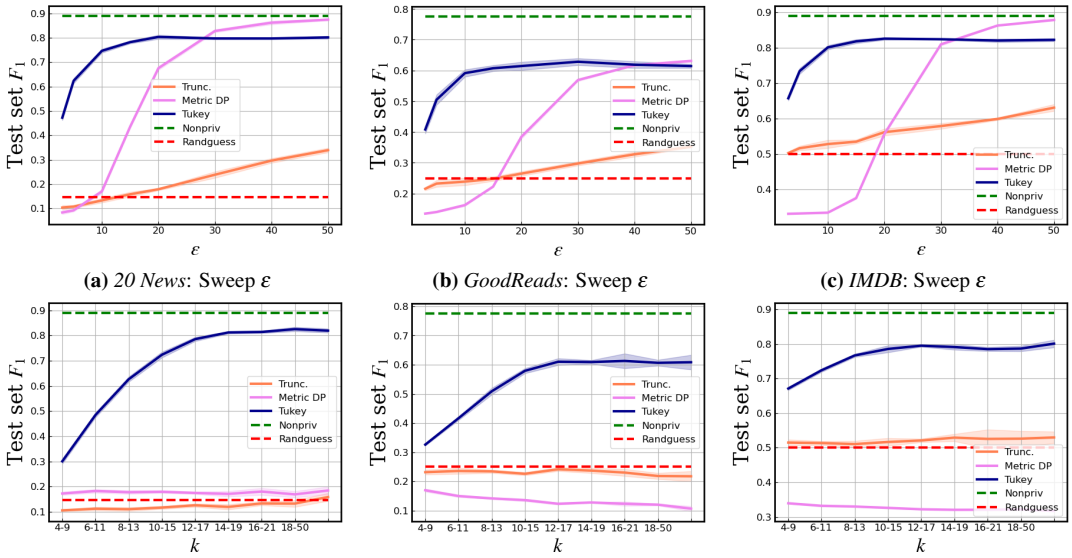
For low $\varepsilon < 10$ (high privacy), about 1% of candidates need to have high depth (≥ 3) in order to be reliably sampled. Note that this is only possible for documents with ≥ 6 sentences. For higher $\varepsilon \geq 10$, $\mathscr{M}_{\text{TD}}$ will reliably sample low depth candidates even if there are only a few.

From these remarks we draw two insights on how DeepCandidate can achieve high utility.

(1) *More sentences* A higher k enables greater depth, and thus a higher probability of sampling

deep candidates with privacy. We explore this effect in our experiments.

(2) *Tuned encoder* By tuning the sentence encoder G for a given domain, we can modify the distribution over document embeddings $\overline{g}(\mu)$ and sentence embeddings $g(\nu_x)$ to encourage deep candidates (high probability u_j for deep j) that are relevant to document x.



(a) *20 News*: Sweep ε **(b)** *GoodReads*: Sweep ε **(c)** *IMDB*: Sweep ε

(d) *20 News*: Sweep k **(e)** *GoodReads*: Sweep k **(f)** *IMDB*: Sweep k

Figure 3.4. Comparison of our mechanism with two baselines: truncation [114] and word-level Metric DP [73] for both sentiment analysis (*IMDB*) and topic classification (*GoodReads*, *20News*) on private, unsupervised embeddings. All plots show test-set macro F_1 scores. The top row shows performance vs. privacy parameter ε (lower is better privacy). The bottom row shows performance vs. number of sentences k with $\varepsilon = 10$. DeepCandidate outperforms both baselines across datasets and tasks. Note that at a given ε, word-level Metric-DP is a significantly weaker privacy guarantee.

3.4.3 Taking advantage of structure: cluster-preserving embeddings

So far, we have identified that deep candidates from F can approximate $\overline{g}(x)$. To produce a good approximation, we need to ensure that 1) there reliably exist deep candidates for any given set of sentence embeddings S_x, and 2) that these deep candidates are good representatives of document x. The general sentence encoder G used may not satisfy this 'out of the box'. If the distribution on document embeddings $\overline{g}(\mu)$ is very scattered around the instance space $\mathbb{R}^{768}$, it can be exceedingly unlikely to have a deep candidate f_i among sentence embeddings

S_x. On the other hand, if distribution $\overline{g}(\mu)$ is tightly concentrated in one region (e.g. 'before training' in Figure 3.3), then we may reliably have many deep candidates, but several will be poor representatives of the document embedding $\overline{g}(x)$.

To prevent this, we propose an unsupervised, efficient, and intuitive modification to the (pretrained) sentence encoder G. We freeze the weights of G and add additional perceptron layers mapping into the same embeddings space $H : \mathbb{R}^d \to \mathbb{R}^d$, producing the extended encoder $G' = H \circ G$. Broadly, we train H to place similar document embeddings close together, and different embeddings far part. To do so, we leverage the assumption that a given domain's distribution over document embeddings $\overline{g}(\mu)$ can be parameterized by n_c clusters, visualized as the black circles in Figure 3.3. H's aim is to recode sentence embeddings such that document embedding clusters are preserved, but spaced apart from each other. By preserving clusters, we are more likely to have deep candidates (increased probability u_j for high depth j). By spacing clusters apart, these deep candidates are more likely to come from the same or a nearby cluster as document x, and thus be good representatives. Note that H is domain-specific: we train separate H encoders for each dataset.

3.4.4 Sampling Algorithm

The final component of DeepCandidate is computing the approximate depth of a candidate for use as utility in the exponential mechanism as in Eq. (3.3). We use a version of the approximation algorithm proposed in [82]. Intuitively, our algorithm computes the one-dimensional depth of each f_i among x's sentence embeddings S_x on each of p random projections. The approximate depth of f_i is then its lowest depth across the p projections. We are guaranteed that $\widehat{\text{TD}}_{S_x}(f_i) \geq \text{TD}_{S_x}(f_i)$. Due to space constraints, we leave the detailed description of the algorithm for the Appendix.

Theorem 3.4.1. $\mathscr{M}_{\text{TD}}$ satisfies ε-Sentence Privacy

Proof follows from the fact that $\widehat{\text{TD}}_{S_x}(f_i)$ has bounded sensitivity (changing one sentence

can only change depth of f_i by one). We expand on this, too, in the Appendix.

3.5 Experiments

3.5.1 Datasets

We produce private, general embeddings of documents from three English-language datasets:

Good Reads [159] 60k book reviews from four categories: fantasy, history, romance, and childrens literature. Train-48k — Val-8k — Test-4k

20 News Groups [110] 11239 correspondences from 20 different affinity groups. Due to similarity between several groups (e.g. `comp.os.ms-windows.misc` and `comp.sys.ibm.pc.hardware`), the dataset is partitioned into nine categories. Train-6743k — Val-2247k — Test-2249k

IMDB [122] 29k movie reviews from the IMDB database, each labeled as a positive or negative review. Train-23k — Val-2k — Test-4k

To evaluate utility of these unsupervised, private embeddings, we check if they are predictive of document properties. For the *Good Reads* and *20 News Groups* datasets, we evaluate how useful the embeddings are for topic classification. For *IMDB* we evaluate how useful the embeddings are for sentiment analysis (positive or negative review). Our metric for performance is test-set macro F_1 score.

3.5.2 Training Details & Setup

For the general encoder, $G : \mathscr{S} \to \mathbb{R}^{768}$, we use SBERT [135], a version of BERT fine-tuned for sentence encoding. Sentence embeddings are generated by mean-pooling output tokens. In all tasks, we freeze the weights of SBERT. The cluster-preserving recoder, H, as well as every classifier is implemented as an instance of a 4-layer MLP taking 768-dimension inputs and only differing on output dimension. We denote an instance of this MLP with output dimension o as $\mathbf{MLP}^o$. We run 5 trials of each experiment with randomness taken over the privacy mechanisms,

and plot the mean along with a $\pm$ 1 standard deviation envelope.

DeepCandidate:

The candidate set F consists of 5k document embeddings from the training set, each containing at least 8 sentences. To train G', we find $n_c = 50$ clusters with k-means. We train a classifier $C_{\text{dc}} = \mathbf{MLP}^r$ on document embeddings $g'(x)$ to predict class, where r is the number of classes (topics or sentiments).

3.5.3 Baselines

We compare the performance of DeepCandidate with 4 baselines: **Non-private**, **Truncation**, **Word-level Metric-DP**, and **Random Guesser**.

Non-private: This demonstrates the usefulness of non-private sentence-mean document embeddings $\overline{g}(x)$. We generate $\overline{g}(x)$ for every document using SBERT, and then train a classifier $C_{\text{nonpriv}} = \mathbf{MLP}^r$ to predict x's label from $\overline{g}(x)$.

Truncation: We adopt the method from **(author?)** 114 to truncate (clip) sentence embeddings within a box in $\mathbb{R}^{768}$, thereby bounding sensitivity as described at the beginning of Section 3.4. Laplace noise is then added to each dimension. Documents with more sentences have proportionally less noise added due to the averaging operation reducing sensitivity.

Word Metric-DP (MDP): The method from **(author?)** 73 satisfies ε-word-level metric DP by randomizing words. We implement MDP to produce a randomized document x', compute $\overline{g}(x')$ with SBERT, and predict class using C_{nonpriv}.

Random Guess: To set a bottom-line, we show the theoretical performance of a random guesser only knowing the distribution of labels.

3.5.4 Results & Discussion

How does performance change with privacy parameter ε?

This is addressed in Figures 3.4a to 3.4c. Here, we observe how the test set macro F_1 score changes with privacy parameter ε (a lower ε offers stronger privacy). Generally speaking, for

local differential privacy, $\varepsilon < 10$ is taken to be a strong privacy regime, $10 \leq \varepsilon < 20$ is moderate privacy, and $\varepsilon \geq 25$ is weak privacy. The **truncation** baseline mechanism does increase accuracy with increasing ε, but never performs much better than the random guesser. This is to be expected with high dimension embeddings, since the standard deviation of noise added increases linearly with dimension.

The word-level **MDP** mechanism performs significantly better than **truncation**, achieving relatively good performance for $\varepsilon \geq 30$. There are two significant caveats, however. First, is the privacy definition: as discussed in the Introduction, for the same ε, word-level MDP is strictly weaker than SentDP. The second caveat is the level of ε at which privacy is achieved. Despite a weaker privacy definition, the MDP mechanism does not achieve competitive performance until the weak-privacy regime of ε. We suspect this is due to two reasons. First, is the fact that the MDP mechanism does not take advantage of contextual information in each sentence as our technique does; randomizing each word independently does not use higher level linguistic information. Second, is the fact that the MDP mechanism does not use domain-specific knowledge as our mechanism does with use of relevant candidates and domain specific sentence encodings.

In comparison, DeepCandidate offers strong utility across tasks and datasets for relatively low values of ε, even into the strong privacy regime. Beyond $\varepsilon = 25$, the performance of DeepCandidate tends to max out, approximately 10-15% below the non-private approach. This is due to the fact that DeepCandidate offers a noisy version of an *approximation* of the document embedding $\overline{g}(x)$ – it cannot perform any better than deterministically selecting the deepest candidate, and even this candidate may be a poor representative of x. We consider this room for improvement, since there are potentially many other ways to tune G' and select the candidate pool F such that deep candidates are nearly always good representatives of a given document x.

How does performance change with the number of sentences k?

This is addressed in Figures 3.4d to 3.4f. We limit the test set to those documents with k in the listed range on the x-axis. We set $\varepsilon = 10$, the limit of the strong privacy regime. Neither baseline offers performance above that of the random guesser at this value of ε. DeepCandidate produces

precisely the performance we expect to see: documents with more sentences result in sampling higher quality candidates, confirming the insights of Section 3.4.2. Across datasets and tasks, documents with more than 10-15 sentences tend to have high quality embeddings.

3.6 Conclusions and Future Work

We introduce a strong and interpretable local privacy guarantee for documents, SentDP, along with DeepCandidate, a technique that combines principles from NLP and robust statistics to generate general ε-SentDP embeddings. Our experiments confirm that such methods can outperform existing approaches even with with more relaxed privacy guarantees. Previous methods have argued that it is "virtually impossible" to satisfy pure local DP [73, 74] at the word level while capturing linguistic semantics. Our work appears to refute this notion at least at the document level.

To follow up, we plan to explore other approaches (apart from k-means) of capturing the structure of the embedding distribution $\overline{g}(\mu)$ to encourage better candidate selection. We also plan to experiment with decoding private embeddings back to documents by using novel candidates produced by a generative model trained on F.

Acknowledgements

KC and CM would like to thank ONR under N00014-20-1-2334. KM gratefully acknowledges funding from an Amazon Research Award and Adobe Unrestricted Research Gifts. We would would also like to thank our reviewers for their insightful feedback.

Chapter 4

Privacy Implications of Shuffling

4.1 Introduction

Differential Privacy (DP) and its local variant (LDP) are the most commonly accepted notions of data privacy. LDP has the significant advantage of not requiring a trusted centralized aggregator, and has become a popular model for commercial deployments, such as those of Microsoft [52], Apple [84], and Google [66, 70, 21]. Its formal guarantee asserts that an adversary cannot infer the value of an individual's private input by observing the noisy output. However in practice, a vast amount of *public auxiliary information*, such as address, social media connections, court records, property records, income and birth dates [109], is available for every individual. An adversary, with access to such auxiliary information, *can* learn about an individual's private data from several *other* participants' noisy responses. We illustrate this as follows.

Problem. An analyst runs a medical survey in Alice's community to investigate how the prevalence of a highly contagious disease changes from neighborhood to neighborhood. Community members report a binary value indicating whether they have the disease.

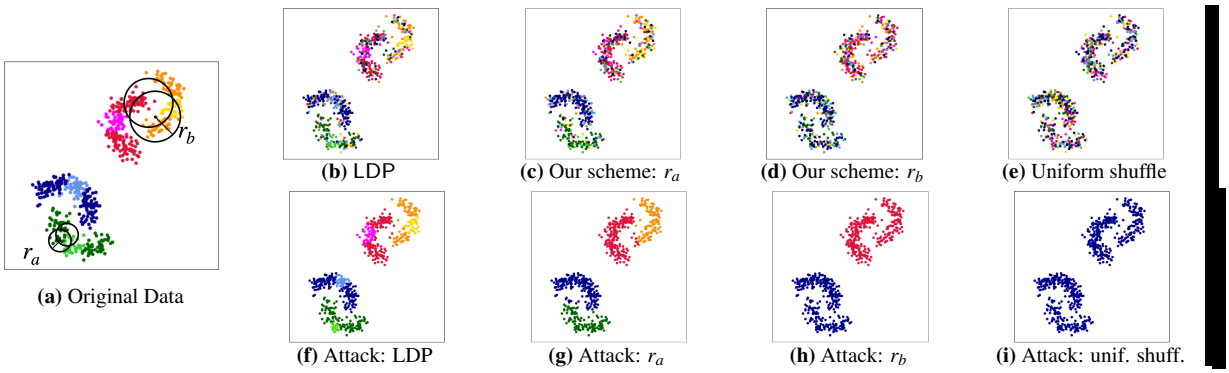
Next, consider the following two data reporting strategies.

Strategy 1. Each data owner passes their data through an appropriate randomizer (that flips the input bit with some probability) in their local devices and reports the noisy output to the untrusted data analyst.

Strategy 2. The noisy responses from the local devices of each of the data owners are collected by an intermediary trusted shuffler which dissociates the device IDs (metadata) from the responses and uniformly randomly shuffles them before sending them to the analyst.

Strategy 1 corresponds to the standard LDP deployment model (for example, Apple and Microsoft's deployments). Here *the order of the noisy responses is informative of the identity of the data owners* – the noisy response at index 1 corresponds to the first data owner and so on. Thus, the noisy responses can be directly linked with its associated device/account ID and subsequently, auxiliary information. This puts Alice's data under the threat of inference attacks. For instance, an adversary[1] may know the home addresses of the participants and use this to identify the responses of all the individuals from Alice's household. Being highly infectious, all or most of them will have the same true value (0 or 1). Hence, the adversary can reliably infer Alice's value by taking a simple majority vote of her and her household's noisy responses. Note that this does not violate the LDP guarantee since the inputs are appropriately randomized when observed in isolation. Additionally, on account of being public, the auxiliary information is known to the adversary (and analyst) *a priori* – no mechanism can prevent their disclosure. For instance, any attempts to include Alice's address as an additional feature of the data and then report via LDP is *futile* – the adversary would simply discard the reported noisy address and use the auxiliary information about the exact addresses to identify the responses of her household members. We call such threats *inference attacks* – recovering an individual's private input using all or a subset of other participants' noisy responses. It is well known that protecting against inference attacks that rely on underlying data correlations is beyond the purview of

[1]The analyst and the adversary could be same, we refer to them separately for the ease of understanding.



(a) Original Data **(b)** LDP **(c)** Our scheme: r_a **(d)** Our scheme: r_b **(e)** Uniform shuffle

(f) Attack: LDP **(g)** Attack: r_a **(h)** Attack: r_b **(i)** Attack: unif. shuff.

Figure 4.1. Demonstration of how our proposed scheme thwarts inference attacks at different granularities. Fig. 4.1a depicts the original sensitive data (such as income bracket) with eight color-coded labels. The position of the points represents public information (such as home address) used to correlate them. There are three levels of granularity: warm vs. cool clusters, blue vs. green and red vs. orange crescents, and light vs. dark within each crescent. Fig. 4.1b depicts $\varepsilon = 2.55$ LDP. Fig. 4.1c and 4.1d correspond to our scheme, each with $\alpha = 1$ (privacy parameter, Def. 4.4.3). The former uses a smaller distance threshold (r_1, used to delineate the granularity of grouping – see Sec. 4.4.2) that mostly shuffles in each crescent. The latter uses a larger distance threshold (r_2) that shuffles within each cluster. Figures in the bottom row demonstrate an inference attack (uses Gaussian process correlation) on all four cases. We see that LDP reveals almost the entire dataset (Fig. 4.1f) while uniform shuffling prevents all classification (4.1i). However, the granularity can be controlled with our scheme (Figs. 4.1g, 4.1h).

DP [105, 153].

Strategy 2 corresponds to the recently introduced shuffle DP model, such as Google's Prochlo [21]. Here, the noisy responses are completely anonymized – the adversary cannot identify which LDP responses correspond to Alice and her household. Under such a model, only information that is completely order agnostic (i.e., symmetric functions that can be computed over just the *bag* of values, such as aggregate statistics) can be extracted. Consequently, the analyst also fails to accomplish their original goal as all the underlying data correlation is destroyed.

Thus, we see that the two models of deployment for LDP present a trade-off between vulnerability to inference attacks and scope of data learnability. In fact, as demonstrated in [103], it is impossible to defend against *all* inference attacks while simultaneously maintaining utility for learning. In the extreme case that the adversary knows *everyone* in Alice's community

has the same true value (but not which one), no mechanism can prevent revelation of Alice's datapoint short of destroying all utility of the dataset. This then begs the question: ***Can we formally suppress specific inference attacks targeting each data owner while maintaining some meaningful learnability of the private data?*** Referring back to our example, can we thwart attacks inferring Alice's data using specifically her households' responses and still allow the medical analyst to learn its target trends? Can we offer this to every data owner participating?

In this paper, we strike a balance and propose a generalized shuffle framework that meets the utility requirements of the above analyst while formally protecting data owners against inference attacks. Our solution is based on the key insight: *the order of the data acts as the proxy for the identity of data owners* as illustrated above. The granularity at which the ordering is maintained formalizes resistance to inference attacks while retaining some meaningful learnability of the private data. Specifically, we guarantee each data owner that their data is shuffled together with a carefully chosen group of other data owners. Revisiting our example, consider uniformly shuffling the responses from Alice's household and her immediate neighbors. Now an adversary cannot use her household's responses to predict her value any better than they could with a random sample of responses from this group. In the same way that LDP prevents reconstruction of her datapoint using specifically *her* noisy response, this scheme prevents reconstruction of her datapoint using specifically *her households'* responses. The real challenge is offering such guarantees *equally* to *every* data owner. Bob, Alice's neighbor, needs his households' responses shuffled in with his neighbors, as does Luis who is a neighbor of Bob but *not* of Alice. Thus, we have n data owners with n distinct, overlapping groups. Our scheme supports arbitrary groupings (overlapping or not), introducing a diverse and tunable class of privacy/utility trade-offs which is not attainable with either LDP or uniform shuffling alone. For the above example, our scheme can formally protect each data owner from inference attacks using specifically their household, while still learning how disease prevalence changes across the neighborhoods of Alice's community.

This work offers two key contributions to the machine learning privacy literature:

- **Novel privacy guarantee.** We propose a novel privacy definition, d_σ-privacy that captures the privacy of the *order* of a data sequence (Sec. 4.4.2) and formalizes the degree of resistance against inference attacks (Sec. 4.4.3). d_σ-privacy allows assigning an arbitrary group, G_i, to each data owner, $\mathsf{DO}_i, i \in [n]$. For instance, the groups can represent individuals in the same age bracket, 'friends' on social media, or individuals living in each other's vicinity (as in case of Alice in our example). Recall that the order is informative of the data owner's identity. Intuitively, d_σ-privacy protects DO_i from inference attacks that arise from knowing the *identity* of the members of their group G_i (Sec. 4.4.3). Additionally, this grouping determines a threshold of learnability – any learning that is order agnostic within a group (disease prevalence in a neighborhood – the data analyst's goal in our example) is utilitarian and allowed; whereas analysis that involves identifying the values of individuals within a group (disease prevalence within specific households – the adversary's goal) is regarded as a privacy threat and protected against. See Fig. 4.1 for a toy demonstration of how our guarantee allows *tuning the granularity at which trends can be learned.*
- **Novel shuffle framework.** We propose a novel mechanism that shuffles the data systematically and achieves d_σ-privacy. This provides a generalized shuffle framework that interpolates between no shuffling (LDP) and uniform random shuffling (shuffle model) in terms of protection against inference attacks and data learnability.

4.2 Related Work

The shuffle model of DP [22, 45, 65] differs from our scheme as follows. These works (1) study DP benefits of shuffling whereas we study the inferential privacy benefits, and (2) only study uniformly random shuffling where ours generalizes this to tunable, non-uniform shuffling (see App. D.1.15).

A steady line of work has studied inferential privacy [98, 103, 81, 47, 60, 153]. Our work departs from those in that we focus on *local* inferential privacy and do so via the new angle of shuffling. Older works such as k-anonymity [149], l-diversity [123], Anatomy [165] and others [163,

150, 170, 46, 54] have studied the privacy risk of non-sensitive auxiliary information or 'quasi identifiers'. These works (1) focus on the setting of dataset release, whereas we focus on dataset collection, and (2) do not offer each data owner formal inferential guarantees, whereas we do. The De Finetti attack [102] shows how shuffling schemes are vulnerable to inference attacks that correlate records to recover the original permutation of sensitive attributes. A strict instance of our privacy guarantee can thwart such attacks (at the cost of no utility, App. D.1.3).

4.3 Background

Notations. Boldface (such as $\mathbf{x} = \langle x_1, \cdots, x_n \rangle$) denotes a data sequence (ordered list); normal font (such as x_1) denotes individual values and $\{\cdot\}$ represents a multiset or bag of values.

4.3.1 Local Differential Privacy

The local model consists of a set of data owners and an untrusted data aggregator (analyst); each individual perturbs their data using a LDP algorithm (randomizers) and sends it to the analyst. The LDP guarantee is formally defined as

Definition 4.3.1. [Local Differential Privacy, LDP [160, 68, 99]] A randomized algorithm $\mathscr{M} : \mathscr{X} \to \mathscr{Y}$ is ε-locally differentially private (or ε-LDP), if for any pair of private values $x, x' \in \mathscr{X}$ and any subset of output,

$$\Pr\big[\mathscr{M}(x) \in \mathscr{W}\big] \leq e^{\varepsilon} \cdot \Pr\big[\mathscr{M}(x') \in \mathscr{W}\big] \tag{4.1}$$

The shuffle model is an extension of the local model where the data owners first randomize their inputs. Additionally, an intermediate trusted shuffler applies a *uniformly random permutation* to all the noisy responses before the analyst can view them. The anonymity provided by the shuffler requires less noise than the local model for achieving the same privacy.

4.3.2 Mallows Model

A permutation of a set S is a bijection $S \mapsto S$. The set of permutations of $[n], n \in \mathbb{N}$ forms a symmetric group S_n. As a shorthand, we use $\sigma(\mathbf{x})$ to denote applying permutation $\sigma \in \mathrm{S}_n$ to a data sequence $\mathbf{x}$ of length n. Additionally, $\sigma(i), i \in [n], \sigma \in \mathrm{S}_n$ denotes the value at index i in σ and σ^{-1} denotes its inverse. For example, if $\sigma = (1\,3\,5\,4\,2)$ and $\mathbf{x} = \langle 21, 33, 45, 65, 67 \rangle$, then $\sigma(\mathbf{x}) = \langle 21, 45, 67, 65, 33 \rangle$, $\sigma(2) = 3, \sigma(3) = 5$ and $\sigma^{-1} = (1\,5\,2\,4\,3)$.

Mallows model is a popular probabilistic model for permutations [124]. The mode of the distribution is given by the reference permutation σ_0 – the probability of a permutation increases as we move 'closer' to σ_0 as measured by rank distance metrics, such as the Kendall's tau distance (Def. D.1.2). The dispersion parameter θ controls how fast this increase happens.

Definition 4.3.2. For a dispersion parameter θ, a reference permutation $\sigma_o \in \mathrm{S}_n$, and a rank distance measure $\delta : \mathrm{S}_n \times \mathrm{S}_n \mapsto \mathbb{R}$, $\mathbb{P}_{\Theta,\delta}(\sigma : \sigma_0) = \frac{1}{\psi(\theta,\delta)} e^{-\theta\delta(\sigma,\sigma_0)}$ is the Mallows model where $\psi(\theta,\delta) = \sum_{\sigma \in \mathrm{S}_n} e^{-\theta\delta(\sigma,\sigma_0)}$ is a normalization term and $\sigma \in \mathrm{S}_n$.

4.4 Data Privacy and Shuffling

In this section, we present d_σ-privacy and a shuffling mechanism capable of achieving the d_σ-privacy guarantee.

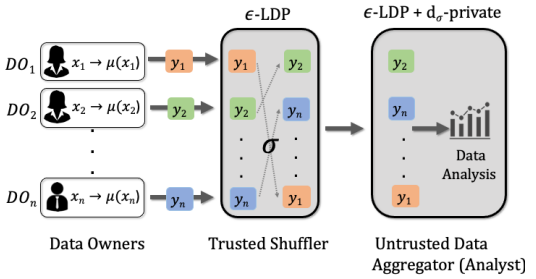


Figure 4.2. Trusted shuffler mediates on $\mathbf{y}$

4.4.1 Problem Setting

In our problem setting, we have n data owners $\mathrm{DO}_i, i \in [n]$ each with a private input $x_i \in \mathscr{X}$ (Fig. 4.2). The data owners first randomize their inputs via a ε-LDP mechanism to generate $y_i = \mathscr{M}(x_i)$. Additionally, just like in the shuffle model, we have a trusted shuffler. It mediates upon the noisy responses $\mathbf{y} = \langle y_1, \cdots, y_n \rangle$ to obtain the final output sequence $\mathbf{z} = \mathscr{A}(\mathbf{y})$ ($\mathscr{A}$ corresponds to Alg. 1) which is sent to the untrusted data analyst. The shuffler can be implemented via trusted execution environments (TEE) just like Google's Prochlo. Next, we formally discuss the notion of order and its implications.

Definition 4.4.1. (Order) The order of a sequence $\mathbf{x} = \langle x_1, \cdots, x_n \rangle$ refers to the indices of its set of values $\{x_i\}$ and is represented by permutations from S_n.

When the noisy response sequence $\mathbf{y} = \langle y_1, \cdots, y_n \rangle$ is represented by the identity permutation $\sigma_I = (1\,2\,\cdots\,n)$, the value at index 1 corresponds to DO_1 and so on. Standard LDP releases the identity permutation w.p. 1. The output of the shuffler, $\mathbf{z}$, is some permutation of the sequence $\mathbf{y}$, i.e.,

$$\mathbf{z} = \sigma(\mathbf{y}) = \langle y_{\sigma(1)}, \cdots, y_{\sigma(n)} \rangle$$

where σ is determined via $\mathscr{A}(\cdot)$. For example, for $\sigma = (4\,5\,2\,3\,1)$, we have $\mathbf{z} = \langle y_4, y_5, y_2, y_3, y_1 \rangle$ which means that the value at index 1 (DO_1) now corresponds to that of DO_4 and so on.

4.4.2 Definition of d_σ-privacy

Inferential risk captures the threat of an adversary who infers DO_i's private x_i using all or a subset of other data owners' released y_j's. Since we cannot prevent all such attacks and maintain utility, our aim is to formally limit *which data owners* can be leveraged in inferring DO_i's private x_i. To make this precise, each DO_i may choose a corresponding group, $G_i \subseteq [n]$, of data owners. d_σ-privacy guarantees that y_j values originating from a data owner's group G_i are shuffled together. In doing so, the LDP values corresponding to subsets of DO_i's group $I \subset G_i$ cannot be reliably identified, and thus cannot be singled out to make inferences about DO_i's x_i. If Alice's group includes her whole neighborhood, LDP data originating from her household cannot be singled out to recover her private x_i.

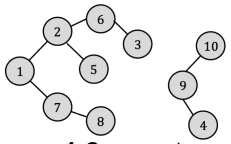


Figure 4.3. An example social media connectivity graph $\mathbf{t}_{e.g}$

Any choice of grouping $\mathscr{G} = \{G_1, G_2, \ldots, G_n\}$ can be accommodated under d_σ-privacy. Each data owner may choose a group large enough to hide anyone they feel sufficient risk from. We outline two systematic approaches to assigning groups as follows:

- Let $\mathbf{t} = \langle t_1, \cdots, t_n \rangle, t_i \in \mathscr{T}$ denote some public auxiliary information about each individual. DO_i's group, G_i, could consist of all those DO_j's who are similar to DO_i w.r.t. the public

auxiliary information t_i, t_j according to some distance measure $d : \mathscr{T} \times \mathscr{T} \to \mathbb{R}$. Here, we define 'similar' as being under a threshold[2] $r \in \mathbb{R}$ such that $G_i = \{j \in [n] \big| d(t_i, t_j) \leq r\}, \forall i \in [n]$. For example, $d(\cdot)$ can be Euclidean distance if $\mathscr{T}$ corresponds to geographical locations, thwarting inference attacks leveraging one's household or immediate neighbors. If $\mathscr{T}$ represents a social media connectivity graph, $d(\cdot)$ can measure the path length between two nodes, thwarting inference attacks using specifically one's close friends. For the example social media connectivity graph depicted in Fig. 4.3, assuming distance metric path length and $r = 2$, the groups are defined as $G_1 = \{1,7,8,2,5,6\}, G_2 = \{2,1,7,5,6,3\}$ and so on.

- Alternatively, the data owners might opt for a group of a specific size $r < n$. Collecting private data from a social media network, we may set $r = 50$, where each G_i is encouraged to include the 50 data owners DO_i interacts with most frequently.

Intuitively, d_σ-privacy protects DO_i against inference attacks that leverages correlations at a finer granularity than G_i. In other words, under d_σ-privacy, one subset of k data owners $\subset G_i$ (e.g. household) is no more useful for targeting x_i than any other subset of k data owners $\subset G_i$ (e.g. some combination of neighbors). This leads to the following key insight for the formal privacy definition.

Key Insight. Formally, our privacy goal is to prevent the leakage of ordinal information from within a group. We achieve this by systematically *bounding the dependence of the mechanism's output on the relative ordering (of data values corresponding to the data owners) within each group*.

First, we introduce the notion of neighboring permutations.

Definition 4.4.2. (Neighboring Permutations) Given a group assignment $\mathscr{G}$, two permutations $\sigma, \sigma' \in \mathrm{S}_n$ are defined to be neighboring w.r.t. a group $G_i \in \mathscr{G}$ (denoted as $\sigma \approx_{G_i} \sigma'$) if $\sigma(j) = \sigma'(j)\ \forall j \notin G_i$.

Neighboring permutations differ only in the indices of its corresponding group G_i. For example, $\sigma = (\underline{1}\,\underline{2}\,4\,5\,\underline{7}\,\underline{6}\,\underline{10}\,\underline{3}\,8\,9)$ and $\sigma' = (\underline{7}\,\underline{3}\,4\,5\,\underline{6}\,\underline{2}\,\underline{1}\,\underline{10}\,8\,9)$ are neighboring w.r.t G_1 (Fig. 4.3)

[2]We could also have different thresholds, r_i, for every data owner, DO_i.

since they differ only in $\sigma(1), \sigma(2), \sigma(5), \sigma(6), \sigma(7)$ and $\sigma(8)$. We denote the set of all neighboring permutations as

$$\mathrm{N}_{\mathscr{G}} = \{(\sigma, \sigma') | \sigma \approx_{G_i} \sigma', \exists G_i \in \mathscr{G}\} \tag{4.2}$$

Now, we formally define d_σ-privacy as follows.

Definition 4.4.3 (d_σ-privacy)**.** For a given group assignment $\mathscr{G}$ on a set of n entities and a privacy parameter $\alpha \in \mathbb{R}_{\geq 0}$, a randomized mechanism $\mathscr{A} : \mathscr{Y}^n \mapsto \mathscr{V}$ is $(\alpha, \mathscr{G})$-d_σ private if for all $\mathbf{y} \in \mathscr{Y}^n$ and neighboring permutations $\sigma, \sigma' \in \mathrm{N}_{\mathscr{G}}$ and any subset of output $O \subseteq \mathscr{V}$, we have

$$\Pr[\mathscr{A}\big(\sigma(\mathbf{y})\big) \in O] \leq e^{\alpha} \cdot \Pr\big[\mathscr{A}\big(\sigma'(\mathbf{y})\big) \in O\big] \tag{4.3}$$

$\sigma(\mathbf{y})$ and $\sigma'(\mathbf{y})$ are defined to be *neighboring sequences*.

d_σ-privacy states that, for any group G_i, the mechanism is (almost) agnostic of the order of the data within the group. Even after observing the output, an adversary cannot learn about the relative ordering of the data within any group. Thus, two neighboring sequences are indistinguishable to an adversary. An important property of d_σ-privacy is that post-processing computations does not degrade privacy. Additionally, when applied multiple times, the privacy guarantee degrades gracefully. Both the properties are analogous to DP and are presented in App. D.1.4.

Note. Any data sequence $\mathbf{x} = \langle x_1, \cdots, x_n \rangle$ can be viewed as a two-tuple, $\big(\{x\}, \sigma\big)$, where $\{x\}$ denotes the *bag* of values and $\sigma \in S_n$ denotes the corresponding indices of the values which represents the *order* of the data. The ε-LDP protects the bag of data values, $\{x\}$, while d_σ-privacy protects the order, σ. Thus, the two privacy guarantees cater to orthogonal parts of a data sequence (see Thm. 5). Also, $\alpha = \infty\ (0), r = 0\ (n)$ represents the standard LDP (shuffle DP) setting.

4.4.3 Privacy Implications

The group assignment $\mathscr{G}$ delineates a threshold of learnability which determines the privacy/utility tradeoff as follows.

- **Learning allowed (Analyst's goal)**. d_σ-privacy can answer queries that are order agnostic within groups, such as aggregate statistics of a group. In Alice's case, the analyst can estimate the disease prevalence in her neighborhood.
- **Learning disallowed (Adversary's goal)**. Adversaries cannot identify (noisy) values of individuals within any group. While they may learn the disease prevalence in Alice's neighborhood, they cannot determine the prevalence within her household and use that to recover her value x_i.

To make this precise, we first formalize the privacy implications of the d_σ guarantee in the standard Bayesian framework, typically used for studying inferential privacy. Next, we formalize the privacy provided by the combination of LDP and d_σ guarantees by way of a decision theoretic adversary.

Bayesian Adversary. Consider a Bayesian adversary with any prior $\mathscr{P}$ on the joint distribution of noisy responses, $\Pr_{\mathscr{P}}[\mathbf{y}]$, which models their beliefs on the correlation between the participants (such as the correlation between Alice and her households' disease status). Their goal is to infer DO_i's private input x_i. As with early DP works [59], we consider an *informed* adversary. Here, the adversary knows (1) the sequence (assignment) of noisy values outside G_i, $\mathbf{y}_{\overline{G}_i}$, and (2) the (unordered) bag of noisy values in G_i, $\{y_{G_i}\}$. d_σ-privacy bounds the prior-posterior odds gap on x_i for such as informed adversary as follows:

Theorem 4. *For a given group assignment $\mathscr{G}$ on a set of n data owners, if a shuffling mechanism $\mathscr{A} : \mathscr{Y}^n \mapsto \mathscr{Y}^n$ is $(\alpha, \mathscr{G})$-$d_\sigma$private, then for each data owner $\mathsf{DO}_i, i \in [n]$,*

$$\max_{\substack{i\in[n]\\ a,b\in\mathscr{X}}} \left| \log \frac{\Pr_{\mathscr{P}}[x_i = a|\mathbf{z}, \{y_{G_i}\}, \mathbf{y}_{\overline{G}_i}]}{\Pr_{\mathscr{P}}[x_i = b|\mathbf{z}, \{y_{G_i}\}, \mathbf{y}_{\overline{G}_i}]} - \log \frac{\Pr_{\mathscr{P}}[x_i = a|\{y_{G_i}\}, \mathbf{y}_{\overline{G}_i}]}{\Pr_{\mathscr{P}}[x_i = b|\{y_{G_i}\}, \mathbf{y}_{\overline{G}_i}]} \right| \leq \alpha$$

for a prior distribution $\mathscr{P}$, where $\mathbf{z} = \mathscr{A}(\mathbf{y})$ and $\mathbf{y}_{\overline{G}_i}$ is the noisy sequence for data owners outside

G_i.

See App D.1.5 for the proof and further discussion on the semantic meaning of the above guarantee.

Decision Theoretic Adversary. Here, we analyse the privacy provided by the combination of LDP and d_σ guarantees. Consider a decision theoretic adversary who aims to identify the noisy responses, $\{z_I\}$, that originated from a specific subset of data owners, $I \subset G_i$ (such as the members of Alice's household). We denote the adversary by a (possibly randomized) function mapping from the output $\mathbf{z}$ sequence to a set of k indices, $\mathscr{D}_{Adv} : \mathscr{Y}^n \to [n]^k$, where $k = |I|$. These k indices, $H \in [n]^k$, represent the elements of $\mathbf{z}$ that $\mathscr{D}_{Adv}$ believes originated from the data owners in I. $\mathscr{D}_{Adv}$ wins if $> k/2$ of the chosen indices indeed originated from I, i.e, $|\sigma(H) \cap I| > k/2$, where $z_i = y_{\sigma(i)}$ and $\sigma(H) = \{\sigma(i) : i \in H\}$. $\mathscr{D}_{Adv}$ loses if most of H did not originate from I, i.e., $|\sigma(H) \cap I| \leq k/2$. We choose the above adversary because this re-identification is a key step in carrying out inference attacks – in failing to reliably re-identify the noisy values originating from I, one cannot make inferences on x_i specifically from the subset $I \subset G_i$.

Theorem 5. *For $\mathscr{A}(\mathscr{M}(\mathbf{x})) = \mathbf{z}$ where $\mathscr{M}(\cdot)$ is ε-LDP and $\mathscr{A}(\cdot)$ is α - $d_\sigma$private, we have*

$$\Pr[\mathscr{D}_{Adv} \textit{ loses}] \geq \left\lfloor \frac{r-k}{k} \right\rfloor e^{-(2k\varepsilon+\alpha)} \cdot \Pr[\mathscr{D}_{Adv} \textit{ wins}]$$

for any input subgroup $I \subset G_i, r = |G_i|$ and $k < r/2$.

The adversary's ability to re-identify the $\{z_I\}$ values comes partially from the *bag of values* (quantified by ε) and partially from the *order* (quantified by α). We highlight two implications of this fact.

- When ε is small ($\ll 1$), an adversary's ability to re-identify the noisy values $\{z_I\}$ originating from I may very well be dominated by α. For instance, if $\varepsilon = 0.2$ and $k = 5$, the adversary's advantage is dominated by α for any $\alpha > 2$. When using LDP alone (no shuffling), $\alpha = \infty$ and the adversary can exactly recover which values came from Alice's household. As such, even a

moderate α value (obtained via d_σ-privacy) significantly reduces the ability to re-identify the values.

- When the loss is dominated by ε ($2k\varepsilon \gg \alpha$), the above expression allows us to disentangle the *source of privacy loss*. In this regime, adversaries get most of their advantage from the bag of values released, not from the order of the release. That is, even if $\alpha = 0$ (uniform random shuffling), participants still suffer a large risk of re-identification simply due to the noisy values being reported. Thus, no shuffling mechanism can prevent re-identification in this regime.

Discussion. In spirit, DP does not guarantee protection against recovering DO_i's private x_i value. It guarantees that – had a user not participated (or equivalently submitted a false value x_i') – the adversary would have about the same ability to learn their true value, potentially from the responses of other data owners. In other words, the choice to participate is unlikely to be responsible for the disclosure of x_i. Similarly, d_σ-privacy does not prevent disclosure of x_i. By requiring indistinguishability of neighboring permutations, it guarantees that – had the data owners of any group G_i completely swapped identities – the adversary would have about the same ability to learn x_i. So most likely, Alice's household is not uniquely responsible for a disclosure of her x_i: had her household swapped identities with any of her neighbors, the adversary would probably draw the same conclusion on x_i. Or, as detailed in Thm.5, an adversary cannot reliably resolve which $\{z\}$ values originated from Alice's household, so they cannot draw conclusions based on her household's responses. In a nutshell,

- Inference attacks can recover a data owner DO_i's private data x_i from the responses of other data owners. The order of the data acts as the proxy for the data owner's identity which can aid an adversary in corralling the subset of other data owners who correlate with DO_i (required to make a reliable inference of x_i).
- DP alleviates concerns that DO_i's choice to share data (y_i) will result in disclosure of x_i, and d_σ-privacy alleviates concerns that DO_i's group's (G_i) choice to share their identity will result in disclosure of x_i.

4.4.4 d_σ-private Shuffling Mechanism

We now describe our novel shuffling mechanism that can achieve d_σ-privacy. In a nutshell, our mechanism samples a permutation from a suitable Mallows model and shuffles the data sequence accordingly. We can characterize the d_σ-privacy guarantee of our mechanism in the same way as that of the DP guarantee of classic mechanisms [62] – with variance and sensitivity. Intuitively, a larger dispersion parameter $\theta \in \mathbb{R}$ (Def. 4.3.2) reduces randomness over permutations, increasing utility and increasing (worsening) the privacy parameter α. The maximum value of θ for a given α guarantee depends on the sensitivity of the rank distance measure $\mathfrak{d}(\cdot)$ over all neighboring permutations $N_{\mathscr{G}}$. Formally, we define the sensitivity as

$$\Delta(\sigma_0 : \mathfrak{d}, \mathscr{G}) = \max_{(\sigma,\sigma') \in N_{\mathscr{G}}} |\mathfrak{d}(\sigma_0\sigma, \sigma_0) - \mathfrak{d}(\sigma_0\sigma', \sigma_0)| \,,$$

the maximum change in distance $\mathfrak{d}(\cdot)$ from the reference permutation σ_0 for any pair of neighboring permutations $(\sigma, \sigma') \in N_{\mathscr{G}}$ permuted by σ_0. The privacy parameter of the mechanism is then proportional to its sensitivity $\alpha = \theta \cdot \Delta(\sigma_0 : \mathfrak{d}, \mathscr{G})$.

Given $\mathscr{G}$ and a reference permutation σ_0, the sensitivity of a rank distance measure $\mathfrak{d}(\cdot)$ depends on the *width*, $\omega^\sigma_{\mathscr{G}}$, which measures how 'spread apart' the members of any group of $\mathscr{G}$ are in σ_0:

$$\omega^\sigma_{G_i} = \max_{(j,k) \in G_i \times G_i} \left| \sigma^{-1}(j) - \sigma^{-1}(k) \right|, i \in [n]; \quad \omega^\sigma_{\mathscr{G}} = \max_{G_i \in \mathscr{G}} \omega^\sigma_{G_i}$$

For example, for $\sigma = (1\,3\,7\,8\,6\,4\,5\,2\,9\,10)$ and $G_1 = \{1,7,8,2,5,6\}$, $\omega^\sigma_{G_1} = |\sigma^{-1}(1) - \sigma^{-1}(2)| = 7$. The sensitivity is an increasing function of the width. For instance, for Kendall's τ distance $\mathfrak{d}_\tau(\cdot)$ we have $\Delta(\sigma_0 : \mathfrak{d}_\tau, \mathscr{G}) = \omega^{\sigma_0}_{\mathscr{G}}(\omega^{\sigma_0}_{\mathscr{G}} + 1)/2$.

If a reference permutation clusters the members of each group closely together (low width), then the groups are more likely to permute within themselves. This has two benefits. First, for the same θ (θ is an indicator of utility as it determines the dispersion of the sampled permutation), a lower value of width gives lower α (better privacy). Second, if a group is likely to shuffle within

itself, it will have better (η,δ)-preservation – a novel utility metric, we propose, for a shuffling mechanism. Intuitively, a mechanism is (η,δ)-preserving w.r.t a subset of indices $S \subset [n]$ if at least $\eta\%$ of its indices are shuffled within itself with probability $(1-\delta)$. The rationale behind this metric is that it captures the utility of the learning allowed by d_σ-privacy – if S is equal to some group $G \in \mathscr{G}$, high (η,δ)-preservation allows overall statistics of G to be captured since $\eta\%$ of the correct data values remain preserved. We present the formal discussion in App. D.1.7.

Unfortunately, minimizing $\omega^\sigma_{\mathscr{G}}$ is an NP-hard problem (Thm. 16 in App. D.1.9). Instead, we estimate the optimal σ_0 using the following heuristic[3] approach based on a graph breadth first search.

Algorithm Description. Alg. 1 above proceeds as follows. We first compute the group assignment, $\mathscr{G}$, based on the public auxiliary information and desired threshold r following discussion in Sec. 4.4.2 (Step 1). Then we construct σ_0 with a breadth first search (BFS) graph traversal.

We translate $\mathscr{G}$ into an undirected graph (V,E), where the vertices are indices $V=[n]$ and two indices i,j are connected by an edge if they are both in some group (Step 2). Next, σ_0 is computed via a breadth first search traversal (Step 4) – if the k-th node in the traversal is i, then $\sigma_0(k)=i$. The rationale is that neighbors of i (members of G_i) would be traversed in close suc-

Algorithm 1: d_σ-private Shuffling Mech.

Input: LDP sequence $\mathbf{y}=\langle y_1,\cdots,y_n\rangle$; Public aux. info. $\mathbf{t}=\langle t_1,\cdots t_n\rangle$; Dist. threshold r; Priv. param. α;

Output: $\mathbf{z}$ - Shuffled output sequence;

2 $\mathscr{G} = ComputeGroupAssignment(\mathbf{t},r)$;

4 Construct graph $\mathbb{G}$ with

5 a) vertices $V=\{1,2,\cdots,n\}$

6 b) edges $E=\{(i,j): j\in G_i, G_i\in\mathscr{G}\}$

8 $root = \arg\max_{i\in[n]} |G_i|$;

10 $\sigma_0 = \mathsf{BFS}(\mathbb{G}, root)$;

12 $\Delta = ComputeSensitivity(\sigma_0,\mathscr{G})$

14 $\theta = \alpha/\Delta$;

16 $\dot{\sigma} \sim \mathbb{P}_{\theta,\delta}(\sigma_0)$;

18 $\sigma^* = \sigma_0^{-1}\dot{\sigma}$;

20 $\mathbf{z} = \langle y_{\sigma^*(1)},\cdots y_{\sigma^*(n)}\rangle$;

22 Return $\mathbf{z}$;

[3]The heuristics only affect σ_0 (and utility). Once σ_0 is fixed, Δ is computed exactly as discussed above.

cession. Hence, a neighboring node j is likely to be traversed at some step h near k which means $|\sigma_0^{-1}(i) - \sigma_0^{-1}(j)| = |h-k|$ would be small (resulting in low width). Additionally, starting from the node with the highest degree (Steps 3-4) which corresponds to the largest group in $\mathscr{G}$ (lower bound for $\omega_{\mathscr{G}}^{\sigma}$ for any σ) helps to curtail the maximum width in σ_0. See App. D.1.16 for evaluations of this heuristic's approximation.

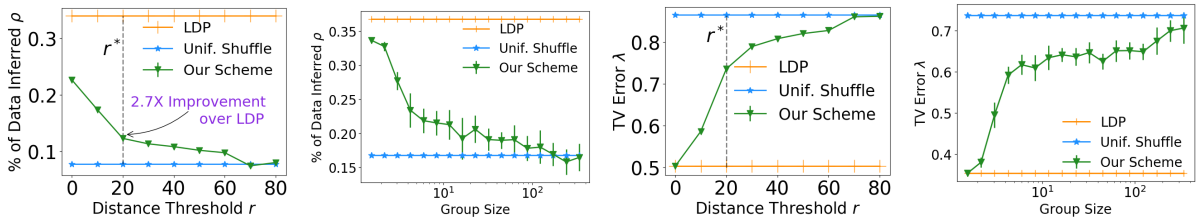
This is followed by the computation of the dispersion parameter, θ, for our Mallows model (Steps 5-6). Next, we sample a permutation from the Mallows model (Step 7) $\dot{\boldsymbol{\sigma}} \sim \mathbb{P}_\theta(\sigma : \sigma_0)$ and we apply the inverse reference permutation to it, $\sigma^* = \sigma_0^{-1}\dot{\boldsymbol{\sigma}}$ to obtain the desired permutation for shuffling. Recall that $\dot{\boldsymbol{\sigma}}$ is (most likely) close to σ_0, which is unrelated to the original order of the data. σ_0^{-1} therefore brings σ^* back to a shuffled version of the original sequence (identity permutation σ_I). Note that since Alg. 1 is publicly known, the adversary/analyst knows σ_0. Hence, even in the absence of this step from our algorithm, the adversary/analyst could perform this anyway. Finally, we permute $\mathbf{y}$ according to σ^* and output the result $\mathbf{z} = \dot{\boldsymbol{\sigma}}(\mathbf{y})$ (Steps 9-10).

Theorem 6. *Alg. 1 is $(\alpha, \mathscr{G})$-d_σ private where $\alpha = \theta \cdot \Delta(\sigma_0 : \delta, \mathscr{G})$.*

The proof is in App. D.1.11. Note that Alg. 1 provides the same level of privacy (α) for any two group assignment $\mathscr{G}, \mathscr{G}'$ as long as they have the same sensitivity, i.e, $\Delta(\sigma_0 : \delta_\tau, \mathscr{G}) = \Delta(\sigma_0 : \delta_\tau, \mathscr{G}')$. This leads to the following theorem which generalizes the privacy guarantee for any group assignment.

Theorem 7. *Alg. 1 satisfies $(\alpha', \mathscr{G}')$-$d_\sigma$privacy for any group assignment $\mathscr{G}'$ with $\alpha' = \alpha \frac{\Delta(\sigma_0:\delta,\mathscr{G}')}{\Delta(\sigma_0:\delta,\mathscr{G})}$ (proof in App. D.1.12.)*

Note. Producing σ^* is completely data ($\mathbf{y}$) independent. It only requires access to the public auxiliary information $\mathbf{t}$. Hence, Steps $1-6$ can be performed in a pre-processing phase and do not contribute to the actual running time. See App. D.1.10 for an illustration of Alg. 1 and runtime analysis.



(a) *PUDF*: Attack **(b)** *Twitch*: Attack **(c)** *PUDF*: Learnability **(d)** *Twitch*: Learnability

Figure 4.4. Our scheme interpolates between standard LDP (orange line) and uniform shuffling (blue line) in both privacy and data learnability. All plots increase group size along x-axis (except (d)). (a) $\rightarrow$ (b): The fraction of participants vulnerable to an inferential attack. (c) $\rightarrow$ (d): The accuracy of a calibration model trained on $\mathbf{z}$ predicting the distribution of LDP outputs at any point $t \in \mathscr{T}$, such as the distribution of medical insurance types used specifically in the Houston area (not possible when uniformly shuffling across Texas).

4.5 Evaluation

The previous sections describe how our shuffling framework interpolates between standard LDP and uniform random shuffling. We now experimentally evaluate this asking the following two questions –

Q1. Does the Alg. 1 mechanism protect against realistic inference attacks?

Q2. How well can Alg. 1 tune a model's ability to learn trends within the shuffled data, i.e., tune *data learnability*?

We evaluate on four datasets. We are not aware of any prior work that provides comparable local inferential privacy. Hence, we baseline our mechanism with the two extremes: standard LDP and uniform random shuffling. For concreteness, we detail our procedure with the *PUDF* dataset [3] (license), which comprises $n \approx 29$k psychiatric patient records from Texas. Each data owner's sensitive value x_i is their medical payment method, which is reflective of socioeconomic class (such as medicaid or charity). Public auxiliary information $t \in \mathscr{T}$ is the hospital's geolocation. Such information is used for understanding how payment methods (and payment amounts) vary from town to town for insurances in practice [64]. Uniform shuffling across Texas precludes such analyses. Standard LDP risks inference attacks, since patients attending hospitals in the same neighborhood have similar socioeconomic standing and use similar payment methods, allowing an adversary to correlate their noisy y_i's. To trade these off,

we apply Alg. 1 with $d(\cdot)$ being distance (km) between hospitals, $\alpha = 4$ and Kendall's τ rank distance measure for permutations.

Our inference attack predicts DO_i's x_i by taking a majority vote of the z_j values of the 25 data owners within r^* of t_i and who are most similar to DO_i w.r.t some additional privileged auxiliary information $t_j^p \in \mathscr{T}_p$. For PUDF, this includes the 25 data owners who attended hospitals that are within r^* km of DO_i's hospital, and are most similar in payment amount t_j^p. Using an $\varepsilon = 2.5$ randomized response mechanism, we resample the LDP sequence $\mathbf{y}$ 50 times, and apply Alg. 1's chosen permutation to each, producing 50 $\mathbf{z}$'s. We then mount the majority vote attack on each x_i for each $\mathbf{z}$. If the attack on a given x_i is successful across $\geq 90\%$ of these LDP trials, we mark that data owner as vulnerable – although they randomize with LDP, there is a $\geq 90\%$ chance that a simple inference attack can recover their true value. We record the fraction of vulnerable data owners as ρ. We report 1-standard deviation error bars over 10 trials.

Additionally, we evaluate *data learnability* – how well the underlying statistics of the dataset are preserved across $\mathscr{T}$. For *PUDF*, this means training a model on the shuffled $\mathbf{z}$ to predict the distribution of payment methods used near, for instance, $t_i =$ Houston for DO_i. For this, we train a calibrated model, $\texttt{Cal} : \mathscr{T} \to \mathscr{D}_x$, on the shuffled outputs where $\mathscr{D}_x$ is the set of all distributions on the domain of sensitive attributes $\mathscr{X}$. We implement `Cal` as a gradient boosted decision tree (GBDT) model [77] calibrated with Platt scaling [131]. For each location t_i, we treat the empirical distribution of x_i values within r^* as the ground truth distribution at t_i, denoted by $\mathscr{E}(t_i) \in \mathscr{D}_x$. Then, for each t_i, we measure the Total Variation error between the predicted and ground truth distributions $\mathrm{TV}\big(\mathscr{E}(t_i), \texttt{Cal}_r(t_i)\big)$. We then report $\lambda(r)$ – the average TV error for distributions predicted at each $t_i \in \mathbf{t}$ normalized by the TV error of naively guessing the uniform distribution at each t_i. With standard LDP, this task can be performed relatively well at the risk of inference attacks. With uniformly shuffled data, it is impossible to make geographically localized predictions unless the distribution of payment methods is identical in every Texas locale.

We additionally perform the above experiments on the following three datasets

- *Twitch* [137]. This dataset, gathered from the *Twitch* social media platform, includes a graph of $\approx 9K$ edges (mutual friendships) along with node features. The user's history of explicit language is private $\mathscr{X} = \{0,1\}$. $\mathscr{T}$ is a user's mutual friendships, i.e. t_i is the i'th row of the graph's adjacency matrix. We do not have any $\mathscr{T}_P$ here and select the 25 neighbors randomly.
- *Syn*. This is a synthetic dataset of size $20K$ which can be classified at three granularities – 8-way, 4-way and 2-way (Fig. 4.1a shows a scaled down version of the dataset). The eight color labels are private $\mathscr{X} = [8]$; the 2D-positions are public $\mathscr{T} = \mathbb{R}^2$. For learnability, we measure the accuracy of 8-way, 4-way and 2-way GBDT models trained on $\mathbf{z}$ on an equal sized test set at each r.
- *Adult* [55]. This dataset is derived from the 1994 Census and has $\approx$ 33K records. Whether DO_i's annual income is $\geq$ 50k is considered private, $\mathscr{X} = \{\geq 50k, < 50k\}$. $\mathscr{T} = [17,90]$ is age and $\mathscr{T}_P$ is the individual's marriage status. Due to lack of space figures are in App. D.1.14.

Experimental Results.

Q1. Our formal guarantee on the inferential privacy loss (Thm. 4) is described w.r.t to a 'strong' adversary (with access to $\{y_{G_i}\}, \mathbf{y}_{\overline{G}_i}$). Here, we test how well does our proposed scheme (Alg. 1) protect against inference attacks on real-world datasets without any such assumptions. Additionally, to make our attack more realistic, the adversary has access to extra privileged auxiliary information $\mathscr{T}_P$ which is *not used* by Alg. 22. Fig. 4.4a$\rightarrow$ 4.4b show that our scheme significantly reduces the attack efficacy. For instance, ρ is reduced by $2.7X$ at the attack distance threshold r^* for *PUDF*.

Additionally, ρ for our scheme varies from that of LDP[4] (minimum privacy) to uniform shuffle (maximum privacy) with increasing r (equivalently group size as in Fig. 4.4b) thereby spanning the entire privacy spectrum. As expected, ρ decreases with de-

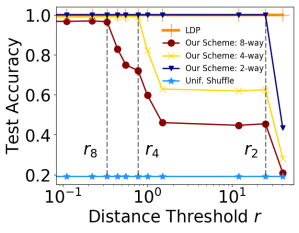


Figure 4.5. *Syn*: Learnability

[4]Our scheme gives lower ρ than LDP at $r = 0$ because the resulting groups are not singletons. For instance, for PUDF, G_i includes all individuals with the same zipcode as DO_i.

creasing privacy parameter α (Fig. D.3b).

Q2. Fig.4.4c $\rightarrow$ 4.4d show that λ varies from that of LDP (maximum learnability) to that of uniform shuffle (minimum learnability) with increasing r (equivalently, group size), thereby providing tunability. Interestingly, for *Adult* our scheme reduces ρ by $1.7X$ at the same λ as that of LDP for $r = 1$ (Fig. D.3c). Fig. 4.5 shows that the distance threshold r defines the granularity at which the data can be classified. LDP allows 8-way classification while uniform shuffling allows none. The granularity of classification can be tuned by our scheme – r_8, r_4 and r_2 mark the thresholds for 8-way, 4-way and 2-way classifications, respectively.

4.6 Conclusion

We have proposed a new privacy definition, d_σ-privacy that casts new light on the inferential privacy benefits of shuffling and a novel shuffling mechanism to achieve the same.

Chapter 5

Location Trace Privacy

5.1 Introduction

Location data is acutely sensitive information, detailing where we live, work, eat, shop, worship, and often when, too. Yet increasingly, location data is being uploaded for smartphone services such as ride hailing and weather forecasting and then being brokered in a thriving user location aftermarket to advertisers and even investors [156]. Users share location 'traces' when they release a sequence of locations, often across a short period of time. These traces are then used by central servers to monitor traffic trends, track individual fitness, target marketing, and even to study the effectiveness of social-distancing ordinances [75]. Here, we aim to provide a *local* privacy guarantee, wherein traces are sanitized at the user level before being transmitted to a centralized service. Note that this requires different guarantees and mechanisms than in *aggregate* applications making queries on large location trace databases.

Specifically, we guarantee a radius r of privacy at any sensitive time point or combination of time points within a given trace. This is challenging due to the fact that the locations within traces are highly inter-dependent. Informally, traces tend to follow relatively smooth trajectories in time. If not sanitized carefully, that knowledge alone may be exploited to infer actual locations from the released version of the trace. This work centers on designing meaningful privacy definitions and corresponding mechanisms that takes this dependence into account.

Broadly speaking, the vast majority of prior work on rigorous data privacy can be divided

into two classes that differ by the kind of guarantee offered: differential and inferential privacy. Differential privacy (DP) guarantees that the participation of a single person in a dataset does not change the probability of any outcome by much. In contrast, inferential privacy guarantees that an adversary who has a certain degree of prior knowledge cannot make certain sensitive inferences.

DP for releasing aggregate statistics of a spatio-temporal dataset has been well studied [69, 31, 171, 1]. There, the idea is to add enough noise to released statistics such that the effect of any user's participation is obscured, even if their locations are highly correlated to each other or to those of other users. Here, such a guarantee does not apply since we aim to release a sanitized version of a single user's trace.

In this local case we cannot rule out the possibility that the data curator knows who each individual is and who participated. Instead, we want to guarantee that event level information *about* each trace remains private. In this work, at any sensitive time t we mask whether the user visited location A or location B for any A,B less than r apart. Without *ad hoc* modifications, standard DP tools are insufficient for achieving this for the primary reasons that 1) the domain of location is virtually unbounded and 2) locations are highly dependent across a short period of time. To see this, consider the following instinctual approaches to achieving location trace privacy.

Approach A:

apply Local Differential Privacy (LDP) to each trace. Imagine a dataset of traces, each from a separate individual. Applying LDP implies that every trace has nearly the same probability of releasing the same sanitized version. This would be robust to arbitrary side information about dependence between locations in any one trace. Unfortunately, the amount of additive noise needed to achieve this would destroy nearly all utility: sanitized traces from California would have almost the same probability of showing up in Connecticut as do those from New York. Even if we constrained the domain to just Manhattan, this definition would not permit enough

utility to perform e.g. traffic monitoring.

Approach B:

apply LDP to each location within a trace. To preserve some utility, imagine a single trace as a dataset of n locations, each of which enjoys ε-LDP guarantees. This alone is not robust to arbitrary dependence between locations. By the logic of group LDP, it does satisfy $k\varepsilon$-LDP regardless of the dependence between any k locations. This approach has two setbacks. First, how to set k is unclear. Technically, all points in the trace are correlated, so to ward off worst-case correlations one might set it to the length of the trace, which is identical to Approach A. Second, even if location is bounded to a single city or county, satisfying this definition would still destroy nearly all utility. We cannot use sanitized traces for traffic monitoring if locations from either side of town have about same probability of being sanitized to the same value.

Approach C:

apply LDP guarantees to each location within a trace, but only within any region less than width r. This definition is known as Geo-Indistinguishability (GI) [8]. GI provides a substitute for restricting the domain of location allowing us to salvage some utility. Here, only locations within r of each other are required to have ε-LDP guarantees. In DP parlance, we might say that 'neighboring traces' have one location altered by $\leq r$ and are identical everywhere else. This gives us the guarantee we want for a trace with one location, but not with more than one location. To see why, compare with Approach B. Analogously, (ε, r)-GI along a trace provides $(k\varepsilon, r)$-GI to any subset of k locations. Like Approach B, setting k is unclear. Yet unlike Approach B, GI is not resistant to arbitrary dependence between any k locations. Any dependence where a change in one or more location(s) by r implies a change in some other location(s) by $\geq r$ breaks the GI guarantee. Even with the simplest models of dependence (e.g. if we know the true trace ought to move in a straight line) this is a problem.

To reiterate, applying LDP to traces or to locations within traces (Approaches A & B) does not provide a principled method for meaningful privacy with reasonable utility. GI adapts

LDP by giving guarantees only within a radius r. But in relaxing LDP, GI compromises the standard DP tools for handling obvious dependences between data-points like group DP. In our eyes, this warrants an *inferentially private* approach. Here, we continue to provide privacy within a radius r, thus allowing for utility. Yet instead of providing resistance to arbitrary dependence across any k locations, we aim to provide resistance to natural models of dependence between all locations. One may view such models as an adversary's prior beliefs about what traces are likely, like the straight-line prior mentioned earlier.

In contrast with differential privacy, providing inferential privacy guarantees is more complex, and has been less studied. It is however appropriate for applications such as ours, where information must be released based on a single person's data, the features of which are private and dependent. [**?**] provide a formal inferential privacy framework called Pufferfish, and design mechanisms for specific Pufferfish instances. As these instances do not apply to our setting, we adapt the Pufferfish framework to location privacy and more broadly to releasing any sequence of real-valued private information.

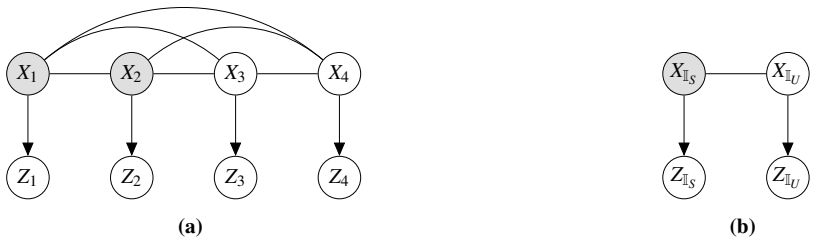


Figure 5.1. (a) An example graphical model of a four point trace X. (b) The more general grouped version of the model in (a), with the secret set $X_{\mathbb{I}_S} = \{X_1, X_2\}$ and the remaining set $X_{\mathbb{I}_U} = \{X_3, X_4\}$.

Contributions:

In this work, we propose an inferentially private approach to guaranteeing a radius r of privacy for sensitive points in location traces in three parts:

- First, we propose an adaptable privacy framework tailored to sequences of highly dependent datapoints that adapts Pufferfish privacy [105] to use Rényi Differential Privacy (RDP)

[128]. Given a model of dependence between points, this framework more appropriately estimates the risk of inference within radius r on points of interest than do vanilla LDP approaches.

- We then demonstrate how to implement our framework for the highly flexible and expressive setting of Gaussian process (GP) priors. These nonparametric models capture the spatiotemporal aspect of location data [115, 118, 41]. GPs have a natural synergy with Rényi privacy enabling an interpretable upper bound on privacy loss for additive Gaussian privacy mechanisms (that add Gaussian noise to each point). Using this, we design a semidefinite program (SDP) that optimizes the correlation of such mechanisms to minimize privacy loss without destroying utility, efficiently thwarting the inference of sensitive locations.

- Finally, we provide experiments on both location trace and home temperature data to demonstrate the advantage of these techniques over Approach C mechanisms like GI. We find that our mechanisms successfully obscure sensitive locations while respecting utility constraints, even when the prior model is misspecified.

Ultimately, by resisting only reasonable kinds of dependence in the data we are able to offer both meaningful privacy and utility. We show that our framework is robust to misspecification of this reasonable dependence and offers a privacy loss that is both tractable and interpretable.

5.2 Preliminaries and Problem Setting

A user transmits a sequence of N 2-dimensional locations along with their corresponding timestamps, collectively forming a 'trace'. We 'unroll' the trace into n real-valued random variables $X = \{X_1, X_2, \ldots, X_n\}$. A trace of 10 2d locations has $n = 2 \times 10 = 20$ random variables X_i. Instead of releasing the raw trace X, the user releases a private version $Z = \{Z_1, Z_2, \ldots, Z_n\}$, by way of an additive noise mechanism $Z = X + G$, where $G = \{G_1, G_2, \ldots, G_n\}$ is random noise produced by a privacy mechanism.

An adversary, receiving the obscured trace Z, then reasons about the true locations at some sensitive time(s). To reference the sensitive times, we use index set $\mathbb{I}_S$. If the sensitive indices are $\mathbb{I}_S = \{1,2\}$, the corresponding location values are $X_{\mathbb{I}_S} = \{X_1, X_2\}$ (e.g. referring to the two coordinates of one location). When inferring the true value of $X_{\mathbb{I}_S}$, the adversary makes use of the remaining points in the trace at indices $\mathbb{I}_U = [n] \backslash \mathbb{I}_S$, denoted $X_{\mathbb{I}_U}$, with obscured values $Z_{\mathbb{I}_U}$. This separation of points into $X_{\mathbb{I}_S}$ and $X_{\mathbb{I}_U}$ is represented in **Figure 5.1**.

We use location as a guiding example, but such inter-dependent traces X could take the form of home temperature time series data or spatial data like 3D facial maps used for identification. Going forward, we will continue to denote $X = \{X_1, X_2, \ldots, X_n\}$ with the understanding that *any* subsequence of d points e.g. $X_{\mathbb{I}_S} = \{X_2, X_6, \ldots\}$ could represent a d-dimensional sensitive value, or Nd points could represent N d-dimensional sensitive values.

For the real-valued distributions considered here, $P_\times(\bullet)$ refers to a density of distribution $\times$ on r.v. $\bullet$ and $P_\times(\bullet|*)$ is its regular conditional density given $*$.

5.2.1 Background

GI limits what can be inferred about the sensitive $X_{\mathbb{I}_S}$ from its corresponding $Z_{\mathbb{I}_S}$, but not from the remaining locations $Z_{\mathbb{I}_U}$. To do so we need a privacy definition that specifies what events of random variable $X_{\mathbb{I}_S}$ we wish to obscure, which realistic priors of inter-dependence to protect against, and a privacy loss.

5.2.2 Basic and Compound Secrets

We borrow heavily from the Pufferfish framework [**?**], and specialize it for the setting of location traces. We define our own set of *secrets* — the collection of events we wish to obscure — and *discriminative pairs*, the pairs of secret events we do not want an adversary to tell between.

Basic Secrets & Pairs

After releasing Z, we do not want an adversary with a reasonable prior on X, $\mathscr{P} \in \Theta$, to have sharp posterior beliefs about the user's location at some sensitive time (e.g. one of the

sensitive times in **Figure E.1** of Appendix E.1.1). As such, the adversary cannot distinguish whether the user visited location A or some nearby location B at that time. Let $x_s \in \mathbb{R}^2$ represent a possible assignments to $X_{\mathbb{I}_S}$, hypothesizing the true sensitive location. Any such assignment is secret, $\mathscr{S} = \{X_{\mathbb{I}_S} = x_s : x_s \in \mathbb{R}^2\}$. Specifically, we want the posterior probability of any two assignments to $X_{\mathbb{I}_S}$ within a radius r to be close: $\mathscr{S}_{\text{pairs}} = \{(x_s, x_s') : \|x_s - x_s'\|_2 \leq r\}$. This protects a single time within a trace of locations. More generally, in the context of spatiotemporal data of any dimension, we call this a *basic secret*.

Compound Secrets & Pairs

Suppose we have three sensitive times (again as in **Figure E.1**). A mechanism that blocks inference on each of these separately does not prevent inference on the combination of them simultaneously. To obscure hypotheses on *all three* of these, we modify our set of secrets to any combination of assignments to each secret location:

$$\mathscr{S} = \Big\{ \{X_{\mathbb{I}_{S1}} = x_{s1}\} \cap \{X_{\mathbb{I}_{S2}} = x_{s2}\} \cap \{X_{\mathbb{I}_{S3}} = x_{s3}\} \\ : x_{si} \in \mathbb{R}^2, i \in [3] \Big\} .$$

Now, the set of discriminative pairs is any two assignments to all three secret locations:

$$\mathscr{S}_{\text{pairs}} = \Big\{ \big(\{x_{s1}, x_{s2}, x_{s3}\}, \{x_{s1}', x_{s2}', x_{s3}'\} \big) \\ : \|x_{si} - x_{si}'\|_2 \leq r,\ i \in [3] \Big\}$$

This protects against compound hypotheses: if daycare and work are within r of each other, this keeps an adversary from inferring $X_{\mathbb{I}_{S1}}$ = 'daycare' *and* $X_{\mathbb{I}_{S2}}$ = 'work' versus $X_{\mathbb{I}_{S1}}$ = 'work' *and* $X_{\mathbb{I}_{S2}}$ = 'daycare'. More generally, in the context of spatiotemporal data of any dimension, we call this a *compound secret*. Intuitively, a mechanism that protects a compound secret of locations close together in time prevents a Bayesian adversary from leveraging the remainder of the trace to infer direction of motion at those sensitive times. Note that bounding the privacy loss

of a compound secret does not bound the privacy loss of its constituent basic secrets.

Going forward, we refer to $\mathbb{I}_S$ as the 'secret set'.

Gaussian Processes

For the purpose of location privacy, it is important to choose a prior class Θ such that the conditional distribution $P_{\mathscr{P}}(X_{\mathbb{I}_U}|X_{\mathbb{I}_S})$ is simple to compute for any secret set $\mathbb{I}_S$ and any prior $\mathscr{P} \in \Theta$. Of course, it is also critical that the prior class naturally models the data, and thus consists of 'reasonable assumptions' for adversaries. GPs satisfy both these requirements. We model a full d-dimensional trace sampled at N times by 'unrolling' it into a $n = dN$ dimensional GP.

Definition 5.2.1. *Gaussian process* A trace X is a Gaussian process if $X_{\mathbb{I}_M}$ has a multivariate normal distribution for any set of indices $\mathbb{I}_M \subset [n]$. If X is a gaussian process, then the function $i \to \mathbb{E}[X_i]$ is called the mean function and the function $(i,j) \to \text{Cov}(X_i, X_j)$ is called the kernel function.

In this work, the kernel uses locations' time stamps to compute their covariance $(t_i, t_j) \to \text{Cov}(X_i, X_j)$, but generally could use any side information provided with each location.

GPs have simple, closed form conditional distributions. Let $X \sim \mathscr{N}(\mu, \Sigma)$, where $\mu \in \mathbb{R}^n$ and $\Sigma \in \mathbb{R}^{n \times n}$. Then, the random variable $X_{\mathbb{I}_U}|\{X_{\mathbb{I}_S} = x_s\} \sim \mathscr{N}(\mu_{u|s}, \Sigma_{u|s})$, where $\mu_{u|s} = \mu_u + \Sigma_{us}\Sigma_{ss}^{-1}(x_s - \mu_s)$ and $\Sigma_{u|s} = \Sigma_{uu} - \Sigma_{us}\Sigma_{ss}^{-1}\Sigma_{su}$. Here, μ_s denotes the mean vector μ accessed at indices $\mathbb{I}_S$ and Σ_{su} denotes the covariance matrix Σ accessed at rows $\mathbb{I}_S$ and columns $\mathbb{I}_U$.

For GP priors, we will use additive noise $G \sim \mathscr{N}(\mathbf{0}, \Sigma^{(g)})$. Thus $Z = X + G$, too, is multivariate normal. Furthermore, the distribution of any set of variables conditioned on any other set of variables in **Figure 5.1** belongs to some multivariate normal distribution.

GPs have been shown to successfully model mobility [41, 115, 118], even in the domain of surveillance video [106]. Furthermore, although these non-parametric models are characterized by second order statistics, GPs are capable of complexity rivaling that of deep neural

networks [112], allowing for scalability to more complex models and domains. Our proposed results and algorithms may be applied regardless of the complexity of the chosen GP.

Rényi Differential Privacy

In the following section, we propose a privacy definition that adapts Rényi Differential Privacy (RDP) [128] to the Pufferfish framework. RDP resembles Differential Privacy [56], except instead of bounding the maximum probability ratio or *max divergence* of the distribution on outputs for two neighboring databases, it bounds the *Rényi divergence* of order λ, defined in Equation (5.1) for distributions $\mathscr{P}_1$ and $\mathscr{P}_2$. The Rényi divergence bears a nice synergy with Gaussian processes. If $\mathscr{P}_1 = \mathscr{N}(\mu_1, \Sigma)$ and $\mathscr{P}_2 = \mathscr{N}(\mu_2, \Sigma)$ — two mean-shifted normal distributions — the Rényi divergence takes on a simple closed form shown in Equation (5.2).

$$D_\lambda\left(\begin{matrix}\mathscr{P}_1\\ \mathscr{P}_2\end{matrix}\right) = \frac{1}{\lambda - 1}\log \mathbb{E}_{x\sim\mathscr{P}_2}\left(\frac{P_{\mathscr{P}_1}(X=x)}{P_{\mathscr{P}_2}(X=x)}\right)^\lambda \tag{5.1}$$

$$= \frac{\lambda}{2}(\mu_1 - \mu_2)^\intercal \Sigma^{-1}(\mu_1 - \mu_2) \tag{5.2}$$

We will make use of this in defining and bounding privacy loss in the next section.

5.3 Conditional Inferential Privacy

We now propose a privacy framework that is tailored to sequences of correlated data, Conditional Inferential Privacy (CIP). CIP guarantees a radius r of indistinguishability for the basic or compound secrets associated with any secret set $\mathbb{I}_S$. Specifically, CIP protects against any adversary with a specific prior on *the shape* of the trace, and is agnostic to their prior on the absolute location of the trace. We call the set of such prior distributions a Conditional Prior Class.

Definition 5.3.1. *Conditional Prior Class* For $X = \{X_1, \ldots, X_n\}$, prior distributions $\mathscr{P}_i, \mathscr{P}_j$ on X are said to belong to the same conditional prior class Θ if a constant shift in the conditioned

x_s results in a constant shift on the distribution of $X_{\mathbb{I}_U}$. Formally, if conditional distributions $P_{\mathscr{P}_i}(X_{\mathbb{I}_U}|X_{\mathbb{I}_S} = x_s) = P_{\mathscr{P}_j}(X_{\mathbb{I}_U} + c^u_{ij\mathbb{I}_S}|X_{\mathbb{I}_S} = x_s + c^s_{ij\mathbb{I}_S})$ for all x_s.

For instance, prior $P_{\mathscr{P}_i}$ may concentrate probability on traces passing through Los Angeles, while $P_{\mathscr{P}_j}$ concentrates on traces passing through London. Conditioning on each secret in the pair (x_s, x'_s) in L.A. is analogous to conditioning on each secret in the pair $(x_s + c^s_{ij\mathbb{I}_S}, x'_s + c^s_{ij\mathbb{I}_S})$ in London. The corresponding pair of conditional distributions on $X_{\mathbb{I}_U}$ in London ($P_{\mathscr{P}_j}$) are copies of those in L.A. ($P_{\mathscr{P}_i}$) shifted by $c^u_{ij\mathbb{I}_S}$. What matters is that the set of all pairs of conditional distributions under $P_{\mathscr{P}_i}$ induced by secret pairs (x_s, x'_s) is identical to those under $P_{\mathscr{P}_j}$ up to a mean shift. See Appendix E.1.5 for a more detailed discussion of conditional prior classes.

Definition 5.3.2. (ε, λ)-*Conditional Inferential Privacy* $(\mathscr{S}_{pairs}, r, \Theta)$ Given compound or basic discriminative pairs $\mathscr{S}_{\text{pairs}}$ associated with $\mathbb{I}_S$, a radius of privacy r, a conditional prior class, Θ, and a privacy parameter, $\varepsilon > 0$, a privacy mechanism $Z = \mathscr{A}(X)$ satisfies (ε, λ)-CIP$(\mathscr{S}_{\text{pairs}}, r, \Theta)$ if for all $(s_i, s_j) \in \mathscr{S}_{\text{pairs}}$, and all prior distributions $\mathscr{P} \in \Theta$, where $P_{\mathscr{P}}(s_i), P_{\mathscr{P}}(s_j) > 0$,

$$D_\lambda \begin{pmatrix} P_{\mathscr{A},\mathscr{P}}(Z|X_{\mathbb{I}_S} = s_i) \\ P_{\mathscr{A},\mathscr{P}}(Z|X_{\mathbb{I}_S} = s_j) \end{pmatrix} \leq \varepsilon \tag{5.3}$$

CIP departs from DP type notions of privacy like Approaches A→C primarily by resisting only a restricted class of inter-dependence — the conditional prior class — as opposed to arbitrary dependence of any k locations. Unlike approaches A and B, we are able to preserve utility for tasks like traffic monitoring. Unlike approach C, CIP is still resistant to realistic models of location inter-dependence.

While this definition borrows heavily from the Pufferfish framework, it has a few key modifications. Pufferfish is generally described from a central, not local model. We specialize the kinds of secrets and discriminative pairs for the case of local location trace privacy. Additionally, we specialize the type of prior distribution class needed for this local setting: the conditional prior class. Finally, we relax the strict max divergence (max log odds) criterion of the Pufferfish

definition to a Rényi divergence. This guarantees that — with high probability on draws of *realistic* traces $Z|X_{\mathbb{I}_S}$ — the log odds will be bounded by ε. As $\lambda \to \infty$, the log odds are bounded for all traces, i.e. the max divergence is bounded. We formalize this in Theorem 5.3.1.

The Rényi criterion of CIP greatly improves its flexibility. Unlike the standard DP Approaches A→C which only take probabilities over the mechanism, we do not have full control over the randomness at play: it is partially from $\mathscr{A}$ defined by us and from $\mathscr{P}$ intrinsic to the data. Unlike max divergence, Rényi divergence is available in closed form for many distributions, allowing for a more flexible privacy framework. The λ parameter helps us tune how strict a CIP definition is and how much noise we need to add. This allows us to design mechanisms that are resistant to natural models of dependence while preserving utility.

5.3.1 Properties

We now identify key properties that make the CIP guarantee interpretable and robust.

Interpretability:

CIP guarantees that a Bayesian adversary with any prior distribution on traces $\mathscr{P}$ in the conditional prior class Θ does not learn much about basic or compound secrets from the released trace Z. For basic secrets, this means that the adversary's posterior beliefs regarding sensitive location $X_{\mathbb{I}_S}$ are not much sharper than their prior beliefs before witnessing Z.

Theorem 5.3.1. *Prior-Posterior Gap:* An (ε, λ)-CIP mechanism with conditional prior class Θ guarantees that for any event O on sanitized trace Z

$$\left| \log \frac{P_{\mathscr{P},\mathscr{A}}(s_i|Z \in O)}{P_{\mathscr{P},\mathscr{A}}(s_j|Z \in O)} - \log \frac{P_{\mathscr{P}}(s_i)}{P_{\mathscr{P}}(s_j)} \right| \leq \varepsilon'$$

for any $\mathscr{P} \in \Theta$ with probability $\geq 1 - \delta$ over draws of $Z|X_{\mathbb{I}_S} = s_i$ or $Z|X_{\mathbb{I}_S} = s_j$, where ε' and δ

are related by

$$\varepsilon' = \varepsilon + \frac{\log 1/\delta}{\lambda - 1} .$$

This holds under the condition that $Z|X_{\mathbb{I}_S} = s_i$ and $Z|X_{\mathbb{I}_S} = s_j$ have identical support.

A CIP mechanism depends only on the conditional prior describing the data, not the data itself. Suppose an adversary's prior beliefs on $X_{\mathbb{I}_S}$ are uniform over some region. For $\lambda = 5$ and $\varepsilon = 0.1$, there is only a $\approx 1\%$ chance that their posterior odds on s_i, s_j will be more than 3.5, and a $\approx 10\%$ chance that they will be more than 2. This 'chance' is over draws of likely remaining locations $X_{\mathbb{I}_U}$ and the additive noise G. Proofs of all results are in Appendix E.1.2.

For additive noise mechanisms like $\mathscr{A}(X) = X + G = Z$, the CIP loss can be split into two terms: one accounting for the direct privacy loss of $Z_{\mathbb{I}_S}$ on $X_{\mathbb{I}_S}$ and a second accounting for the inferential privacy loss of $Z_{\mathbb{I}_U}$ on $X_{\mathbb{I}_S}$ via $X_{\mathbb{I}_U}$.

Lemma 8. Conditional Independence *For an additive noise mechanism, a fully dependent trace as in* ***Figure 5.1a****, and any prior $\mathscr{P}$ on X the CIP loss may be expressed as*

$$\begin{aligned} &D_\lambda \begin{pmatrix} P_{\mathscr{A},\mathscr{P}}(Z|X_{\mathbb{I}_S} = s_i) \\ P_{\mathscr{A},\mathscr{P}}(Z|X_{\mathbb{I}_S} = s_j) \end{pmatrix} \\ &= \sum_{i \in \mathbb{I}_S} \left[D_\lambda \begin{pmatrix} P_{\mathscr{A}}(Z_i|X_i = s_i) \\ P_{\mathscr{A}}(Z_i|X_i = s_j) \end{pmatrix} \right] + D_\lambda \begin{pmatrix} P_{\mathscr{A},\mathscr{P}}(Z_{\mathbb{I}_U}|X_{\mathbb{I}_S} = s_i) \\ P_{\mathscr{A},\mathscr{P}}(Z_{\mathbb{I}_U}|X_{\mathbb{I}_S} = s_j) \end{pmatrix} \end{aligned} \tag{5.4}$$

One interpretation of GI is that it assumes all locations X_i are independent. In this case, the second term vanishes and the privacy loss only depends on randomness of the mechanism, not the prior.

Robustness:

[104] show that it is impossible to achieve both utility and privacy resistant to all priors. CIP provides resistance to a reasonable class of priors $\mathscr{P} \in \Theta$, but it is possible that the true distribution $\mathscr{Q} \notin \Theta$. In this case, the privacy guarantees degrade gracefully as the divergence

between $\mathscr{Q}$ and $\mathscr{P} \in \Theta$ grows.

Theorem 5.3.2. *Robustness to Prior Misspecification* Mechanism $\mathscr{A}$ satisfies $\varepsilon(\lambda)$-CIP for prior class Θ. Suppose the finite mean true distribution $\mathscr{Q}$ is not in Θ. The CIP loss of $\mathscr{A}$ against prior $\mathscr{Q}$ is bounded by

$$D_\lambda \begin{pmatrix} P_{\mathscr{A},\mathscr{Q}}(Z|X_{\mathbb{I}_S} = s_i) \\ P_{\mathscr{A},\mathscr{Q}}(Z|X_{\mathbb{I}_S} = s_j) \end{pmatrix} \leq \varepsilon'(\lambda)$$

where

$$\varepsilon'(\lambda) = \frac{\lambda - \frac{1}{2}}{\lambda - 1}\,\Delta(2\lambda) + \Delta(4\lambda - 3) + \frac{2\lambda - \frac{3}{2}}{2\lambda - 2}\,\varepsilon(4\lambda - 2)$$

and where $\Delta(\lambda)$ is

$$\inf_{\mathscr{P}\in\Theta} \sup_{s_i \in \mathscr{S}} \max \left\{ D_\lambda \begin{pmatrix} P_{\mathscr{P}}(X_{\mathbb{I}_U}|X_{\mathbb{I}_S} = s_i) \\ P_{\mathscr{Q}}(X_{\mathbb{I}_U}|X_{\mathbb{I}_S} = s_i) \end{pmatrix}, D_\lambda \begin{pmatrix} P_{\mathscr{Q}}(X_{\mathbb{I}_U}|X_{\mathbb{I}_S} = s_i) \\ P_{\mathscr{P}}(X_{\mathbb{I}_U}|X_{\mathbb{I}_S} = s_i) \end{pmatrix} \right\}$$

As long as the conditional distribution on $X_{\mathbb{I}_U}|X_{\mathbb{I}_S} = s_i$ of prior $\mathscr{Q}$ is close to that of some $\mathscr{P} \in \Theta$, the privacy guarantees should change only marginally. This bound is tightest when $\varepsilon(\lambda)$ does not grow quickly with order λ.

5.3.2 CIP for Gaussian Process Priors

A *GP conditional prior class* is the set of all GP prior distributions with the same kernel function $(i, j) \to \mathrm{Cov}(X_i, X_j)$ and any mean function $i \to \mathbb{E}[X_i]$. With an additive Gaussian mechanism $G \sim \mathscr{N}(\mathbf{0}, \Sigma^{(g)})$, the CIP loss of Equation (5.4) can be bounded for any GP conditional prior class. See Appendix E.1.5 for further discussion of the GP conditional prior class.

Theorem 5.3.3. *CIP loss bound for GP conditional priors:* Let Θ be a GP conditional prior class. Let Σ be the covariance matrix for X produced by its kernel function. Let $\mathscr{S}$ be the basic or compound secret associated with $\mathbb{I}_S$, and S be the number of unique times in $\mathbb{I}_S$. The mechanism

$\mathscr{A}(X) = X + G = Z$, where $G \sim \mathscr{N}(\mathbf{0}, \Sigma^{(g)})$, then satisfies (ε, λ)-Conditional Inferential Privacy $(\mathscr{S}_{\text{pairs}}, r, \Theta)$, where

$$\varepsilon \leq \frac{\lambda}{2} S r^2 \Big(\frac{1}{\sigma_s^2} + \alpha^* \Big) \tag{5.5}$$

where σ_s^2 is the variance of each $G_i \in G_{\mathbb{I}_S}$ (diagonal entries of $\Sigma_{ss}^{(g)}$) and α^* is the maximum eigenvalue of $\Sigma_{\text{eff}} = \big(\Sigma_{us}\Sigma_{ss}^{-1}\big)^{\mathsf{T}} \big(\Sigma_{u|s} + \Sigma_{uu}^{(g)}\big)^{-1} \big(\Sigma_{us}\Sigma_{ss}^{-1}\big)$.

The above bound is tight for basic secrets ($S = 1$). The two terms of Equation (5.5) represent the direct $(\frac{1}{\sigma_s^2})$ and inferential (α^*) loss terms of Equation (5.4). We assume that each diagonal entry of $\Sigma_{ss}^{(g)}$ equals some σ_s^2, so that each $X_i \in X_{\mathbb{I}_S}$ experiences identical direct privacy loss, which is optimal under utility constraints.

The above bound composes gracefully when multiple traces of an individual are released.

Corollary 5.3.3.1. *Graceful Composition in Time* Suppose a user releases two traces X and $\dot{X}$ with additive noise $G \sim \mathscr{N}(\mathbf{0}, \Sigma^{(g)})$ and $\dot{G} \sim \mathscr{N}(\mathbf{0}, \dot{\Sigma}^{(g)})$, respectively. Then basic or compound secret $X_{\mathbb{I}_S}$ of X enjoys $(\bar{\varepsilon}, \lambda)$-CIP, where

$$\bar{\varepsilon} \leq \frac{\lambda}{2} S r^2 \Big(\frac{1}{\sigma_s^2} + \bar{\alpha}^* \Big)$$

and where $\bar{\alpha}^*$ is the maximum eigenvalue of $\bar{\Sigma}_{\text{eff}} = \big(\Sigma_{us}\Sigma_{ss}^{-1}\big)^{\mathsf{T}} \big(\Sigma_{u|s} + \bar{\Sigma}_{uu}^{(g)}\big)^{-1} \big(\Sigma_{us}\Sigma_{ss}^{-1}\big)$. Σ is the covariance matrix of the joint distribution on $X, \dot{X}$ and

$$\bar{\Sigma}^{(g)} = \begin{bmatrix} \Sigma^{(g)} & 0 \\ 0 & \dot{\Sigma}^{(g)} \end{bmatrix}.$$

This bound is identical to that of Theorem 5.3.3, only using the joint distribution over X, $\dot{X}$ and

$G, \dot{G}$. This provides some insight to the fact that, unlike DP, even parallel composition guarantees are not automatic. Composition depends on the conditional prior. In the GP setting, if the chosen kernel function decays over time, we can expect composition to have minimal effects on privacy for traces separated by long durations.

To reduce the upper bound of Theorem 5.3.3, we optimize the correlation (off-diagonal) of $\Sigma^{(g)}$ to minimize α^*, and optimize its variance (diagonal) to balance a noise budget between lowering inferential (α^*) and direct ($\frac{1}{\sigma_s^2}$) loss.

5.4 Optimized Privacy Mechanisms

Theorem 5.3.3 characterizes the privacy loss for GP conditional priors. We next show how to use this Theorem to design mechanisms that can strategically reduce CIP loss given a utility constraint. We measure 'utility loss' as the total mean squared error (MSE) between the released (Z) and true (X) traces: $\text{MSE}(\Sigma^{(g)}) = \sum_{i=1}^{n} \mathbb{E}[Z_i - X_i] = \mathbf{tr}(\Sigma^{(g)})$. We bound the utility loss by $\mathbf{tr}(\Sigma^{(g)}) \leq no_t$, where o_t is the average per-point utility loss.

It can be shown that optimizing the privacy loss under this utility constraint can be described by a semidefinite program (SDP) (formalization/derivation of SDPs in Appendix E.1.3). For a given trace X, define its covariance matrix Σ using the the kernel of the GP conditional prior $\Sigma_{ij} = k(i, j)$. Then pass Σ, the secret set $\mathbb{I}_S$, and the utility constraint o_t to our first program, SDP_A, which returns noise covariance $\Sigma^{(g)}$. This defines an additive noise mechanism $G \sim \mathcal{N}(0, \Sigma^{(g)})$ that minimizes CIP loss to $\mathbb{I}_S$.

$$\Sigma^{(g)} = \text{SDP}_\text{A}(\Sigma, \mathbb{I}_S, o_t)$$

We can thus use a SDP to minimize the CIP loss to any single compound or basic secret. However, a trace may contain multiple locations or combinations thereof that one wishes to protect. It remains to produce a single mechanism $\Sigma^{(g)}$ that bounds the CIP loss to multiple basic and/or compound secrets in a single trace.

For this we propose SDP_B, which uses the fact that if $\Sigma^{(g)'} \succ \Sigma^{(g)}$ it will have lower CIP loss (see Appendix E.1.3). SDP_B takes in a set of covariance matrices $\mathscr{F} = \{\Sigma_1^{(g)}, \ldots, \Sigma_m^{(g)}\}$, each designed to minimize CIP loss for a single compound or basic secret $\mathbb{I}_{S_i}$. It then returns a single covariance matrix $\Sigma^{(g)} \succeq \Sigma_i^{(g)}, i \in [m]$ that maintains the privacy guarantee each $\Sigma_i^{(g)}$ offered its corresponding $\mathbb{I}_{S_i}$, while minimizing utility loss.

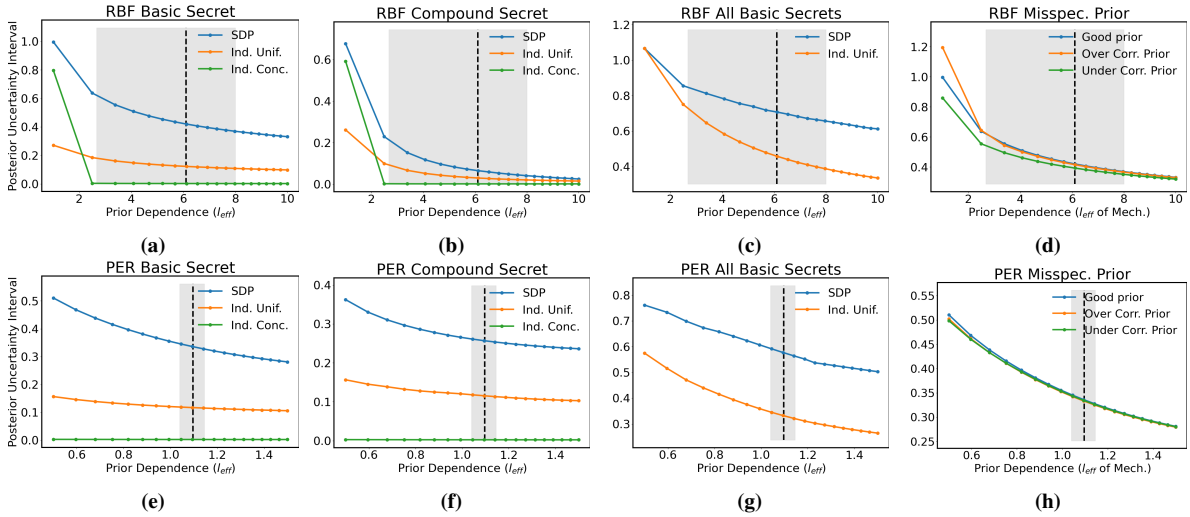


Figure 5.2. [1]Posterior uncertainty interval (higher=better privacy) on $X_{\mathbb{I}_S}$ of a GP Bayesian adversary. A larger l_{eff} corresponds to greater inter-dependence and reduces posterior uncertainty. The gray interval depicts the middle 50% of the MLE l_{eff} among traces in each dataset, and the black dotted line the median l_{eff}. **(a)**→**(c)**, **(e)**→**(g)** show SDP mechanisms (blue) maintaining relatively high uncertainty compared to two GI (Approach C) baselines of equal utility (MSE). **(d)**, **(h)** show the (minor) change in posterior uncertainty when the prior covariance Σ used in SDP_A is misspecified: when it is identical to the true covariance Σ^* known to the adversary (blue), is more correlated (orange), or is less correlated (green).

In our experiments, we use Algorithm 2 to design a single mechanism that protects all locations in the trace — all basic secrets — while minimizing utility loss.

5.5 Experiments

Here, we aim to empirically answer: **1)** Do our SDP mechanisms maintain high posterior uncertainty of sensitive locations? How do they compare to Approach C baselines of equal MSE? **2)** How robust is the SDP_A mechanism when the prior covariance Σ is misspecified?

Algorithm 2: Multiple Secrets

Input: $\mathbb{I}_{S1}, \ldots, \mathbb{I}_{Sm}, o_t, \Sigma$
Output: $\Sigma^{(g)}$

$\mathscr{F} = \emptyset$;
for $i \in [m]$ **do**
  $\Sigma_i^{(g)} = \mathrm{SDP_A}(\Sigma, \mathbb{I}_{Si}, o_t)$;
  $\mathscr{F} = \mathscr{F} \cup \Sigma_i^{(g)}$;
end for
$\Sigma^{(g)} = \mathrm{SDP_B}(\mathscr{F})$;
return $\Sigma^{(g)}$;

Methods

To answer these questions, we look at the range of conditional prior classes that fit real-world data. For location trace data, we use the GeoLife GPS Trajectories dataset [185] containing 10k human mobility traces after preprocessing (see Appendix E.1.4 for details). We also consider the privacy risk of room temperature data [130], using the SML2010 dataset [180], which contains approximately 40 days of room temperature data sampled every 15 minutes.

For the location data, having observed that the correlation between latitude and longitude is low (≈ 0.06) we treat each dimension as independent. By way of Corollary E.1.0.1, this allows us to bound privacy loss and design mechanisms for each dimension separately. Furthermore, having observed that each dimension fits nearly the same conditional prior, we treat our dataset of 10k 2-dimensional traces as a dataset of 20k 1-dimensional traces, where each trace represents one dimension of a 2d location trajectory.

We model the location trace data with a Radial Basis Function (RBF) kernel GP and the temperature series data with a periodic kernel GP:

$$k_{\mathrm{RBF}}(t_i, t_j) = \sigma_x^2 \exp\Big(-\frac{(t_i - t_j)^2}{2l^2}\Big)$$
$$k_{\mathrm{PER}}(t_i, t_j) = \sigma_x^2 \exp\Big(\frac{-2\sin^2(\pi|t_i - t_j|/p)}{l^2}\Big)$$

In both kernels, the intrinsic degree of dependence between points is captured by the lengthscale l. However, the fact that sampling rates vary significantly between traces means that traces with equal length scales can have very different degrees of correlation. To encapsulate both of these effects, we study the empirical distribution of *effective* length scale of each trace

$$l_{\text{eff},x} = \frac{l_x}{P} \quad l_{\text{eff},y} = \frac{l_y}{P}$$

where P is the trace's sampling period and l_x, l_y are the its optimal length scales for each dimension.

$l_{\text{eff},x}, l_{\text{eff},y}$ tell us the average number of neighboring locations that are highly correlated, instead of time period. For instance, a given trace with an optimal $l_{\text{eff},x} = 8$ tells us that every eight neighboring location samples in the x dimension have correlation > 0.8. The empirical distribution of effective length scales across all traces describes – over a range of logging devices (sampling rates), users, and movement patterns – how many neighboring points are highly correlated in location trace data. After this preprocessing, we are able to use the kernels that take indices (not time) as arguments:

$$k_{\text{RBF}}(i,j) = \exp\Big(-\frac{(i-j)^2}{2l_{\text{eff}}^2}\Big)$$
$$k_{\text{PER}}(i,j) = \exp\Big(\frac{-2\sin^2(\pi|i-j|/p)}{l_{\text{eff}}^2}\Big)$$

See Appendix E.1.4 for a more detailed discussion of how the empirical distribution of l_{eff} across traces is measured.

To impart the range of realistic conditional priors the gray interval of each plot depicts the middle 50% of the empirical l_{eff} among traces in each dataset. The dashed vertical line reports the median l_{eff}.

Each figure increases the degree of dependence, l_{eff}, used by the kernel to compute the prior covariance $\Sigma(l_{\text{eff}})$. $\Sigma(l_{\text{eff}})$ is then used in one of the SDP routines of Section 5.4 to produce

a mechanism $\Sigma^{(g)}(l_{\text{eff}})$ that protects a basic secret (SDP_A), a compound secret (SDP_A), or the union of all basic secrets (Multiple Secrets). We then observe the 68% confidence interval of the Gaussian posterior on sensitive points $X_{\mathbb{I}_S}$ (blue line). This is the 2σ uncertainty of a Bayesian adversary with a GP prior represented by $\Sigma(l_{\text{eff}})$ (see Appendix E.1.4 for how this is computed). As l_{eff} increases, their posterior uncertainty will reduce. Our aim is to mitigate this as much as possible with the given utility constraint. For scale, recall that prior variance $\mathbf{diag}(\Sigma)$ is normalized to one. In the case of all basic secrets, we report the average posterior uncertainty over locations.

We compare the SDP mechanisms with two mechanisms using the logic of Approach C (all three of equal MSE utility loss): *independent/uniform* and *independent/concentrated*. The uniform approach adds independent Gaussian noise evenly along the whole trace regardless of $\mathbb{I}_S$, $\Sigma^{(g)} = o_t I$. The concentrated approach allocates the entire noise budget to the sensitive set $\mathbb{I}_S$.

Results

For our first question, see **Figures 5.2a→5.2c, 5.2e→5.2g**. For both location and temperature data, our SDP mechanisms maintain higher posterior uncertainty than the baselines with identical utility cost for a single basic secret, a compound secret, and all basic secrets. By actively considering the conditional prior class parametrized by Σ, the SDP mechanisms can strategize to both correlate noise samples and concentrate noise power such that posterior inference is thwarted at the sensitive set $\mathbb{I}_S$. For an intuitive illustration of the chosen $\Sigma^{(g)}$'s, see Appendix E.1.1.

To answer our second question, see **Figures 5.2d** and **5.2h**. When the prior covariance Σ does not represent the true data distribution known to the adversary, a smaller posterior uncertainty may be achieved. The orange line indicates the uncertainty interval of an adversary who knows the data is *less* correlated than we believe i.e. the true $\Sigma^* = \Sigma(0.5l_{\text{eff}})$. The blue line represents an adversary who knows the data is *more* correlated than we believe i.e. the true $\Sigma^* = \Sigma(1.5l_{\text{eff}})$. Both plots confirm the robustness of our privacy guarantees stated by Theorem

5.3.2. Particularly around the median l_{eff} we see that the change in posterior uncertainty with this change in prior is indeed marginal.

5.6 Discussion

Related Work

Few works have proposed solutions to the *local* guarantee when releasing individual traces. A mechanism offered in [19] releases synthesized traces satisfying the notion of *plausible deniability* [20], but this is distinctly different from providing a radius of privacy to sensitive locations. Meanwhile, the frameworks proposed in [166] and [30] nicely characterize the risk of inference in location traces, but use only first-order Markov models of correlation between points, do not offer a radius of indistinguishability as in this work, and are not suited to continuous-valued spatiotemporal traces.

Perhaps more technically similar to this work, [145] provide a general mechanism that applies to any Pufferfish framework, as well as a more computationally efficient mechanism that applies when the joint distribution of an individual's features can be described by a graphical model. The first is too computationally intensive. The second is for discrete settings, and cannot accommodate spatiotemporal effects.

Conclusion

This work proposes a framework for both identifying and quantifying the *inferential* privacy risk for highly dependent sequences of spatiotemporal data. As a starting point, we have provided a simple bound on the privacy loss for Gaussian process priors, and an SDP-based privacy mechanism for minimizing this bound without destroying utility. We hope to extend this work to other data domains with different conditional priors, and different sets of secrets.

Acknowledgements

KC and CM would like to thank ONR under N00014-20-1-2334 and UC Lab Fees under LFR 18-548554 for research support. We would also like to thank our reviewers for their

insightful feedback.

Concluding Remarks

The above chapters provide a diverse set of examples of how privacy risks can be measured or mitigated in different scenarios. While highly different from each other, these examples all highlight the following three guiding principles for data privacy.

No privacy definition is a 'gold standard'

Differential privacy (DP) is often touted as the 'gold standard' of data privacy. The cases studied in Chapters 3-5 challenge this by proposing entirely different privacy definitions for entirely different settings and risks. Chapter 3 proposes sentence privacy, which is DP-like but uses a different neighboring notion. Chapter 4, on the other hand, proposes a shuffling-based privacy definition that is almost orthogonal to DP. That is because the correlation adversary considered in that setting cannot be thwarted by DP alone. However, the broad population trends that we wish to learn are still accessible under our semi-random shuffling approach. Similarly, Chapter 5 analyzes the threat of correlation adversaries in the domain of location traces. Here, we see that a DP-based definition has to add *more noise* in order to thwart these adversaries.

Taken together, we see that sometimes DP definitions are effective, and other times they require one to choose between meaningful privacy and utility. If one 'gold standard' definition were effective in all of these settings, we would not need to propose so many contrasting privacy definitions and methods.

No Free Lunch

The goal of data privacy is to allow the release of high-level information (*e.g.* data distribution) while obscuring low-level information (*e.g.* individuals' data features). It is natural to wonder whether it is possible to design a privacy definition under which we can release highly level information and defend against *any* adversary. The answer is unequivocally, *no*. This fact is known as the No-Free-Lunch theorem, made precise in [103]. The Theorem shows that releasing any information about a dataset that is useful to one person can be leveraged by an adversary to learn fine-grained information. The No-Free-Lunch theorem is instructive, because it shifts our attention from the question of whether we can provide air-tight privacy (impossible) to whether the adversaries our definition allows are *realistic* in our setting.

The No-Free-Lunch principle is fundamental to the approaches of Chapters 4 and 5 in particular. Here, we propose novel privacy definitions that are adversary-focused. Note that in both of these papers, we consider limited classes of adversaries. As stated above, it is impossible to block the inferences of *all adversaries* while still sharing useful information derived from the sensitive data. By practically evaluating what prior knowledge an adversary might have like a correlation prior, we can formalize a privacy definition that gives strong guarantees in realistic settings.

Perfect is the enemy of good

Chapters 1 and 2 offer no formal privacy definition or provably private mechanism. Instead, they offer statistical tests to empirically evaluate a model's memorization of its training data, and thereby risk of exposing that data. In both Chapters, we then examine how model selection can effect the degree of memorization as detected by our tests. While our proposed test statistics do not confer any formal privacy guarantees, they guide practitioners towards models that memorize less. In many cases, our tests showed that it is possible to find models which have significantly less memorization at little to no cost in utility.

While formal privacy definitions are a valuable goal they make up only a small part of an ML practitioners privacy toolkit. To preserve privacy, we as researchers ought to put equal effort into methodical empirical privacy tests as we do formally private algorithms. These tests tend to be far more accessible to practitioners and allow them to significantly improve model privacy. Although empirical privacy tests are imperfect, the practical benefits to be gained by proposing them are undoubtedly a positive good. Do not let perfect privacy be the enemy of good privacy.

Appendix A

A.0.1 Limitations and societal impact

Limitations.

Our work sets out to define, quantify, and visualize data memorization in SSL. Our tests guide us towards potential mitigation strategies. However, note that these strategies are distinct from provable privacy (e.g. DP), and do not guarantee that data is not memorized. It is possible that — even if our tests detect no memorization — data is being memorized in some other fashion, and could be detected with a different test. Furthermore, we focus on detecting image memorization with a curated, de-duplicated dataset (ImageNet-1k), which may over- or underestimate data memorization in practice. We chose this in order to claim the learning algorithm as the cause for memorization as opposed to the dataset itself. It is possible that models exhibit different memorization behavior on larger, less curated datasets. With orders of magnitude more data it is possible that memorization is reduced, but with more data duplication it also may be exacerbated.

Societal Impact.

Our work's findings have a critical societal impact from a privacy perspective. We show that it is possible for SSL—an increasingly popular learning paradigm—to memorize training images, which could have significant privacy implications. This direction of research is important if we want to understand how we can train such models without exposing user data. Additionally,

our proposed mitigation strategies point to the possibility of having strong privacy without significant loss in utility. Ultimately, we open a promising direction towards making SSL vision models more secure.

A.0.2 Experimental details

Details on dataset splits

Imagenet1k provides bounding box annotations of foreground objects to a subset of examples in each class. Private sets $\mathscr{A}$ and $\mathscr{B}$ contain shared examples, $\mathscr{A} \cap \mathscr{B}$, without bounding box annotations, and unique examples with bounding box annotations. Denote the unique examples in each set as $\overline{\mathscr{A}} = \mathscr{A} \backslash (\mathscr{A} \cap \mathscr{B})$ and $\overline{\mathscr{B}} = \mathscr{B} \backslash (\mathscr{A} \cap \mathscr{B})$. To identify memorization, our tests only attempt to infer the labels of the unique examples $\overline{\mathscr{A}}$ and $\overline{\mathscr{B}}$ that differentiate the two private sets. The periphery crop, $\text{crop}(A_i)$, is computed as the largest possible crop that does not intersect with the foreground object bounding box. In some instances the largest periphery crop is small, and not high enough resolution to get a meaningful embedding. To circumvent this, we only run the test on bounding box examples where the periphery crop is at least 100×100 pixels.

Each size of training set, 100k to 500k, includes an equal number of examples per class in both sets $\mathscr{A}$ and $\mathscr{B}$. The total bounding box annotated examples of each class are evenly divided between $\overline{\mathscr{A}}$ and $\overline{\mathscr{B}}$. The remaining examples in each class are the shared examples $\mathscr{A} \cap \mathscr{B}$. Shared examples are necessary due to a limit number of bounding box examples and a limited number of total images. However, we reiterate that the bounding box examples in set $\mathscr{A}$ are *unique* to set $\mathscr{A}$, and thus can only be memorized by SSL_A.

The disjoint public set, X, contains ground truth labels but no bounding-box annotations. The size and content of X remains fixed for all tests.

Details on the training setup

Model Training:

We use PyTorch [133] with FFCV-SSL [26]. All models are trained for 1000 epochs with model checkpoints taken at 50, 100, 250, 500, 750, and 1000 epochs. We note that 1000 epochs is used in the original papers of both VICReg and SimCLR. All sweeps of epochs use the 300k dataset. All sweeps of datasets use the final, 1000 epoch checkpoint. We use a batch size of 1024, and LARS optimizer [177] for all SSL models. All models use Resnet101 for the backbone. As seen in Appendix A.0.3, a Resnet50 backbone results in *déjà vu* consistent with that of Resnet101.

VICReg Training:

VICReg is trained with the 3-layer fully connected projector used in the original paper with layer dimensions 8192-8192-8192. The invariance, variance, and covariance parameters are set to $\lambda = 25, \mu = 25, \nu = 1$, respectively, which are used in the original paper [14]. The LARS base learning rate is set to 0.2, and weight decay is set to 1e-6.

SimCLR Training:

SimCLR is trained with the 2-layer fully connected projector used in the original paper with layer dimensions 2048-256. The temperature parameter is set to $\tau = 0.15$. The LARS base learning rate is set to 0.3, and weight decay is set to 1e-6.

Supervised Training:

Unlike the SSL models, the supervised model is trained with label access using cross-entropy loss. To keep architectures as similar as possible, the supervised model also uses a Resnet101 backbone and the same projector as VICReg. A final batchnorm, ReLU, and linear layer is added to bring the 8192 dimension projector output to 1000-way classification activations. We use these activations as the supervised model's projector embedding. The supervised model uses the LARS optimizer with learning rate 0.2.

Details on the evaluation setup

KNN:

For each test, we build two KNN's: one using the target model, SSL_A (or CLF_A), and one using the reference model SSL_B (or CLF_B). As depicted in Figure 1.2, each KNN is built using the projector embeddings of all images in the public set $\mathscr{X}$ as the neighbor set. When testing for memorization on an image $A_i \in \mathscr{A}$, we first embed $\text{crop}(A_i)$ using SSL_A, and find its $K = 100$ L_2 nearest neighbors within the SSL_A embeddings of $\mathscr{X}$. See section A.0.3 for a discussion on selection of K. We then take the majority vote of the neighbors' labels to determine the class of A_i. This entire pipleline is repeated using reference model SSL_B and its KNN to compute reference model accuracy.

In practice, all of our quantitative tests are repeated once with SSL_A as the target model (recovering labels of images in set $\mathscr{A}$) and again with SSL_B as the target model (recovering labels of images in set $\mathscr{B}$). All results shown are the average of these two tests. Throughout the paper, we describe SSL_A as the target model and SSL_B as the reference model for ease of exposition.

RCDM:

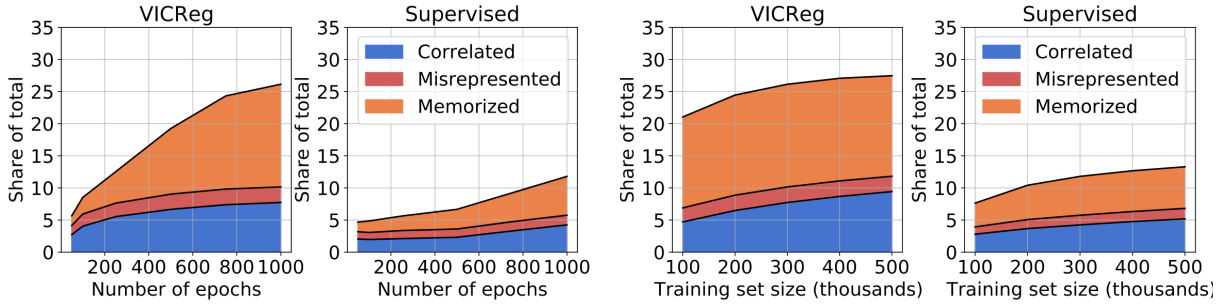
The RCDM is trained on a face-blurred version of ImageNet [50] and is used to decode the SSL backbone embedding of an image back into an approximation of the original image. All RCDMs are trained on the public set of images $\mathscr{X}$ used for the KNN. A separate RCDM must be trained for each SSL model, since each model has a unique mapping from image space to embedding space.

At inference time, the RCDM is used to reconstruct the foreground object given only the periphery cropping. To produce this reconstruciton, the RCDM needs an approximation of the backbone embedding of the original image. The backbone of image A_i is approximated by **1)** computing crop embedding $\text{SSL}_A^{\text{proj}}(\text{crop}(A_i))$, **2)** finding the five public set nearest neighbors of the crop embedding, and **3)** averaging the five nearest neighbors' backbone embeddings. In

practice, these public set nearest neighbors are often a very good approximation of the original image, capturing aspects like object class, position, subspecies, etc..

A.0.3 Additional quantitative experiments

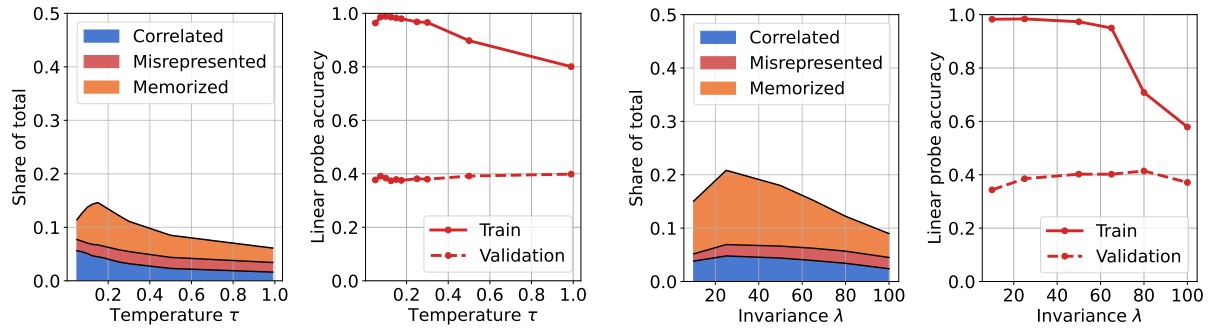
Sample-level memorization



(a) Categories of training samples vs. number of epochs

(b) Categories of training samples vs. training set size

Figure A.1. Partition of samples $A_i \in \mathscr{A}$ into the four categories: unassociated (not shown), memorized, misrepresented and correlated. The memorized samples—ones whose labels are predicted by KNN_A but not by KNN_B—occupy a significantly larger share for VICReg compared to the supervised model, indicating that sample-level *déjà vu* memorization is more prevalent in VICReg.



(a) SimCLR

(b) VICReg

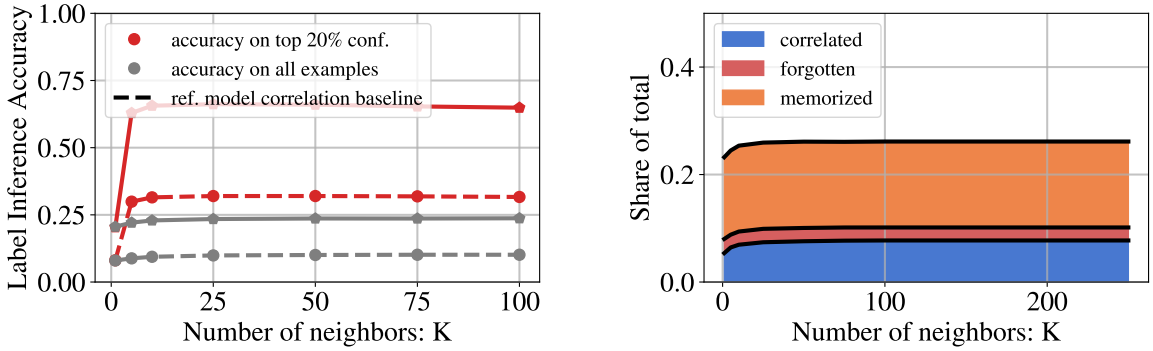
Figure A.2. Effect of SSL hyperparameter on *déjà vu* memorization. The left plot of Figures A.8a and A.8b show the size of the memorized set as a function of the temperature parameter for SimCLR and invariance parameter for VICReg, respectively. *Déjà vu* memorization is the highest within a narrow band of hyperparameters, and one can mitigate against *déjà vu* memorization by selecting hyperparameters outside of this band. Doing so has negligible effect on the quality of SSL embeddings as indicated by the linear probe accuracy on ImageNet validation set.

Many SSL algorithms contain hyperparameters that control how similar the embeddings of different views should be in the training objective. We show that these hyperparameters directly affect *déjà vu* memorization. Figure A.8 shows the size of the memorized set for SimCLR (left) and VICReg (right) as a function of their respective hyperparameters, τ and λ.

We observe that the memorized set is largest within a relatively narrow band of hyperparameter values, indicating strong *déjà vu* memorization. By selecting hyperparameters outside this band, *déjà vu* memorization sharply decreases while the linear probe validation accuracy on ImageNet remains roughly the same.

Selection of K for KNN

In this section, we describe the impact of K on the KNN label inference accuracy.



(a) Vicreg, Accuracy

(b) Vicreg, Share of memorized examples

Figure A.3. Impact of K on label inference accuracy for target and reference models. **Left:** the population-level label inference accuracy experiment of Section 1.4.1 on VICReg vs. K. **Right:** the individualized memorization test of Section 1.4.2 on VICReg vs. K. In both cases, we see that our tests are relatively robust to choice of K beyond $K = 50$.

Figure A.3 above shows how the tests of Section 1.4 change with number of public set nearest neighbors K used to make label inferences. Both tests are relatively robust to any choice of K. Results are shown on VICReg trained for 1k epochs on the 300k dataset. We see that any choice of K greater than 50 and less than the number of examples per class (300, in this case) appears to have good performance. Since our smallest dataset has 100 images per class, we chose to set $K = 100$ for all experiments.

Effect of SSL criteria

We repeat the quantitative memorization tests of Section 1.4 on different models: VICReg[14], Barlow-Twins[181], Dino[37], Byol[85], SimCLR[182] and a supervised model in Figure A.4. We observe differences between SSL training criteria with respect to Dejavu memorization. The easy ones to attack are VICReg and Barlow Twins whereas SimCLR and Byol are more robust to these attacks. While the degree of memorization appears to be reduced for SimCLR compared with VICReg, it is still stronger than the supervised baseline.

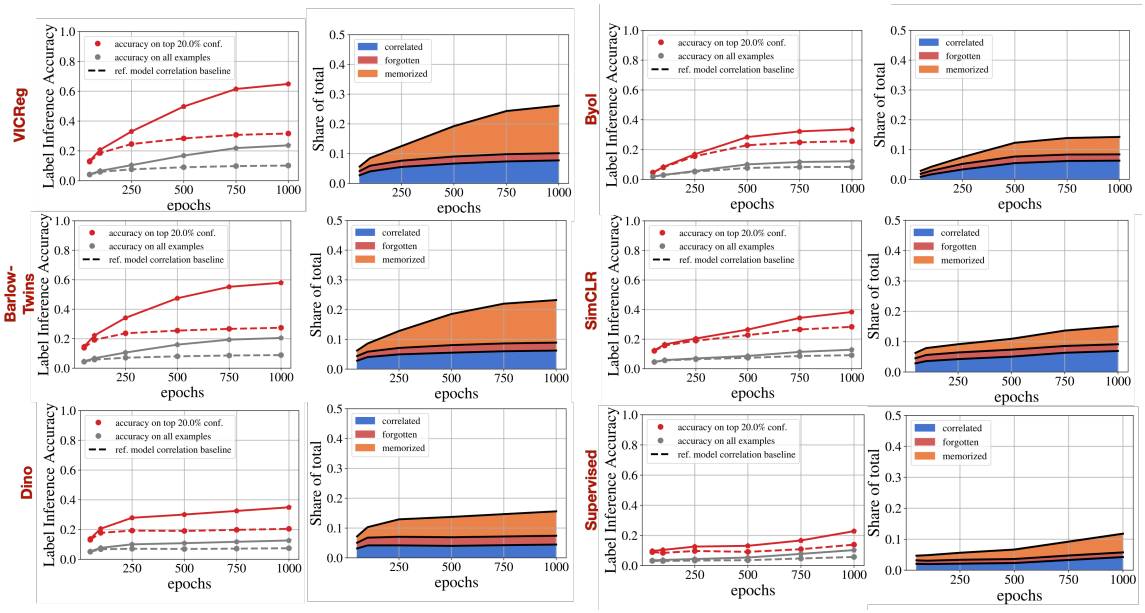


Figure A.4. Comparison of *déjà vu* memorization for VICReg, Barlow Twins, Dino, Byol, SimCLR, and a supervised model. All tests are described in Section 1.4. We are showing *déjà vu* vs. number of training epochs. We see that SimCLR (center row) shows less *déjà vu* than VICReg, yet marginally more than the supervised model. Even with this reduced degree of memorization, we are able to produce detailed reconstructions of training set images, as shown in Figures 1.6 and **??**.

Effect of Model Architecture and Complexity

Results shown in the main paper use Resnet101 for the model backbone. To understand the relationship between *déjà vu* and overparameterization, we compare with the smaller Resnet50 and Resnet18 in Figure A.5. Overall, we find that increasing the number of parameters of the model leads to higher degree of *déjà vu* memorization. The same trend holds when using Vision Transformers (VIT-Tiny, -Small, -Base, and -Large with patch size of 16) of various sizes as the SSL backbone, instead of a Resnet. This highlights that *déjà vu* memorization is not unique to convolution architectures.

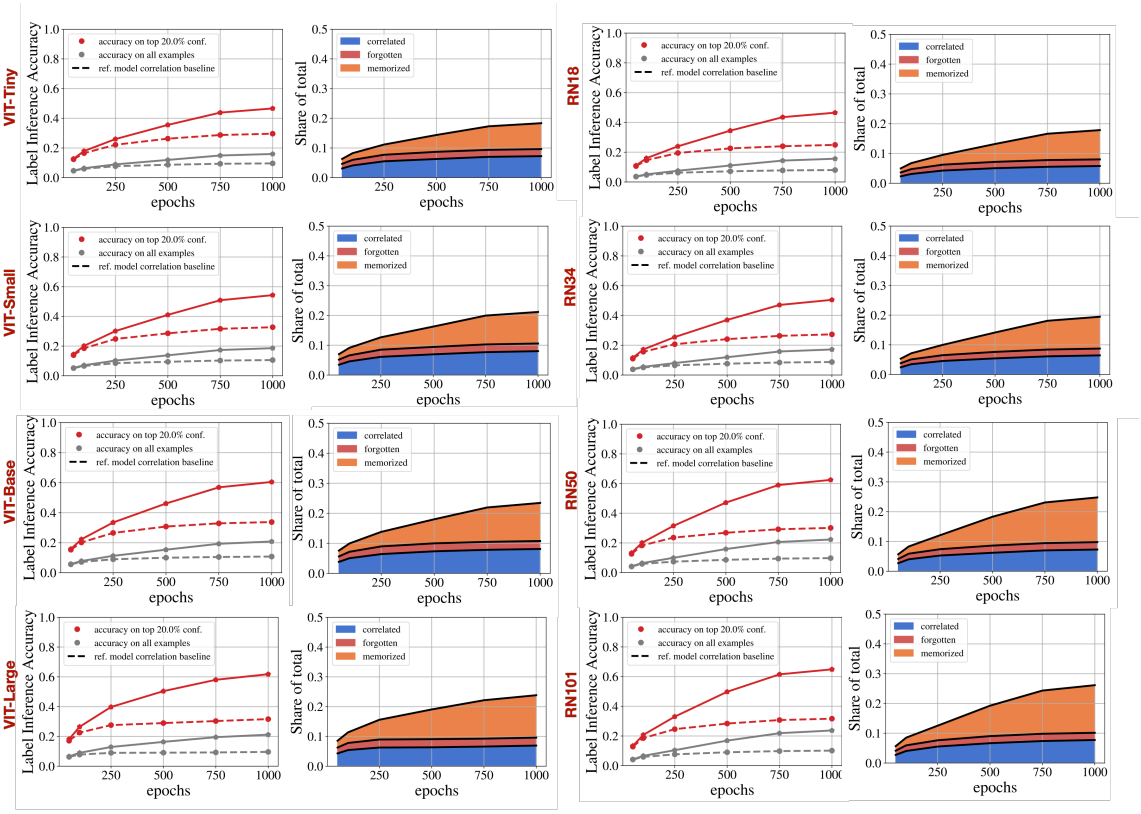
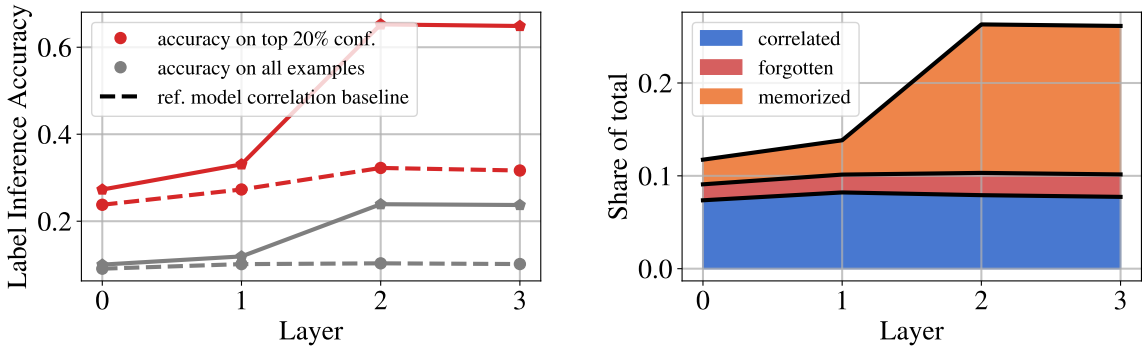


Figure A.5. Comparison of VICReg *déjà vu* memorization for different architectures and model sizes. On the left, we present deja vu memorization using VIT architectures (from vit-tiny in the first row to vit-base in the last row). On the right, we use Resnet based architectures (from resnet18 in the first row to resnet101 in the last row). All tests are described in Section 1.4, with the plots showing *déjà vu* vs. number of training epochs. Reducing model complexity from Resnet101 to Resnet18 or from Vit-Large to Vit-tiny has a significant impact on the degree of memorization.

The impact of Guillotine Regularization on Deja Vu

In our experiments, we show *déjà vu* using the projector representation. The SSL loss directly incentivizes the projector representation to be invariant to random crops of a particular image. As such, we expect the projector to be the *most* overfit and produce the strongest *déjà vu* . Here, we study whether earlier representations between the projector and backbone exhibit less *déjà vu* memorization. This phenomenon – 'guillotine regularization' – has recently been studied from the perspective of generalization in **(author?)** [24]. Here, we study it from the perspective of *memorization*.

To show how guillotine regularization impacts *déjà vu* , we repeat the tests of Section 1.4 on each layer of the VICReg projector: the 2048-dimension backbone (layer 0) up to the projector output (layer 3). We evaluate whether memorization is indeed reduced for the more *regularized* layers between the projector output and the backbone.



(a) population level accuracy

(b) share of memorized examples

Figure A.6. *déjà vu* memorization versus layer from backbone (0) to projector output (3). The memorization tests of Section 1.4 are evaluated at each level of the VICReg projector. We see that *déjà vu* is significantly stronger closer to the projector output and nearly zero near the backbone. Interestingly, most memorization appears to occur in the final two layers of VICReg.

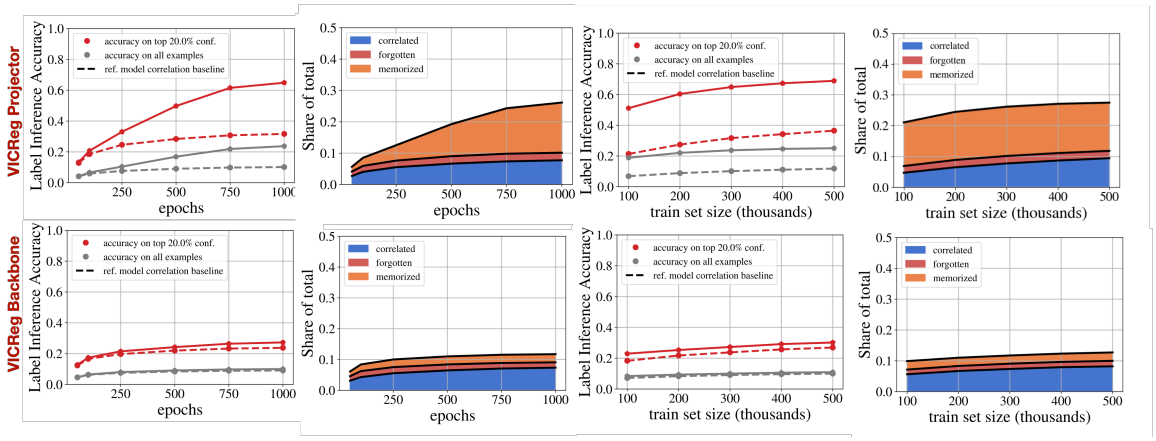
Figure A.6 shows how guillotine regularization significantly reduces the degree of memorization in VICReg. The vast majority of VICReg's *déjà vu* appears to occur in the final two layers of the projector (2,3): in earlier layers (0,1), the label inference accuracy of the target model and reference model are comparable. This suggests that – like the hyperparameter selection

of Section 1.7 – guillotine regularization can also significantly mitigate *déjà vu* memorization. In the following, we extend this result to SimCLR and supervised models by measuring the degree of *déjà vu* in the backbone (layer 0) versus training epochs and dataset size.

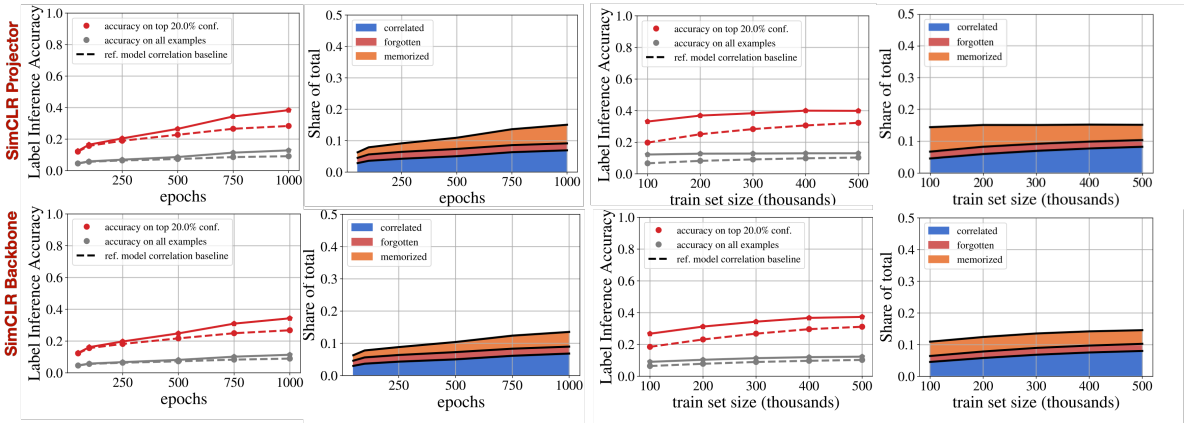
Comparison of *déjà vu* in projector and backbone vs. epochs and dataset size

Since the backbone is mostly used at inference time, we now evaluate how much *déjà vu* exists in the backbone representation for VICReg and SimCLR. We repeat the tests of Section 1.4 versus training epochs and train set size.

Figure A.7 shows that, indeed, *déjà vu* is significantly reduced in the backbone representation. For SimCLR, however, we see that backbone memorization is comparable with projector memorization. In light of the Guillotine regularization results above, this makes some sense since SimCLR uses fewer layers in its projector. Given that we were able to generate accurate reconstructions with the SimCLR projector (see Figures **??** and 1.6), we now evaluate whether we can produce accurate reconstructions of training examples using the SimCLR backbone alone.



(a) VICReg

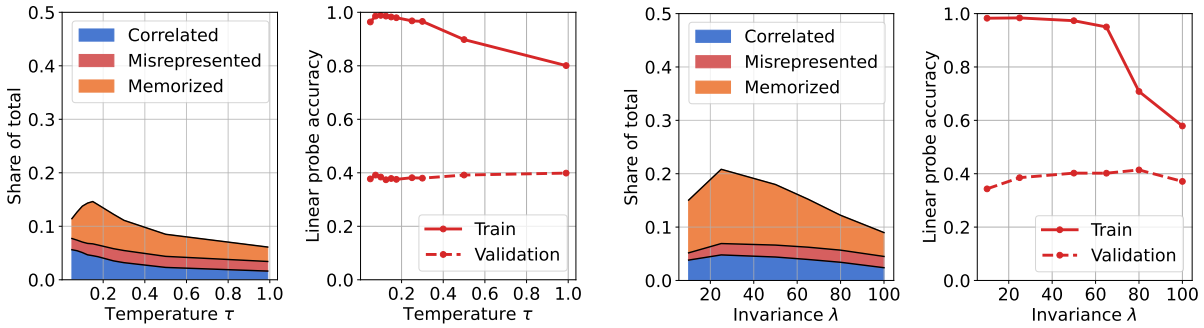


(b) SimCLR

Figure A.7. Accuracy of label inference on VICReg and SimCLR using projector and backbone representations. **First two columns:** Effect of training epochs on memorization for each representation. **Last two columns:** Effect of training set size on memorization for each representation. In contrast with VICReg, the *déjà vu* memorization detected in SimCLR's projector and backbone representations is quite similar. While SimCLR's projector memorization appears weaker than that of VICReg, its backbone memorization is markedly stronger. This kind be easily explained as a byproduct of Guillotine Regularization [24], i.e. removing layers close to the objective reduce the bias of the network with respect to the training task. Since SimCLR's projector has fewer layers than VICReg's, the impact of Guillotine Regularization is less salient.

A.0.4 Additional reconstruction examples

The two reconstruction experiments of Section 1.5 are each exemplified within one class. However, we see strong reconstructions using SSL_A in several classes, and similar experimental results. To demonstrate this, we repeat the experiments 1.5 using the *yellow garden spider* class and the and the *aircraft carrier* class.



(a) SimCLR **(b)** VICReg

Figure A.8. Effect of SSL hyperparameter on *déjà vu* memorization. The left plot of Figures A.8a and A.8b show the size of the memorized set as a function of the temperature parameter for SimCLR and invariance parameter for VICReg, respectively. *Déjà vu* memorization is the highest within a narrow band of hyperparameters, and one can mitigate against *déjà vu* memorization by selecting hyperparameters outside of this band. Doing so has negligible effect on the quality of SSL embeddings as indicated by the linear probe accuracy on ImageNet validation set.

Selection of Memorized and Correlated Images:

The images of Figure 1.6 and **??** were chosen methodically as follows.

Image selection:

The 20 images of Figures 1.6 and **??** are selected deterministically using label inference accuracy and KNN confidence score. The 10 most correlated images are those images in the correlated set (both models infer label correctly) of $\mathscr{A}$ with the highest confidence agreement between models SSL_A and SSL_B. To measure confidence agreement we take the minimum confidence of the two models. The 10 most memorized images are those images in the memorized set (only target model infers the label correctly) of $\mathscr{A}$ with the highest confidence difference between models SSL_A and SSL_B.

Class selection:

To find classes with a high degree of *déjà vu* , classes were sorted by the label inference accuracy gap between the target and reference model. We selected the class based on a handful of criteria. First, we prioritized classes without images of human faces, thereby removing classes like 'basketball', 'bobsled', 'train station', and even 'tench' which is a fish often depicted in the hands of a fisherman. Second, we prioritized classes that include at least ten images with a high confidence difference between the target and reference models ('most memorized' images described above) and at least ten images with high confidence agreement ('most correlated' images described above). This led us to the *dam* and *yellow garden spider* classes.

Selection of Beyond-Label-Inference Images:

The images of Figure 1.7 and **??** were chosen methodically as follows.

Image selection:

The four images of Figures 1.7 and **??** are selected using KNN confidence score, and, necessarily, hand picked selection for unlabeled features. Within a given class, we look at the top 40 images with highest target model KNN confidence scores. We then filter through these images to identify a distinguishable feature like different species within the same class or different object positions within the same class. This step is necessary because we are looking for features that are not labeled by ImageNet. We then choose two of these top 40 with one feature (e.g. American badger) and two with the alternative feature (e.g. European badger).

Class selection:

To find classes with a high degree of *déjà vu* , classes were sorted by the target model's top-40 KNN confidence values within each class. As in the memorization vs. correlation experiment, we prioritized classes without images of human faces.

Reconstructions using SimCLR Backbone

The label inference results in Appendix A.0.3 show that the SimCLR backbone exhibits a similar degree of *déjà vu* memorization as the projector does. To evaluate the risk of such memorization, we repeat the reconstruction experiment of Section 1.5 on the *dam* class using the SimCLR backbone instead of its projector.

Figure A.9 demonstrates that we are able to reconstruct training set images using the SimCLR backbone alone. This indicates that *déjà vu* memorization can be leveraged to make detailed inferences about training set images without *any* access to the projector. As such, withholding the projector for model release may not be a strong enough mitigation against *déjà vu* memorization.

(a) First memorized dam example

(b) Second memorized dam example

Figure A.9. Instances of *déjà vu* memorization by the SimCLR backbone representation. Here, the backbone embedding of the crop is used instead of the projector embedding on the same training images used in Figure 1.6. Interestingly, we see that *déjà vu* memorization is still present in the SimCLR backbone representation. Here, the nearest neighbor set recovers dam given an uninformative crop of still or running water. Even without projector access, we are able to reconstruct images in set $\mathscr{A}$ using SSL_A, and are unable using SSL_B.

A.0.5 Detecting *Déjà vu* without Bounding Box Annotations

The memorization tests presented critically depend on bounding box annotations in order to separate the foreground object from the periphery crop. Since such annotations are often not available, we propose a heuristic test that simply uses the lower left corner of an image as a surrogate for the periphery crop. Since foreground objects tend to be near the center of the image, the corner crop usually excludes the foreground object and does not require a bounding box annotation.

Figure A.10 demonstrates that this heuristic test can successfully capture the trends of the original tests (seen in Figure A.4) *without* access to bounding box annotations. However, as compared to Figure A.4, the heuristic tends to slightly underestimate the degree of memorization. This is likely due to the fact that some corner crops partially include the foreground object, thus enabling the KNN to successfully recover the label with the reference model where it would have failed with a proper periphery crop that excludes the foreground object.

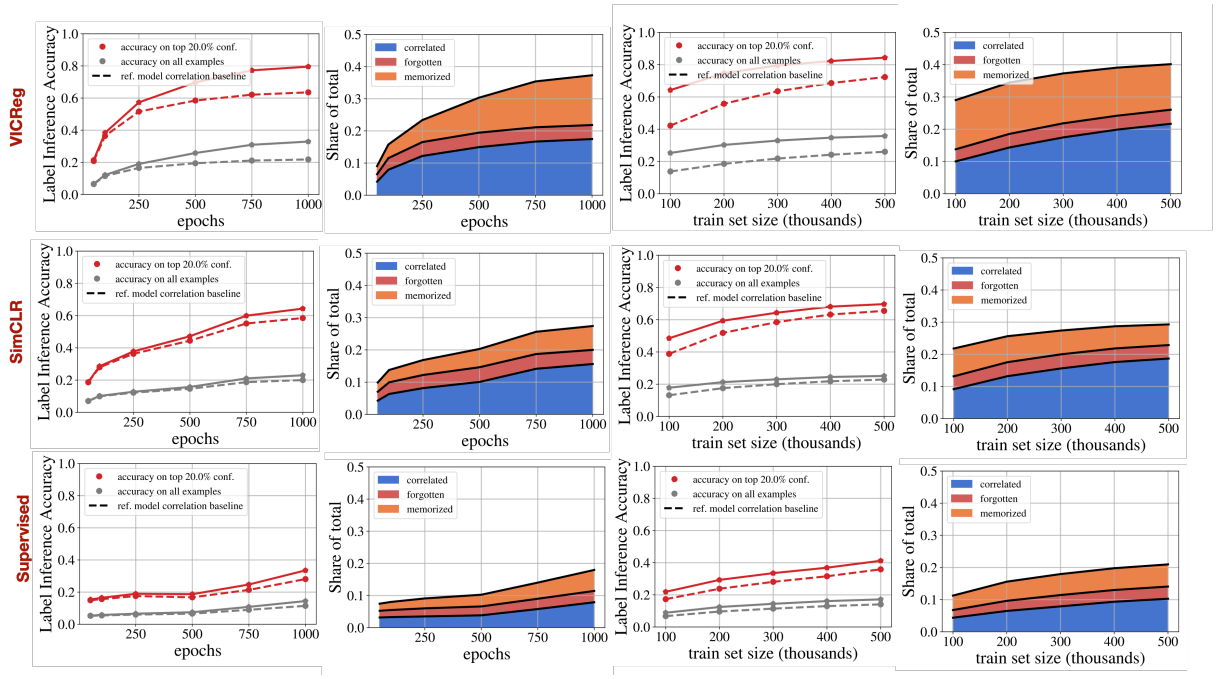


Figure A.10. Deja Vu Memorization using a simple corner crop instead of the periphery crop extracted using bounding box annotations. While the heuristic overall underestimates the degree of *déjà vu* , it roughly follows the same trends versus dataset size and training epochs. This is crucial, since it allows us to estimate *déjà vu* without access to bounding box annotations.

Appendix B

B.0.1 Proof of Theorem 1

A restatement of the theorem:

For true distribution P, model distribution Q, and distance metric $d : \mathscr{X} \to \mathbb{R}$, the estimator $\frac{1}{mn}U_{Q_m} \to_P \Delta(P,Q)$ according to the concentration inequality

$$\Pr\big(\big|\frac{1}{mn}U_{Q_m} - \Delta(P,Q)\big| \geq t\big) \leq \exp\bigg(-\frac{2t^2 mn}{m+n}\bigg)$$

Proof. We establish consistency using the following nifty lemma

Lemma 9. *(Bounded Differences Inequality) Suppose $X_1,\ldots,X_n \in \mathscr{X}$ are independent, and $f : \mathscr{X}^n \to \mathbb{R}$. Let $c_1,\ldots,c_n$ satisfy*

$$\sup_{x_1,\ldots,x_n,x_i'} \big|f(x_1,\ldots,x_i,\ldots,x_n) - f(x_1,\ldots,x_i',\ldots,x_n)\big| \leq c_i$$

for $i = 1,\ldots,n$. Then we have for any $t > 0$

$$\Pr\big(\big|f - \mathbb{E}[f]\big| \geq t\big) \leq \exp\bigg(\frac{-2t^2}{\sum_{i=1}^n c_i^2}\bigg) \tag{B.1}$$

This directly equips us to prove the Theorem.

It is relatively straightforward to apply Lemma 9 to the normalized $\overline{U} = \frac{1}{mn} U_{Q_m}$. First, think of it as a function of m independent samples of $X \sim Q$ and n independent samples of $Y \sim P$, $\overline{U}(X_1, \ldots, X_m, Y_1, \ldots, Y_n) = \frac{1}{mn} \sum_{ij} \mathbb{1}_{d(X_i) > d(Y_j)}$

$$\overline{U} : (\mathbb{R}^d)^{mn} \to \mathbb{R}$$

Let b_i bound the change in $\overline{U}$ after substituting any X_i with X_i', and c_j bound the change in $\overline{U}$ after substituting any Y_j with Y_j'. Specifically

$$\begin{aligned}
\sup_{x_1,\ldots,x_m,y_1,\ldots,y_n,\, x_i'} & \Big| \overline{U}(x_1, \ldots, x_i, \ldots, x_m, y_1, \ldots, y_n) \\
& - \overline{U}(x_1, \ldots, x_i', \ldots, x_m, y_1, \ldots, y_n) \Big| \\
& \leq b_i \\
\sup_{x_1,\ldots,x_m,y_1,\ldots,y_n,\, y_j'} & \Big| \overline{U}(x_1, \ldots, x_m, y_1, \ldots, y_j, \ldots y_n) \\
& - \overline{U}(x_1, \ldots, x_m, y_1, \ldots, y_j', \ldots y_n) \Big| \\
& \leq c_i
\end{aligned}$$

We then know that $b_i = \frac{n}{mn} = \frac{1}{m}$ for all i, with equality when $d(x_i') < d(y_j) < d(x_i)$ for all $j \in [n]$. In this case, substituting x_i with x_i' flips n of the indicator comparisons in $\overline{U}$ from 1 to 0, and is then normalized by mn. By a similar argument, $c_j = \frac{m}{nm} = \frac{1}{n}$ for all j.

Equipped with b_i and c_j, we may simply substitute into Equation B.1 of the Bounded

Differences Inequality, giving us

$$\begin{aligned}\Pr\left(\left|\overline{U}-\mathbb{E}[\overline{U}]\right|\geq t\right) &= \Pr\left(\left|\frac{1}{mn}U_{Q_m}-\Delta(\mu_p,\mu_q)\right|\geq t\right)\\ &\leq \exp\left(\frac{-2t^2}{\sum_{i=1}^{m}b_i^2+\sum_{j=1}^{n}c_j^2}\right)\\ &= \exp\left(\frac{-2t^2}{\sum_{i=1}^{m}\frac{1}{m^2}+\sum_{j=1}^{n}\frac{1}{n^2}}\right)\\ &= \exp\left(\frac{-2t^2}{\frac{1}{m}+\frac{1}{n}}\right)=\exp\left(\frac{-2t^2mn}{m+n}\right)\end{aligned}$$

∎

B.0.2 Proof of Theorem 2

A restatement of the theorem:

When $Q=P$, and the corresponding distance distribution $L(Q)=L(P)$ is non-atomic,

$$\mathbb{E}\Big[\frac{1}{mn}\overline{U}\Big]=\frac{1}{2}$$

Proof. For random variables $A\sim L(P)$ and $B\sim L(P)$, we can partition the event space of $A\times B$ into three disjoint events:

$$\begin{aligned}&\Pr(A>B)+\Pr(A<B)\\ &\qquad+\Pr(A=B)=1\end{aligned}$$

Since $Q=P$, the first two events have equal probability, $\Pr(A>B)=\Pr(A<B)$, so

$$2\Pr(A>B)+\Pr(A=B)=1$$

And since the distributions of A and B are non-atomic (i.e. $\Pr\big(B=b\big)=0,\quad\forall\, b\in\mathbb{R}$) we have

that $\Pr(A = B) = 0$, and thus

$$2\Pr(A > B) = 1$$
$$\Pr(A > B) = \Delta(P, Q) = \frac{1}{2}$$

∎

B.0.3 Proof of Lemma 3

Lemma 3 *For the kernel density estimator (2.1), the maximum-likehood choice of σ, namely the maximizer of $\mathbb{E}_{X\sim P}[\log q_\sigma(X)]$, satisfies*

$$\mathbb{E}_{X\sim P}\left[\sum_{t\in T} Q_\sigma(t|X)\|X-t\|^2\right] = \mathbb{E}_{Y\sim Q_\sigma}\left[\sum_{t\in T} Q_\sigma(t|Y)\|Y-t\|^2\right]$$

Proof. We have

$$\begin{aligned}
&\mathbb{E}_{X\sim P}\left[\ln q_\sigma(X)\right] \\
&= \mathbb{E}_{X\sim P}\left[-\ln((2\pi)^{k/2}|T|\sigma^k) + \ln\sum_{t\in T}\exp\left(-\frac{\|x-t\|^2}{2\sigma^2}\right)\right] \\
&= \text{constant} - k\ln\sigma + \mathbb{E}_{X\sim P}\left[\ln\sum_{t\in T}\exp\left(-\frac{\|x-t\|^2}{2\sigma^2}\right)\right]
\end{aligned}$$

Setting the derivative of this to zero and simplifying, we find that the maximum-likelihood σ satisfies

$$\sigma^2 = \frac{1}{k}\,\mathbb{E}_{X\sim P}\left[\sum_{t\in T} Q_\sigma(t|X)\|X-t\|^2\right]. \tag{B.2}$$

Now, interpreting Q_σ as a mixture of $|T|$ Gaussians, and using the notation $t \in_R T$ to mean that t

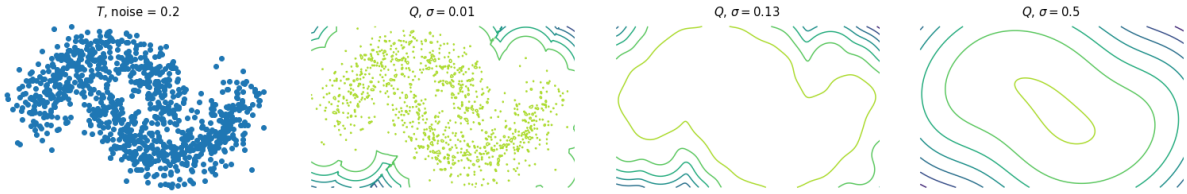
is chosen uniformly at random from T, we have

$$\mathbb{E}_{Y\sim Q_\sigma}\left[\sum_{t\in T} Q_\sigma(t|Y)\|Y-t\|^2\right]$$
$$= \mathbb{E}_{t\in_R T}\mathbb{E}_{Y\sim N(t,\sigma^2 I_k)}\left[\|Y-t\|^2\right] = k\sigma^2.$$

Combining this with (B.2) yields the lemma. ∎

B.0.4 Procedural Details of Experiments

Moons Dataset, and Gaussian KDE



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

Figure B.1. Contour plots of KDE fit on T: a) training T sample, b) over-fit 'data copying' KDE, c) max likelihood KDE, d) underfit KDE

moons dataset

'Moons' is a synthetic dataset consisting of two curved interlocking manifolds with added configurable noise. We chose to use this dataset as a proof of concept because it is low dimensional, and thus KDE friendly and easy to visualize, and we may have unlimited train, test, and validation samples.

Gaussian KDE

We use a Gaussian KDE as our preliminary generative model Q because its likelihood is theoretically related to our non-parametric test. Perhaps more importantly, it is trivial to control the degree of data-copying with the bandwidth parameter σ. **Figures B.1b, B.1c, B.1d** provide contour plots of of a Gaussian KDE Q trained on the moons dataset with progressively larger σ.

With $\sigma = 0.01$, Q will effectively resample the training set. $\sigma = 0.13$ is nearly the MLE model. With $\sigma = 0.5$, the KDE struggles to capture the unique definition of T.

Moons Experiments

Our experiments that examined whether several baseline tests could detect data-copying (Section 2.4.1), and our first test of our own metric (Section 2.4.2) use the moons dataset. In both of these, we fix a training sample, T of 2000 points, a test sample P_n of 1000 points, and a generated sample Q_m of 1000 points. We regenerate Q_m 10 times, and report the average statistic across these trials along with a single standard deviation. If the standard deviation buffer along the line is not visible, it is because the standard deviation is relatively small. We artificially set the constraint that $m, n \ll |T|$, as is true for big natural datasets, and more elaborate models that are computationally burdensome to sample from.

Section 2.4.1 Methods

Here are the routines we used for the four baseline tests:

- **Frechét Inception Distance (FID)** [92]: Normally, this test is run on two samples of images (T and Q_m) that are first embedded into a perceptually meaningful latent space using a discriminative neural net, like the Inception Network. By 'meaningful' we mean points that are closer together are more perceptually alike to the human eye. Unlike images in pixel space, the samples of the moons dataset require no embedding, so we run the Frechét test directly on the samples.

  First, we fit two MLE Gaussians: $\mathcal{N}(\mu_T, \Sigma_T)$ to T, and $\mathcal{N}(\mu_Q, \Sigma_Q)$ to Q_m, by collecting their respective MLE mean and covariance parameters. The statistic reported is the Frechét distance between these two Gaussians, denoted $\mathrm{Fr}(\bullet, \bullet)$, which for Gaussians has a closed

form:

$$\mathrm{Fr}\big(\mathcal{N}(\mu_T,\Sigma_T),\mathcal{N}(\mu_Q,\Sigma_Q)\big) = \\ \|\mu_T-\mu_Q\| + \mathbf{Tr}\big(\Sigma_T+\Sigma_Q-2(\Sigma_T\Sigma_Q)^{1/2}\big)$$

Naturally, if Q is data-copying T, its MLE mean and covariance will be nearly identical, rendering this test ineffective for capturing this kind of overfitting.

- **Binning Based Evaluation** [136]: This test, takes a hypothesis testing approach for evaluating mode collapse and deletion. The test bears much similarity to the test described in Section 2.3.2. The basic idea is as follows. Split the training set into partition Π using k-means; the number of samples falling into each bin is approximately normally distributed if it has ¿20 samples. Check the null hypothesis that the normal distribution of the fraction of the training set in bin π, $T(\pi)$, equals the normal distribution of the fraction of the generated set in bin π, $Q_m(\pi)$. Specifically:

$$Z_\pi = \frac{Q_m(\pi)-T(\pi)}{\sqrt{\widehat{p}\big(1-\widehat{p}\big)\big(\frac{1}{|T|}+\frac{1}{m}\big)}}$$

where $\widehat{p} = \frac{|T|T(\pi)+mQ_m(\pi)}{|T|+m}$. We then perform a one-sided hypothesis test, and compute the number of positive Z_π values that are greater than the significance level of 0.05. We call this the number of statistically different bins or NDB. The NDB/k ought to equal the significance level if $P=Q$.

- **Two-Sample Nearest-Neighbor** [121]: In this test — our primary baseline — we report the three LOO NN values discussed in [169]. The generated sample Q_m and training sample (subsampled to have equal size, m), $\widetilde{T}\subseteq T$, are joined together create sample $S=\widetilde{T}\cup Q_m$ of size $2m$, with training samples labeled '1' and test samples labeled '0'. One then fits a 1-Nearest-Neighbor classifier to S, and reports the accuracy in predicting the

training samples ('1's), the accuracy in predicting the generated samples ('0's), and the average.

One can expect that — when Q collapses to a few mode centers of T — the training accuracy is low, and the generated accuracy is high, thus indicating over-representation. Additionally, one could imagine that when the training and generated accuracies are near 0, we have extreme data-copying. However, as explained in Experiments section, when we are forced to subsample T, it is unlikely that a given copied training point $t \in T$ is used in the test, thus making the test result unclear.

- **Precision and Recall** [140]: This method offers a clever technique for scaling classical precision and recall statistics to high dimensional, complex spaces. First, all samples are embedded to Inception Network Pool3 features. Then, the author's use the following insight: for distribution's Q and P, the precision and recall curve is approximately given by the set of points:

$$\widehat{\text{PRD}}(Q,P) = \{(\alpha(\lambda),\beta(\lambda)|\lambda \in \Lambda\}$$

where

$$\Lambda = \{\tan\big(\frac{i}{r+1}\frac{\pi}{2}\big)|i \in [r]\}$$
$$\alpha(\lambda) = \sum_{\pi\in\Pi} \min\big(\lambda P(\pi), Q(\pi)\big)$$
$$\beta(\lambda) = \sum_{\pi\in\Pi} \min\big(P(\pi), \frac{Q(\pi)}{\lambda}\big)$$

and where r is the 'resolution' of the curve, the set Π is a partition of the instance space and $P(\pi), Q(\pi)$ are the fraction of samples falling in cell π. Π is determined by running k-means on the combination of the training and generated sets. In our tests here, we set $k = 5$, and report the average PRD curve measured over 10 k-means clusterings (and then

re-run 10 times for 10 separate trials of Q_m).

MNIST Experiments

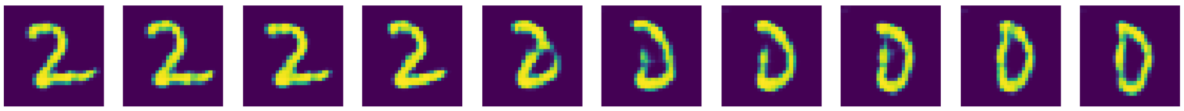
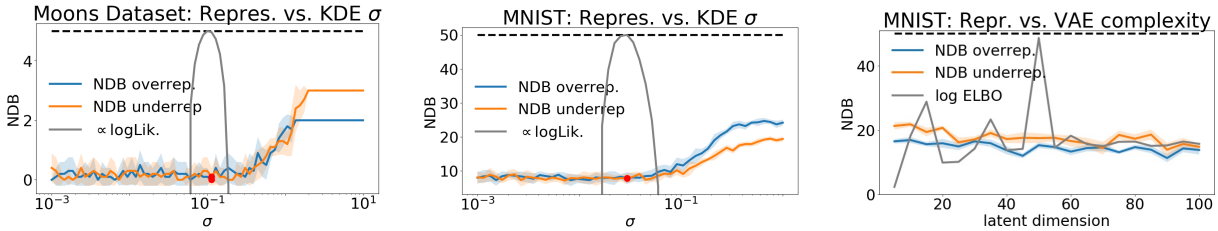


Figure B.2. Interpolating between two points in the latent space to demonstrate L_2 perceptual significance

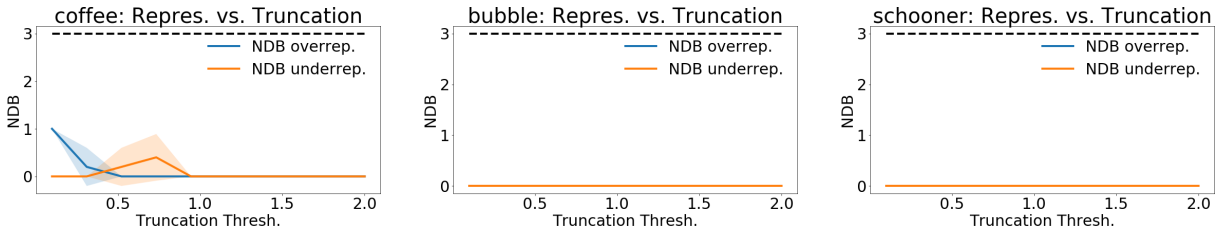


(a) **(b)** **(c)**

Figure B.3. Number of statistically different bins, both those over and under the significance level of 0.05. The black dotted line indicates the total number of cells or 'bins'. **(a,b)** KDEs tend to start misrepresenting with $\sigma \gg \sigma_{\text{MLE}}$, which makes sense as they become less and less dependent on training set. **(c)** it makes sense that the VAE over- and under-represents across all latent dimensions due to its reverse KL loss. There is slightly worse over- and under-representation for simple models with low latent dimension.

The experiments of Sections 2.4.2 and 2.4.3 use the MNIST digit dataset [111]. We use a training sample, T, of size $|T| = 50,000$, a test sample P_n of size $n = 10,000$, a validation sample V_l of $l = 10,000$, and create generated samples of size $m = 10,000$.

Here, for a meaningful distance metric, we create a custom embedding using a convolutional autoencoder trained using a VGG perceptual loss proposed by [183]. The encoder and decoder each have four convolutional layers using batch normalization, two linear layers using dropout, and two max pool layers. The autoencoder is trained for 100 epochs with a batch size of 128 and Adam optimizer with learning rate 0.001. For each training sample $t \in \mathbb{R}^{784}$, the



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

Figure B.4. This GAN model produces relatively equal representation according to our clustering for all three classes. It makes sense that a low truncation level tends to over-represent for one class, as a lower truncation threshold causes less variance. Even though it places samples into all cells, some cells are data-copying much more aggressively than others.

encoder compresses to $z \in \mathbb{R}^{64}$, and decoder expands back up to $\widehat{t} \in \mathbb{R}^{784}$. Our loss is then

$$L(t,\widehat{t}) = \gamma(t,\widehat{t}) + \lambda \max\{\|z\|_2^2 - 1, 0\}$$

where $\gamma(\bullet,\bullet)$ is the VGG perceptual loss, and $\lambda \max\{\|z\|_2^2 - 1, 0\}$ provides a linear hinge loss outside of a unit L_2 ball. The hinge loss encourages the encoder to learn a latent representation within a bounded domain, hopefully augmenting its ability to interpolate between samples. It is worth noting that the perceptual loss is not trained on MNIST, and hopefully uses agnostic features that help keep us from overfitting. We opt to use a standard autoencoder instead of a stochastic autoencoder like a VAE, because we want to be able to exactly data-copy the training set T. Thus, we want the encoder to create a near-exact encoding and decoding of the training samples specifically. **Figure B.2** provides an example of linearly spaced steps between two training samples. While not perfect, we observe that half-way between the '2' and the '0' is a sample that appears perceptually to be almost almost a '2' and almost a '0'. As such, we consider the distance metric $d(x)$ on this space used in our experiments to be meaningful.

KDE tests:

In the MNIST KDE experiments, we fit each KDE Q on the 64-d latent representations of the training set T for several values of σ; we gather all statistical tests in this space, and effectively only decode to visaully inspect samples. We gather the average and standard deviation

(a) $Z_U = -1.05$ **(b)** $Z_U = +1.32$

Figure B.5. Example of data-copied and underfit cell of ImageNet 'schooner' instance space, from 'BigGan' with trunc. threshold = 2. We note here, that — limited to only 50 training samples — the insufficient $k = 3$ clustering is perhaps not fine grain enough for this class. Notice that the generated samples falling into the underfit cell (mostly training images of either masts or fronts of boats) are hardly any different from those of the over-fit cell. They are likely on the boundary of the two cells. With that said, the samples of the data-copied cell **(a)** are certainly close to the training samples in this region.

of each data point across 5 trials of generating Q_m. For the Two-Sample Nearest-Neighbor test, it is computationally intense to compute the nearesnt neighbor in a 64-dimensional dataset of 20,000 points $\widetilde{T} \cup Q_m$ 20,000 times. To limit this, we average each of the training and generated NN accuracy over 500 training and generated samples. We find this acceptable, since the test results depicted in **Figure 2.3f** are relatively low variance.

VAE experiments:

In the MNIST VAE experiments, we only use the 64-d autoencoder latent representation in computing the C_T and 1-NN test scores, and not at all in training. Here, we experiment with twenty standard, fully connected, VAEs using binary cross entropy reconstruction loss. The twenty models have three hidden layers and latent dimensions ranging from $d = 5$ to $d = 100$ in steps of 5. The number of neurons in intermediate layers is approximately twice the number of the layer beneath it, so for a latent space of 50-d, the encoder architecture is $784 \to 400 \to 200 \to 100 \to 50$, and the decoder architecture is the opposite.

To sample from a trained VAE, we sample from a standard normal with dimensionality equivalent to the VAEs latent dimension, and pass them through the VAE decoder to the 784-d image space. We then encode these generated images to the agnostic 64-d latent space of the perceptual autoencoder described at the beginning of the section, where L_2 distance is meaningful.

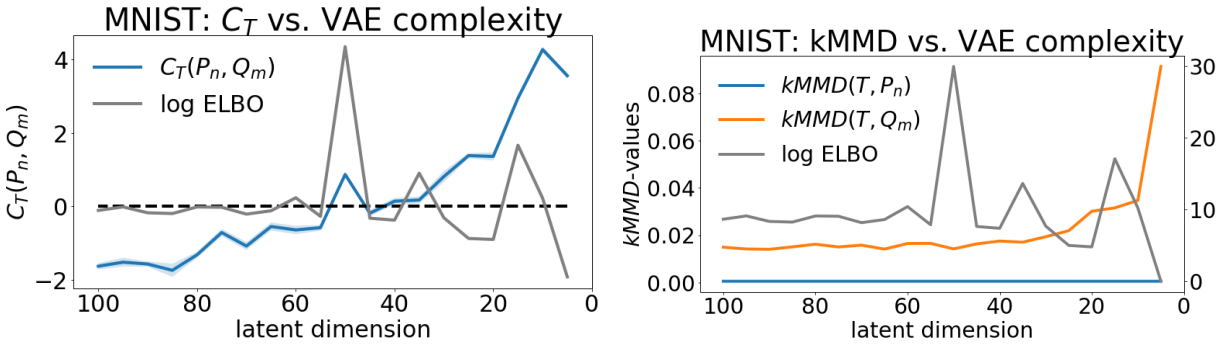
We also encode the training sample T and test sample P_n to this space, and then run the C_T and two-sample NN tests. We again compute the nearest neighbor accuracies for 500 of the training and generated samples (the 1-NN classifier is fit on the 20,000 sample set $\widetilde{T} \cup Q_m$), which appears to be acceptable due to low test variance.

ImageNet Experiments

Here, we have chosen three of the one thousand ImageNet12 classes that 'BigGan' produces. To reiterate, a conditional GAN can output samples from a specific class by conditioning on a class code input. We acknowledge that conditional GANs combine features from many classes in ways not yet well understood, but treat the GAN of each class as a uniquely different generative model trained on the training samples from that class. So, for the 'coffee' class, we treat the GAN as a coffee generator Q, trained on the 1300 'coffee' class samples. For each class, we have 1300 training samples $|T|$, 2000 generated samples m, and 50 test samples n. Being atypically training sample starved ($m > |T|$), we subsample Q_m (not T!), to produce equal size samples for the two-sample NN test. As such, all training samples used are in the combined set S. We also note that the 50 test samples provided in each class is highly limiting, only allowing us to split the instance space into about three cells and keep a reasonable number of test samples in each cell. As the number of test samples grows, so can the number of cells and the resolution of the partition. **Figure B.5** provides an example of where this clustering might be limited; the generated samples of the underfit cell seem hardly any different from those of the over-fit cell. A finer-grain partition is likely needed here. However, the data-copied cell to the left does appear to be very close to the training set, potentially too close according to Z_U.

In performing these experiments, we gather the $C_T(P_n, Q_m)$ statistic for a given class of images. In an attempt to embed the images into a lower dimensional latent space with L_2 significance, we pass each image through an InceptionV3 network and gather the 2048-dimension feature embeddings after the final average pooling layer (Pool3). We then project all inception-space images (T, P_n, Q_m) onto the 64 principal components of the training set

embeddings. Finally, we use k-means to partition the points of each sample into one of $k = 3$ cells. The number of cells is limited by the 50 test images available per class. Any more cells would strain the Central Limit Theorem assumption in computing Z_U. Finally, we gather the C_T and two-sample NN baseline statistics on this 64-d space.



(a) $C_T(P_n, Q_m)$ **(b)** kernel MMD

Figure B.6. Comparison of the $C_T(P_n, Q_m)$ test presented in this paper alongside the three sample kMMD test.

B.0.5 Comparison with three-sample Kernel-MMD:

Another three-sample test not shown in the main body of this work is the three-sample kernel MMD test introduced by [27] intended more for model comparison than for checking model overfitting. For samples $X \sim P$ and $Y \sim Q$, we can estimate the squared kernel MMD between P and Q under kernel k by empirically estimating

$$\begin{aligned}&\mathrm{MMD}^2(P, Q)\\ &= \mathbb{E}_{x,x' \sim P}[k(x, x')] - 2\mathbb{E}_{x \sim P, y \sim Q}[k(x, y)] + \mathbb{E}_{y,y' \sim Q}[k(y, y')]\end{aligned}$$

More recent works such as [67] have repurposed this test for measuring generative model overfitting. Intuitively, if the model is overfitting its training set, the empirical kMMD between training and generated data may be smaller than that between training and test sets. This may be triggered by the data-copying variety of overfitting.

This test provides an interesting benchmark to consider in addition to those in the main body. **Figure B.6** demonstrates some preliminary experimental results repeating the MNIST VAE experiment **Figure 2.4b**. To implement the kMMD, we used code posted by [27] https://github.com/eugenium/MMD, and ran the three sample RBF-kMMD test on twenty MNIST VAEs with decreasing complexity (latent dimension). **Figure B.6b** attached plots the kMMD distance to training set for both the generated (orange) and test samples (blue). We observe that this test does not appear sensitive to over-parametrized VAEs ($d > 50$) in the same way our proposed test (Figure 1a attached) is. It is sensitive to underfitting ($d << 50$), however. The p-values of that work's corresponding hypothesis test (figure not shown) similarly did not respond to data-copying. We suspect that this insensitivity to data-copying is due to the global nature of this test, incapable of capturing data-copying at smaller scales.

Appendix C

C.0.1 Privacy Mechanism

We now describe in detail our instance of the exponential mechanism $\mathscr{M}_{\text{TD}}$. Recall from Definition 3.2.2 that the exponential mechanism samples candidate $f_i \in F$ with probability

$$\Pr[\mathscr{M}(x) = f_i] \propto \exp\left(\frac{\varepsilon u(x, f_i)}{2\Delta u}\right) .$$

Thus, $\mathscr{M}_{\text{TD}}$ is fully defined by its utility function, which, as listed in Equation (3.3), is approximate Tukey Depth,

$$u(x, f_i) = \widehat{\text{TD}}_{S_x}(f_i) \quad .$$

We now describe our approximation algorithm of Tukey Depth $\widehat{\text{TD}}_{S_x}(f_i)$, which is an adaptation of the general median hypothesis algorithm proposed by **(author?)** [82].

Note that we can precompute the projections on line 10. The runtime is $O(mkp)$: for each of m candidates and on each of p projections, we need to compute the scalar difference with k sentence embeddings. Sampling from the multinomial distribution defined by P_F then takes $O(m)$ time.

Additionally note from lines 13 and 15 that utility has a maximum of 0 and a minimum of $-\frac{k}{2}$, which is a semantic change from the main paper where maximum utility is $\frac{k}{2}$ and minimum

Algorithm 3: $\mathscr{M}_{\text{TD}}$ compute probabilities

Input : m candidates F,
sentence embs. $S_x = (s_1, \ldots, s_k)$,
number of projections p

Output : probability of sampling each candidate $P_F = [P_{f_1}, \ldots, P_{f_m}]$

$v_1, \ldots, v_p \leftarrow$ random vecs. on unit sphere
// Project all embeddings
for $i \in [k]$ **do**
  for $j \in [p]$ **do**
    $s_i^j \leftarrow s_i^\intercal v_j$
  end for
end for
for $i \in [m]$ **do**
  for $j \in [p]$ **do**
    $f_i^j \leftarrow f_i^\intercal v_j$
    /* Compute depth of f_i on projection v_j */
    $h_j(x, f_i) \leftarrow \#\{s_l^j : s_l^j \geq f_i^j, l \in [k]\}$
    $u_j(x, f_i) \leftarrow -\left|h_j(x, f_i) - \frac{k}{2}\right|$
  end for
  $u(x, f_i) \leftarrow \max_{j \in [p]} u_j(x, f_i)$ $\dot{P}_{f_i} \leftarrow \exp(\varepsilon u(x, f_i)/2)$
end for
$\Psi \leftarrow \sum_{i=1}^m \dot{P}_{f_i}$
for $i \in [m]$ **do**
  $P_{f_i} \leftarrow \frac{1}{\Psi} \dot{P}_{f_i}$
end for
return P_F

is 0.

C.0.2 Proof of Privacy

Theorem 3.4.1 *$\mathscr{M}_{TD}$ satisfies ε-Sentence Privacy*

Proof. It is sufficient to show that the sensitivity,

$$\Delta u = \max_{x,x',f_i} |u(x,f_i) - u(x',f_i)| \leq 1 \quad .$$

Let us expand the above expression using the terms in Algorithm 3.

$$\begin{aligned}
\Delta u &= \max_{x,x',f_i} |\max_{j\in[p]} u_j(x,f_i) - \max_{j'\in[p]} u_{j'}(x',f_i)| \\
&= \max_{x,x',f_i} |\min_{j\in[p]} \left|h_j(x,f_i) - \frac{k}{2}\right| \\
&- \min_{j'\in[p]} \left|h_{j'}(x',f_i) - \frac{k}{2}\right||
\end{aligned}$$

$$\begin{aligned}
&\leq \max_{f_i} |\min_{j\in[p]} \left|h_j(x,f_i) - \frac{k}{2}\right| \\
&- \left(\min_{j'\in[p]} \left|h_{j'}(x,f_i) - \frac{k}{2}\right| - 1\right)| \\
&\leq 1
\end{aligned}$$

The last step follows from the fact that $|h_j(x,f_i) - h_j(x',f_i)| \leq 1$ for all $j \in [p]$. In other words, by modifying a single sentence embedding, we can only change the number of embeddings greater than f_i^j on projection j by 1. So, the distance of $h_j(x,f_i)$ from $\frac{k}{2}$ can only change by 1 on each projection. In the 'worst case', the distance $\left|h_j(x,f_i) - \frac{k}{2}\right|$ reduces by 1 on every projection v_j. Even then, the minimum distance from $\frac{k}{2}$ across projections (the worst case depth) can only change by 1, giving us a sensitivity of 1. ∎

C.0.3 Experimental Details

Here, we provide an extended, detailed version of section 3.5.

For the general encoder, $G : \mathscr{S} \to \mathbb{R}^{768}$, we use SBERT [135], a version of BERT fine-tuned for sentence encoding. Sentence embeddings are generated by mean-pooling output tokens. In all tasks, we freeze the weights of SBERT. The cluster-preserving recoder, H, as well as every classifier is implemented as an instance of a 4-layer MLP taking 768-dimension inputs and only differing on output dimension. We denote an instance of this MLP with output dimension o as $\mathbf{MLP}^o$. We run 5 trials of each experiment with randomness taken over the privacy mechanisms, and plot the mean along with a $\pm$ 1 standard deviation envelope.

Non-private:

For our non-private baseline, we demonstrate the usefulness of sentence-mean document embeddings. First, we generate the document embeddings $\overline{g}(x_i)$ for each training, validation, and test set document using SBERT, G. We then train a classifier $C_{\text{nonpriv}} = \mathbf{MLP}^r$ to predict each document's topic or sentiment, where r is the number of classes. The number of training epochs is determined with the validation set.

DeepCandidate:

We first collect the candidate set F by sampling 5k document embeddings from the subset of the training set containing at least 8 sentences. We run k-means with $n_c = 50$ cluster centers, and label each training set document embedding $t_i \in T_G$ with its cluster. The sentence recoder, $H = \mathbf{MLP}^{768}$ is trained on the training set along with the linear model L with the Adam optimizer and cross-entropy loss. For a given document x, its sentence embeddings S_x are passed through H, averaged together, and then passed to L to predict x's cluster. L's loss is then back-propagated through H. A classifier $C_{\text{dc}} = \mathbf{MLP}^r$ is trained in parallel using a separate instance of the Adam optimizer to predict class from the recoded embeddings, where r is the number of classes (topics or sentiments). The number of training epochs is determined using the validation set. At test time, (generating private embeddings using $\mathscr{M}_{\text{TD}}$), the optimal number of projections p is empirically chosen for each ε using the validation set.

Truncation:

The truncation baseline [114] requires first constraining the embedding instance space. We do so by computing the 75% median interval on each of the 768 dimensions of training document embeddings T_G. Sentence embeddings are truncated at each dimension to lie in this box. In order to account for this distribution shift, a new classifier $C_{\text{trunc}} = \mathbf{MLP}^r$ is trained on truncated mean embeddings to predict class. The number of epochs is determined with the validation set. At test time, a document's sentence embeddings S_x are truncated and averaged. We then add Laplace noise to each dimension with scale factor $\frac{768w}{k\varepsilon}$, where w is the width of the box on that dimension (*sensitivity* in DP terms). Note that the standard deviation of noise added is inversely proportional to the number of sentences in the document, due to the averaging operation reducing sensitivity.

Word Metric-DP:

Our next baseline satisfies ε-word-level metric DP and is adopted from [73]. The corresponding mechanism $\text{MDP} : \mathscr{X} \to \mathscr{X}$ takes as input a document x and returns a private version, x', by randomizing each word individually. For comparison, we generate document embeddings by first randomizing the document $x' = \text{MDP}(x)$ as prescribed by [73], and then computing its document embedding $\overline{g}(x')$ using SBERT. At test time, we classify the word-private document embedding using C_{nonpriv}.

Random Guess:

To set a bottom-line, we show the theoretical performance of a random guesser. The guesser chooses class i with probability q_i equal to the fraction of i labels in the training set. The performance is then given by $\sum_{i=1}^{r} q_i^2$.

C.0.4 Reproducability Details

We plan to publish a repo of code used to generate the exact figures in this paper (random seeds have been set) with the final version. Since we do not train the BERT base model G, our

algorithms and training require relatively little computational resouces. Our system includes a single Nvidia GeForce RTX 2080 GPU and a single Intel i9 core. All of our models complete an epoch training on all datasets in less than one minute. We never do more than 20 epochs of training. All of our classifier models train (including linear model) have less than 11 million parameters. The relatively low amount of parameters is due to the fact that we freeze the underlying language model. The primary hyperparameter tuned is the number of projections p. We take the argmax value on the validation set between 10 and 100 projections. We repeat this for each value of ε.

Dataset preprocessing:

For all datasets, we limit ourselves to documents with at least 2 sentences.

IMDB: This dataset has pre-defined train/test splits. We use the entire training set and form the test set by randomly sampling 4,000 from the test set provided. We do this for efficiency in computing the Metric-DP baseline, which is the slowest of all algorithms performed. Since the Metric-DP baseline randomizes first, we cannot precompute the sentence embeddings $G(s_i)$ – we need to compute the sentence embeddings every single time we randomize. Since we randomize for each sentence of each document at each ε and each k over 5 trials – this takes a considerable amount of time.

Good Reads: This dataset as provided is quite large. We randomly sample 15000 documents from each of 4 classes, and split them into 12K training examples, 2K validation examples, and 1K test examples per class.

20 News Groups: We preprocess this dataset to remove all header information, which may more directly tell information about document class, and only provide the model with the sentences from the main body. We use the entire dataset, and form the Train/Val/Test splits by random sampling.

Appendix D

D.1 Appendix

D.1.1 Background Cntd.

D.1.2 Local Inferential Privacy

Local inferential privacy captures what information a Bayesian adversary [105], with some prior, can learn in the LDP setting. Specifically, it measures the largest possible ratio between the adversary's posterior and prior beliefs about an individual's data after observing a mechanism's output .

Definition D.1.1. (Local Inferential Privacy Loss [105]) Let $\mathbf{x} = \langle x_1, \cdots, x_n \rangle$ and let $\mathbf{y} = \langle y_1, \cdots, y_n \rangle$ denote the input (private) and output sequences (observable to the adversary) in the LDP setting. Additionally, the adversary's auxiliary knowledge is modeled by a prior distribution $\mathscr{P}$ on $\mathbf{x}$. The inferential privacy loss for the input sequence $\mathbf{x}$ is given by

$$\mathbb{L}_{\mathscr{P}}(\mathbf{x}) = \max_{\substack{i \in [n] \\ a,b \in \mathscr{X}}} \left(\log \frac{\Pr_{\mathscr{P}}[\mathbf{y}|x_i = a]}{\Pr_{\mathscr{P}}[\mathbf{y}|x_i = b]} \right) = \max_{\substack{i \in [n] \\ a,b \in \mathscr{X}}} \left(\left| \log \frac{\Pr_{\mathscr{P}}[x_i = a|\mathbf{y}]}{\Pr_{\mathscr{P}}[x_i = b|\mathbf{y}]} - \log \frac{\Pr_{\mathscr{P}}[x_i = a]}{\Pr_{\mathscr{P}}[x_i = b]} \right| \right) \tag{D.1}$$

Bounding $\mathbb{L}_{\mathscr{P}}(\mathbf{x})$ would imply that the adversary's belief about the value of any x_i does not change by much even after observing the output sequence $\mathbf{y}$. This means that an informed adversary does not learn much about the individual i's private input upon observation of the entire private dataset $\mathbf{y}$.

Here we define two rank distance measures

Definition D.1.2 (Kendall's τ Distance)**.** For any two permutations, $\sigma, \pi \in \mathrm{S}_n$, the Kendall's τ distance $\mathfrak{d}_\tau(\sigma,\pi)$ counts the number of pairwise disagreements between σ and π, i.e., the number of item pairs that have a relative order in one permutation and a different order in the other. Formally,

$$\mathfrak{d}_\tau(\sigma,\pi) = \left| \left\{ (i,j) : i < j, \left[\sigma(i) > \sigma(j) \wedge \pi(i) < \pi(j)\right] \vee \left[\sigma(i) < \sigma(j) \wedge \pi(i) > \pi(j)\right] \right\} \right| \quad \text{(D.2)}$$

For example, if $\sigma = (1\ 2\ 3\ 4\ 5\ 6\ 7\ 8\ 9\ 10)$ and $\pi = (1\,2\,3\,\underline{6}\,5\,\underline{4}\,7\,8\,9\,10)$, then $\mathfrak{d}_\tau(\sigma,\pi) = 3$.

Next, Hamming distance measure is defined as follows.

Definition D.1.3 (Hamming Distance)**.** For any two permutations, $\sigma, \pi \in \mathrm{S}_n$, the Hamming distance $\mathfrak{d}_H(\sigma,\pi)$ counts the number of positions in which the two permutations disagree. Formally,

$$\mathfrak{d}_H(\sigma,\pi) = \left| \left\{ i \in [n] : \sigma(i) \neq \pi(i) \right\} \right|$$

Repeating the above example, if $\sigma = (1\ 2\ 3\ 4\ 5\ 6\ 7\ 8\ 9\ 10)$ and $\pi = (1\,2\,3\,\underline{6}\,5\,\underline{4}\,7\,8\,9\,10)$, then $\mathfrak{d}_H(\sigma,\pi) = 2$.

D.1.3 d_σ-privacy and the De Finetti attack

We now show that a strict instance of $d_\sigma$privacy is sufficient for thwarting any de Finetti attack [102] on individuals. The de Finetti attack involves a Bayesian adversary, who, assuming some degree of correlation between data owners, attempts to recover the true permutation from the shuffled data. As written, the de Finetti attack assumes the sequence of sensitive attributes and side information $(x_1,t_1),\ldots,(x_n,t_n)$ are *exchangeable*: any ordering of them is equally likely. By the de Finetti theorem, this implies that they are i.i.d. conditioned on some latent measure

θ. To balance privacy with utility, the $\mathbf{x}$ sequence is non-uniformly randomly shuffled w.r.t. the $\mathbf{t}$ sequence producing a shuffled sequence $\mathbf{z}$, which the adversary observes. Conditioning on $\mathbf{z}$ the adversary updates their posterior on θ (i.e. posterior on a model predicting $x_i|t_i$), and thereby their posterior predictive on the true $\mathbf{x}$. The definition of privacy in [102] holds that the adversary's posterior beliefs are close to their prior beliefs by some metric on distributions in $\mathscr{X}$, $\delta(\cdot,\cdot)$:

$$\delta\Big(\Pr[x_i], \Pr[x_i|\mathbf{z}]\Big) \leq \alpha$$

We now translate the de Finetti attack to our setting. First, to align notation with the rest of the paper we provide privacy to the sequence of LDP values $\mathbf{y}$ since we shuffle those instead of the $\mathbf{x}$ values as in [102]. We use max divergence (multiplicative bound on events used in DP) for δ:

$$\Pr[y_i \in O] \leq e^{\alpha} \Pr[y_i \in O|\mathbf{z}]$$
$$\Pr[y_i \in O|\mathbf{z}] \leq e^{\alpha} \Pr[y_i \in O]$$

which, for compactness, we write as

$$\Pr[y_i \in O] \approx_{\alpha} \Pr[y_i \in O|\mathbf{z}] \quad . \tag{D.3}$$

We restrict ourselves to shuffling mechanisms, where we only randomize the order of sensitive values. By learning the unordered values $\{y\}$ alone, an adversary may have arbitrarily large updates to its posterior (e.g. if all values are identical), breaking the privacy requirement above. With this in mind, we assume the adversary already knows the unordered sequence of values $\{y\}$ (which they will learn anyway), and has a prior on permutations σ allocating values from that sequence to individuals. We then generalize the de Finetti problem to an adversary with an

arbitrary prior on the true permutation σ, and observes a randomize permutation σ' from the shuffling mechanism. We require that the adversary's prior belief that $\sigma(i) = j$ is close to their posterior belief for all $i, j \in [n]$:

$$\Pr[\sigma \in \Sigma_{i,j}] \approx_\alpha \Pr[\sigma \in \Sigma_{i,j} | \sigma'] \quad \forall i, j \in [n], \forall \sigma' \in S_n \quad , \tag{D.4}$$

where $\Sigma_{i,j} = \{\sigma \in S_n : \sigma(i) = j\}$, the set of permutations assigning element j to DO_i. Conditioning on any unordered sequence $\{y\}$ with all unique values, the above condition is necessary to satisfy Eq. (D.3) for events of the form $O = \{y_i = a\}$, since $\{y_i = a\} = \{\Sigma_{i,j}\}$ for some $j \in [n]$. For any $\{y\}$ with repeat values, it is sufficient since $\Pr[y_i = a]$ is the sum of probabilities of disjoint events of the form $\Pr[\sigma \in \Sigma_{i,k}]$ for various $k \in [n]$ values.

We now show that a strict instance of d_σ-privacy satisfies Eq. (D.4). Let $\widehat{\mathscr{G}}$ be any group assignment such that at least one $G_i \in \widehat{\mathscr{G}}$ includes all data owners, $G_i = \{1, 2, \ldots, n\}$.

Property 1. *A $(\widehat{\mathscr{G}}, \alpha)$-$d_\sigma$-private shuffling mechanism $\sigma' \sim \mathscr{A}$ satisfies*

$$\Pr[\sigma \in \Sigma_{i,j}] \approx_\alpha \Pr[\sigma \in \Sigma_{i,j} | \sigma']$$

for all $i, j \in [n]$ and all priors on permutations $\Pr[\sigma]$.

Proof.

Lemma 10. *For any prior $\Pr[\sigma]$, Eq. (D.4) is equivalent to the condition*

$$\frac{\sum_{\dot{\sigma} \in \overline{\Sigma}_{i,j}} \Pr[\dot{\sigma}] \Pr[\sigma' | \dot{\sigma}]}{\sum_{\dot{\sigma} \in \Sigma_{i,j}} \Pr[\dot{\sigma}] \Pr[\sigma' | \dot{\sigma}]} \approx_\alpha \frac{\sum_{\dot{\sigma} \in \overline{\Sigma}_{i,j}} \Pr[\dot{\sigma}]}{\sum_{\dot{\sigma} \in \Sigma_{i,j}} \Pr[\dot{\sigma}]} \tag{D.5}$$

where the set $\overline{\Sigma}_{i,j}$ is the complement of $\Sigma_{i,j}$.

Under grouping $\dot{\mathscr{G}}$, every permutation $\sigma_a \in \Sigma_{i,j}$ neighbors every permutation $\sigma_b \in \overline{\Sigma}_{i,j}$, $\sigma_a \approx_{\dot{\mathscr{G}}} \sigma_b$, for any i, j. By the definition of d_σ-privacy, we have that for any observed permutation

σ' output by the mechanism:

$$\Pr[\sigma'|\sigma = \sigma_a] \approx_\alpha \Pr[\sigma'|\sigma = \sigma_b] \quad \forall \sigma_a \in \Sigma_{i,j}, \sigma_b \in \overline{\Sigma}_{i,j}, \sigma' \in S_n \quad .$$

This implies Eq. D.5. Thus, $(\widehat{\mathscr{G}}, \alpha)$-$d_\sigma$-privacy implies Eq. D.5, which implies Eq. D.4, thus proving the property. ∎

Using Lemma 10, we may also show that this strict instance of d_σ-privacy is *necessary* to block all de Finetti attacks:

Property 2. *A $(\widehat{\mathscr{G}}, \alpha)$-$d_\sigma$-private shuffling mechanism $\sigma' \sim \mathscr{A}$ is necessary to satisfy*

$$\Pr[\sigma \in \Sigma_{i,j}] \approx_\alpha \Pr[\sigma \in \Sigma_{i,j}|\sigma']$$

for all $i, j \in [n]$ and all priors on permutations $\Pr[\sigma]$.

Proof. If our mechanism $\mathscr{A}$ is not $(\widehat{\mathscr{G}}, \alpha)$-$d_\sigma$-private, then for some pair of true (input) permutations $\sigma_a \neq \sigma_b$ and some released permutation $\sigma' \sim \mathscr{A}$, we have that

$$\Pr[\sigma'|\sigma_b] \geq e^\alpha \Pr[\sigma'|\sigma_a] \quad .$$

Under $\mathscr{G}^j$, all permutations neighbor each other, so $\sigma_a \approx_{\mathscr{G}^j} \sigma_b$. Since $\sigma_a \neq \sigma_b$, then for some $i, j \in [n]$, $\sigma_a \in \Sigma_{i,j}$ and $\sigma_b \in \overline{\Sigma}_{i,j}$: one of the two permutations assigns some j to some DO_i and the other does not. Given this, we may construct a bimodal prior on the true σ that assigns half its probability mass to σ_a and the rest to σ_b,

$$\Pr[\sigma_a] = \Pr[\sigma_b] = \frac{1}{2} \quad .$$

Therefore, for released permutation σ', the RHS of Eq. D.5 is 1, and the LHS is

$$\frac{\sum_{\dot{\sigma}\in\overline{\Sigma}_{i,j}} \Pr[\dot{\sigma}]\Pr[\sigma'|\dot{\sigma}]}{\sum_{\dot{\sigma}\in\Sigma_{i,j}} \Pr[\dot{\sigma}]\Pr[\sigma'|\dot{\sigma}]} = \frac{1/2\Pr[\sigma'|\sigma_b]}{1/2\Pr[\sigma'|\sigma_a]}$$
$$\geq e^{\alpha} \quad ,$$

violating Eq. D.5, thus violating Eq. D.4, and failing to prevent de Finetti attacks against this bimodal prior. ∎

Ultimately, unless we satisfy d_σ-privacy shuffling the entire dataset, there exists some prior on the true permutation $\Pr[\sigma]$ such that after observing the shuffled $\mathbf{z}$ permuted by σ', the adversary's posterior belief on one permutation is larger than their prior belief by a factor $\geq e^{\alpha}$. If we suppose that the set of values $\{y\}$ are all distinct, this means that for some $a \in \{y\}$, the adversary's belief that $y_i = a$ is signficantly larger after observing $\mathbf{z}$ than it was before.

Now to prove Lemma 10:

Proof.

$$\Pr[\sigma \in \Sigma_{i,j}] \approx_\alpha \Pr[\sigma \in \Sigma_{i,j}|\sigma']$$
$$\Pr[\sigma \in \Sigma_{i,j}] \approx_\alpha \frac{\Pr[\sigma'|\sigma \in \Sigma_{i,j}]\Pr[\sigma \in \Sigma_{i,j}]}{\sum_{\dot{\sigma}\in S_n} \Pr[\dot{\sigma}]\Pr[\sigma'|\dot{\sigma}]}$$
$$\sum_{\dot{\sigma}\in S_n} \Pr[\dot{\sigma}]\Pr[\sigma'|\dot{\sigma}] \approx_\alpha \Pr[\sigma'|\sigma \in \Sigma_{i,j}]$$
$$\sum_{\dot{\sigma}\in S_n} \Pr[\dot{\sigma}]\Pr[\sigma'|\dot{\sigma}] \approx_\alpha \Pr[\sigma \in \Sigma_{i,j}]^{-1} \sum_{\dot{\sigma}\in\Sigma_{i,j}} \Pr[\dot{\sigma}]\Pr[\sigma'|\dot{\sigma}]$$
$$\sum_{\dot{\sigma}\in\Sigma_{i,j}} \Pr[\dot{\sigma}]\Pr[\sigma'|\dot{\sigma}] + \sum_{\dot{\sigma}\in\overline{\Sigma}_{i,j}} \Pr[\dot{\sigma}]\Pr[\sigma'|\dot{\sigma}] \approx_\alpha \Pr[\sigma \in \Sigma_{i,j}]^{-1} \sum_{\dot{\sigma}\in\Sigma_{i,j}} \Pr[\dot{\sigma}]\Pr[\sigma'|\dot{\sigma}]$$
$$\sum_{\dot{\sigma}\in\overline{\Sigma}_{i,j}} \Pr[\dot{\sigma}]\Pr[\sigma'|\dot{\sigma}] \approx_\alpha \sum_{\dot{\sigma}\in\Sigma_{i,j}} \Pr[\dot{\sigma}]\Pr[\sigma'|\dot{\sigma}]\Big(\frac{1}{\Pr[\sigma \in \Sigma_{i,j}]} - 1\Big)$$
$$\frac{\sum_{\dot{\sigma}\in\overline{\Sigma}_{i,j}} \Pr[\dot{\sigma}]\Pr[\sigma'|\dot{\sigma}]}{\sum_{\dot{\sigma}\in\Sigma_{i,j}} \Pr[\dot{\sigma}]\Pr[\sigma'|\dot{\sigma}]} \approx_\alpha \frac{\sum_{\dot{\sigma}\in\overline{\Sigma}_{i,j}} \Pr[\dot{\sigma}]}{\sum_{\dot{\sigma}\in\Sigma_{i,j}} \Pr[\dot{\sigma}]}$$

∎

As such, a strict instance of d_σ-privacy can defend against any de Finetti attack (i.e. for any prior $\Pr[\sigma]$ on permutations), wherein at least one group $G_i \in \mathscr{G}$ includes all data owners. Furthermore, it is necessary. This makes sense. In order to defend against any prior, we need to significantly shuffle the entire dataset. Without a restriction of priors as in Pufferfish [105], the de Finetti attack (i.e. uninformed Bayesian adversaries) is an indelicate metric for evaluating the privacy of shuffling mechanisms: to achieve significant privacy, we must sacrifice all utility. This in many regards is reminiscent of the no free lunch for privacy theorem established in [103]. As such, there is a need for more flexible privacy definitions for shuffling mechanisms.

D.1.4 Additional Properties of d_σ-privacy

Lemma 11 (Convexity)**.** *Let $\mathscr{A}_1, \dots \mathscr{A}_k : \mathscr{Y}^n \mapsto \mathscr{V}$ be a collection of k $(\alpha, \mathscr{G})$-$d_\sigma$private mechanisms for a given group assignment $\mathscr{G}$ on a set of n entities. Let $\mathscr{A} : \mathscr{Y}^n \mapsto \mathscr{V}$ be a convex combination of these k mechanisms, where the probability of releasing the output of mechanism $\mathscr{A}_i$ is p_i, and $\sum_{i=1}^k p_i = 1$. $\mathscr{A}$ is also $(\alpha, \mathscr{G})$-$d_\sigma$private w.r.t. $\mathscr{G}$.*

Proof. For any $(\sigma, \sigma') \in \mathrm{N}_{\mathscr{G}}$ and $\mathbf{y} \in \mathscr{Y}$:

$$\begin{aligned}
\Pr[\mathscr{A}\big(\sigma(\mathbf{y})\big) \in O] &= \sum_{i=1}^{k} p_i \Pr[\mathscr{A}_i\big(\sigma(\mathbf{y})\big) \in O] \\
&\leq e^{\alpha} \sum_{i=1}^{k} p_i \Pr[\mathscr{A}_i\big(\sigma'(\mathbf{y})\big) \in O] \\
&= \Pr[\mathscr{A}\big(\sigma'(\mathbf{y})\big) \in O]
\end{aligned}$$

∎

Theorem 12 (Post-processing)**.** *Let $\mathscr{A} : \mathscr{Y}^n \mapsto \mathscr{V}$ be $(\alpha, \mathscr{G})$-$d_\sigma$private for a given group assignment $\mathscr{G}$ on a set of n entities. Let $f : \mathscr{V} \mapsto \mathscr{V}'$ be an arbitrary randomized mapping. Then $f \circ \mathscr{A} : \mathscr{Y}^n \mapsto \mathscr{V}'$ is also $(\alpha, \mathscr{G})$-$d_\sigma$private.*

Proof. Let $g : \mathscr{V} \to \mathscr{V}'$ be a deterministic, measurable function. For any output event $\mathscr{Z} \subset \mathscr{V}'$, let $\mathscr{W}$ be its preimage:
$\mathscr{W} = \{v \in \mathscr{V} | g(v) \in \mathscr{Z}\}$. Then, for any $(\sigma, \sigma') \in \mathrm{N}_{\mathscr{G}}$,

$$\begin{aligned}
\Pr\Big[g\Big(\mathscr{A}\big(\sigma(\mathbf{y})\big)\Big) \in \mathscr{Z}\Big] &= \Pr\Big[\mathscr{A}\big(\sigma(\mathbf{y})\big) \in \mathscr{W}\Big] \\
&\leq e^{\alpha} \cdot \Pr\Big[\mathscr{A}\big(\sigma'(\mathbf{y})\big) \in \mathscr{W}\Big] \\
&= e^{\alpha} \cdot \Pr\Big[g\Big(\mathscr{A}\big(\sigma'(\mathbf{y})\big)\Big) \in \mathscr{Z}\Big]
\end{aligned}$$

This concludes our proof because any randomized mapping can be decomposed into a convex combination of measurable, deterministic functions [62], and as Lemma 11 shows, a convex combination of $(\alpha, \mathscr{G})$-$d_\sigma$private mechanisms is also $(\alpha, \mathscr{G})$-$d_\sigma$private. ∎

Theorem 13 (Sequential Composition)**.** *If $\mathscr{A}_1$ and $\mathscr{A}_2$ are $(\alpha_1, \mathscr{G})$- and $(\alpha_2, \mathscr{G})$-$d_\sigma$private mechanisms, respectively, that use independent randomness, then releasing the outputs $\big(\mathscr{A}_1(\mathbf{y}), \mathscr{A}_2(\mathbf{y})\big)$ satisfies $(\alpha_1 + \alpha_2, \mathscr{G})$-$d_\sigma$privacy.*

Proof. We have that $\mathscr{A}_1 : \mathscr{Y}^n \to \mathscr{V}'$ and $\mathscr{A}_1 : \mathscr{Y}^n \to \mathscr{V}''$ each satisfy d_σ-privacy for different α values. Let $\mathscr{A} : \mathscr{Y}^n \to (\mathscr{V}' \times \mathscr{V}'')$ output $\big(\mathscr{A}_1(\mathbf{y}), \mathscr{A}_2(\mathbf{y})\big)$. Then, we may write any event $\mathscr{Z} \in (\mathscr{V}' \times \mathscr{V}'')$ as $\mathscr{Z}' \times \mathscr{Z}''$, where $\mathscr{Z}' \in \mathscr{V}'$ and $\mathscr{Z}'' \in \mathscr{V}''$. We have for any $(\sigma, \sigma') \in \mathrm{N}_{\mathscr{G}}$,

$$\begin{aligned}
\Pr\big[\mathscr{A}\big(\sigma(\mathbf{y})\big) \in \mathscr{Z}\big] &= \Pr\big[\big(\mathscr{A}_1\big(\sigma(\mathbf{y})\big), \mathscr{A}_2\big(\sigma(\mathbf{y})\big)\big) \in \mathscr{Z}\big] \\
&= \Pr\big[\{\mathscr{A}_1\big(\sigma(\mathbf{y})\big) \in \mathscr{Z}'\} \cap \{\mathscr{A}_2\big(\sigma(\mathbf{y})\big) \in \mathscr{Z}''\}\big] \\
&= \Pr\big[\{\mathscr{A}_1\big(\sigma(\mathbf{y})\big) \in \mathscr{Z}'\}\big] \Pr\big[\{\mathscr{A}_2\big(\sigma(\mathbf{y})\big) \in \mathscr{Z}''\}\big] \\
&\leq e^{\alpha_1+\alpha_2} \Pr\big[\{\mathscr{A}_1\big(\sigma'(\mathbf{y})\big) \in \mathscr{Z}'\}\big] \Pr\big[\{\mathscr{A}_2\big(\sigma'(\mathbf{y})\big) \in \mathscr{Z}''\}\big] \\
&= e^{\alpha_1+\alpha_2} \cdot \Pr\big[\mathscr{A}\big(\sigma'(\mathbf{y})\big) \in \mathscr{Z}\big]
\end{aligned}$$

∎

D.1.5 Proof for Thm. 4

Theorem 4 *For a given group assignment $\mathscr{G}$ on a set of n data owners, if a shuffling mechanism $\mathscr{A} : \mathscr{Y}^n \mapsto \mathscr{Y}^n$ is $(\alpha, \mathscr{G})$-$d_\sigma$private, then for each data owner $DO_i, i \in [n]$,*

$$\max_{\substack{i\in[n]\\ a,b\in\mathscr{X}}} \left| \log \frac{\Pr_{\mathscr{P}}[x_i = a|\mathbf{z}, \{y_{G_i}\}, \mathbf{y}_{\overline{G}_i}]}{\Pr_{\mathscr{P}}[x_i = b|\mathbf{z}, \{y_{G_i}\}, \mathbf{y}_{\overline{G}_i}]} - \log \frac{\Pr_{\mathscr{P}}[x_i = a|\{y_{G_i}\}, \mathbf{y}_{\overline{G}_i}]}{\Pr_{\mathscr{P}}[x_i = b|\{y_{G_i}\}, \mathbf{y}_{\overline{G}_i}]} \right| \leq \alpha$$

for a prior distribution $\mathscr{P}$, where $\mathbf{z} = \mathscr{A}(\mathbf{y})$ and $\mathbf{y}_{\overline{G}_i}$ is the noisy sequence for data owners outside G_i.

Proof. We prove the above by bounding the following equivalent expression for any $i \in [n]$ and $a, b \in \mathscr{X}$.

$$\begin{aligned}
&\frac{\Pr_{\mathscr{P}}[\mathbf{z}|x_i = a, \{y_{G_i}\}, \mathbf{y}_{\overline{G}_i}]}{\Pr_{\mathscr{P}}[\mathbf{z}|x_i = b, \{y_{G_i}\}, \mathbf{y}_{\overline{G}_i}]} \\
&= \frac{\int \Pr_{\mathscr{P}}[\mathbf{y}|x_i = a, \{y_{G_i}\}, \mathbf{y}_{\overline{G}_i}] \Pr_{\mathscr{A}}[\mathbf{z}|\mathbf{y}] d\mathbf{y}}{\int \Pr_{\mathscr{P}}[\mathbf{y}|x_i = b, \{y_{G_i}\}, \mathbf{y}_{\overline{G}_i}] \Pr_{\mathscr{A}}[\mathbf{z}|\mathbf{y}] d\mathbf{y}} \\
&= \frac{\sum_{\sigma\in \mathrm{S}_r} \Pr_{\mathscr{P}}[\sigma(\mathbf{y}^*_{G_i})|x_i = a, \mathbf{y}_{\overline{G}_i}] \Pr_{\mathscr{A}}[\mathbf{z}|\sigma(\mathbf{y}^*_{G_i}), \mathbf{y}_{\overline{G}_i}]}{\sum_{\sigma\in \mathrm{S}_r} \Pr_{\mathscr{P}}[\sigma(\mathbf{y}^*_{G_i})|x_i = b, \mathbf{y}_{\overline{G}_i}] \Pr_{\mathscr{A}}[\mathbf{z}|\sigma(\mathbf{y}^*_{G_i}), \mathbf{y}_{\overline{G}_i}]} \\
&\leq \max_{\{\sigma,\sigma'\in \mathrm{S}_r\}} \frac{\Pr_{\mathscr{A}}[\mathbf{z}|\sigma(\mathbf{y}^*_{G_i}), \mathbf{y}_{\overline{G}_i}]}{\Pr_{\mathscr{A}}[\mathbf{z}|\sigma'(\mathbf{y}^*_{G_i}), \mathbf{y}_{\overline{G}_i}]} \\
&\leq \max_{\{\sigma,\sigma'\in \mathrm{N}_{G_i}\}} \frac{\Pr_{\mathscr{A}}[\mathbf{z}|\sigma(\mathbf{y})]}{\Pr_{\mathscr{A}}[\mathbf{z}|\sigma'(\mathbf{y})]} \\
&\leq e^{\alpha}
\end{aligned}$$

The second line simply marginalizes out the full noisy sequence $\mathbf{y}$. The third line reduces this to a sum over permutations of of $\mathbf{y}_{G_i}$, where $r = |G_i|$ and $\mathbf{y}^*_{G_i}$ is any fixed permutation of values $\{y_{G_i}\}$. This is possible since we are given the values outside the group, $\mathbf{y}_{\overline{G}_i}$, and the unordered set of values inside the group, $\{y_{G_i}\}$. Note that the permutations σ written here are possible permutations of the LDP input, not permutations output by the mechanism applied to the input as sometimes written in other parts of this document.

The fourth line uses the fact that the numerator and denominator are both convex combinations of $\Pr_{\mathscr{A}}[\mathbf{z}|\sigma(\mathbf{y}^*_{G_i}), \mathbf{y}_{\overline{G_i}}]$ over all $\sigma \in \mathrm{S}_r$.

The fifth line uses the fact that for any $\mathbf{y}_{\overline{G_i}}$,

$$(\sigma(\mathbf{y}^*_{G_i}), \mathbf{y}_{\overline{G_i}}) \approx_{G_i} (\sigma'(\mathbf{y}^*_{G_i}), \mathbf{y}_{\overline{G_i}}) \ .$$

This allows a further upper bound over all neighboring sequences w.r.t. G_i, and thus over any permutation of $\mathbf{y}_{\overline{G_i}}$, as long as it is the same in the numerator and denominator. ∎

Discussion

The above Bayesian analysis measures what can be learned about DO_i's x_i from observing the private release $\mathbf{z}$ relative to some other known information (the conditioned information). Under d_σ-privacy, we condition on the bag of LDP values in Alice's group $\{y_{G_i}\}$ as well as the sequence (order and value) of LDP values outside her group $\mathbf{y}_{\overline{G_i}}$. This implies that releasing the shuffled sequence $\mathbf{z}$ cannot provide much more information about Alice's x_i than would releasing the LDP values outside her neighborhood (her group) and the unordered bag of LDP values inside her neighborhood, regardless of the adversary's prior knowledge $\mathscr{P}$. This is a communicable guarantee: if Alice feels comfortable with the data collection knowing that her entire neighborhood's responses will be uniformly shuffled together (including those of her household), then she ought to be comfortable with d_σ-privacy. Now, we have to provide this guarantee to Bob, a neighbor of Alice, as well as Luis, a neighbor of Bob but *not* of Alice. Thus, Bob, Alice and Luis have *distinct* and *overlapping* groups (neighborhoods). Hence, the trivial solution of uniformly shuffling the noisy responses of every group separately does not work in this case. d_σ-privacy, however, offers the above guarantee to each user (knowing that their entire neighborhood is *nearly* uniformly shuffled) while still maintaining utility (estimate disease prevalence within neighborhoods). Semantically, this is very powerful, since it implies that the noisy responses specific to one's household cannot be leveraged to infer one's disease state x_i.

D.1.6 Proof of Theorem 5

Theorem 5

For $\mathscr{A}(\mathscr{M}(\mathbf{x})) = \mathbf{z}$ where $\mathscr{M}(\cdot)$ is ε-LDP and $\mathscr{A}(\cdot)$ is α - $d_\sigma$private, we have

$$\Pr[\mathscr{D}_{Adv} \textit{ loses}] \geq \lfloor \frac{r-k}{k} \rfloor e^{-(2k\varepsilon+\alpha)} \cdot \Pr[\mathscr{D}_{Adv} \textit{ wins}]$$

for any input subgroup $I \subset G_i, r = |G_i|$ and $k < r/2$.

Proof. We first focus on deterministic adversaries and then expand to randomized adversaries afterwards using the fact that randomized adversaries are mixtures of deterministic ones.

Our adversary $\mathscr{D}_{Adv}$ is then defined by a deterministic decision function $\eta : \mathscr{Y}^n \to [n]^k$. Upon observing $\mathbf{z}$, $\eta(\mathbf{z})$ selects k elements in $\mathbf{z}$ which it believes originated from $I \subset G_i$.

In the following, let $\Pr_{\mathbf{z}}$ be the probability of events conditioned on the shuffled output sequence $\mathbf{z}$, where randomness is over the ε-LDP mechanism $\mathscr{M}$ and the α-d_σ-private shuffling mechanism $\mathscr{A}$. [1]

The adversary wins if it reidentifies $> \frac{k}{2}$ of the LDP values originating from I. Let $H = \eta(\mathbf{z})$ be the indices of elements in $\mathbf{z}$ selected by η. Let $W = \{\sigma \in \mathrm{S}^n : |\sigma(H) \cap I| > \frac{k}{2}\}$ be the set of permutations where the adversary wins and let $L = \{\sigma \in \mathrm{S}^n : \sigma(H) \cap I| \leq \frac{k}{2}\}$ be the set of permutations where the adversary loses.

$$\Pr_{\mathbf{z}}[\eta(\mathbf{z}) \text{ wins}] = \Pr_{\mathbf{z}}[\sigma \in W]$$
$$\Pr_{\mathbf{z}}[\eta(\mathbf{z}) \text{ loses}] = \Pr_{\mathbf{z}}[\sigma \in L]$$

where σ is the shuffling permutation produced by $\mathscr{A}$, $\mathbf{z} = \sigma(\mathbf{y})$ i.e. $z_i = y_{\sigma(i)}$. Concretely, this is equivalent to DO_i releasing $\mathsf{DO}_{\sigma(i)}$'s LDP response. Since the permutation and LDP outputs are

[1] As an abuse of notation, we assume the output space of the LDP randomizers, $\mathscr{Y}$, have outcomes with non-zero measure e.g. randomized response. The following analysis can be expanded to continuous outputs (with outcomes of zero measure) by simply replacing the output sequence $\mathbf{z} \in \mathscr{Y}^n$ with an output event $\mathbf{Z} \subseteq \mathscr{Y}^n$.

randomized, many subgroups of size k in G_i could have produced the LDP values $(z_{H_1}, \ldots, z_{H_k})$ and then been mapped to H by a permutation. Concretely, there is a reasonable probability that Alice's household output the LDP values of another k-member household in her neighborhood and they output her household's LDP values. In the worst case, this is $e^{-2k\varepsilon}$ less likely than without swapping values, by group DP guarantees. Since both households are part of the same group G_i, the permutation that maps her household to elements H in the output is close in probability to that which maps the other household to elements H in the output. As such, we have in the worst case a $e^{-(2k\varepsilon+\alpha)}$ reduction in probability of the other household having swapped LDP values with Alice's and permuting to subset H.

The above provides intuition on how we could get the same output $\mathbf{z}$ many different ways, and how Alice's household could or could not contribute to elements H. It does not, however, explain why an adversary who is given output $\mathbf{z}$ has limited advantage in choosing a subset H such that they recover *most* of Alice's household's values. We formalize this fact as follows.

We may rewrite the probabilities of winning or losing by marginalizing out all possible LDP sequences $\mathbf{y}$. Conditioning on the output sequence $\mathbf{z}$, the only possible LDP sequences $\mathbf{y}$ are permutations of $\mathbf{z}$. Note that the probability of any sequence $\mathbf{y}$ is determined by the input $\mathbf{x}$ and the LDP mechanism $\mathscr{M}$:

$$\begin{aligned}\Pr_{\mathbf{z}}[\eta(\mathbf{z}) \text{ loses}] &= \Pr_{\mathbf{z}}[\sigma \in W] \\ &= \sum_{\sigma \in W} \Pr[\mathscr{A}(\mathbf{x}) = \mathbf{y} = \sigma^{-1}(\mathbf{z})] \Pr[\sigma|\mathbf{y}] / \Pr[\mathbf{z}]\end{aligned}$$

Note that $\Pr_{\mathbf{z}}[\sigma|\mathbf{y}] = \Pr_{\mathbf{z}}[\sigma]$ for the mallows mechanism, which chooses its permutations independently of $\mathbf{y}$. Now consider when $\eta(\mathbf{z})$ loses. By similar arguments as above:

$$\begin{aligned}\Pr_{\mathbf{z}}[\eta(\mathbf{z}) \text{ loses}] &= \Pr_{\mathbf{z}}[\sigma \in L] \\ &= \sum_{\sigma \in L} \Pr[\mathscr{A}(\mathbf{x}) = \mathbf{y} = \sigma^{-1}(\mathbf{z})] \Pr[\sigma|\mathbf{y}] / \Pr[\mathbf{z}]\end{aligned}$$

The odds of losing versus winning is given by

$$\frac{\Pr_{\mathbf{z}}[\eta(\mathbf{z}) \text{ loses}]}{\Pr_{\mathbf{z}}[\eta(\mathbf{z}) \text{ wins}]} = \frac{\sum_{\sigma' \in L} \Pr[\mathscr{A}(\mathbf{x}) = \mathbf{y} = \sigma'^{-1}(\mathbf{z})] \Pr[\sigma'|\mathbf{y}]}{\sum_{\sigma \in W} \Pr[\mathscr{A}(\mathbf{x}) = \mathbf{y} = \sigma^{-1}(\mathbf{z})] \Pr[\sigma|\mathbf{y}]}$$

We now show that for each σ in the denominator, we may construct $m = \lfloor \frac{r-k}{k} \rfloor$ distinct permutations σ' in the numerator that are close in probability to it.

Lemma 14. *For every $\sigma \in W$ there exists a set of $m = \lfloor \frac{r-k}{k} \rfloor$ permutations, $E(\sigma)$, such that*

1. $E(\sigma) \subseteq L$
2. $\sigma^{-1} \approx_{G_i} \sigma'^{-1}$
3. $E(\sigma_a) \cap E(\sigma_b) = \emptyset$ *for any pair* $\sigma_a, \sigma_b \in W$
4. $\Pr[\mathscr{A}(\mathbf{x}) = \mathbf{y} = \sigma^{-1}(\mathbf{z})] \leq e^{2k\varepsilon} \Pr[\mathscr{A}(\mathbf{x}) = \mathbf{y} = \sigma'^{-1}(\mathbf{z})]$ *for any* $\mathbf{x} \in \mathscr{X}^n$ *and any* $\mathbf{z} \in \mathscr{Y}^n$

Proof. Given $\sigma \in W$, we construct $E(\sigma)$ by first taking the inverse σ^{-1}. Recall that, since $\sigma \in W$, we have that $|\sigma^{-1}(I) \cap H| > \frac{k}{2}$. ($\sigma^{-1}(i) = j$ could be interpreted as data owner i's LDP value will be output at position j). We then divide the remainder of the group $G_i \backslash I$ into m disjoint subsets of size k each, $J_1, J_2, \ldots, J_m$. These represent the other distinct subsets of size k that Alice's household could swap LDP values with. We then produce m permutations, $\sigma_1'^{-1}, \ldots, \sigma_m'^{-1}$, by making $\sigma_i'^{-1}(I) = \sigma^{-1}(J_i)$ and $\sigma_i'^{-1}(J_i) = \sigma^{-1}(I)$ (preserving order within those subsets) and $\sigma'^{-1} = \sigma^{-1}$ everywhere else.

On the first point, we know that every $\sigma' \in E(\sigma)$ is also in L. We know this because $\sigma_i'^{-1}(I) = \sigma^{-1}(J_i)$. Since $\sigma \in W$, we have that $|\sigma^{-1}(J_i) \cap H| < \frac{k}{2}$ since $|\sigma^{-1}(I) \cap H| \geq \frac{k}{2}$ and $I \cap J_i = \emptyset$ by definition. Thus, $|\sigma_i'^{-1}(I) \cap H| < \frac{k}{2}$, so $|\sigma_i'(H) \cap I| < \frac{k}{2}$ and $\sigma_i' \in L$.

On the second point, we know that the inverse permutations are neighboring $\sigma^{-1} \approx_{G_i} \sigma'^{-1}$ simply by construction – they only differ on elements in G_i.

On the third point, we know that the sets $E(\sigma_a)$ and $E(\sigma_b)$ are distinct since we can map any permutation $\sigma' \in E(\sigma_a)$ uniquely back to σ_a for any $\sigma_a \in W$. We do so by taking its inverse $\sigma^{'-1}$, finding which subset J_i has majority elements from H i.e. $|\sigma^{'-1}(J_i) \cap H| > \frac{k}{2}$. Swap elements back: $\sigma^{'-1}(J_i)$ with $\sigma^{'-1}(I)$. Invert back to σ_a.

On the fourth point, we know that $\sigma^{-1}(\mathbf{z})$ and $\sigma^{'-1}(\mathbf{z})$ differ on at most $2k$ indices. As such, by group DP guarantees, we know that their probabilities must be close to a factor of $e^{-2k\varepsilon}$ regardless of $\mathbf{z}$ and $\mathbf{x}$. ∎

Using the above Lemma we may bound the odds of losing vs. winning.

$$\begin{aligned}
\frac{\Pr_{\mathbf{z}}[\eta(\mathbf{z}) \text{ loses}]}{\Pr_{\mathbf{z}}[\eta(\mathbf{z}) \text{ wins}]} &= \frac{\sum_{\sigma' \in L} \Pr[\mathscr{A}(\mathbf{x}) = \mathbf{y} = \sigma^{'-1}(\mathbf{z})] \Pr[\sigma'|\mathbf{y}]}{\sum_{\sigma \in W} \Pr[\mathscr{A}(\mathbf{x}) = \mathbf{y} = \sigma^{-1}(\mathbf{z})] \Pr[\sigma|\mathbf{y}]} \\
&\geq \frac{\sum_{\sigma \in W} \sum_{\sigma' \in E(\sigma)} \Pr[\mathscr{A}(\mathbf{x}) = \mathbf{y} = \sigma^{'-1}(\mathbf{z})] \Pr[\sigma'|\mathbf{y}]}{\sum_{\sigma \in W} \Pr[\mathscr{A}(\mathbf{x}) = \mathbf{y} = \sigma^{-1}(\mathbf{z})] \Pr[\sigma|\mathbf{y}]} \\
&\geq \min_{\sigma \in W} \frac{\sum_{\sigma' \in E(\sigma)} \Pr[\mathscr{A}(\mathbf{x}) = \mathbf{y} = \sigma^{'-1}(\mathbf{z})] \Pr[\sigma'|\mathbf{y}]}{\Pr[\mathscr{A}(\mathbf{x}) = \mathbf{y} = \sigma^{-1}(\mathbf{z})] \Pr[\sigma|\mathbf{y}]} \\
&\geq \lfloor \frac{r-k}{k} \rfloor e^{-(2k\varepsilon+\alpha)}
\end{aligned}$$

where the last line follows from the fourth point of the above Lemma (for the $2k\varepsilon$ term) and the fact that the inverse permutations $\sigma'^{-1}, \sigma^{-1}$ are neighboring (second point of the Lemma) so the probabilities of the mechanism to produce σ vs. σ' to reach $\mathbf{z}$ from these neighboring permutations must be close by a factor of e^{α}.

Since the above holds for any $\mathbf{z}$ and $\mathbf{x}$, the bound holds on average across all outcomes $\mathbf{z}$, thus

$$\Pr[\eta \text{ loses}] \geq \lfloor \frac{r-k}{k} \rfloor e^{-(2k\varepsilon+\alpha)} \cdot \Pr[\eta \text{ wins}]$$

for any deterministic adversary with decision function η. Finally, we may write any probabilistic adversary as mixture of decision functions. By convexity (same argument used in Lemma 11),

the above bound still holds. As such,

$$\Pr[\mathscr{D}_{Adv} \text{ loses}] \geq \lfloor \frac{r-k}{k} \rfloor e^{-(2k\varepsilon+\alpha)} \cdot \Pr[\mathscr{D}_{Adv} \text{ wins}]$$

∎

D.1.7 Utility of Shuffling Mechanism

We now introduce a novel metric, (η, δ)-preservation, for assessing the utility of any shuffling mechanism. Let $S \subseteq [n]$ correspond to a set of indices in $\mathbf{y}$. The metric is defined as follows.

Definition D.1.4. ((η, δ)-preservation) A shuffling mechanism $\mathscr{A} : \mathscr{Y}^n \mapsto \mathscr{Y}^n$ is defined to be (η, δ)-preserving ($\eta, \delta \in [0,1]$) w.r.t to a given subset $S \subseteq [n]$, if

$$\Pr\left[|S_\sigma \cap S| \geq \eta \cdot |S|\right] \geq 1 - \delta, \sigma \in \mathrm{S}_n \tag{D.6}$$

where $\mathbf{z} = \mathscr{A}(\mathbf{y}) = \sigma(\mathbf{y})$ and $S_\sigma = \{\sigma(i) | i \in S\}$.

For example, consider $S = \{1,4,5,7,8\}$. If $\mathscr{A}(\cdot)$ permutes the output according to $\sigma = (\underline{5}\,3\,2\,\underline{6}\,\underline{7}\,9\,\underline{8}\,\underline{1}\,4\,10)$, then $S_\sigma = \{5,6,7,8,1\}$ which preserves 4 or 80% of its original indices. This means that for any data sequence $\mathbf{y}$, at least η fraction of its data values corresponding to the subset S overlaps with that of shuffled sequence $\mathbf{z}$ with high probability $(1-\delta)$. Assuming, $\{y_S\} = \{y_i | i \in S\}$ and $\{z_S\} = \{z_i | i \in S\} = \{y_{\sigma(i)} | i \in S\}$ denotes the set of data values corresponding to S in data sequences $\mathbf{y}$ and $\mathbf{z}$ respectively, we have $\Pr\left[|\{y_S\} \cap \{z_S\}| \geq \eta \cdot |S|\right] \geq 1 - \delta$, $\forall \mathbf{y}$. For example, let S be the set of individuals from Nevada. Then, for a shuffling mechanism that provides $(\eta = 0.8, \delta = 0.1)$-preservation to S, with probability ≥ 0.9, $\geq 80\%$ of the values that are reported to be from Nevada in $\mathbf{z}$ are genuinely from Nevada. The rationale behind this metric is that it captures the utility of the learning allowed by d_σ-privacy – if S is equal to some group $G \in \mathscr{G}$, (η, δ) preservation

allows overall statistics of G to be captured. Note that this utility metric is *agnostic of both the data distribution and the analyst's query*. Hence, it is a conservative analysis of utility which serves as a lower bound for learning from $\{z_S\}$. We suspect that with the knowledge of the data distribution and/or the query, a tighter utility analysis is possible.

A formal utility analysis of Alg. 22 is presented in App. D.1.13. Empirical evaluation of (η, δ) - preservation is presented in App. D.1.14.

D.1.8 Discussion on Properties of Mallows Mechanism

Property 3. *For group assignment $\mathscr{G}$, a mechanism $\mathscr{A}(\cdot)$ that shuffles according to a permutation sampled from the Mallows model $\mathbb{P}_{\theta,\mathfrak{d}}(\cdot)$, satisfies $(\alpha, \mathscr{G})$-$d_\sigma$privacy where*

$$\Delta(\sigma_0 : \mathfrak{d}, \mathscr{G}) = \max_{(\sigma,\sigma') \in N_{\mathscr{G}}} |\mathfrak{d}(\sigma_0\sigma, \sigma_0) - \mathfrak{d}(\sigma_0\sigma', \sigma_0)|$$

and

$$\alpha = \theta \cdot \Delta(\sigma_0 : \mathfrak{d}, \mathscr{G})$$

We refer to $\Delta(\sigma_0 : \mathfrak{d}, \mathscr{G})$ as the sensitivity of the rank-distance measure $\mathfrak{d}(\cdot)$

Proof. Consider two permutations of the initial sequence $\mathbf{y}$, $\sigma_1(\mathbf{y}), \sigma_2(\mathbf{y})$ that are neighboring w.r.t. some group $G_i \in \mathscr{G}$, $\sigma_1 \approx_{G_i} \sigma_2$. Additionally consider any fixed released shuffled sequence $\mathbf{z}$. Let Σ_1, Σ_2 be the set of permutations that turn $\sigma_1(\mathbf{y}), \sigma_2(\mathbf{y})$ into $\mathbf{z}$, respectively:

$$\Sigma_1 = \{\sigma \in \mathrm{S}_n : \sigma\sigma_1(\mathbf{y}) = \mathbf{z}\}$$
$$\Sigma_2 = \{\sigma \in \mathrm{S}_n : \sigma\sigma_2(\mathbf{y}) = \mathbf{z}\} \quad .$$

In the case that $\{y\}$ consists entirely of unique values, Σ_1, Σ_2 will each contain exactly one permutation, since only one permutation can map $\sigma_i(\mathbf{y})$ to $\mathbf{z}$.

Lemma 15. *For each permutation $\sigma_1' \in \Sigma_1$ there exists a permutation in $\sigma_2' \in \Sigma_2$ such that*

$$\sigma_1' \approx_{G_i} \sigma_2' \quad .$$

Proof follows from the fact that — since only the elements $j \in G_i$ differ in $\sigma_1(\mathbf{y})$ and $\sigma_2(\mathbf{y})$ — only those elements need to differ to achieve the same output permutation. In other words, we may define σ_1', σ_2' at all inputs $i \notin G_i$ identically, and then define all inputs $i \in G_i$ differently as needed. As such, they are neighboring w.r.t. G_i.

Recalling that Alg. 1 applies σ_0^{-1} to the sampled permutation, we must sample $\sigma_0\sigma_1'$ (for some $\sigma_1' \in \Sigma_1$) for the mechanism to produce $\mathbf{z}$ from $\sigma_1(\mathbf{y})$. Formally, since $\sigma_1'\sigma_1(\mathbf{y}) = \mathbf{z}$ we must sample $\sigma_0\sigma_1'$ to get $\mathbf{z}$ since we are going to apply σ_0^{-1} to the sampled permutation.

$$\Pr\left[\mathscr{A}\left(\sigma_1(\mathbf{y})\right) = \mathbf{z}\right] = \mathbb{P}_{\theta,\delta}\left(\sigma_0\sigma', \sigma' \in \Sigma_1 : \sigma_0\right)$$
$$\Pr\left[\mathscr{A}\left(\sigma_2(\mathbf{y})\right) = \mathbf{z}\right] = \mathbb{P}_{\theta,\delta}\left(\sigma_0\sigma', \sigma' \in \Sigma_2 : \sigma_0\right)$$

Taking the odds, we have

$$\begin{aligned}
\frac{\mathbb{P}_{\theta,\delta}\left(\sigma_0\sigma', \sigma' \in \Sigma_1 : \sigma_0\right)}{\mathbb{P}_{\theta,\delta}\left(\sigma_0\sigma'', \sigma'' \in \Sigma_2 : \sigma_0\right)} &= \frac{\sum_{\sigma' \in \Sigma_1} \mathbb{P}_{\Theta,\delta}(\sigma_0\sigma' : \sigma_0)}{\sum_{\sigma'' \in \Sigma_2} \mathbb{P}_{\Theta,\delta}(\sigma_0\sigma'' : \sigma_0)} \\
&= \frac{\sum_{\sigma' \in \Sigma_1} e^{-\theta\delta(\sigma_0\sigma', \sigma_0)}}{\sum_{\sigma'' \in \Sigma_2} e^{-\theta\delta(\sigma_0\sigma'', \sigma_0)}} \\
&\leq \frac{e^{-\theta\delta(\sigma_0\sigma_a, \sigma_0)}}{e^{-\theta\delta(\sigma_0\sigma_b, \sigma_0)}} \\
&\leq e^{\theta|\delta(\sigma_0\sigma_a, \sigma_0) - \delta(\sigma_0\sigma_b, \sigma_0)|} \\
&\leq e^{\theta\Delta}
\end{aligned}$$

where

$$\sigma_a = \arg\max_{\sigma' \in \Sigma_1} e^{-\theta \delta(\sigma_0 \sigma', \sigma_0)} \text{ and}$$

$$\sigma_a = \arg\min_{\sigma'' \in \Sigma_2} e^{-\theta \delta(\sigma_0 \sigma'', \sigma_0)} .$$

Therefore, setting $\alpha = \theta \cdot \Delta$, we achieve $(\alpha, \mathscr{G})$-$d_\sigma$privacy. ∎

Property 4. *The sensitivity of a rank-distance is an increasing function of the width* $\omega_{\mathscr{G}}^{\sigma_0}$. *For instance, for Kendall's* τ *distance* $\delta_\tau(\cdot)$, *we have* $\Delta(\sigma_0 : \delta_\tau, \mathscr{G}) = \frac{\omega_{\mathscr{G}}^{\sigma_0}(\omega_{\mathscr{G}}^{\sigma_0}+1)}{2}$.

To show the sensitivity of Kendall's τ, we make use of its triangle inequality.

Proof. Recall from the proof of the previous property that the expression $\delta(\sigma, \sigma_0) = \delta\big(\sigma_0 \sigma, \sigma_0\big)$, where δ is the actual rank distance measure e.g. Kendall's τ. As such, we require that

$$\big|\delta(\sigma_0 \sigma_a, \sigma_0) - \delta(\sigma_0 \sigma_b, \sigma_0)\big| \leq \frac{\omega_{\mathscr{G}}^{\sigma_0}(\omega_{\mathscr{G}}^{\sigma_0} + 1)}{2}$$

for any pair of permutations $(\sigma_a, \sigma_b) \in N_{\mathscr{G}}$.

For any group $G_i \in \mathscr{G}$, let $W_i \subseteq n$ represent the smallest contiguous subsequence of indices in σ_0 that contains all of G_i.

For instance, if $\sigma_0 = [2,4,6,8,1,3,5,7]$ and $G_i = \{2,6,8\}$, then $W_i = \{2,4,6,8\}$. Then the group width width is $\omega_i = |W_i| - 1 = 3$. Now consider two permutations neighboring w.r.t. G_i, $\sigma_a \approx_{G_i} \sigma_b$, so only the elements of G_i are shuffled between them. We want to bound

$$\big|\delta(\sigma_0 \sigma_a, \sigma_0) - \delta(\sigma_0 \sigma_b, \sigma_0)\big|$$

For this, we use a pair of triangle inequalities:

$$\mathfrak{d}(\sigma_0\sigma_a, \sigma_0\sigma_b) \geq \mathfrak{d}(\sigma_0\sigma_a, \sigma_0) - \mathfrak{d}(\sigma_0\sigma_b, \sigma_0) \quad \& \quad \mathfrak{d}(\sigma_0\sigma_a, \sigma_0\sigma_b) \geq \mathfrak{d}(\sigma_0\sigma_b, \sigma_0) - \mathfrak{d}(\sigma_0\sigma_a, \sigma_0)$$

so,

$$\left|\mathfrak{d}(\sigma_0\sigma_a, \sigma_0) - \mathfrak{d}(\sigma_0\sigma_b, \sigma_0)\right| \leq \mathfrak{d}(\sigma_0\sigma_a, \sigma_0\sigma_b)$$

Since $\sigma_0\sigma_a$ and $\sigma_0\sigma_b$ only differ in the contiguous subset W_i, the largest number of discordant pairs between them is given by the maximum Kendall's τ distance between two permutations of size $\omega_i + 1$:

$$|\mathfrak{d}(\sigma_0\sigma_a, \sigma_0\sigma_b)| \leq \frac{\omega_i(\omega_i+1)}{2}$$

Since $\omega_{\mathscr{G}}^{\sigma_0} \geq \omega_i$ for all $G_i \in \mathscr{G}$, we have that

$$\Delta(\sigma_0 : \mathfrak{d}, \mathscr{G}) \leq \frac{\omega_{\mathscr{G}}^{\sigma_0}(\omega_{\mathscr{G}}^{\sigma_0}+1)}{2}$$

∎

D.1.9 Hardness of Computing The Optimum Reference Permutation

Theorem 16. *The problem of finding the optimum reference permutation, i.e.,* $\sigma_0^* = \arg\min_{\sigma \in \mathrm{S}_n} \omega_{\mathscr{G}}^{\sigma}$ *is NP-hard.*

Proof. We start with the formal representation of the problem as follows.

Optimum Reference Permutation Problem. Given n subsets $\mathscr{G} = \{G_i \in 2^{[n]}, i \in [n]\}$, find the permutation $\sigma_0^* = \arg\min_{\sigma \in \mathrm{S}_n} \omega_{\mathscr{G}}^{\sigma}$.

Now, consider the following job-shop scheduling problem.

Job Shop Scheduling. There is one job J with n operations $o_i, i \in [n]$ and n machines such that o_i needs to run on machine M_i. Additionally, each machine has a sequence dependent processing time p_i. Let S be the sequence till There are n subsets $S_i \subseteq [n]$, each corresponding to a set of operations that need to occur in contiguous machines, else the processing times incur penalty as follows. Let p_i denote the processing time for the machine running the i-th operation scheduled. Let $\mathbb{S}_i$ be the prefix sequence with i schedulings. For instance, if the final scheduling is 1 3 4 5 9 8 10 6 7 2 then $\mathbb{S}_4 = 1345$. Additionally, let $P^j_{\mathbb{S}_i}$ be the shortest subsequence such of $\mathbb{S}_i$ such that it contains all the elements in $S_j \cap \{\mathbb{S}_i\}$. For example for $S_1 = \{3,5,7\}$, $P^1_{\mathbb{S}_4} = 345$.

$$p_i = \max_{i \in [n]} (|P^j_{\mathbb{S}_i}| - |S_j \cap \{\mathbb{S}_i\}|) \tag{D.7}$$

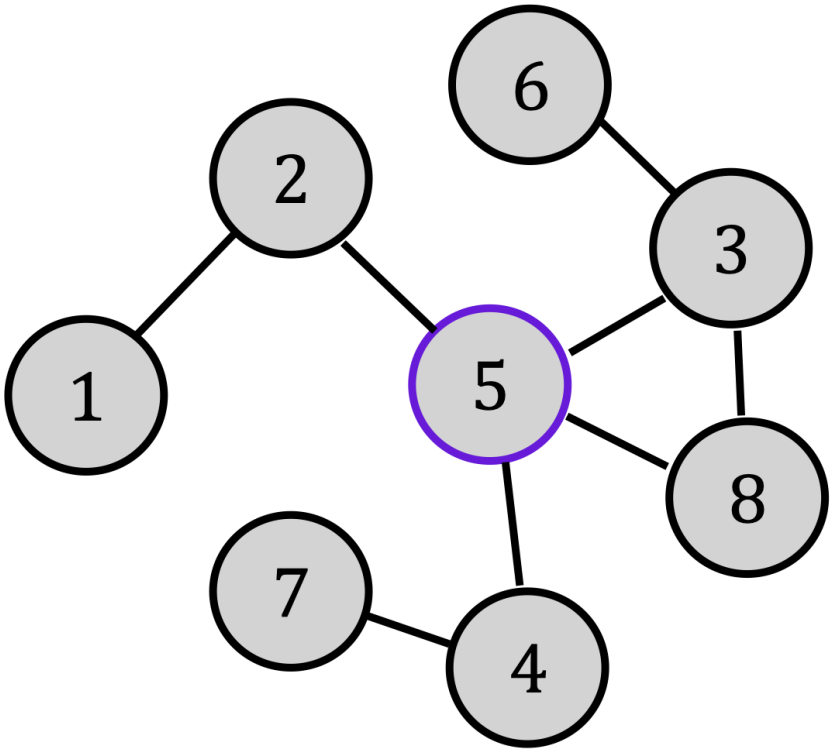
The objective is to find a scheduling for J such that it minimizes the makespan, i.e., the completion time of the job. Note that $p_n = \max_i p_i$, hence the problem reduces to minimizing p_n.

Lemma 17. *The aforementioned job shop scheduling problem with sequence-dependent processing time is NP-hard.*

Proof. Consider the following instantiation of the sequence-dependent job shop scheduling problem where the processing time is given by p_i=$p_{i-1} + w_{kl}, p_1 = 0$ where $\mathbb{S}_i[i-1] = k$, $\mathbb{S}_i[i] = l$ and $w_{ij}, j \in S_i$ represents some associated weight. This problem is equivalent to the travelling salesman problem (TSP) [10] and is therefore, NP-hard. Thus, our aforementioned job shop scheduling problem is also clearly NP-hard. ∎

Reduction: Let the n subsets S_i correspond to the groups in $\mathcal{G}$. Clearly, minimizing $\omega^\sigma_\mathcal{G}$ minimizes p_n. Hence, the optimal reference permutation gives the solution to the scheduling problem as well.

∎



(a) Group graph

	$\sigma_0(1)$	$\sigma_0(2)$	$\sigma_0(3)$	$\sigma_0(4)$	$\sigma_0(5)$	$\sigma_0(6)$	$\sigma_0(7)$	$\sigma_0(8)$
	5	2	3	8	4	1	6	7
queue pos'n 0	2	3	8	4	1	6	7	
1	3	8	4	1	6	7		
2	8	4	1	6	7			
3	4	1	6					

(b) BFS reference permutation σ_0

Figure D.1. Illustration of Alg. 1

D.1.10 Illustration of Alg. 1

We now provide a small-scale step-by-step example of how Alg. 1 operates.

Fig. D.1a is an example of a grouping $\mathscr{G}$ on a dataset of $n = 8$ elements. The group of DO_i includes i and its neighbors. For instance, $G_8 = \{8,3,5\}$. To build a reference permutation, Alg. 1 starts at the index with the largest group, $i = 5$ (highlighted in purple), with $G_5 = \{5,2,3,8,4\}$. As shown in Figure D.1b, the σ_0 is then constructed by following a BFS traversal from $i = 5$. Each $j \in G_5$ is visited, queuing up the neighbors of each $j \in G_5$ that haven't been visited along the way, and so on. The algorithm completes after the entire graph has been visited.

The goal is to produce a reference permutation in which the width of each group in the reference permutation ω_i is small. In this case, the width of the largest group G_5 is as small as it can be $\omega_5 = 5 - 1 = 4$. However, the width of $G_4 = \{4,5,7\}$ is the maximum possible since $\sigma^{-1}(5) = 1$ and $\sigma^{-1}(7) = 8$, so $\omega_4 = 7$. This is difficult to avoid when the maximum group size is large as compared to the full dataset size n. Realistically, we expect n to be significantly larger, leading to relatively smaller groups.

With the reference permutation in place, we compute the sensitivity:

$$\begin{aligned}\Delta(\sigma_0 : \delta, \mathscr{G}) &= \frac{\omega_4(\omega_4+1)}{2} \\ &= 28\end{aligned}$$

Which lets us set $\theta = \frac{\alpha}{28}$ for any given α privacy value. To reiterate, lower θ results in more randomness in the mechanism.

We then sample the permutation $\dot{\boldsymbol{\sigma}} = \mathbb{P}_{\theta,\delta}(\sigma_0)$. Suppose

$$\dot{\boldsymbol{\sigma}} = [3\ 2\ 5\ 4\ 8\ 1\ 7\ 6]$$

Then, the released $\mathbf{z}$ is given as

$$\begin{aligned}\mathbf{z} = \sigma^* &= \sigma^{-1}\dot{\boldsymbol{\sigma}}(\mathbf{y}) \\ &= [y_1\ y_2\ y_5\ y_8\ y_3\ y_7\ y_6\ y_4]\end{aligned}$$

One can think of the above operation as follows. What was 5 in the reference permutation ($\sigma_0(1) = 5$) is 3 in the sampled permutation ($\dot{\boldsymbol{\sigma}}(1) = 3$). So, index 5 corresponding to DO_5 now holds DO_3's noisy data y_3. As such, we shuffle mostly between members of the same group, and minimally between groups.

The runtime of this mechanism is dominated by the Repeated Insertion Model sampler

[53], which takes $\mathcal{O}(n^2)$ time. It is very possible that there are more efficient samplers available, but RIM is a standard and simple to implement for this first proposed mechanism. Additionally, the majority of this is spent computing sampling parameters which can be stored in advanced with $\mathcal{O}(n^2)$ memory. Furthermore, sampling from a Mallows model with some reference permutation σ_0 is equivalent to sampling from a Mallows model with the identity permutation and applying it to σ_0. As such, permutations may be sampled in advanced, and the runtime is dominated by computation of σ_0 which takes $\mathcal{O}(|V|+|E|)$ time (the number of vertices and edges in the graph).

A future direction could be exploring even better heuristics for computing σ_0. One possibility could be based on ranked enumeration of the permutations [49, 48].

D.1.11 Proof of Thm. 6

Theorem 6 *Alg. 1 is* $(\alpha,\mathcal{G})$*-*d_σ *private.*

Proof. The proof follows from Prop. 3. Having computed the sensitivity of the reference permutation σ_0, Δ, and set $\theta = \alpha/\Delta$, we are guaranteed by Property 3 that shuffling according to the permutation $\dot{\boldsymbol{\sigma}}$ guarantees $(\alpha,\mathcal{G})$-$d_\sigma$privacy.

∎

.

D.1.12 Proof of Thm. 7

Theorem 7 *Alg. 1 satisfies* $(\alpha',\mathcal{G}')$*-*d_σ*privacy for any group assignment* $\mathcal{G}'$ *where* $\alpha' = \alpha\frac{\Delta(\sigma_0:\mathfrak{d},\mathcal{G}')}{\Delta(\sigma_0:\mathfrak{d},\mathcal{G})}$

Proof. Recall from Property 3 that we satisfy $(\alpha,\mathcal{G})$ d_σ-privacy by setting $\theta = \alpha/\Delta(\sigma_0:\mathfrak{d},\mathcal{G})$.

Given alternative grouping $\mathscr{G}'$ with sensitivity $\Delta(\sigma_0 : \mathfrak{d}, \mathscr{G}')$, this same mechanism provides

$$\begin{aligned}\alpha' &= \frac{\theta}{\Delta(\sigma_0 : \mathfrak{d}, \mathscr{G}')} \\ &= \frac{\alpha/\Delta(\sigma_0 : \mathfrak{d}, \mathscr{G})}{\Delta(\sigma_0 : \mathfrak{d}, \mathscr{G}')} \\ &= \alpha \frac{\Delta(\sigma_0 : \mathfrak{d}, \mathscr{G}')}{\Delta(\sigma_0 : \mathfrak{d}, \mathscr{G})}\end{aligned}$$

∎

D.1.13 Formal Utility Analysis of Alg. 1

Theorem 18. *For a given set $S \subset [n]$ and Hamming distance metric, $\mathfrak{d}_H(\cdot)$, Alg. 1 is (η, δ)-preserving for $\delta = \frac{1}{\psi(\theta, \mathfrak{d}_H)} \sum_{h=2k+1}^{n} (e^{-\theta \cdot h} \cdot c_h)$ where $k = \lceil (1-\eta) \cdot |S| \rceil$ and c_h is the number of permutations with hamming distance h from the reference permutation that do not preserve η% of S and is given by*

$$\begin{aligned}c_h = \sum_{j=k+1}^{\max(l_s, \lfloor h/2 \rfloor)} \binom{l_s}{j} \cdot \binom{n-l_s}{j} \cdot \Bigg[\sum_{i=0}^{\min(l_s - j, h-2j)} \binom{l_s - j}{i} \\ \cdot \binom{i+j}{j} \cdot f(i,j) \cdot \binom{n-l_s-j}{h-2j-i} \cdot f(h-2j-i, j)! \Bigg]\end{aligned}$$

$$\begin{aligned}f(i,0) &= !i, f(0,q) = q! \\ f(i,j) &= \sum_{q=0}^{\min(i,j)} \left[\binom{i}{q} \cdot \binom{j}{j-q} \cdot j! \cdot f(i-q, q) \right] \\ l_s &= |S|, k = (1-\eta) \cdot l_s, !n = \lfloor \frac{n!}{e} + \frac{1}{2} \rfloor\end{aligned}$$

Proof. Let $l_s = |S|$ denote the size of the set S and $k = \lceil (1-\eta) \cdot l_S \rceil$ denote the maximum number of correct values that can be missing from S. Now, for a given permutation $\sigma \in \mathrm{S}_n$, let h denote its Hamming distance from the reference permutation σ_0, i.e, $h = \mathfrak{d}_H(\sigma, \sigma_0)$. This means that σ

and σ_0 differ in h indices. Now, h can be analysed in the the following two cases,

Case I. $h \leq 2k+1$

For $(1-\eta)$ fraction of indices to be removed from S, we need at least $k+1$ indices from S to be replaced by $k+1$ values from outside S. This is clearly not possible for $h \leq 2k+1$. Hence, here $c_h = 0$.

Case II. $h > 2k$

For the following analysis we consider we treat the permutations as strings (multi-digit numbers are treated as a single string character). Now, Let $\mathbb{S}_{\sigma_0}$ denote the non-contiguous substring of σ_0 such that it consists of all the elements of S, i.e.,

$$|\mathbb{S}| = l_S \tag{D.8}$$

$$\forall i \in [l_S], \mathbb{S}_{\sigma_0}[i] \in S \tag{D.9}$$

Let $\mathbb{S}_\sigma$ denote the substring corresponding to the positions occupied by $\mathbb{S}_{\sigma_0}$ in σ. Formally,

$$|\mathbb{S}_\sigma| = l_S \tag{D.10}$$

$$\forall i \in [l_S], \mathbb{S}_{\sigma_0}[i] = \sigma(\sigma_0^{-1}(\mathbb{S}_{\sigma_0}[i])) \tag{D.11}$$

For example, for $\sigma_0 = (1\,2\,3\,5\,4\,7\,8\,10\,9\,6), \sigma = (1\,3\,2\,7\,8\,5\,4\,6\,10\,9)$ and $S = \{2,4,5,8\}$, we have $\mathbb{S}_{\sigma_0} = 2548$ and $S_\sigma = 3784$ where $h = \mathfrak{d}_H(\sigma, \sigma_0) = 9$. Let $\{\mathbb{S}_\sigma\}$ denote the set of the elements of string $\mathbb{S}_\sigma$. Let A be the set of characters in $\mathbb{S}_\sigma$ such that they do not belong to S, i.e, $A = \{\mathbb{S}_\sigma[i] | \mathbb{S}_\sigma[i] \notin S, i \in [l_S]\}$. Let B be the set of characters in $\mathbb{S}_\sigma$ that belong to S but differ from $\mathbb{S}_{\sigma_0}$ in position, i.e., $B = \{\mathbb{S}_\sigma[i] | \mathbb{S}_\sigma[i] \in S, \mathbb{S}_\sigma[i] \neq \mathbb{S}_{\sigma_0}[i], i \in [l_S]\}$. Additionally, let $C = S - \{\mathbb{S}_\sigma\}$. For instance, in the above example, $A = \{3,7\}, B = \{4,8\}, C = \{2,5\}$. Now consider an initial arrangement of $p+m$ distinct objects that are subdivided into two types – p objects of Type A and m objects of Type B. Let $f(p,m)$ denote the number of permutations of these $p+m$ objects such that the m Type B objects can occupy any position but no object of Type A can occupy its

original position. For example, for $f(p,0)$ this becomes the number of derangements [2] denoted as $!p = \lfloor \frac{p!}{e} + \frac{1}{2} \rfloor$. Therefore, $f(|B|,|A|)$ denotes the number of permutations of $\mathbb{S}_\sigma$ such that $\mathfrak{d}_H(\mathbb{S}_{\sigma_0}, \mathbb{S}_\sigma) = |A| + |B|$. This is because if elements of B are allowed to occupy their original position then this will reduce the Hamming distance.

Now, let $\bar{\mathbb{S}}_\sigma$ ($\bar{\mathbb{S}}_{\sigma_0}$) denote the substring left out after extracting from $\mathbb{S}_\sigma$ ($\mathbb{S}_{\sigma_0}$) from σ (σ_0). For example, $\bar{\mathbb{S}}_\sigma = 1256109$ and $\bar{\mathbb{S}}_{\sigma_0} = 1371096$ in the above example. Let D be the set of elements outside of S and A that occupy different positions in $\bar{\mathbb{S}}_\sigma$ and $\bar{\mathbb{S}}_{\sigma_0}$ (thereby contributing to the hamming distance), i.e., $D = \{\bar{\mathbb{S}}_{\sigma_0[i]} | \bar{\mathbb{S}}_{\sigma_0[i]} \notin S, \bar{\mathbb{S}}_{\sigma_0[i]} \neq \bar{\mathbb{S}}_{\sigma[i]}, i \in [n - l_S]\}$. For instance, in the above example $D = \{9,6,10\}$. Hence, $h = \mathfrak{d}_H(\sigma, \sigma_0) = |A| + |B| + |C| + |D|$ and clearly $f(|D|,|C|)$ represents the number of permutations of $\bar{\mathbb{S}}_\sigma$ such that $\mathfrak{d}_H(\bar{\mathbb{S}}_\sigma, \bar{\mathbb{S}}_{\sigma_0}) = |C| + |D|$. Finally, we have

$$c_h = \sum_{j=k+1}^{\max(l_s, \lfloor h/2 \rfloor)} \underbrace{\binom{l_s}{j}}_{\text{\# ways of selecting set } C} \cdot \underbrace{\binom{n-l_s}{j}}_{\text{\# ways of selecting set } A} \cdot \Bigg[\sum_{i=0}^{\min(l_s - j, h-2j)} \underbrace{\binom{l_s - j}{i}}_{\text{\# ways of selecting set } B} \cdot f(i,j) \cdot \underbrace{\binom{n - l_s - j}{h - 2j - i}}_{\text{\# ways of selecting set } D} \cdot f(h - 2j - i, j) \Bigg]$$

Now, for $f(i,j)$ let E be the set of original positions of Type A that are occupied by Type B objects in the resulting permutation. Additionally, let F be the set of the original positions of Type B objects that are still occupied by some Type B object. Clearly, Type B objects can occupy these $|E| + |F| = m$ in any way they like. However, the type A objects can only result in

$f(p-q,q)$ permutations. Therefore, $f(p,m)$ is given by the following recursive function

$$f(p,0) = !p$$

$$f(0,m) = m!$$

$$f(p,m) = \sum_{q=0}^{\min p,m} \Bigg(\underbrace{\binom{p}{q}}_{\text{\# ways of selecting set } E} \cdot \underbrace{\binom{m}{m-q}}_{\text{\# ways of selecting set } F} \cdot m! \cdot f(p-q,q) \Bigg)$$

Thus, the total probability of failure is given by

$$\delta = \frac{1}{\psi(\theta, \mathfrak{d}_H)} \sum_{h=2k+2}^{n} (e^{-\theta \cdot h} \cdot c_h) \tag{D.12}$$

∎

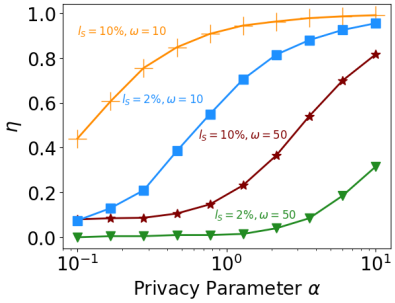
D.1.14 Additional Experimental Details

Evaluation of (η, δ)-preservation

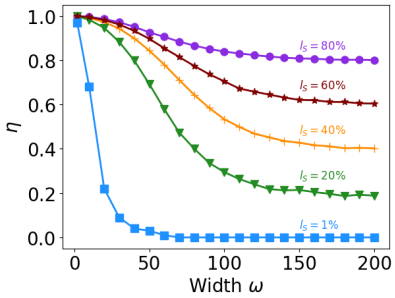
In this section, we evaluate the characteristics of the (η, δ)-preservation for Kendall's τ distance $\mathfrak{d}_\tau(\cdot,\cdot)$.

Each sweep of Fig. D.2 fixes $\delta = 0.01$, and observes η. We consider a dataset of size $n = 10K$ and a subset S of size l_S corresponding to the indices in the middle of the reference permutation σ_0 (the actual value of the reference permutation is not significant for measuring preservation). For the rest of the discussion, we denote the width of a permutation by ω for notational brevity. For each value of the independent axis, we generate 50 trials of the permutation σ from a Mallows model with the appropriate θ (given the ω and α parameters). We then report the largest η (fraction of subset preserved) that at least 99% of trials satisfy.

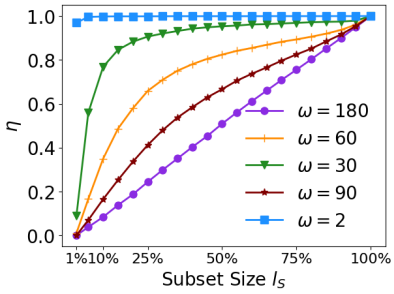
In Fig. D.2a, we see that preservation is highest for higher α and increases gradually with declining width ω and increasing subset size l_s.



(a) Variation with α



(b) Variation with ω; $\alpha = 3$



(c) Variation with l_S; $\alpha = 3$

Figure D.2. (η, δ)-Preservation Analysis

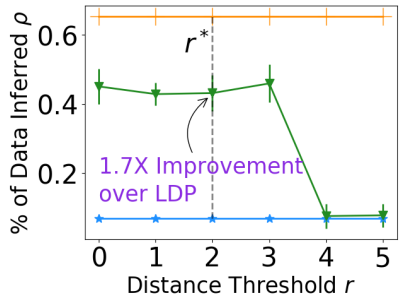
Fig. D.2b demonstrates that preservation declines with increasing width. Δ increases quadratically with width ω for δ_τ, resulting in declining θ and increasing randomness. We also see that larger subset sizes result in a more gradual decline in η. This is due to the fact that the worst-case preservation (uniform random shuffling) is better for larger subsets. i.e. we cannot do worse than 80% preservation for a subset that is 80% of indices.

Finally, Fig. D.2c demonstrates how preservation grows rapidly with increasing subset size. For large widths, we are nearly uniformly randomly permuting, so preservation will equal the size of the subset relative to the dataset size. For smaller widths, we see that preservation offers diminishing returns as we grow subset size past some critical l_s. For $\omega = 30$, we see that subset sizes much larger than a quarter of the dataset gain little in preservation.

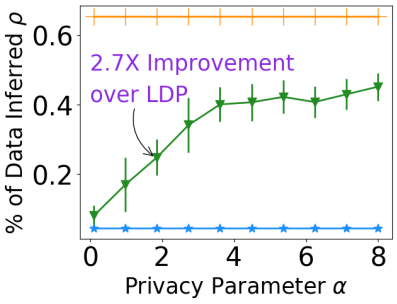
Adult Dataset

D.1.15 Additional Related Work

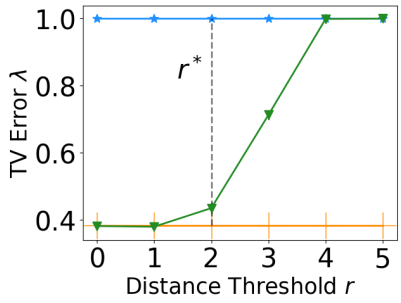
In this section, we discuss the relevant existing work.



(a) *Adult*: Attack



(b) *Adult*: Attack (α)



(c) *Adult*: Learnability

Figure D.3. Adult dataset experiments

The anonymization of noisy responses to improve differential privacy was first proposed by Bittau et al. [22] who proposed a principled system architecture for shuffling. This model was formally studied later in [65, 45]. Erlingsson et al. [65] showed that for arbitrary ε-LDP randomizers, random shuffling results in privacy amplification. Cheu et al. [45] formally defined the shuffle DP model and analyzed the privacy guarantees of the binary randomized response in this model. The shuffle DP model differs from our approach in two ways. First, it focuses completely on the DP guarantee. The privacy amplification is manifested in the from of a lower ε (roughly a factor of $\sqrt{n}$) when viewed in an alternative DP model known as the central DP model. [65, 45, 12, 72, 22, 11]. However, our result caters to local inferential privacy. Second, the shuffle model involves an uniform random shuffling of the entire dataset. In contrast, our approach the granularity at which the data is shuffled is tunable which delineates a threshold for the learnability of the data.

A steady line of work has sudied the inferential privacy setting [98, 103, 81, 47, 60, 153]. Kifer et al. [103] formally studied privacy degradation in the face of data correlations and

later proposed a privacy framework, Pufferfish [105, 146, 91], for analyzing inferential privacy. Subsequently, several other privacy definitions have also been proposed for the inferential privacy setting [117, 172, 42, 188, 15]. For instance, Gehrke et al. proposed a zero-knowledge privacy [79, 78] which is based on simulation semantics. Bhaskar et al. proposed noiseless privacy [17, 86] by restricting the set of prior distributions that the adversary may have access to. A recent work by Zhang et al. proposes attribute privacy [184] which focuses on the sensitive properties of a whole dataset. In another recent work, Ligett et al. study a relaxation of DP that accounts for mechanisms that leak some additional, bounded information about the database [116]. Some early work in local inferential privacy include profile-based privacy [80] by Gehmke et al. where the problem setting comes with a graph of data generating distributions, whose edges encode sensitive pairs of distributions that should be made indistinguishable. In another work by Kawamoto et al., the authors propose distribution privacy [100] – local differential privacy for probability distributions. The major difference between our work and prior research is that we provide local inferential privacy through a new angle – data shuffling.

Finally, older works such as k-anonymity [149], l-diversity [123], and Anatomy [165] and other [163, 150, 170, 46, 54] have studied the privacy risk of non-sensitive auxiliary information, or 'quasi identifiers' (QIs). In practice, these works focus on the setting of dataset release, where we focus on dataset collection. As such, QIs can be manipulated and controlled, whereas we place no restriction on the amount or type of auxiliary information accessible to the adversary, nor do we control it. Additionally, our work offers each individual formal inferential guarantees against informed adversaries, whereas those works do not. We emphasize this last point since formalized guarantees are critical for providing meaningful privacy definitions. As established by Kifer and Lin in *An Axiomatic View of Statistical Privacy and Utility* (2012), privacy definitions ought to at least satisfy post-processing and convexity properties which our formal definition does.

D.1.16 Evaluation of Heuristic

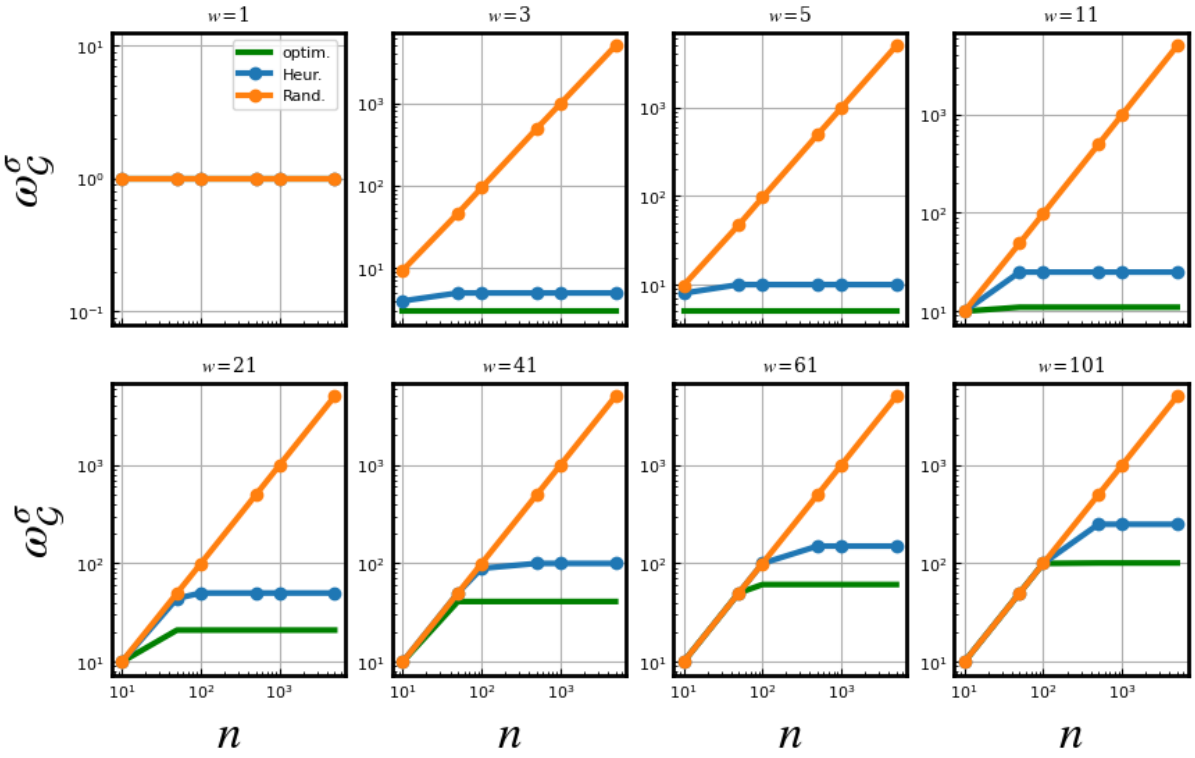


Figure D.4. Comparison of our heuristic's performance with that of an optimal reference permutation σ_0^*. An optimal σ_0^* is generated with every group having size w. A graph is generated from this optimal σ_0^* from which our heuristic (blue) attempts to reconstruct the optimal permutation. For baselining, the performance of a random σ_0 selection is plotted (orange). We observe that at worst, our heuristic picks a reference permutation with width $2.5\times$ that of the optimal reference permutation (green). See Section 4.4.4 for definition of terms.

Algorithm 22 is designed to find a reference permutation σ_0 with low width $\omega_{\mathcal{G}}^{\sigma}$ w.r.t. the given grouping $\mathcal{G}$. A low width is desirable, since it leads to low sensitivity $\Delta(\sigma_0 : \delta, \mathcal{G})$, which in turn leads to higher dispersion parameter $\theta = \alpha/\Delta$, and thus less randomness over permutations (higher utility). Theorem 16 proves that computing the optimal reference permutation (minimum width) is NP-hard. As such, we propose a BFS-based heuristic.

Comparison with optimal reference permutation

To demonstrate the value of the heuristic used in Alg. 22, we provide two evaluations of its performance. For our first evaluation, we compare the performance of our heuristic BFS reference

permutation selection (σ_0) with that of the optimal reference permutation and that of a random reference permutation. As identified by Theorem 16, finding the optimal reference permutation for a given grouping $\mathscr{G}$ is NP-hard. For these experiments, we first create an optimal reference permutation, where each group $G_i \in \mathscr{G}$ is equally sized w and maximally compact. The optimal width, $\omega^{\sigma}_{\mathscr{G}}$, is then $\min(n, w)$. We then generate a graph from this optimal reference permutation. Finally, we run the BFS reference permutation computation described in Alg. 22 attempting to approximate the optimal σ^*_0, and compute its width.

To compare with a naive approach, we also plot the performance of a randomly chosen reference permutation. We expect the maximum width across groups $\omega^{\sigma}_{\mathscr{G}}$ to be large for this technique. If one of the n groups has a single entry low (near 0) in σ_0 and a single entry high (near n) in σ_0, the width will be near n. The random baseline is averaged over 10 trials with a 1 standard deviation envelope plotted (but difficult to see, since the variance is low).

Figure D.4 depicts our findings. Each plot has a different group size w, listed at the top, used in the optimal reference permutation. We find that the random baseline (orange) consistently chooses a reference permutation such that $\omega^{\sigma}_{\mathscr{G}}$ is near n, as expected. Our method (blue), on the other hand, closely tracks the optimal solution (green). We find that in the worst case, our algorithm's solution has a width $\leq 2.5\times$ larger than the optimal. Note that for $r = 0$ (upper left), all methods trivially have a width of one, since the corresponding graph has no edges. While there may be room for improvement, we find this to be sufficient for the present work.

Performance on randomly generated graphs

For our second evaluation, we observe how well our BFS heuristic (in Algorithm 22) performs on randomly generated graphs. Here, we sample n points uniformly on the unit interval. We then say that the ith point's group, G_i, consists of all other points within r of it. As r increases, so does the groups size. Since computing the optimal reference permutation is NP-hard (Theorem 16), we do not show the optimal width. Instead, we show a loose lower bound of the optimal width (green) by plotting the average group size for a given r (recall that the width is greater than or equal to the largest group size, so we expect this to be a loose lower bound, solely for

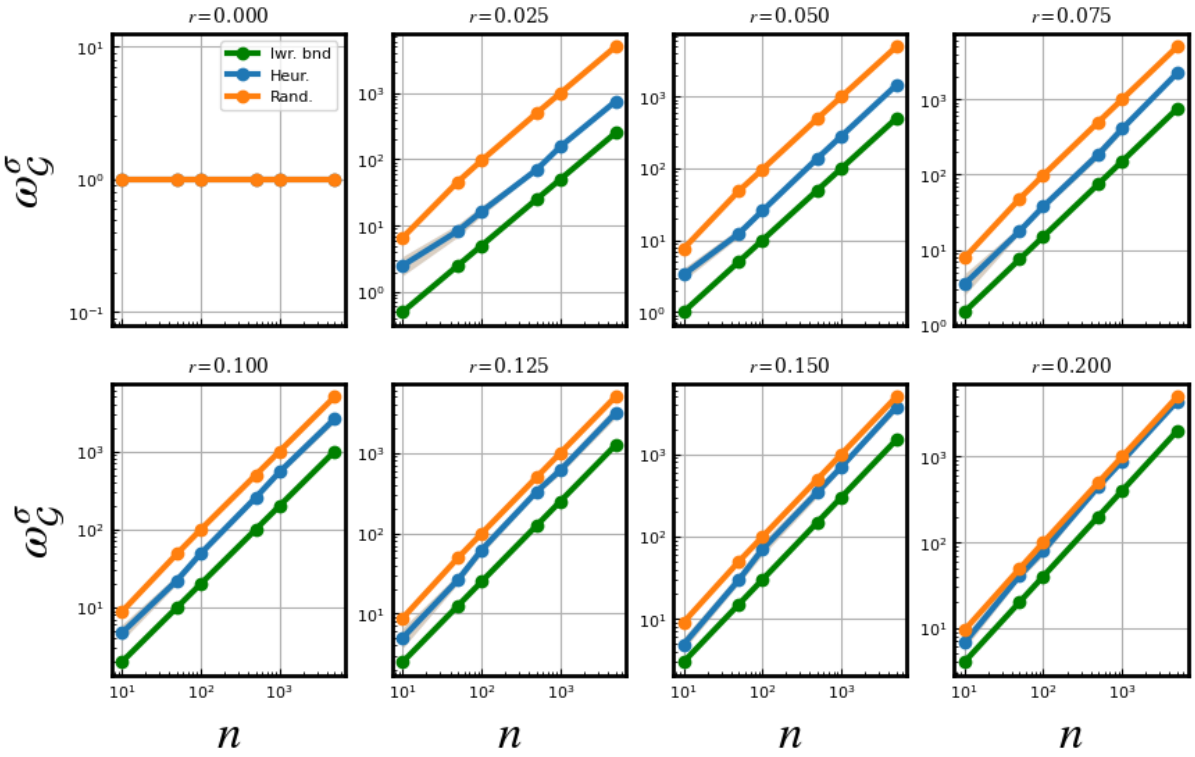


Figure D.5. example of our heuristic's performance on randomly generated graphs. As r increases, so does the connectivity of the random graphs and the average group size (green). As shown by Theorem 16, computing the optimal $\omega^{\sigma}_{\mathscr{G}}$ is NP-hard. The average group size (green) in $\mathscr{G}$ is a loose lower bound on the optimal $\omega^{\sigma}_{\mathscr{G}}$. The performance of a random σ_0 assignment (orange) is also plotted for reference. Our heuristic BFS algorithm (blue) consistently outperforms the random baseline.

reference). For comparison, we evaluate the performance of a random σ_0 choice as well. For both of these methods, we run 10 trials of generating a random graph (and picking a random σ_0) at each value of n and plot the mean along with a 1 standard deviation envelope, which is difficult to see due to low variance.

Figure D.5 depicts our findings. We find that — across values of n and r — our heuristic (blue) significantly outperforms the random baseline (orange). Additionally, we observe the trends we expect. For a low r values, our heuristic BFS algorithm chooses a σ_0 with width close to the lower bound (green) of the optimal width $\omega^{\sigma}_{\mathscr{G}}$. As r increases, the graph become significantly more connected. Both the lower bound and our heuristic move closer to the width

of the random baseline. Note that for $r = 0$ (upper left), all methods trivially have a width of one, since the corresponding graph has no edges. Ultimately, these findings indicate that our heuristic for computing σ_0 significantly outperforms a naive random choice, and follows the same trend as the lower bound of the optimal.

Appendix E

E.1 Appendix

For documented code demonstrating our SDP mechanisms used to generate the plots of **Figure 5.2** please visit our repo: https://github.com/casey-meehan/location_trace_privacy

The following sections will include proofs of results, derivations of algorithms, and explanations of experimental procedures.

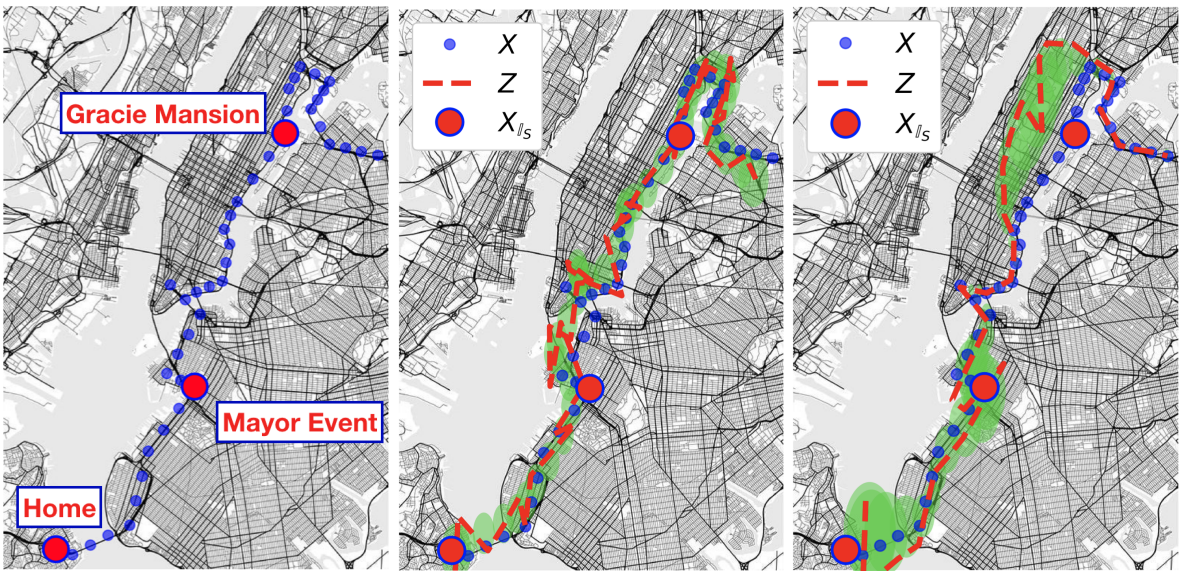
E.1.1 Illustrations

NYC Mayoral Staff Member Location Trace

Juxtaposition of Mechanisms' Covariance Matrices

The following figures aim to illustrate the difference between the covariance matrices used in the experimental baselines (indep./uniform and indep./concentrated) and those chosen by our SDP algorithms for both the RBF and periodic prior. Note that here we presume the different dimensions of location to be independent and — by Corollary E.1.0.1 — are able to treat a 2d location trace as two 1d traces. As such, the following examples are demonstrating mechanism covariance matrices and additive noise samples used for either a single dimension of location data (for RBF kernel) or for the one dimension of temperature data (for periodic kernel).

The first figure **(a)** shows the covariance of the Approach C baselines used in the experiments. The second figure **(b)** shows the covariance of our SDP mechanisms for the RBF

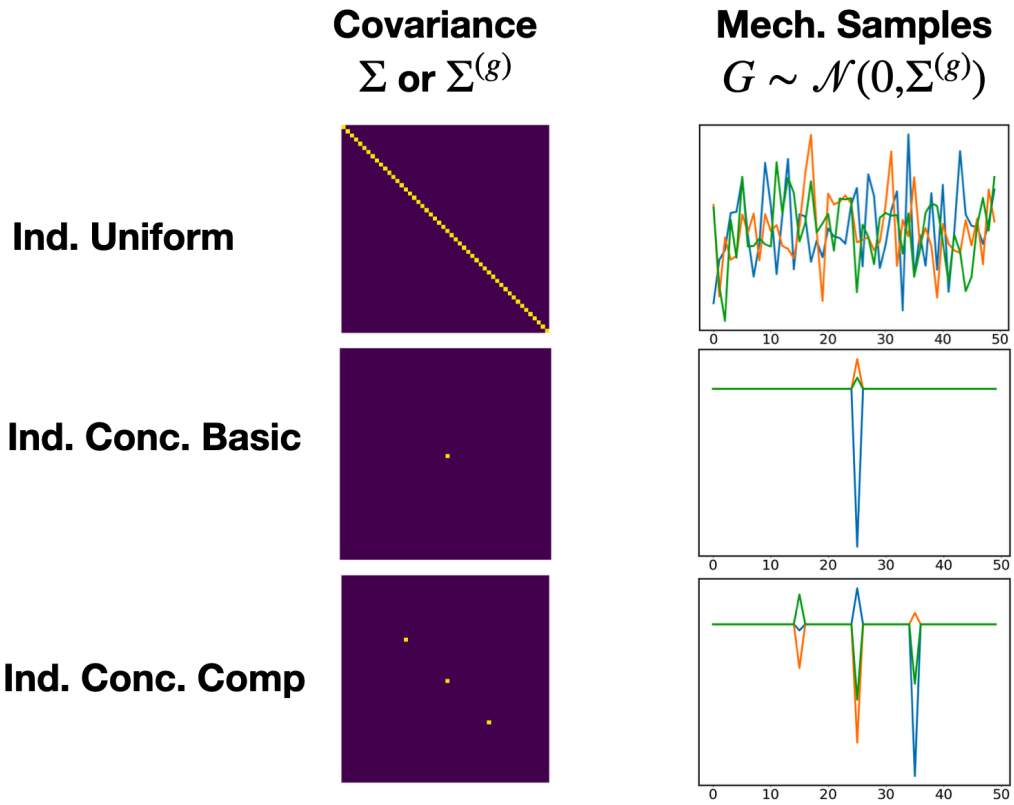


(a) (b) (c)

Figure E.1. Example of sensitive location trace of NYC mayoral staff member exposed by [156]. (b) and (c) depict the posterior uncertainty (green) $P_{\mathscr{A},\mathscr{P}}(X_i|Z)$ for each 2d location. (a) depicts three sensitive times (red with blue outline): Gracie Mansion (Mayor's home), an event on Staten Island that the mayor attended, and finally the staff member's home on long island. (b) provides an example of Approach C: adding independent Gaussian noise to each location (red dotted line). A GP posterior still maintains high confidence within a small radius along the trace, including at the sensitive times. (c) provides an example of the optimized noise of Multiple Secrets of identical aggregate MSE as (b). By focusing *correlated* noise around the three sensitive times, there is high uncertainty at sensitive times and high confidence elsewhere.

kernel used on location data. The third figure **(c)** shows the covariance of our SDP mechanisms for the periodic kernel used for temperature data.

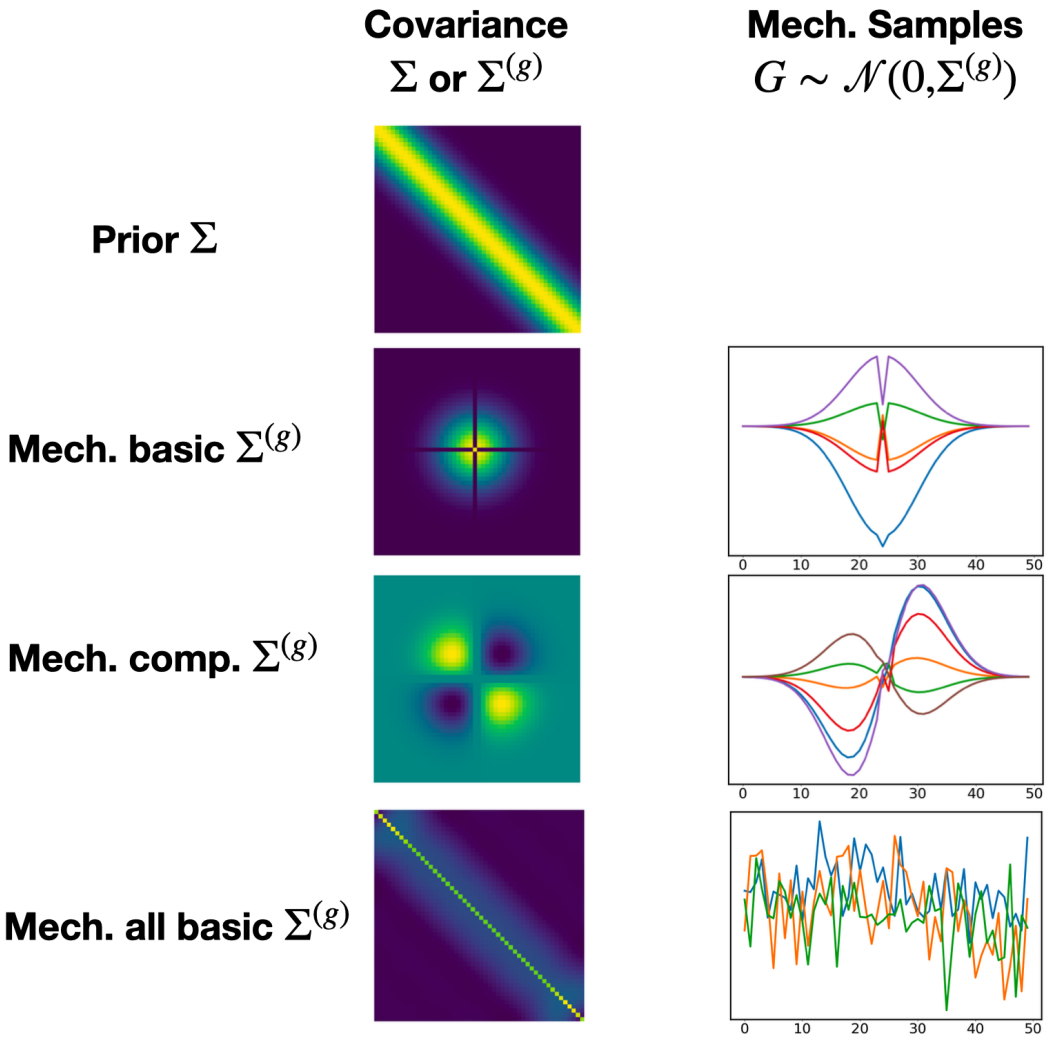
In each figure the covariance matrix is depicted as a heat map with warmer colors indicating higher values (normalized to largest and smallest value in the covariance matrix). The drawn noise samples G are plotted against their time index. So, the sequence of plotted (x,y) values is $\big[(1,G_1),(2,G_2),\ldots,(n,G_n)\big]$, where $n=50$ for the RBF case and $n=48$ for the periodic case.



(a) Covariance matrices and mechanism samples for the baselines used in experiments.

The first figure demonstrates the uniform approach that distributes the independent Gaussian noise budget along the entire trace, regardless of $\mathbb{I}_S$.

The second and third show the concentrated approach that allocates the entire noise budget to only the sensitive locations in $\mathbb{I}_S$: first for a basic secret (one location) and then for a compound secret of 3 evenly spaced locations.

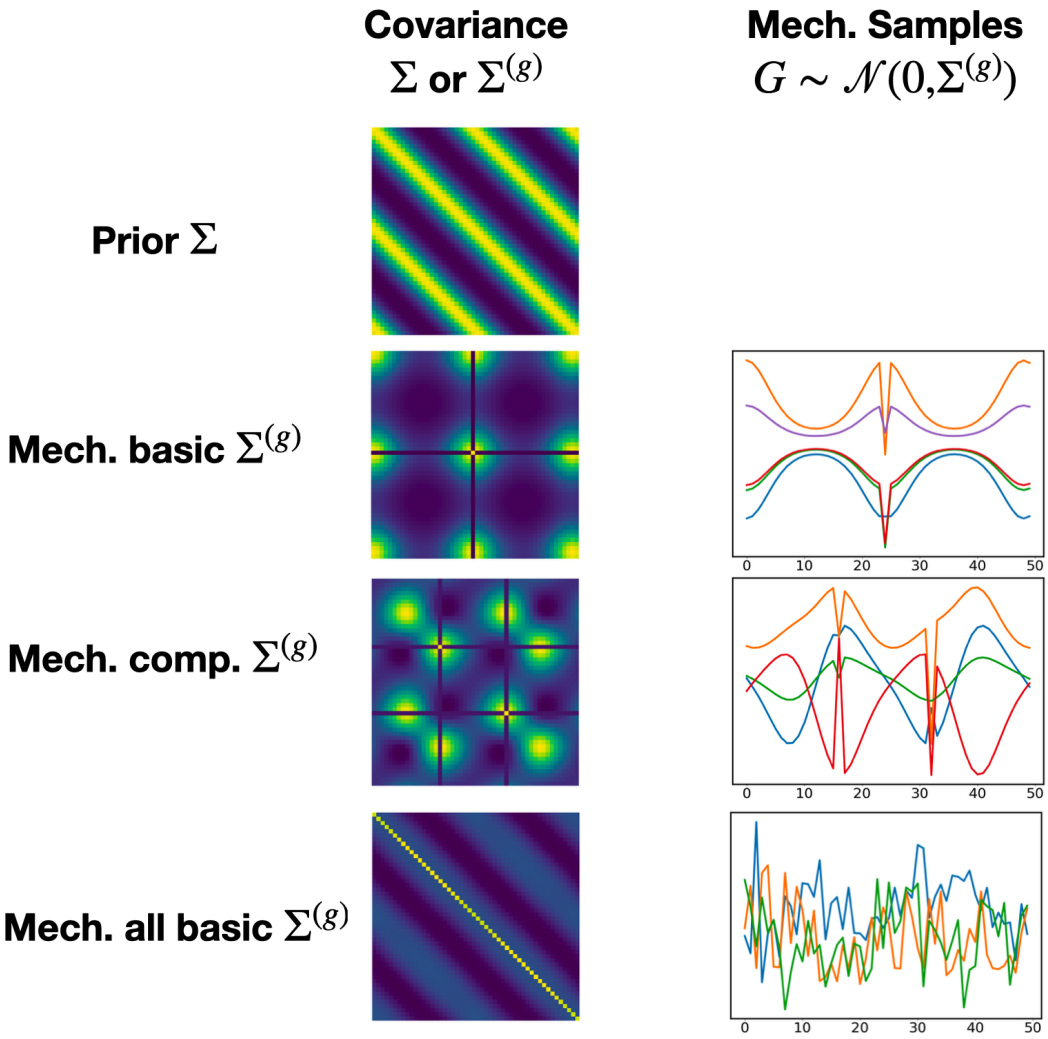


(b) Covariance matrices and mechanism samples for the median RBF prior ($l_{\text{eff}} \approx 6$).

The first noise mechanism (Mech. basic) demonstrates the covariance matrix chosen by SDP_A for a basic secret of a single location X_i in the middle of the trace. The uncorrelated dot in the middle of the covariance matrix, $\Sigma^{(g)}_{ii}$, represents the independent noise G_i added at the sensitive location to mitigate *direct* loss. To mitigate *inferential* loss, the SDP optimizes the remainder of the matrix to be positively correlated with maximum variance allocated to locations near X_i in time. This thwarts GP inference of the true location at time t_i.

The second mechanism (Mech. comp.) depicts the covariance chosen by SDP_A to protect a compound secret of two adjacent locations in the trace (visible as the uncorrelated '+' through the middle consuming 2 rows/columns). Recall that a compound secret ought to protect directional information: *did the user visit B first and then A, or A and then B?* That is precisely what this mechanism does by randomizing the angle of approach to the two locations in the middle with positively and negatively correlated noise. Also note that the SDP does not allocate a large share of noise budget to the actual locations themselves. This highlights the fact that protecting a compound secret does not protect its constituent basic secrets.

The third and final mechanism (Mech. all basic) is the noise covariance chosen by SDP_B in the Multiple Secrets algorithm. To protect all basic secrets with a utility constraint, the SDP converges to a mechanism that looks similar to the uniform baseline. However, this mechanism adds a subtle degree of off-diagonal correlation along with greater noise power towards the beginning and end of the trace. The off-diagonal correlation is noticeable when the samples are compared to those of the uniform baseline in the previous figure. While this change appears to be minor, it makes a significant change in the posterior confidence of a GP adversary (as seen in **Figure 5.2c**).



(c) Covariance matrices and mechanism samples for the median periodic prior ($l_{\text{eff}} \approx 1.1$), and a period of half the trace length.

The first noise mechanism (Mech. Basic) shows the covariance chosen by SDP_A to protect a single location (temperature) in the middle of the trace. As in the RBF case, significant noise power is allocated to the sensitive location itself, X_i, to limit *direct* privacy loss. However, the noise added to the remainder of the trace is significantly different. It is tailored to thwart inference by a periodic prior, wherein the location one period away has correlation 1.

The second noise mechanism (Mech. comp.) shows the covariance chosen by SDP_A to protect a compound secret of two locations, X_i, X_j, 16 timesteps apart (not quite a full period). Here, we see the SDP randomize the phase of the additive noise such that periodic inference cannot tell directional information like $X_i > X_j$ or vice versa.

The third noise mechanism (Mech. all basic) is identical to the all basic secrets mechanism chosen for the RBF case above, except using a periodic prior Σ. The mechanism chosen looks similar to the uniform baseline, except with slightly periodic off-diagonal correlation imitating the prior covariance. Additionally, noise power is mitigated towards the middle and ends of the trace. Again, **Figure 5.2g** indicates that this subtle change makes a significant difference in thwarting Bayesian adversaries.

E.1.2 Proof of results

Proof of Theorem 5.3.2

Theorem 5.3.2 Prior-Posterior Gap: *An (ε, λ)-CIP mechanism with conditional prior class Θ guarantees that for any event O on sanitized trace Z*

$$\left| \log \frac{P_{\mathscr{P},\mathscr{A}}(s_i | Z \in O)}{P_{\mathscr{P},\mathscr{A}}(s_j | Z \in O)} - \log \frac{P_{\mathscr{P}}(s_i)}{P_{\mathscr{P}}(s_j)} \right| \leq \varepsilon'$$

for any $\mathscr{P} \in \Theta$ with probability $\geq 1 - \delta$ over draws of $Z|X_{\mathbb{I}_S} = s_i$ or $Z|X_{\mathbb{I}_S} = s_j$, where ε' and δ are related by

$$\varepsilon' = \varepsilon + \frac{\log 1/\delta}{\lambda - 1} .$$

This holds under the condition that $Z|X_{\mathbb{I}_S} = s_i$ and $Z|X_{\mathbb{I}_S} = s_j$ have identical support.

Proof. This result makes use of a Rényi divergence property identified in [128]:

Lemma 19. *Let $\mathscr{P}, \mathscr{Q}$ be two distributions on X of identical support such that*

$$\max \left\{ D_\lambda \begin{pmatrix} P_{\mathscr{P}}(X) \\ P_{\mathscr{Q}}(X) \end{pmatrix}, D_\lambda \begin{pmatrix} P_{\mathscr{Q}}(X) \\ P_{\mathscr{P}}(X) \end{pmatrix} \right\} \leq \varepsilon$$

Then for any event O,

$$P_{\mathscr{P}}(X \in O) \leq \max \left\{ e^{\varepsilon'} P_{\mathscr{Q}}(X \in S), \delta \right\}$$

and

$$P_{\mathscr{Q}}(X \in O) \leq \max \left\{ e^{\varepsilon'} P_{\mathscr{P}}(X \in S), \delta \right\}$$

where

$$\varepsilon' = \varepsilon + \frac{\log 1/\delta}{\lambda - 1}$$

CIP guarantees that for all $\mathscr{P} \in \Theta$ and all discriminative pairs $(s_i, s_j) \in \mathscr{S}_{\text{pairs}}$ (which also includes (s_j, s_i))

$$D_\lambda \left(\begin{matrix} P_{\mathscr{P},\mathscr{A}}(Z|X_{\mathbb{I}_S} = s_i) \\ P_{\mathscr{P},\mathscr{A}}(Z|X_{\mathbb{I}_S} = s_j) \end{matrix} \right) \leq \varepsilon$$

and thus by Lemma 19 we have for any event O on Z

$$P_{\mathscr{P},\mathscr{A}}(Z \in O|X_{\mathbb{I}_S} = s_i) \leq \max \left\{ e^{\varepsilon'} P_{\mathscr{P},\mathscr{A}}(Z \in O|X_{\mathbb{I}_S} = s_j), \delta \right\}$$

and

$$P_{\mathscr{P},\mathscr{A}}(Z \in O|X_{\mathbb{I}_S} = s_j) \leq \max \left\{ e^{\varepsilon'} P_{\mathscr{P},\mathscr{A}}(Z \in O|X_{\mathbb{I}_S} = s_i), \delta \right\}$$

As such, given that $X_{\mathbb{I}_S} = s_i$ the probability of some event $\{Z \in W\}$ such that

$$P_{\mathscr{P},\mathscr{A}}(Z \in W|X_{\mathbb{I}_S} = s_i) \geq e^{\varepsilon'} P_{\mathscr{P},\mathscr{A}}(Z \in W|X_{\mathbb{I}_S} = s_j)$$

is no more than δ. The same is true swapping s_j for s_i. So, over draws of $Z|X_{\mathbb{I}_S} = s_i$ or $Z|X_{\mathbb{I}_S} = s_j$ we have that

$$\frac{P_{\mathscr{P},\mathscr{A}}(Z \in O|X_{\mathbb{I}_S} = s_i)}{P_{\mathscr{P},\mathscr{A}}(Z \in O|X_{\mathbb{I}_S} = s_j)} \leq e^{\varepsilon'} \quad \text{and} \quad \frac{P_{\mathscr{P},\mathscr{A}}(Z \in O|X_{\mathbb{I}_S} = s_j)}{P_{\mathscr{P},\mathscr{A}}(Z \in O|X_{\mathbb{I}_S} = s_i)} \leq e^{\varepsilon'}$$

with probability $\geq 1-\delta$, which is equivalent to the statement that

$$-\varepsilon' \leq \log \frac{P_{\mathscr{P},\mathscr{A}}(Z \in O | X_{\mathbb{I}_S} = s_i)}{P_{\mathscr{P},\mathscr{A}}(Z \in O | X_{\mathbb{I}_S} = s_j)} \leq \varepsilon'$$
$$\left| \log \frac{P_{\mathscr{P},\mathscr{A}}(s_i | Z \in O)}{P_{\mathscr{P},\mathscr{A}}(s_j | Z \in O)} - \log \frac{P_{\mathscr{P}}(s_i)}{P_{\mathscr{P}}(s_j)} \right| \leq \varepsilon'$$

∎

Proof of Lemma 8

Lemma 8 (CIP loss for additive mechanisms) *For an additive noise mechanism, a fully dependent trace as in **Figure 5.1b**, and any prior $\mathscr{P}$ on X the CIP loss may be expressed as*

$$D_\lambda \begin{pmatrix} P_{\mathscr{A},\mathscr{P}}(Z|X_{\mathbb{I}_S} = s_i) \\ P_{\mathscr{A},\mathscr{P}}(Z|X_{\mathbb{I}_S} = s_j) \end{pmatrix} = \sum_{i \in \mathbb{I}_S} \left[D_\lambda \begin{pmatrix} P_{\mathscr{A}}(Z_i|X_i = s_i) \\ P_{\mathscr{A}}(Z_i|X_i = s_j) \end{pmatrix} \right] + D_\lambda \begin{pmatrix} P_{\mathscr{A},\mathscr{P}}(Z_{\mathbb{I}_U}|X_{\mathbb{I}_S} = s_i) \\ P_{\mathscr{A},\mathscr{P}}(Z_{\mathbb{I}_U}|X_{\mathbb{I}_S} = s_j) \end{pmatrix}$$

Proof.

$$\begin{aligned}
D_\lambda \begin{pmatrix} P_{\mathscr{A},\mathscr{P}}(Z|X_{\mathbb{I}_S} = x_s) \\ P_{\mathscr{A},\mathscr{P}}(Z|X_{\mathbb{I}_S} = x'_s) \end{pmatrix} &= D_\lambda \begin{pmatrix} P_{\mathscr{A}}(Z_{\mathbb{I}_S}|X_{\mathbb{I}_S} = x_s) P_{\mathscr{A},\mathscr{P}}(Z_{\mathbb{I}_U}|X_{\mathbb{I}_S} = x_s) \\ P_{\mathscr{A}}(Z_{\mathbb{I}_S}|X_{\mathbb{I}_S} = x'_s) P_{\mathscr{A},\mathscr{P}}(Z_{\mathbb{I}_U}|X_{\mathbb{I}_S} = x'_s) \end{pmatrix} && (1) \\
&= D_\lambda \begin{pmatrix} P_{\mathscr{A}}(Z_{\mathbb{I}_S}|X_{\mathbb{I}_S} = x_s) \\ P_{\mathscr{A}}(Z_{\mathbb{I}_S}|X_{\mathbb{I}_S} = x'_s) \end{pmatrix} + D_\lambda \begin{pmatrix} P_{\mathscr{A},\mathscr{P}}(Z_{\mathbb{I}_U}|X_{\mathbb{I}_S} = x_s) \\ P_{\mathscr{A},\mathscr{P}}(Z_{\mathbb{I}_U}|X_{\mathbb{I}_S} = x'_s) \end{pmatrix} && (2) \\
&= D_\lambda \begin{pmatrix} \prod_{i \in \mathbb{I}_S} P_{\mathscr{A}}(Z_i|X_i = x_i) \\ \prod_{i \in \mathbb{I}_S} P_{\mathscr{A}}(Z_i|X_i = x'_i) \end{pmatrix} + D_\lambda \begin{pmatrix} P_{\mathscr{A},\mathscr{P}}(Z_{\mathbb{I}_U}|X_{\mathbb{I}_S} = x_s) \\ P_{\mathscr{A},\mathscr{P}}(Z_{\mathbb{I}_U}|X_{\mathbb{I}_S} = x'_s) \end{pmatrix} && (3) \\
&= \sum_{i \in \mathbb{I}_S} \left[D_\lambda \begin{pmatrix} P_{\mathscr{A}}(Z_i|X_i = x_i) \\ P_{\mathscr{A}}(Z_i|X_i = x'_i) \end{pmatrix} \right] + D_\lambda \begin{pmatrix} P_{\mathscr{A},\mathscr{P}}(Z_{\mathbb{I}_U}|X_{\mathbb{I}_S} = x_s) \\ P_{\mathscr{A},\mathscr{P}}(Z_{\mathbb{I}_U}|X_{\mathbb{I}_S} = x'_s) \end{pmatrix} && (4)
\end{aligned}$$

Where line (1) uses the conditional independence seen in the graphical model of **Figure 5.1**. Line (2) is due to the fact that the two terms in line (1) are conditionally independent, allowing for separating into the sum of two separate divergences (which is an easily verifiable property of Rényi divergence evident from its definition in Equation 5.1). Line (3) is again from the conditional independence between the Z_i for each $i \in \mathbb{I}_S$ when conditioned on $X_{\mathbb{I}_S}$. Line (4) uses the same property of Rényi divergence used in Line (2): the terms in the product are conditionally

independent allowing for the separation into the sum of multiple divergences.

∎

Proof of Theorem 5.3.2

Thoerem 5.3.2 Robustness to Prior Misspecification *Mechanism $\mathscr{A}$ satisfies $\varepsilon(\lambda)$-CIP for prior class Θ. Suppose the finite mean true distribution $\mathscr{Q}$ is not in Θ. The CIP loss of $\mathscr{A}$ against prior $\mathscr{Q}$ is bounded by*

$$D_\lambda\left(\begin{matrix}P_{\mathscr{A},\mathscr{Q}}(Z|X_{\mathbb{I}_S}=s_i)\\P_{\mathscr{A},\mathscr{Q}}(Z|X_{\mathbb{I}_S}=s_j)\end{matrix}\right)\leq\varepsilon'(\lambda)$$

where

$$\varepsilon'(\lambda)=\frac{\lambda-\frac{1}{2}}{\lambda-1}\,\Delta(2\lambda)+\Delta(4\lambda-3)+\frac{2\lambda-\frac{3}{2}}{2\lambda-2}\,\varepsilon(4\lambda-2)$$

and where $\Delta(\lambda)$ is

$$\inf_{\mathscr{P}\in\Theta}\sup_{s_i\in\mathscr{S}}\max\left\{D_\lambda\left(\begin{matrix}P_{\mathscr{P}}(X_{\mathbb{I}_U}|X_{\mathbb{I}_S}=s_i)\\P_{\mathscr{Q}}(X_{\mathbb{I}_U}|X_{\mathbb{I}_S}=s_i)\end{matrix}\right),D_\lambda\left(\begin{matrix}P_{\mathscr{Q}}(X_{\mathbb{I}_U}|X_{\mathbb{I}_S}=s_i)\\P_{\mathscr{P}}(X_{\mathbb{I}_U}|X_{\mathbb{I}_S}=s_i)\end{matrix}\right)\right\}$$

Proof. By 'finite mean' distribution $\mathscr{Q}$, we mean that all conditionals of $\mathscr{Q}$ given some $X_{\mathbb{I}_S}$ have finite mean. Since a conditional prior class contains conditionals of one distribution with any offset (any mean value), this guarantees that $\Delta(\lambda)$ is achieved for some $\mathscr{P}\in\Theta$. Intuitively, this prevents the pathological case of $\inf_{\mathscr{P}\in\Theta}$ being a limit as the mean of $\mathscr{P}\to\infty$, only asymptotically approaching $\Delta(\lambda)$. If the mean of $\mathscr{Q}$ is finite, then the closest $\mathscr{P}\in\Theta$ (in Rényi divergence) must also have finite mean, since any mean is attainable in a conditional prior class Θ.

With this in mind, we make use of the following triangle inequality provided in [128]:

Lemma 20. *For distributions $\mathscr{P}$, $\mathscr{Q}$, $\mathscr{R}$ on X with common support we have*

$$D_\lambda\begin{pmatrix}P_{\mathscr{P}}(X)\\P_{\mathscr{Q}}(X)\end{pmatrix} \leq \frac{\lambda-\frac{1}{2}}{\lambda-1}D_{2\lambda}\begin{pmatrix}P_{\mathscr{P}}(X)\\P_{\mathscr{R}}(X)\end{pmatrix} + D_{2\lambda-1}\begin{pmatrix}P_{\mathscr{R}}(X)\\P_{\mathscr{Q}}(X)\end{pmatrix}$$

In our case, we assume that the mechanism $\mathscr{A}$ gives $Z|X_{\mathbb{I}_S} = x_s$ identical support for all $\mathbb{I}_S, x_s$. Using this, we have

$$D_\lambda\begin{pmatrix}P_{\mathscr{A},\mathscr{Q}}(Z_{\mathbb{I}_U}|X_{\mathbb{I}_S}=x_s)\\P_{\mathscr{A},\mathscr{Q}}(Z_{\mathbb{I}_U}|X_{\mathbb{I}_S}=x'_s)\end{pmatrix} \leq \frac{\lambda-\frac{1}{2}}{\lambda-1}D_{2\lambda}\begin{pmatrix}P_{\mathscr{A},\mathscr{Q}}(Z_{\mathbb{I}_U}|X_{\mathbb{I}_S}=x_s)\\P_{\mathscr{A},\mathscr{P}}(Z_{\mathbb{I}_U}|X_{\mathbb{I}_S}=x_s)\end{pmatrix} + D_{2\lambda-1}\begin{pmatrix}P_{\mathscr{A},\mathscr{P}}(Z_{\mathbb{I}_U}|X_{\mathbb{I}_S}=x_s)\\P_{\mathscr{A},\mathscr{Q}}(Z_{\mathbb{I}_U}|X_{\mathbb{I}_S}=x'_s)\end{pmatrix}.$$

By a data processing inequality, the divergence of the first term is bounded by $\Delta(2\lambda)$ and the blue term may be bounded by a second application of the triangle inequality:

$$D_{2\lambda-1}\begin{pmatrix}P_{\mathscr{A},\mathscr{P}}(Z_{\mathbb{I}_U}|X_{\mathbb{I}_S}=x_s)\\P_{\mathscr{A},\mathscr{Q}}(Z_{\mathbb{I}_U}|X_{\mathbb{I}_S}=x'_s)\end{pmatrix} \leq \frac{2\lambda-\frac{3}{2}}{2\lambda-2}D_{4\lambda-2}\begin{pmatrix}P_{\mathscr{A},\mathscr{P}}(Z_{\mathbb{I}_U}|X_{\mathbb{I}_S}=x_s)\\P_{\mathscr{A},\mathscr{P}}(Z_{\mathbb{I}_U}|X_{\mathbb{I}_S}=x'_s)\end{pmatrix} + D_{4\lambda-3}\begin{pmatrix}P_{\mathscr{A},\mathscr{P}}(Z_{\mathbb{I}_U}|X_{\mathbb{I}_S}=x'_s)\\P_{\mathscr{A},\mathscr{Q}}(Z_{\mathbb{I}_U}|X_{\mathbb{I}_S}=x'_s)\end{pmatrix}$$

The first divergence is bounded by $\varepsilon(4\lambda-2)$ and the second divergence is bounded by $\Delta(4\lambda-3)$. Putting all this together we have the following upper bound

$$D_\lambda\begin{pmatrix}P_{\mathscr{A},\mathscr{Q}}(Z_{\mathbb{I}_U}|X_{\mathbb{I}_S}=x_s)\\P_{\mathscr{A},\mathscr{Q}}(Z_{\mathbb{I}_U}|X_{\mathbb{I}_S}=x'_s)\end{pmatrix} \leq \frac{\lambda-\frac{1}{2}}{\lambda-1}\,\Delta(2\lambda)+\Delta(4\lambda-3)+\frac{2\lambda-\frac{3}{2}}{2\lambda-2}\,\varepsilon(4\lambda-2)$$

Proof of Theorem 5.3.3

Theorem 5.3.3 CIP loss bound for GP conditional priors: *Let Θ be a GP conditional prior class. Let Σ be the covariance matrix for X produced by its kernel function. Let $\mathscr{S}$ be the basic or compound secret associated with $\mathbb{I}_S$, and S be the number of unique times in $\mathbb{I}_S$. The mechanism $\mathscr{A}(X) = X + G = Z$, where $G \sim \mathscr{N}(\mathbf{0}, \Sigma^{(g)})$, then satisfies (ε, λ)-Conditional*

Inferential Privacy $(\mathscr{S}_{pairs}, r, \Theta)$*, where*

$$\varepsilon \leq \frac{\lambda}{2} S r^2 \Big(\frac{1}{\sigma_s^2} + \alpha^* \Big)$$

where σ_s^2 *is the variance of each* $G_i \in G_{\mathbb{I}_S}$ *(diagonal entries of* $\Sigma_{ss}^{(g)}$*) and* α^* *is the maximum eigenvalue of* $\Sigma_{eff} = \big(\Sigma_{us}\Sigma_{ss}^{-1}\big)^\intercal \big(\Sigma_{u|s} + \Sigma_{uu}^{(g)}\big)^{-1} \big(\Sigma_{us}\Sigma_{ss}^{-1}\big)$.

Proof. Again, the conditional prior class Θ is defined by a kernel function $i, j \to \mathrm{Cov}(i, j)$, which – given the indices of the trace X – induces a covariance matrix Σ between all X_i, X_j. In practice, when the sampling rate of locations is non-uniform the kernel function may use the time-stamps of the points in the trace to assign high correlation to X_i that are close in time and low correlation to X_i that are far apart in time. Of course, correlation between X_i that are different dimension (e.g. latitude and longitude) must be designed for the given application and may be completely independent. The kernel function can encode this as well.

Recall from Equation 5.1 that the Rényi divergence between two mean-shifted multivariate normal distributions, $\mathscr{P}_1 = \mathscr{N}(\mu_1, \Sigma)$ and $\mathscr{P}_2 = \mathscr{N}(\mu_2, \Sigma)$ is

$$D_\lambda \begin{pmatrix} \mathscr{P}_1 \\ \mathscr{P}_2 \end{pmatrix} = \frac{\lambda}{2} (\mu_1 - \mu_2)^\intercal \Sigma^{-1} (\mu_1 - \mu_2)$$

Now, for any prior $\mathscr{P} \in \Theta$, we have that $X \sim \mathscr{N}(\mu, \Sigma)$ for some μ and for Σ defined by the kernel function. Again, $G \sim \mathscr{N}(\mathbf{0}, \Sigma^{(g)})$. $\mathbb{I}_S$ encodes the indices of a single location basic secret or a multi-location compound secret. Then, the divergence to bound for (ε, λ)-CIP$(\mathscr{S}_{\text{pairs}}, r, \Theta)$ is

$$D_\lambda \begin{pmatrix} P_{\mathscr{A},\mathscr{P}}(Z | X_{\mathbb{I}_S} = s_i) \\ P_{\mathscr{A},\mathscr{P}}(Z | X_{\mathbb{I}_S} = s_j) \end{pmatrix}$$

for any

$$(s_i, s_j) \in \mathscr{S}_{\text{pairs}} = \{(x_s, x'_s) : \|x_s - x'_s\|_2 \leq 2r\}$$

if $\mathbb{I}_S$ encodes a basic secret, or for any

$$(s_i, s_j) \in \mathscr{S}_{\text{pairs}} = \left\{ \left(\{x_{s1}, x_{s2}, \dots\}, \{x'_{s1}, x'_{s2}, \dots\} \right) : \|x_{sk} - x'_{sk}\|_2 \leq 2r, \forall\, k \right\}$$

if $\mathbb{I}_S$ encodes a compound secret. A discriminative pair (s_i, s_j) is two real valued vectors $\in \mathbb{R}^{|\mathbb{I}_S|}$, representing two hypotheses about the true values of $X_{\mathbb{I}_S}$. We denote the m^{th} element as s_{im}, s_{jm}. Let $f : \mathbb{I}_S \to [|\mathbb{I}_S|]$ be a mapping from each index $w \in \mathbb{I}_S$ to its corresponding position in the vector s_i or s_j (where the value of X_w is hypothesized). By Lemma 8, the divergence can be written as

$$D_\lambda \begin{pmatrix} P_{\mathscr{A},\mathscr{P}}(Z|X_{\mathbb{I}_S} = s_i) \\ P_{\mathscr{A},\mathscr{P}}(Z|X_{\mathbb{I}_S} = s_j) \end{pmatrix} = \sum_{w \in \mathbb{I}_S} \left[D_\lambda \begin{pmatrix} P_{\mathscr{A}}(Z_w|X_w = s_{i f(w)}) \\ P_{\mathscr{A}}(Z_w|X_w = s_{j f(w)}) \end{pmatrix} \right] + D_\lambda \begin{pmatrix} P_{\mathscr{A},\mathscr{P}}(Z_{\mathbb{I}_U}|X_{\mathbb{I}_S} = x_s) \\ P_{\mathscr{A},\mathscr{P}}(Z_{\mathbb{I}_U}|X_{\mathbb{I}_S} = x'_s) \end{pmatrix}$$

where $P_{\mathscr{A}}(Z_w|X_w = x) = \mathscr{N}(x, \sigma_s^2)$ for all $w \in \mathbb{I}_S$. Recall from the statement of the Theorem that we assume the diagonal entries of Σ_{ss} all equal some value σ_s^2: we add the same noise variance to each point in the secret set, which is optimal under MSE constraints. Additionally, note that for the hypothesis $X_{\mathbb{I}_S} = x_s$, we know the distribution of $X_{\mathbb{I}_U}|X_{\mathbb{I}_S} = x_s \sim \mathscr{N}(\mu_{u|s}, \Sigma_{u|s})$, where $\mu_{u|s} = \mu_u + \Sigma_{us}\Sigma_{ss}^{-1}(x_s - \mu_s)$ and $\Sigma_{u|s} = \Sigma_{uu} - \Sigma_{us}\Sigma_{ss}^{-1}\Sigma_{su}$. Notice that only $\mu_{u|s}$ depends on the actual value of x_s, and $\Sigma_{u|s}$ depends only on the indices of $\mathbb{I}_S$. Being the sum of two normally distributed variables, we have that $(Z_{\mathbb{I}_U}|X_{\mathbb{I}_S} = x_s) \overset{d}{=} (X_{\mathbb{I}_U}|X_{\mathbb{I}_S} = x_s) + G_{\mathbb{I}_U} = \mathscr{N}(\mu_{u|s}, \Sigma_{u|s} + \Sigma_{uu}^{(g)})$.

Substituting this into the divergences above sum of divergences:

$$D_\lambda\begin{pmatrix}P_{\mathscr{A},\mathscr{P}}(Z|X_{\mathbb{I}_S}=s_i)\\P_{\mathscr{A},\mathscr{P}}(Z|X_{\mathbb{I}_S}=s_j)\end{pmatrix}=\sum_{m=1}^{|\mathbb{I}_S|}\left[D_\lambda\begin{pmatrix}\mathscr{N}(s_{im},\sigma_s^2)\\\mathscr{N}(s_{jm},\sigma_s^2)\end{pmatrix}\right]+D_\lambda\begin{pmatrix}\mathscr{N}(\mu_{u|s_i},\Sigma_{u|s}+\Sigma_{uu}^{(g)})\\\mathscr{N}(\mu_{u|s_j},\Sigma_{u|s}+\Sigma_{uu}^{(g)})\end{pmatrix} \quad (1)$$

$$=\frac{\lambda}{2}\sum_{m=1}^{|\mathbb{I}_S|}\frac{1}{\sigma_s^2}(s_{im}-s_{jm})^2+\frac{\lambda}{2}(\mu_{u|s_i}-\mu_{u|s_j})^\intercal(\Sigma_{u|s}+\Sigma_{uu}^{(g)})^{-1}(\mu_{u|s_i}-\mu_{u|s_j}) \quad (2)$$

$$=\frac{\lambda}{2\sigma_s^2}(s_i-s_j)^\intercal(s_i-s_j)+\frac{\lambda}{2}\left(\Sigma_{us}\Sigma_{ss}^{-1}(s_i-s_j)\right)^\intercal(\Sigma_{u|s}+\Sigma_{uu}^{(g)})^{-1}\left(\Sigma_{us}\Sigma_{ss}^{-1}(s_i-s_j)\right) \quad (3)$$

$$=\frac{\lambda}{2\sigma_s^2}(s_i-s_j)^\intercal(s_i-s_j)+\frac{\lambda}{2}(s_i-s_j)^\intercal\Sigma_{ss}^{-1}\Sigma_{su}(\Sigma_{u|s}+\Sigma_{uu}^{(g)})^{-1}\Sigma_{us}\Sigma_{ss}^{-1}(s_i-s_j) \quad (4)$$

Line (1) substitutes in the normal distributions given by our mechanism and conditional prior class. Line (2) substitutes in the closed-form expression for Rényi divergence between two mean-shifted normal distributions given in Equation 5.1. Line (3) substitutes in the expression for $\mu_{u|s}$ given above, and simplifies. To expand out this simplification in explicit steps:

$$\begin{aligned}(\mu_{u|s_i}-\mu_{u|s_j})&=\left(\mu_u+\Sigma_{us}\Sigma_{ss}^{-1}(s_i-\mu_s)-[\mu_u+\Sigma_{us}\Sigma_{ss}^{-1}(s_j-\mu_s)]\right)\\&=\left(\Sigma_{us}\Sigma_{ss}^{-1}s_i-\Sigma_{us}\Sigma_{ss}^{-1}s_j\right)\\&=\Sigma_{us}\Sigma_{ss}^{-1}(s_i-s_j)\end{aligned}$$

Line (4) distributes the transpose in the right term of line (3):

$$\begin{aligned}\left(\Sigma_{us}\Sigma_{ss}^{-1}(s_i-s_j)\right)^\intercal&=(s_i-s_j)^\intercal\left(\Sigma_{us}\Sigma_{ss}^{-1}\right)^\intercal\\&=(s_i-s_j)^\intercal\left(\Sigma_{ss}^{-1}\right)^\intercal\Sigma_{us}^\intercal\\&=(s_i-s_j)^\intercal\Sigma_{ss}^{-1}\Sigma_{su}\end{aligned}$$

where that final step is a consequence of Σ being symmetric. Σ_{ss} is also a symmetric matrix (so its inverse is symmetric) and $\Sigma_{us}^\intercal=\Sigma_{su}$.

Returning to line (4) above, simplify this expression by substituting $\Delta = s_i - s_j$:

$$D_\lambda \begin{pmatrix} P_{\mathscr{A},\mathscr{P}}(Z|X_{\mathbb{I}_S} = s_i) \\ P_{\mathscr{A},\mathscr{P}}(Z|X_{\mathbb{I}_S} = s_j) \end{pmatrix} = \frac{\lambda}{2\sigma_s^2}\Delta^\intercal\Delta + \frac{\lambda}{2}\Delta^\intercal\Sigma_{ss}^{-1}\Sigma_{su}(\Sigma_{u|s} + \Sigma_{uu}^{(g)})^{-1}\Sigma_{us}\Sigma_{ss}^{-1}\Delta \tag{5}$$

$$= \frac{\lambda}{2\sigma_s^2}\|\Delta\|_2^2 + \frac{\lambda}{2}\Delta^\intercal\Sigma_{\text{eff}}\Delta \tag{6}$$

Where $\Sigma_{\text{eff}} = \Sigma_{ss}^{-1}\Sigma_{su}(\Sigma_{u|s} + \Sigma_{uu}^{(g)})^{-1}\Sigma_{us}\Sigma_{ss}^{-1}$. The left term of line (6) attributes the direct loss of $Z_{\mathbb{I}_S}$ on $X_{\mathbb{I}_S}$ and the right term attributes the indirect loss of $Z_{\mathbb{I}_U}$ on $X_{\mathbb{I}_S}$.

We are interested in bounding the expression of line (6) for all $(s_i, s_j) \in \mathscr{S}_{\text{pairs}}$. We do this by bounding it for all vectors $\Delta \in \mathscr{D}$

$$\mathscr{D} = \{s_i - s_j : \|s_i - s_j\|_2 \leq \sqrt{S}\, r\}$$

, where S is the number of basic secrets (locations) contained in $\mathbb{I}_S$ which may be a basic or compound secret set. For a basic secret ($S = 1$), this bound is tight, since $\mathscr{D} = \{s_i - s_j : (s_i, s_j) \in \mathscr{S}_{\text{pairs}}\}$. The set of $\Delta \in \mathscr{D}$ is exactly any two hypothesis (s_i, s_j) that are within any circle of radius r. For a compound secret, this bound is not guaranteed to be tight. Recall once again that the set of $\mathscr{S}_{\text{pairs}}$ for a compound secret is given by the set of (s_i, s_j) in

$$\mathscr{S}_{\text{pairs}} = \left\{ \left(\{x_{s1}, x_{s2}, \dots\}, \{x'_{s1}, x'_{s2}, \dots\} \right) : \|x_{sk} - x'_{sk}\|_2 \leq r, \forall\, k \right\}$$

For concreteness, consider the 2d location trace example in **Figure E.1**, where we have a compound secret of $S = 3$ locations. Here, $s_i, s_j \in \mathbb{R}^6$, where 6 comes from the fact that we have three 2d locations. So, (s_i, s_j) represents a pair of hypotheses on all three locations. s_i's hypothesis of the first secret location — written as $x_{s1} \in \mathbb{R}^2$ above — is within r of the s_j's hypothesis of the first secret location — written as $x'_{s_1} \in \mathbb{R}^2$ above. The same goes for the second

and third locations. So, the L_2 norm of $\Delta = s_i - s_j$ is no greater than

$$\begin{aligned}
\sup_{(s_i,s_j)\in\mathscr{S}_{\text{pairs}}} \|s_i - s_j\|_2 &= \sup_{(s_i,s_j)\in\mathscr{S}_{\text{pairs}}} \sqrt{\sum_{m=1}^{6} (s_{im} - s_{jm})^2} \\
&= \sup_{(s_i,s_j)\in\mathscr{S}_{\text{pairs}}} \sqrt{\sum_{k=1}^{3} \|x_{sk} - x'_{sk}\|_2^2} \\
&= \sqrt{\sum_{k=1}^{3} r^2} \\
&= \sqrt{3}\, r
\end{aligned}$$

For compound secrets, $\mathscr{D}$ represents the L_2 ball enclosing all $\Delta \in \{s_i - s_j : (s_i,s_j) \in \mathscr{S}_{\text{pairs}}\}$. However, $\mathscr{D}$ also includes some values of $\Delta = s_i - s_j$ not covered by $\mathscr{S}_{\text{pairs}}$. Suppose an adversary considers the hypotheses

$$s_i = \{x_{s1}, x_{s2}, x_{s3}\}, s_j = \{x'_{s1}, x'_{s2}, x'_{s3}\}$$

where $x_{s1} = 0, x'_{s1} = \sqrt{3}\, r, x_{s2} = x'_{s2}, x_{s3} = x'_{s3}$. Since x_{s1}, x'_{s1} are not within r of each other, this is not in $\mathscr{S}_{\text{pairs}}$. However, it is covered by $\mathscr{D}$, and thus is covered by our bound on CIP loss and our mechanisms.

With $\mathscr{D}$ defined, we may return to bounding the expression in line (6):

$$\begin{aligned}
D_\lambda \begin{pmatrix} P_{\mathscr{A},\mathscr{P}}(Z|X_{\mathbb{I}_S} = s_i) \\ P_{\mathscr{A},\mathscr{P}}(Z|X_{\mathbb{I}_S} = s_j) \end{pmatrix} &\leq \sup_{\Delta\in\mathscr{D}} \left(\frac{\lambda}{2\sigma_s^2} \|\Delta\|_2^2 + \frac{\lambda}{2} \Delta^\intercal \Sigma_{\text{eff}} \Delta \right) && (7) \\
&\leq \frac{\lambda}{2} \left(\frac{1}{\sigma_s^2} S r^2 + S r^2 \text{maxeig}(\Sigma_{\text{eff}}) \right) && (8) \\
&= \frac{\lambda}{2} S r^2 \left(\frac{1}{\sigma_s^2} + \alpha^* \right) && (9)
\end{aligned}$$

where line (8) distributes the supremum. For the right term, this is given by the maximum magnitude of all $\Delta \in \mathscr{D}$ times the maximum eigenvalueof Σ_{eff} which equals $Sr^2\text{maxeig}(\Sigma_{\text{eff}})$.

Line (9) simply substitutes $\alpha^* = \text{maxeig}(\Sigma_{\text{eff}})$.

∎

Proof of Corollary 5.3.3.1

Corollary 5.3.3.1 Graceful Composition in Time *Suppose a user releases two traces X and $\dot{X}$ with additive noise $G \sim \mathcal{N}(\mathbf{0}, \Sigma^{(g)})$ and $\dot{G} \sim \mathcal{N}(\mathbf{0}, \dot{\Sigma}^{(g)})$, respectively. Then basic or compound secret $X_{\mathbb{I}_S}$ of X enjoys $(\bar{\varepsilon}, \lambda)$-CIP, where*

$$\bar{\varepsilon} \leq \frac{\lambda}{2} Sr^2 \Big(\frac{1}{\sigma_s^2} + \bar{\alpha}^* \Big)$$

and where $\bar{\alpha}$ is the maximum eigenvalue of $\bar{\Sigma}_{eff} = \big(\Sigma_{us}\Sigma_{ss}^{-1}\big)^{\intercal} \big(\Sigma_{u|s} + \bar{\Sigma}_{uu}^{(g)}\big)^{-1} \big(\Sigma_{us}\Sigma_{ss}^{-1}\big)$. Σ is the covariance matrix of the joint distribution on $X, \dot{X}$ and

$$\bar{\Sigma}^{(g)} = \begin{bmatrix} \Sigma^{(g)} & 0 \\ 0 & \dot{\Sigma}^{(g)} \end{bmatrix}.$$

Proof. Here, we record two traces (presumably) far apart in time

$$(X_1, \ldots, X_n) \text{ and } (\dot{X}_1, \ldots, \dot{X}_m)$$

And release

$$(Z_1, \ldots, Z_n) = (X_1, +G_1, \ldots, X_n + G_n) \text{ and } (\dot{Z}_1, \ldots, \dot{Z}_m) = (\dot{X}_1, +\dot{G}_1, \ldots, \dot{X}_m, +\dot{G}_m)$$

the first trace protects secret locations $X_{\mathbb{I}_S}$ and the second protects $\widehat{X_{\mathbb{I}_S}}$, so we have that

$$D_\lambda \begin{pmatrix} P_{\mathscr{A},\mathscr{P}}(Z | X_{\mathbb{I}_S} = s_i) \\ P_{\mathscr{A},\mathscr{P}}(Z | X_{\mathbb{I}_S} = s_j) \end{pmatrix} \leq \varepsilon$$
$$D_\lambda \begin{pmatrix} P_{\mathscr{A},\mathscr{P}}(\dot{Z} | \widehat{X_{\mathbb{I}_S}} = \dot{s}_i) \\ P_{\mathscr{A},\mathscr{P}}(\dot{Z} | \widehat{X_{\mathbb{I}_S}} = \dot{s}_j) \end{pmatrix} \leq \dot{\varepsilon}$$

We aim to update the losses:

$$D_\lambda\begin{pmatrix}P_{\mathscr{A},\mathscr{P}}(Z,\dot{Z}|X_{\mathbb{I}_S}=s_i)\\ P_{\mathscr{A},\mathscr{P}}(Z,\dot{Z}|X_{\mathbb{I}_S}=s_j)\end{pmatrix}\leq\varepsilon'$$
$$D_\lambda\begin{pmatrix}P_{\mathscr{A},\mathscr{P}}(\dot{Z},Z|\widehat{X_{\mathbb{I}_S}}=\dot{s}_i)\\ P_{\mathscr{A},\mathscr{P}}(\dot{Z},Z|\widehat{X_{\mathbb{I}_S}}=\dot{s}_j)\end{pmatrix}\leq\dot{\varepsilon}'$$

Fortunately, our framework is pretty friendly to figuring this out, and can be done simply by updating the 'inferential loss term' α^* and $\dot{\alpha}^*$ of each, the max eigenvalues used to compute each of ε and $\dot{\varepsilon}$, respectively. Let's focus on ε', since the same analysis follows for $\dot{\varepsilon}'$.

Recall that α^* is given by the max eigenvalue of Σ_{eff} which is

$$\Sigma_{\text{eff}}=\left(\Sigma_{us}\Sigma_{ss}^{-1}\right)^\intercal\left(\Sigma_{u|s}+\Sigma_{uu}^{(g)}\right)^{-1}\left(\Sigma_{us}\Sigma_{ss}^{-1}\right)$$

Where Σ is the covariance matrix of $X_1,\ldots,X_n$ and $\Sigma^{(g)}$ is the noise covariance matrix added. Simply augment Σ to become the joint covariance matrix Σ_J of $X,\dot{X}$, and augment $\Sigma^{(g)}$ to become

$$\Sigma_J^{(g)}=\begin{bmatrix}\Sigma^{(g)} & 0\\ 0 & \dot{\Sigma}^{(g)}\end{bmatrix}$$

then update Σ_{eff} to $\Sigma_{\text{eff},J}$ which uses both Σ_J and $\Sigma_J^{(g)}$. Using the corresponding max eigenvalue α_J^* in the loss expression of Theorem 3.2 gives us ε'.

Note that for kernels like RBF, $\varepsilon'\to\varepsilon$ as the traces X and $\dot{X}$ move apart further and further in time. This is not the case for traces using a purely periodic kernel with not time decay, and we should expect much worse composition. ∎

Traces with Independent Dimensions

In many cases, the different dimensions of the trace may be probabilistically independent, and it may be more convenient to make separate privacy mechanisms for each. For a 2d trace

X, suppose $\mathbb{I}_x$ and $\mathbb{I}_y$ store the indices of the latitude points $X_{\mathbb{I}_x}$ and longitude points $X_{\mathbb{I}_y}$, such that $X = X_{\mathbb{I}_x} \cup X_{\mathbb{I}_y}$. If latitude and longitude are independent, it may be more convenient to characterize the conditional priors of $X_{\mathbb{I}_x}$ abd $X_{\mathbb{I}_y}$ separately. The question is whether privacy guarantees remain for the full trace X. To answer this, we provide the following corollary:

Corollary E.1.0.1. *CIP loss of independent dimensions* Let Θ be a GP conditional prior class on a 2d trace X such that the dimensions are independent. Let $\mathbb{I}_S$ be some secret set of time indices corresponding to some basic or compound secret. For the trace $X = X_{\mathbb{I}_x} \cup X_{\mathbb{I}_y}$, the Gaussian mechanism $\mathscr{A}(X) = Z_{\mathbb{I}_x} \cup Z_{\mathbb{I}_y}$ where $Z_{\mathbb{I}_x} = \mathscr{A}_x(X_{\mathbb{I}_x}) = X_{\mathbb{I}_x} + G_{\mathbb{I}_x}$ and $Z_{\mathbb{I}_y} = \mathscr{A}_y(X_{\mathbb{I}_y}) = X_{\mathbb{I}_y} + G_{\mathbb{I}_y}$ satisfies (ε, λ)-CIP where

$$\varepsilon \leq \frac{\lambda}{2} Sr^2 \big(\frac{1}{\sigma_s^2} + \alpha_x^* + \alpha_y^* \big)$$

when $\mathscr{A}_x$ and $\mathscr{A}_y$ provide $\frac{\lambda}{2} Sr^2 \big(\frac{1}{\sigma_s^2} + \alpha_x^* \big)$ and $\frac{\lambda}{2} Sr^2 \big(\frac{1}{\sigma_s^2} + \alpha_y^* \big)$ to $\mathbb{I}_S \cap \mathbb{I}_x$ and $\mathbb{I}_S \cap \mathbb{I}_y$, respectively.

The gist of this corollary is that a mechanism can be designed to achieve the bound of Theorem 5.3.3 to each dimension independently and released with still-meaningful privacy guarantees. The reason is that this still includes all secret pairs $\mathscr{S}_{\text{pairs}}$

Proof. By independence, $X_{\mathbb{I}_x}$ and $X_{\mathbb{I}_y}$ can be treated as two unconnected traces of the type seen in **Figure 5.1**. As such the privacy guarantee of Theorem 5.3.3 can be upheld for each. The question is whether bounding CIP loss to the one-dimensional basic or compound secret associated with secret sets $\mathbb{I}_S \cap \mathbb{I}_x$ and $\mathbb{I}_S \cap \mathbb{I}_y$ still provides guarantees for the full secret set $\mathbb{I}_S$.

Without loss of generality, we will demonstrate for a basic and a compound secret. Consider the basic secret set $\mathbb{I}_S = \{X_{10}, X_{11}\}$, where $\mathbb{I}_S \cap \mathbb{I}_x = \{X_{10}\}$ (latitude) and $\mathbb{I}_S \cap \mathbb{I}_y = \{X_{11}\}$ (longitude). We again assume that independent gaussian noise of variance σ_s^2 is added to all $X_{\mathbb{I}_S}$, since this is optimal under utility constraints. We have now bounded the Rényi divergence when

conditioning on pairs of hypotheses on latitude and longitude separately.

$$\mathscr{S}_{\text{pairs}_x} = \mathscr{S}_{\text{pairs}_y} = \{(x_s, x'_s) : x_s \in \mathbb{R}, \|x_s - x'_s\|_2 \leq r\}$$

By independence, this also bounds the Rényi divergence conditioning on pairs of hypotheses on latitude and longitude jointly:

$$\mathscr{S}_{\text{pairs}_{xy}} = \{(x_s, x'_s) : x_s \in \mathbb{R}^2, \|x_s - x'_s\|_2 \leq r\}$$

In effect, we have guaranteed privacy for any pair of hypotheses (s_i, s_j) in the square circumscribing the circle of radius r that we with to provide. The analysis on the direct privacy loss is exactly the same as it was in the more general case. Since the Rényi divergences of $X_{\mathbb{I}_U} \cap X_{\mathbb{I}_x}$ and of $X_{\mathbb{I}_U} \cap X_{\mathbb{I}_y}$ add, the α^*'s add.

The same goes for a compound secret. Consider three location compound secret pairs given by

$$\mathscr{S}_{\text{pairs}_{xy}} = \left\{ \left(\{x_{s1}, x_{s2}, \dots\}, \{x'_{s1}, x'_{s2}, \dots\} \right) : x_{si} \in \mathbb{R}^2, \|x_{sk} - x'_{sk}\|_2 \leq r, \forall\, k \right\}$$

Instead, we bound privacy loss for

$$\mathscr{S}_{\text{pairs}_x} = \mathscr{S}_{\text{pairs}_y} = \left\{ \left(\{x_{s1}, x_{s2}, \dots\}, \{x'_{s1}, x'_{s2}, \dots\} \right) : x_{si} \in \mathbb{R}, \|x_{sk} - x'_{sk}\|_2 \leq r, \forall\, k \right\}$$

Separately, giving us α_x^* and α_y^*. This again includes any two hypotheses on the three locations such that each pair of x_{sk}, x'_{sk} is within a square circumscribing a circle of radius r. We achieve this by bounding privacy loss for all Δ_x in a 3d L_2 ball of radius $\sqrt{S}\, r$, as with Δ_y.

This corollary can be extended to all traces of all dimensions that are probabilistically independent. ∎

We make use of the above proof in the Experiments section.

E.1.3 Derivation of Algorithms

In this section, we derive the three SDP-based algorithms of Section 5.4 and their properties.

Derivation of SDP_A

SDP_A minimizes the privacy loss bound of Theorem 5.3.3 for any compound or basic secret encoded by secret set $\mathbb{I}_S$. As is clarified in its proof (Appendix E.1.2), the bound is tight when $\mathbb{I}_S$ encodes a basic secret. If $\mathbb{I}_S$ encodes a compound secret, the tightness depends on the conditional prior class Θ.

Our variable for minimizing this bound is the noise covariance matrix $\Sigma^{(g)}$. Due to the conditional independence exhibited by Lemma 8, $G_{\mathbb{I}_S}$ and $G_{\mathbb{I}_U}$ may be independent. The additive noise $G_i \in G_{\mathbb{I}_S}$ are all independent Gaussian with variance σ_s^2. This is because — conditioning on $\{X_{\mathbb{I}_S} = x_s\}$ — $Z_{\mathbb{I}_S}$ is independent of $X_{\mathbb{I}_U}$ and $Z_{\mathbb{I}_U}$. So, $G_{\mathbb{I}_S} \sim \mathcal{N}(\mathbf{0}, \sigma_s^2 I)$, and $\Sigma^{(g)}_{ss} = \sigma_s^2 I$. The additive noise $G_i \in G_{\mathbb{I}_U}$ are all dependent as described by $\Sigma^{(g)}_{uu}$, and $G_{\mathbb{I}_U} \sim \mathcal{N}(\mathbf{0}, \Sigma^{(g)}_{uu})$. Consequently, $\Sigma^{(g)}$ is completely characterized by $\Sigma^{(g)}_{uu}$ and σ_s^2.

To see how the bound of Theorem 5.3.3 can be redrafted as an SDP, first notice that its two terms may be written as the maximum eigenvalue of a matrix product. Here, $\Sigma_{\text{eff}} = A^\intercal B A$, where $A = \Sigma_{us}\Sigma_{ss}^{-1}$ and $B = \left(\Sigma_{u|s} + \Sigma^{(g)}_{uu}\right)^{-1}$

$$\frac{1}{\sigma_s^2} + \alpha^* = \text{maxeig}\big(\frac{1}{\sigma_s^2} I + A^\intercal B A\big) = \text{maxeig}\left(\begin{bmatrix} I & A \end{bmatrix} \begin{bmatrix} \frac{1}{\sigma_s^2} I & 0 \\ 0 & B \end{bmatrix} \begin{bmatrix} I \\ A \end{bmatrix} \right) = \text{maxeig}\big(\tilde{A}^\intercal \tilde{B} \tilde{A}\big)$$

This expression uses all parameters of $\Sigma^{(g)}$: σ_s^2 parametrizes $\Sigma^{(g)}_{ss}$ and $\Sigma^{(g)}_{uu} = B^{-1} - \Sigma_{u|s}$, where $\Sigma_{u|s}$ is given by the kernel function of Θ.

Before casting this as an SDP, we provide a formal definition from [158]:

Definition E.1.1. *Semidefinite Program* The problem of minimizing a linear function of a

variable $x \in \mathbb{R}^n$ subject to a matrix inequality:

$$\begin{aligned} \min_{x \in \mathbb{R}^n} \; & c^\intercal x \\ \text{s.t. } & F_0 + \sum_{i=1}^{n} x_i F_i \succeq 0 \\ & Ax = b \end{aligned}$$

where the $F_i \in \mathbb{R}^{n \times n}$ are all symmetric and $A \in \mathbb{R}^{p \times n}$ is a *semidefinite program*, or SDP.

The task of minimizing $\text{maxeig}\left(\tilde{A}^\intercal \tilde{B} \tilde{A}\right)$ under MSE constraints can almost be formulated as an SDP:

$$\begin{aligned} \min_{B \succeq 0, 1/\sigma_s^2 \geq 0} \; & \beta^* \\ \text{s.t. } & \beta^* I \succeq \tilde{A}^\intercal \tilde{B} \tilde{A} \\ & B \preceq \Sigma_{u|s}^{-1} \\ & \mathbf{tr}(\Sigma_{uu}^{(g)}) + |\mathbb{I}_S| \sigma_s^2 \leq n o_t \end{aligned}$$

Here, the first constraint guarantees that the maximum eigenvalue of $\tilde{A}^\intercal \tilde{B} \tilde{A}$ is bounded by β^*, which the objective minimizes. At program completion, we set $\Sigma_{uu}^{(g)} = B^{-1} - \Sigma_{u|s}$, and the second constraints ensures that this is still PSD. The final constraint bounds the MSE of the mechanism $\Sigma^{(g)}$. Note that $\mathbf{tr}(\Sigma_{uu}^{(g)}) + |\mathbb{I}_S| \sigma_s^2 = \mathbf{tr}(\Sigma^{(g)})$. The trouble lies the last constraint. Our program variable is B, but the final linear constraint requires $\Sigma^{(g)}$, which is expressed using the inverse of B. This is not immediately available in the SDP framework.

To make the final linear constraint available, we invert the above program using the observation that the maximum eigenvalue of $\tilde{A}^\intercal \tilde{B} \tilde{A}$ is the inverse of the minimum eigenvalue of $(\tilde{A}^\intercal \tilde{B} \tilde{A})^{-1}$. Instead of optimizing over B and $1/\sigma_s^2$, we optimize over B^{-1} and σ_s^2. Since $B^{-1} = \Sigma_{u|s} + \Sigma_{uu}^{(g)}$, we may now have a utility constraint directly on the trace of $\Sigma^{(g)}$. To make B^{-1} our program variable, we approximate $(\tilde{A}^\intercal \tilde{B} \tilde{A})^{-1}$ with $\tilde{A}^{-1} \tilde{B}^{-1} \tilde{A}^{-\intercal}$. First note that $\tilde{A} \in \mathbb{R}^{n \times |\mathbb{I}_S|}$,

and has full column rank for the covariances we work with. So, $\tilde{A}^{-1} = (\tilde{A}^\intercal \tilde{A})^{-1}\tilde{A}^\intercal \in \mathbb{R}^{(|\mathbb{I}_S| \times n)}$ is the left inverse of $\tilde{A}$ and is the least squares solution to $\tilde{A}^{-1}\tilde{A} = \tilde{A}^\intercal \tilde{A}^{-\intercal} = I$ (we denote its transpose as $\tilde{A}^{-\intercal}$). It is also the least squares solution to $\tilde{A}\tilde{A}^{-1} = \tilde{A}^{-\intercal}\tilde{A}^\intercal = I$. Thus, we have an approximation of the inverse $(\tilde{A}^\intercal \tilde{B}\tilde{A})^{-1}$:

$$\begin{aligned}(\tilde{A}^\intercal \tilde{B}\tilde{A})\,(\tilde{A}^{-1}\tilde{B}^{-1}\tilde{A}^{-\intercal}) &\approx \tilde{A}^\intercal \tilde{B}\tilde{B}^{-1}\tilde{A}^{-\intercal} \\ &= \tilde{A}^\intercal \tilde{A}^{-\intercal} \\ &\approx I\end{aligned}$$

We now can optimize in terms of B^{-1} with the augmented matrix $\tilde{B}^{-1}$:

$$\tilde{B}^{-1} = \begin{bmatrix} \sigma_s^2 I & 0 \\ 0 & B^{-1} \end{bmatrix}$$

We then optimize the following SDP:

$$\begin{aligned}\max_{B^{-1}\succeq 0, \sigma_s^2 \geq 0} \quad & \beta^* \\ \text{s.t. } & \beta^* I \preceq \tilde{A}^{-1}\tilde{B}^{-1}\tilde{A}^{-\intercal} \\ & B^{-1} \succeq \Sigma_{u|s} \\ & \mathbf{tr}(\tilde{B}) - \mathbf{tr}(\Sigma_{u|s}) \leq no_t\end{aligned}$$

Upon program completion we recover σ_s^2 and $\Sigma_{uu}^{(g)} = B^{-1} - \Sigma_{u|s}$ which we know is PSD due to the second constraint. The first constraint guarantees that the minimum eigenvalue of the approximated inverse is $\geq \beta^*$, which the objective maximizes. If the minimum eigenvalue of the approximate inverse is close to that of the true inverse, then we successfully minimize the maximum eigenvalue of $\tilde{A}^\intercal \tilde{B}\tilde{A}$, and thus minimize the direct and indirect privacy loss. The

third constraint limits the MSE of $\Sigma^{(g)}$ since $\mathbf{tr}(\tilde{B}) - \mathbf{tr}(\Sigma_{u|s}) = (\mathbf{tr}(\Sigma_{uu}^{(g)}) + |\mathbb{I}_S|\sigma_s^2 + \mathbf{tr}(\Sigma_{u|s})) - \mathbf{tr}(\Sigma_{u|s}) = \mathbf{tr}(\Sigma^{(g)})$. By inverting $\tilde{A}^\intercal \tilde{B} \tilde{A}$, this constraint is available in the SDP framework.

By expressing the above program in terms of the variable $\Sigma^{(g)}$ instead of indirectly via B^{-1} and σ_s^2, we get SDP_A:

$$\begin{aligned} \mathbf{SDP_A}: \quad & \underset{\Sigma^{(g)} \succeq 0}{\arg\max}\ \beta^* \\ & \text{s.t. } \tilde{A}^{-1}\tilde{B}^{-1}\tilde{A}^{-\intercal} \succeq \beta^* \mathbf{I} \\ & \mathbf{tr}(\Sigma^{(g)}) \leq no_t \end{aligned}$$

It is straightforward to write this SDP in the form seem in Definition E.1.1. The program variables x would be the diagonal and upper or lower triangular part of $\Sigma^{(g)}$ along with β^*. With some linear algebra, the first constraint can be written in the form of $F_0 + \sum_{i=1}^{n} x_i F_i \succeq 0$, and the second constraint can be written as $Ax = b$. With the use of contemporary convex programming tools like CVXOPT [157] rewriting into this form is unnecessary.

Derivation of SDP_B

SDP_B takes a set of covariance matrices $\mathscr{F} = \{\Sigma_1, \ldots, \Sigma_k\}$, each of which is designed to protect some secret set $\mathbb{I}_{Si}$, and returns a covariance matrix $\Sigma^{(g)}$ that preserves the privacy loss bound of each Σ_i to each $\mathbb{I}_{Si}$. It does so while minimizing the utility loss of $\Sigma^{(g)}$. This algorithm is also expressed as an SDP. It is based on the following corollary, which we have omitted from the main text:

Corollary E.1.0.2. *More PSD, More Private:* For a basic or compound secret denoted by indices $\mathbb{I}_S$, the CIP loss bound of Equation 5.5 provided by a Gaussian noise mechanism with covariance $\Sigma^{(g)}$ is lower than it would be for any $\Sigma^{(g)'} \prec \Sigma^{(g)}$.

Proof. First note that if $\Sigma^{(g)} \succ \Sigma^{(g)'}$, then the same is true for its sub-matrices:

$$\Sigma_{ss}^{(g)} \succ \Sigma_{ss}^{(g)'} \qquad \Sigma_{uu}^{(g)} \succ \Sigma_{uu}^{(g)'}$$

Recall the privacy loss bound of Equation 5.5:

$$\varepsilon \leq \frac{\lambda}{2} S r^2 \Big(\frac{1}{\sigma_s^2} + \alpha^* \Big)$$

Also recall that $\Sigma_{ss}^{(g)} = \sigma_s^2 I$ and $\Sigma_{ss}^{(g)'} = \sigma_s^{2\prime} I$. Since $\Sigma_{ss}^{(g)} \succ \Sigma_{ss}^{(g)'}$, we already know that $\sigma_s^2 > \sigma_s^{2\prime}$, and thus the first term of Equation 5.5 is lower for $\Sigma^{(g)}$.

It remains to show that the second term is also lower, $\alpha^* < \alpha^{*\prime}$. Starting with what we're given,

$$\Sigma_{uu}^{(g)} \succ \Sigma_{uu}^{(g)'}$$

$$\Sigma_{uu}^{(g)} + \Sigma_{u|s} \succ \Sigma_{uu}^{(g)'} + \Sigma_{u|s}$$

$$(\Sigma_{uu}^{(g)} + \Sigma_{u|s})^{-1} \prec (\Sigma_{uu}^{(g)'} + \Sigma_{u|s})^{-1}$$

$$B \prec B'$$

$$A^\intercal B A \prec A^\intercal B' A$$

$$\max \mathrm{eig}(A^\intercal B A) < \max \mathrm{eig}(A^\intercal B' A)$$

$$\alpha^* < \alpha^{*\prime}$$

Therefore $\frac{1}{\sigma_s^2} + \alpha^* < \frac{1}{\sigma_s^{2\prime}} + \alpha^{*\prime}$, and the CIP bound of Equation 5.5 is lower for $\Sigma^{(g)}$ than it is for $\Sigma^{(g)'}$. ∎

With Corollary E.1.0.2 in mind, $\mathrm{SDP_B}$ is natural:

$$\textbf{SDP}_\textbf{B}: \quad \arg\min_{\Sigma^{(g)}} \mathbf{tr}(\Sigma^{(g)})$$
$$\text{s.t. } \Sigma^{(g)} \succeq \Sigma_i^{(g)},\ \forall \Sigma_i^{(g)} \in \mathscr{F}$$

SDP_B attempts to minimize, but does not constrain, the utility loss of the chosen $\Sigma^{(g)}$. To provide an upper bound on the resulting utility loss, we provided the following claim in the main text:

Claim Utility loss of SDP_B: *The utility loss of* $\Sigma^{(g)} = SDP_B(\mathscr{F})$ *is no greater than* $\sum_{\Sigma_i \in \mathscr{F}} \mathbf{tr}(\Sigma_i)$.

Proof. The covariance $\Sigma^{(g)'} = \sum_{\Sigma_i^{(g)} \in \mathscr{F}} \Sigma_i^{(g)}$ with MSE $\sum_{\Sigma_i^{(g)} \in \mathscr{F}} \mathbf{tr}(\Sigma_i^{(g)})$ is in the feasible set of SDP_B problem since $\Sigma^{(g)'} \succeq \Sigma_i^{(g)},\ \forall \Sigma_i^{(g)} \in \mathscr{F}$. Unless $\Sigma^{(g)'}$ has the lowest MSE of all $\Sigma^{(g)}$ in the feasible set, a covariance matrix with better utility will be chosen. ∎

Derivation of Algorithm 2, Multiple Secrets

Multiple Secrets combines SDP_A and SDP_B to minimize the privacy loss to each basic secret within a trace. The basic mechanism is useful in cases when inferences at each time within the trace — each basic secret — is sensitive.

Let $\mathbb{I}_{Si}$ be the secret set representing basic secret i, of which there are N (e.g. if location is sampled at N times). Then $\mathbb{I}_{\mathscr{S}_b} = \{\mathbb{I}_{S1}, \ldots, \mathbb{I}_{SN}\}$ contains the indices corresponding to each. Multiple Secrets works by first producing N covariance matrices, $\Sigma_i^{(g)} = \text{SDP}_\text{A}(\mathbb{I}_{Si}, \Sigma, o_t)$ on each basic secret. It then uses $\text{SDP}_\text{B}(\mathscr{F} = \{\Sigma_1^{(g)}, \ldots, \Sigma_N^{(g)}\})$ to produce a single covariance matrix $\Sigma^{(g)}$ that preserves the privacy loss to each basic secret (note that, being basic secrets, the privacy loss bound that SIG OPT optimizes is tight).

By virtue of using SDP_B, the MSE of the resultant $\Sigma^{(g)}$ is minimized but not constrained. To bound the MSE of the Basic Mechanism by O, we may simply bound the MSE of each $\Sigma_i^{(g)}$

by $o_t = O/N$. Then, by the above Claim, the MSE of the solution cannot be greater than O. In practice, this bound may be too loose. We hope to tighten it in future work.

E.1.4 Experimental details

We use a 2d location trace and a 1d home temperature dataset. For the location data, having observed that the correlation between latitude and longitude is low (≈ 0.06) we treat each dimension as independent. By way of Corollary E.1.0.1, this allows us to bound privacy loss and design mechanisms for each dimension separately. Furthermore, having observed that each dimension fits the nearly the same conditional prior, we treat our dataset of 10k 2-dimensional traces as a dataset of 20k 1-dimensional traces, where each trace represents one dimension of a 2d location trajectory.

The one-dimensional traces of temperature and location are indexed by timestamps, for which we would use the following kernel functions:

$$k_{\text{RBF}}(t_i,t_j) = \sigma_x^2 \exp\Big(-\frac{(t_i-t_j)^2}{2l^2}\Big) \qquad k_{\text{PER}}(t_i,t_j) = \sigma_x^2 \exp\Big(\frac{-2\sin^2(\pi|t_i-t_j|/p)}{l^2}\Big) \quad \text{(E.1)}$$

to determine the covariance between two points sampled at times t_i and t_j. The parameters including variance σ_x^2 and length scale l. The lengthscale determines the window of time in which two sampled points are highly correlated.

Preprocessing of location data

We first limit the dataset to traces of under 50 locations that are between 4.5 and 5.5 minutes in duration. Caring only about the conditional dependence between locations, we then de-mean each trace and normalize its variance to one. Normalizing the variance of traces implicitly sets $\sigma_x^2 = 1$ in the above RBF kernel, in essence assuming that the adversary has a decent prior for the user's average speed in a given trace, and could do the same operation.

Fitting of location data

We then find the maximum likelihood RBF kernel for each distinct trace. Having fixed the variance σ_x^2, this amounts to fitting only the length scale for each dimension, l_x and l_y, individually. The length scale represents the average window of time during which neighboring locations are highly correlated (i.e. correlation > 0.8). Relatively smooth traces will have large length scales and chaotic traces will have low length scales. However, the fact that sampling rates vary significantly between traces means that traces with equal length scales can have very different degrees of correlation. To encapsulate both of these effects, we study the empirical distribution of *effective* length scale of each trace

$$l_{\text{eff},x} = \frac{l_x}{P} \quad l_{\text{eff},y} = \frac{l_y}{P}$$

where P is the trace's sampling period and l_x, l_y are the its optimal length scales. $l_{\text{eff},x}, l_{\text{eff},y}$ tell us the average number of neighboring locations that are highly correlated, instead of time period. For instance, a given trace with an optimal $l_{\text{eff},x} = 8$ tells us that every eight neighboring location samples in the x dimension have correlation > 0.8. The empirical distribution of effective length scales across all traces describes – over a range of logging devices (sampling rates), users, and movement patterns – how many neighboring points are highly correlated in location trace data. After this preprocessing, we are able to use the kernels that take indices (not time) as arguments.

$$k_{\text{RBF}}(i,j) = \exp\Big(-\frac{(i-j)^2}{2l_{\text{eff}}^2}\Big) \qquad k_{\text{PER}}(i,j) = \exp\Big(\frac{-2\sin^2(\pi|i-j|/p)}{l_{\text{eff}}^2}\Big)$$

In each plot we then observed a spectrum of conditional priors by sweeping the effective length scale and plotting posterior uncertainty for various noise mechanisms of equal utility loss. This ranges from a prior assuming nearly independent location samples (chaotic trace) on the left up to highly dependent location samples (traveling in a straight line or standing still) on the right.

To understand how realistic these conditional prior parameters are, we displayed the middle 50% of the empirical distribution of l_{eff} (x and y together) from the GeoLife dataset. Note that the distribution of l_{eff_x} and l_{eff_y} are nearly identical.

To compute posterior uncertainty, we consider a 50-point one-dimensional location trace. The basic secret is a single index in the middle of the trace, and the compound secret consists of two neighboring indices also in the middle of trace. For each value of l_{eff}, we compute the $\mathbb{R}^{50\times 50}$ conditional prior covariance matrix Σ using the RBF kernel above. We then compare the posterior uncertainty when $\Sigma^{(g)}$ is an Approach C baseline, or an optimized covariance matrix using one of the three algorithms. We re-optimize $\Sigma^{(g)}$ for each l_{eff}, since each l_{eff} represents a different conditional prior class. The MSE is fixed in all figures except the two exhibiting "All Basic Secrets", where SDP_{B} is used. Recall that this algorithm minimizes utility loss while maintaining a series of privacy guarantees. Here, the MSE is identical across mechanisms for each l_{eff}, but changes from one l_{eff} to another.

For the temperature data, our preprocessing steps were nearly identical, except we use the periodic kernel instead of the RBF kernel, and we did not need to remove any traces from the dataset, as the data was much cleaner.

Computation of Posterior Uncertainty Interval

Each of the plots in **Figure 5.2** shows the 2σ uncertainty interval on $X_{\mathbb{I}_S}$ of a Gaussian process Bayesian adversary with prior covariance Σ and any mean function

The posterior covariance is computed using standard formulas for linear Gaussian systems. Knowing that $Z = X + G$, we may write the joint precision matrix Λ (inverse of covariance matrix) of (X,Z) as

$$\Lambda^{(X,Z)} = \begin{bmatrix} \Sigma^{-1} + {\Sigma^{(g)}}^{-1} & -{\Sigma^{(g)}}^{-1} \\ -{\Sigma^{(g)}}^{-1} & {\Sigma^{(g)}}^{-1} \end{bmatrix}$$

It is then a well known result that the conditional covariance matrix is given by

$$\begin{aligned}\Sigma_{x|z} &= \Lambda_{xx}^{-1} \\ &= \left(\Sigma^{-1} + {\Sigma^{(g)}}^{-1}\right)^{-1}\end{aligned}$$

This provides the posterior covariance of all locations X given any released trace Z that uses a Gaussian mechanism with covariance $\Sigma^{(g)}$. Note that the CIP guarantee naturally keeps posterior uncertainty large since the posterior density at any two x_s close together must be similar. For these Gaussian posteriors, 2σ tells us the adversary's 68% confidence interval on $X_{\mathbb{I}_S}$ after obvserving Z.

For basic secrets (one location), we simply report twice the posterior standard deviation at the sensitive index i, given by

$$2\sqrt{\Sigma_{x|z,ii}}\,.$$

For compound secrets involving multiple locations the posterior distribution is a length $|\mathbb{I}_S|$ multivariate normal with covariance $\Sigma_{x|z,ss}$. Intuitively, we wish to find the direction of the vector $X_{\mathbb{I}_S}$ in which the posterior interval is the *shortest*. This is the worst case posterior interval on the compound secret. We do this by reporting

$$2\sqrt{\text{mineig }\Sigma_{x|z,ss}}\,.$$

E.1.5 Discussion of GP Conditional Prior Class

Recall that a conditional prior class requires for any $P_{\mathscr{P}_i}, P_{\mathscr{P}_j} \in \Theta$ that

$$P_{\mathscr{P}_i}(X_{\mathbb{I}_U}|X_{\mathbb{I}_S} = x_s) = P_{\mathscr{P}_j}(X_{\mathbb{I}_U} + c^u_{ij\mathbb{I}_S}|X_{\mathbb{I}_S} = x_s + c^s_{ij\mathbb{I}_S})$$

for all x_s. Notice that the mapping $(x_s, x_s') + c^s_{ij\mathbb{I}_S}$ is a bijection from $\mathscr{S}_{\text{pairs}}$ onto itself. As such, each pair of conditional distributions,

$$\left(P_{\mathscr{P}_j}(X_{\mathbb{I}_U}|X_{\mathbb{I}_S} = x_s), P_{\mathscr{P}_j}(X_{\mathbb{I}_U}|X_{\mathbb{I}_S} = x_s')\right)$$

induced by $(x_s, x_s') \in \mathscr{S}_{\text{pairs}}$ is a mean-shifted version of the pair of distributions

$$\left(P_{\mathscr{P}_i}(X_{\mathbb{I}_U}|X_{\mathbb{I}_S} = x_s - c^s_{ij\mathbb{I}_S}), P_{\mathscr{P}_i}(X_{\mathbb{I}_U}|X_{\mathbb{I}_S} = x_s' - c^s_{ij\mathbb{I}_S})\right)$$

induced by $(x_s, x_s') - c^s_{ij\mathbb{I}_S} \in \mathscr{S}_{\text{pairs}}$. Since the Rényi divergence between two distributions and two mean-shifted versions thereof is unchanged, we may use one additive noise mechanism for all priors in class Θ.

To see how this applies to the GP prior class, recall the formula for a conditional multivariate Gaussian distribution:

$$P(X_{\mathbb{I}_U}|X_{\mathbb{I}_S} = x_s) = \mathscr{N}(\mu_{u|s}, \Sigma_{u|s})$$

where,

$$\mu_{u|s} = \mu_u + \Sigma_{us}\Sigma_{ss}^{-1}(x_s - \mu_s)$$
$$\Sigma_{u|s} = \Sigma_{uu} - \Sigma_{us}\Sigma_{ss}^{-1}\Sigma_{su}$$

A GP prior class includes all GP distributions with a fixed kernel $k(t_i, t_j)$ and any mean function $\mu(t)$. For a fixed set of time points, this corresponds to a fixed covariance matrix Σ and any mean parameters $\boldsymbol{\mu}$:

$$X \sim \mathscr{N}(\boldsymbol{\mu}, \Sigma)$$

Let $P_{\mathscr{P}_i} = \mathcal{N}(\bar{\boldsymbol{\mu}}, \Sigma)$ and $P_{\mathscr{P}_j} = \mathcal{N}(\dot{\boldsymbol{\mu}}, \Sigma)$, then conditioned on some sensitive points $X_{\mathbb{I}_S}$ the distribution on $X_{\mathbb{I}_U}$ has the same covariance $\Sigma_{u|s}$ and conditional means

$$\begin{aligned}
\bar{\mu}_{u|s} &= \bar{\mu}_u + \Sigma_{us}\Sigma_{ss}^{-1}(x_s - \bar{\mu}_s) \\
&= (\bar{\mu}_u - \Sigma_{us}\Sigma_{ss}^{-1}\bar{\mu}_s) + \Sigma_{us}\Sigma_{ss}^{-1}x_s \\
\dot{\mu}_{u|s} &= \dot{\mu}_u + \Sigma_{us}\Sigma_{ss}^{-1}(x_s - \dot{\mu}_s) \\
&= (\dot{\mu}_u - \Sigma_{us}\Sigma_{ss}^{-1}\dot{\mu}_s) + \Sigma_{us}\Sigma_{ss}^{-1}x_s
\end{aligned}$$

which implies that the conditional distributions are identical up to a mean shift for the *same* x_s value.

$$P_{\mathscr{P}_i}(X_{\mathbb{I}_U}|X_{\mathbb{I}_S} = x_s) = P_{\mathscr{P}_j}(X_{\mathbb{I}_U} + c^u_{ij\mathbb{I}_S}|X_{\mathbb{I}_S} = x_s)$$

for all x_s. Here, $c^u_{ij\mathbb{I}_S} = (\bar{\mu}_u - \Sigma_{us}\Sigma_{ss}^{-1}\bar{\mu}_s) - (\dot{\mu}_u - \Sigma_{us}\Sigma_{ss}^{-1}\dot{\mu}_s)$, and $c^s_{ij\mathbb{I}_S} = 0$.

To see how this allows a single additive mechanism to work for all mean functions, notice that we also have

$$P_{\mathscr{P}_i}(X_{\mathbb{I}_U}|X_{\mathbb{I}_S} = x'_s) = P_{\mathscr{P}_j}(X_{\mathbb{I}_U} + c^u_{ij\mathbb{I}_S}|X_{\mathbb{I}_S} = x'_s)$$

for x'_s, so the divergences

$$\begin{aligned}
D_\lambda\begin{pmatrix} P_{\mathscr{P}_i}(X_{\mathbb{I}_U}|X_{\mathbb{I}_S} = x_s) \\ P_{\mathscr{P}_i}(X_{\mathbb{I}_U}|X_{\mathbb{I}_S} = x'_s)\end{pmatrix} &= D_\lambda\begin{pmatrix} P_{\mathscr{P}_j}(X_{\mathbb{I}_U} + c^u_{ij\mathbb{I}_S}|X_{\mathbb{I}_S} = x_s) \\ P_{\mathscr{P}_j}(X_{\mathbb{I}_U} + c^u_{ij\mathbb{I}_S}|X_{\mathbb{I}_S} = x'_s)\end{pmatrix} \\
&= D_\lambda\begin{pmatrix} P_{\mathscr{P}_j}(X_{\mathbb{I}_U}|X_{\mathbb{I}_S} = x_s) \\ P_{\mathscr{P}_j}(X_{\mathbb{I}_U}|X_{\mathbb{I}_S} = x'_s)\end{pmatrix}
\end{aligned}$$

are equal. The same goes for the noisy trace $X_{\mathbb{I}_U} + Z_{\mathbb{I}_U}|X_{\mathbb{I}_S} = x_s$, when Z is drawn independently of X, allowing us to bound privacy loss for all $P \in \Theta$.

Bibliography

[1] Dependence Makes You Vulnerable: Differential Privacy Under Dependent Tuples. San Diego, CA.

[2] Derangement. https://en.wikipedia.org/wiki/Derangement.

[3] Hospital discharge data public use data file. https://www.dshs.state.tx.us/THCIC/Hospitals/Download.shtm.

[4] Martín Abadi, Andy Chu, Ian Goodfellow, H. Brendan McMahan, Ilya Mironov, Kunal Talwar, and Li Zhang. Deep Learning with Differential Privacy. *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security*, pages 308–318, October 2016. arXiv: 1607.00133.

[5] Mohamed Abdalla, Moustafa Abdalla, Graeme Hirst, and Frank Rudzicz. Exploring the Privacy-Preserving Properties of Word Embeddings: Algorithmic Validation Study. *Journal of Medical Internet Research*, 22(7):e18055, July 2020.

[6] Emily Alsentzer, John Murphy, William Boag, Wei-Hung Weng, Di Jindi, Tristan Naumann, and Matthew McDermott. Publicly available clinical bert embeddings. In *Proceedings of the 2nd Clinical Natural Language Processing Workshop*, pages 72–78, 2019.

[7] Mário Alvim, Konstantinos Chatzikokolakis, Catuscia Palamidessi, and Anna Pazii. Invited Paper: Local Differential Privacy on Metric Spaces: Optimizing the Trade-Off with Utility. In *2018 IEEE 31st Computer Security Foundations Symposium (CSF)*, pages 262–267, July 2018. ISSN: 2374-8303.

[8] Miguel E Andrés, Nicolás E Bordenabe, Konstantinos Chatzikokolakis, and Catuscia Palamidessi. Geo-indistinguishability: Differential privacy for location-based systems. *arXiv preprint arXiv:1212.1984*, 2012.

[9] Eugene Bagdasaryan, Omid Poursaeed, and Vitaly Shmatikov. Differential privacy has disparate impact on model accuracy. *Advances in Neural Information Processing Systems*, 32:15479–15488, 2019.

[10] Simonetti N. Vazacopoulos A. Balas, E. Job shop scheduling with setup times, deadlines and precedence constraints. *J Sched*, 11:253–262, 2008.

[11] Victor Balcer and Albert Cheu. Separating local and shuffled differential privacy via histograms. In *ITC*, 2020.

[12] Borja Balle, James Bell, Adrià Gascón, and Kobbi Nissim. The privacy blanket of the shuffle model. In Alexandra Boldyreva and Daniele Micciancio, editors, *Advances in Cryptology – CRYPTO 2019*, pages 638–667, Cham, 2019. Springer International Publishing.

[13] Borja Balle, Giovanni Cherubin, and Jamie Hayes. Reconstructing training data with informed adversaries. *arXiv preprint arXiv:2201.04845*, 2022.

[14] Adrien Bardes, Jean Ponce, and Yann LeCun. Vicreg: Variance-invariance-covariance regularization for self-supervised learning. In *The Tenth International Conference on Learning Representations, ICLR 2022, Virtual Event, April 25-29, 2022*. OpenReview.net, 2022.

[15] R. Bassily, A. Groce, J. Katz, and A. Smith. Coupled-worlds privacy: Exploiting adversarial uncertainty in statistical data privacy. In *2013 IEEE 54th Annual Symposium on Foundations of Computer Science*, pages 439–448, 2013.

[16] Amos Beimel, Shay Moran, Kobbi Nissim, and Uri Stemmer. Private Center Points and Learning of Halfspaces. *arXiv:1902.10731 [cs, stat]*, February 2019. arXiv: 1902.10731.

[17] Raghav Bhaskar, Abhishek Bhowmick, Vipul Goyal, Srivatsan Laxman, and Abhradeep Thakurta. Noiseless database privacy. In Dong Hoon Lee and Xiaoyun Wang, editors, *Advances in Cryptology – ASIACRYPT 2011*, pages 215–232, Berlin, Heidelberg, 2011. Springer Berlin Heidelberg.

[18] Federico Bianchi, Silvia Terragni, and Dirk Hovy. Pre-training is a hot topic: Contextualized document embeddings improve topic coherence. *arXiv preprint arXiv:2004.03974*, 2020.

[19] Vincent Bindschaedler and Reza Shokri. Synthesizing Plausible Privacy-Preserving Location Traces. In *2016 IEEE Symposium on Security and Privacy (SP)*, pages 546–563, May 2016. ISSN: 2375-1207.

[20] Vincent Bindschaedler, Reza Shokri, and Carl A. Gunter. Plausible deniability for privacy-preserving data synthesis. *Proceedings of the VLDB Endowment*, 10(5):481–492, January 2017.

[21] Andrea Bittau, Úlfar Erlingsson, Petros Maniatis, Ilya Mironov, Ananth Raghunathan,

David Lie, Mitch Rudominer, Ushasree Kode, Julien Tinnes, and Bernhard Seefeld. Prochlo: Strong privacy for analytics in the crowd. In *Proceedings of the 26th Symposium on Operating Systems Principles*, SOSP '17, pages 441–459, New York, NY, USA, 2017. ACM.

[22] Andrea Bittau, Úlfar Erlingsson, Petros Maniatis, Ilya Mironov, Ananth Raghunathan, David Lie, Mitch Rudominer, Ushasree Kode, Julien Tinnes, and Bernhard Seefeld. Prochlo: Strong privacy for analytics in the crowd. In *Proceedings of the 26th Symposium on Operating Systems Principles*, SOSP '17, page 441–459, New York, NY, USA, 2017. Association for Computing Machinery.

[23] Piotr Bojanowski, Edouard Grave, Armand Joulin, and Tomas Mikolov. Enriching word vectors with subword information. *Transactions of the Association for Computational Linguistics*, 5:135–146, 2017.

[24] Florian Bordes, Randall Balestriero, Quentin Garrido, Adrien Bardes, and Pascal Vincent. Guillotine regularization: Improving deep networks generalization by removing their head, 2022.

[25] Florian Bordes, Randall Balestriero, and Pascal Vincent. High fidelity visualization of what your self-supervised representation knows about. *Transactions on Machine Learning Research*, 2022.

[26] Florian Bordes, Randall Balestriero, and Pascal Vincent. Towards democratizing joint-embedding self-supervised learning, 2023.

[27] Wacha Bounliphone, Eugene Belilovsky, Matthew B. Blaschko, Ioannis Antonoglou, and Arthur Gretton. A test of relative similarity for model selection in generative models. In *4th International Conference on Learning Representations, ICLR 2016, San Juan, Puerto Rico, May 2-4, 2016, Conference Track Proceedings*, 2016.

[28] Alican Bozkurt, Babak Esmaeili, Dana H Brooks, Jennifer G Dy, and Jan-Willem van de Meent. Can vaes generate novel examples? 2018.

[29] Andrew Brock, Jeff Donahue, and Karen Simonyan. Large scale gan training for high fidelity natural image synthesis, 2018.

[30] Yang Cao, Yonghui Xiao, Li Xiong, and Liquan Bai. PriSTE: From Location Privacy to Spatiotemporal Event Privacy. In *2019 IEEE 35th International Conference on Data Engineering (ICDE)*, pages 1606–1609, April 2019. ISSN: 2375-026X.

[31] Yang Cao, Masatoshi Yoshikawa, Yonghui Xiao, and Li Xiong. Quantifying Differential Privacy under Temporal Correlations. In *2017 IEEE 33rd International Conference on Data Engineering (ICDE)*, pages 821–832, April 2017. ISSN: 2375-026X.

[32] Nicholas Carlini, Jamie Hayes, Milad Nasr, Matthew Jagielski, Vikash Sehwag, Florian Tramer, Borja Balle, Daphne Ippolito, and Eric Wallace. Extracting training data from diffusion models. *arXiv preprint arXiv:2301.13188*, 2023.

[33] Nicholas Carlini, Chang Liu, Úlfar Erlingsson, Jernej Kos, and Dawn Song. The secret sharer: Evaluating and testing unintended memorization in neural networks. In *28th USENIX Security Symposium (USENIX Security 19)*, pages 267–284, 2019.

[34] Nicholas Carlini, Florian Tramer, Eric Wallace, Matthew Jagielski, Ariel Herbert-Voss, Katherine Lee, Adam Roberts, Tom Brown, Dawn Song, Ulfar Erlingsson, et al. Extracting training data from large language models. In *30th USENIX Security Symposium (USENIX Security 21)*, pages 2633–2650, 2021.

[35] Nicholas Carlini, Florian Tramer, Eric Wallace, Matthew Jagielski, Ariel Herbert-Voss, Katherine Lee, Adam Roberts, Tom Brown, Dawn Song, Ulfar Erlingsson, Alina Oprea, and Colin Raffel. Extracting Training Data from Large Language Models. *arXiv:2012.07805 [cs]*, December 2020. arXiv: 2012.07805.

[36] Mathilde Caron, Ishan Misra, Julien Mairal, Priya Goyal, Piotr Bojanowski, and Armand Joulin. Unsupervised learning of visual features by contrasting cluster assignments. In *NeurIPS*, 2020.

[37] Mathilde Caron, Hugo Touvron, Ishan Misra, Herve Jegou, and Julien Mairal Piotr Bojanowski Armand Joulin. Emerging properties in self-supervised vision transformers. In *ICCV*, 2021.

[38] Ricardo Silva Carvalho, Theodore Vasiloudis, and Oluwaseyi Feyisetan. Tem: High utility metric differential privacy on text. *arXiv preprint arXiv:2107.07928*, 2021.

[39] Timothy M Chan. An optimal randomized algorithm for maximum tukey depth. In *SODA*, volume 4, pages 430–436, 2004.

[40] Tong Che, Yanran Li, Athul Paul Jacob, Yoshua Bengio, and Wenjie Li. Mode regularized generative adversarial networks. *CoRR*, abs/1612.02136, 2016.

[41] Jie Chen, Kian Hsiang Low, Yujian Yao, and Patrick Jaillet. Gaussian Process Decentralized Data Fusion and Active Sensing for Spatiotemporal Traffic Modeling and Prediction in Mobility-on-Demand Systems. *IEEE Transactions on Automation Science and Engineering*, 12(3):901–921, July 2015. Conference Name: IEEE Transactions on Automation Science and Engineering.

[42] Rui Chen, Benjamin C. Fung, Philip S. Yu, and Bipin C. Desai. Correlated network data publication via differential privacy. *The VLDB Journal*, 23(4):653–676, August 2014.

[43] Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey E. Hinton. A simple framework for contrastive learning of visual representations. In *ICML*, 2020.

[44] Xinlei Chen and Kaiming He. Exploring simple siamese representation learning. In *CVPR*, 2020.

[45] Albert Cheu, Adam Smith, Jonathan Ullman, David Zeber, and Maxim Zhilyaev. Distributed differential privacy via shuffling. In Yuval Ishai and Vincent Rijmen, editors, *Advances in Cryptology – EUROCRYPT 2019*, pages 375–403, Cham, 2019. Springer International Publishing.

[46] Krzysztof M Choromanski, Tony Jebara, and Kui Tang. Adaptive anonymity via *b*-matching. *Advances in Neural Information Processing Systems*, 26:3192–3200, 2013.

[47] Tore Dalenius. Towards a methodology for statistical disclosure control. *Statistik Tidskrift*, 15:429–444, 1977.

[48] Shaleen Deep, Xiao Hu, and Paraschos Koutris. Enumeration algorithms for conjunctive queries with projection. In *24th International Conference on Database Theory (ICDT 2021)*. Schloss Dagstuhl-Leibniz-Zentrum für Informatik, 2021.

[49] Shaleen Deep and Paraschos Koutris. Ranked enumeration of conjunctive query results. In *24th International Conference on Database Theory (ICDT 2021)*. Schloss Dagstuhl-Leibniz-Zentrum für Informatik, 2021.

[50] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pages 248–255. Ieee, 2009.

[51] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, 2019.

[52] Bolin Ding, Janardhan Kulkarni, and Sergey Yekhanin. Collecting telemetry data privately. In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems 30*, pages 3571–3580. Curran Associates, Inc., 2017.

[53] Jean-Paul Doignon, Aleksandar Pekeč, and Michel Regenwetter. The repeated insertion model for rankings: Missing link between two subset choice models. *Psychometrika*, 69(1):33–54, March 2004.

[54] Katerina Doka, Mingqiang Xue, Dimitrios Tsoumakos, and Panagiotis Karras. k-anonymization by freeform generalization. In *Proceedings of the 10th ACM Symposium on Information, Computer and Communications Security*, pages 519–530, 2015.

[55] Dheeru Dua and Casey Graff. UCI machine learning repository, 2017.

[56] Cynthia Dwork. *Differential Privacy*, volume 4052. July 2006.

[57] Cynthia Dwork, Krishnaram Kenthapadi, Frank McSherry, Ilya Mironov, and Moni Naor. Our Data, Ourselves: Privacy Via Distributed Noise Generation. In Serge Vaudenay, editor, *Advances in Cryptology - EUROCRYPT 2006*, volume 4004, pages 486–503. Springer Berlin Heidelberg, Berlin, Heidelberg, 2006. Series Title: Lecture Notes in Computer Science.

[58] Cynthia Dwork, Frank McSherry, Kobbi Nissim, and Adam Smith. Calibrating noise to sensitivity in private data analysis. In *Theory of cryptography*, pages 265–284. Springer, 2006.

[59] Cynthia Dwork, Frank McSherry, Kobbi Nissim, and Adam Smith. *Calibrating Noise to Sensitivity in Private Data Analysis*, volume 3876. March 2006.

[60] Cynthia Dwork and Moni Naor. On the difficulties of disclosure prevention in statistical databases or the case for differential privacy. January 2010.

[61] Cynthia Dwork and Aaron Roth. The algorithmic foundations of differential privacy. *Theoretical Computer Science*, 9(3-4):211–407, 2013.

[62] Cynthia Dwork and Aaron Roth. The algorithmic foundations of differential privacy. *Found. Trends Theor. Comput. Sci.*, pages 211–407, August 2014.

[63] Cynthia Dwork, Aaron Roth, et al. The algorithmic foundations of differential privacy. *Found. Trends Theor. Comput. Sci.*, 9(3-4):211–407, 2014.

[64] Gary Claxton Eric Lopez. Comparing private payer and medicare payment rates for select inpatient hospital services. *Kaiser Family Foundation*, Jul 2020.

[65] Úlfar Erlingsson, Vitaly Feldman, Ilya Mironov, Ananth Raghunathan, Kunal Talwar, and Abhradeep Thakurta. Amplification by shuffling: From local to central differential privacy via anonymity. In *Proceedings of the Thirtieth Annual ACM-SIAM Symposium on Discrete Algorithms*, SODA '19, page 2468–2479, USA, 2019. Society for Industrial and Applied Mathematics.

[66] Úlfar Erlingsson, Vasyl Pihur, and Aleksandra Korolova. Rappor: Randomized aggregatable privacy-preserving ordinal response. In *CCS*, 2014.

[67] Cristóbal Esteban, Stephanie L Hyland, and Gunnar Rätsch. Real-valued (medical) time series generation with recurrent conditional gans. *arXiv preprint arXiv:1706.02633*, 2017.

[68] Alexandre Evfimievski, Johannes Gehrke, and Ramakrishnan Srikant. Limiting privacy breaches in privacy preserving data mining. In *Proceedings of the Twenty-second ACM SIGMOD-SIGACT-SIGART Symposium on Principles of Database Systems*, PODS '03, pages 211–222, New York, NY, USA, 2003. ACM.

[69] Liyue Fan, Li Xiong, and Vaidy Sunderam. Differentially private multi-dimensional time series release for traffic monitoring. In *IFIP Annual Conference on Data and Applications Security and Privacy*, pages 33–48. Springer, 2013.

[70] Giulia Fanti, Vasyl Pihur, and Úlfar Erlingsson. Building a rappor with the unknown: Privacy-preserving learning of associations and data dictionaries, 2015.

[71] Vitaly Feldman. Does learning require memorization? a short tale about a long tail. In *Proceedings of the 52nd Annual ACM SIGACT Symposium on Theory of Computing*, pages 954–959, 2020.

[72] Vitaly Feldman, Audra McMillan, and Kunal Talwar. Hiding among the clones: A simple and nearly optimal analysis of privacy amplification by shuffling, 2020.

[73] Oluwaseyi Feyisetan, Borja Balle, Thomas Drake, and Tom Diethe. Privacy- and Utility-Preserving Textual Analysis via Calibrated Multivariate Perturbations. October 2019.

[74] Oluwaseyi Feyisetan and Shiva Kasiviswanathan. Private release of text embedding vectors. In *Proceedings of the First Workshop on Trustworthy Natural Language Processing*, pages 15–27, 2021.

[75] Geoffrey A. Fowler. Perspective | Smartphone data reveal which Americans are social distancing (and not). *Washington Post*, 2020.

[76] Matthew Fredrikson, Eric Lantz, Somesh Jha, Simon Lin, David Page, and Thomas Ristenpart. Privacy in pharmacogenetics: An {End-to-End} case study of personalized warfarin dosing. In *23rd USENIX Security Symposium (USENIX Security 14)*, pages 17–32, 2014.

[77] Jerome H Friedman. Greedy function approximation: a gradient boosting machine. *Annals of statistics*, pages 1189–1232, 2001.

[78] Johannes Gehrke, Michael Hay, Edward Lui, and Rafael Pass. Crowd-blending privacy. In Reihaneh Safavi-Naini and Ran Canetti, editors, *Advances in Cryptology – CRYPTO 2012*, pages 479–496, Berlin, Heidelberg, 2012. Springer Berlin Heidelberg.

[79] Johannes Gehrke, Edward Lui, and Rafael Pass. Towards privacy for social networks: A zero-knowledge based definition of privacy. In *Proceedings of the 8th Conference on Theory of Cryptography*, TCC'11, page 432–449, Berlin, Heidelberg, 2011. Springer-Verlag.

[80] Joseph Geumlek and Kamalika Chaudhuri. Profile-based privacy for locally private computations. In *IEEE International Symposium on Information Theory, ISIT 2019, Paris, France, July 7-12, 2019*, pages 537–541. IEEE, 2019.

[81] Arpita Ghosh and Robert Kleinberg. Inferential privacy guarantees for differentially private mechanisms. *CoRR*, abs/1603.01508, 2016.

[82] Ran Gilad-Bachrach and Chris J. C. Burges. The Median Hypothesis. June 2012.

[83] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. In *Advances in neural information processing systems*, pages 2672–2680, 2014.

[84] Andy Greenberg. Apple's 'differential privacy' is about collecting your data—but not *your* data. *Wired*, Jun 13 2016.

[85] Jean-Bastien Grill, Florian Strub, Florent Altché, Corentin Tallec, Pierre H. Richemond, Elena Buchatskaya, Carl Doersch, Bernardo Avila Pires, Zhaohan Daniel Guo, Mohammad Gheshlaghi Azar, Bilal Piot, Koray Kavukcuoglu, Rémi Munos, and Michal Valko. Bootstrap your own latent: A new approach to self-supervised learning. In *NeurIPS*, 2020.

[86] Krzysztof Grining and Marek Klonowski. Towards extending noiseless privacy: Dependent data and more practical approach. In *Proceedings of the 2017 ACM on Asia Conference on Computer and Communications Security*, ASIA CCS '17, page 546–560, New York, NY, USA, 2017. Association for Computing Machinery.

[87] Ishaan Gulrajani, Colin Raffel, and Luke Metz. Towards GAN benchmarks which require generalization. *CoRR*, abs/2001.03653, 2020.

[88] Chuan Guo, Brian Karrer, Kamalika Chaudhuri, and Laurens van der Maaten. Bounding training data reconstruction in private (deep) learning. *arXiv preprint arXiv:2201.12383*, 2022.

[89] Prakhar Gupta, Matteo Pagliardini, and Martin Jaggi. Better word embeddings by disentangling contextual n-gram information. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 933–939, 2019.

[90] Kaiming He, Xinlei Chen, Saining Xie, Yanghao Li, Piotr Dollár, and Ross Girshick.

Masked autoencoders are scalable vision learners. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 16000–16009, June 2022.

[91] Xi He, Ashwin Machanavajjhala, and Bolin Ding. Blowfish privacy: Tuning privacy-utility trade-offs using policies. In *Proceedings of the 2014 ACM SIGMOD International Conference on Management of Data*, SIGMOD '14, page 1447–1458, New York, NY, USA, 2014. Association for Computing Machinery.

[92] Martin Heusel, Hubert Ramsauer, Thomas Unterthiner, Bernhard Nessler, and Sepp Hochreiter. Gans trained by a two time-scale update rule converge to a local nash equilibrium. 2017.

[93] Yangsibo Huang, Zhao Song, Danqi Chen, Kai Li, and Sanjeev Arora. TextHide: Tackling data privacy in language understanding tasks. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 1368–1382, Online, November 2020. Association for Computational Linguistics.

[94] Ganesh Jawahar, Benoît Sagot, and Djamé Seddah. What does bert learn about the structure of language? In *ACL 2019-57th Annual Meeting of the Association for Computational Linguistics*, 2019.

[95] Bargav Jayaraman and David Evans. Are attribute inference attacks just imputation? *arXiv preprint arXiv:2209.01292*, 2022.

[96] Peter Kairouz, H Brendan McMahan, Brendan Avent, Aurélien Bellet, Mehdi Bennis, Arjun Nitin Bhagoji, Kallista Bonawitz, Zachary Charles, Graham Cormode, Rachel Cummings, et al. Advances and open problems in federated learning. *arXiv preprint arXiv:1912.04977*, 2019.

[97] Gautam Kamath, Jerry Li, Vikrant Singhal, and Jonathan Ullman. Privately Learning High-Dimensional Distributions. In *Conference on Learning Theory*, pages 1853–1902. PMLR, June 2019. ISSN: 2640-3498.

[98] S. Kasiviswanathan and A. Smith. On the 'semantics' of differential privacy: A bayesian formulation. *J. Priv. Confidentiality*, 6, 2014.

[99] S. P. Kasiviswanathan, H. K. Lee, K. Nissim, S. Raskhodnikova, and A. Smith. What can we learn privately? In *2008 49th Annual IEEE Symposium on Foundations of Computer Science*, pages 531–540, 2008.

[100] Yusuke Kawamoto and Takao Murakami. Differentially private obfuscation mechanisms for hiding probability distributions. *CoRR*, abs/1812.00939, 2018.

[101] Gavin Kerrigan, Dylan Slack, and Jens Tuyls. Differentially Private Language Models Benefit from Public Pre-training. *arXiv:2009.05886 [cs]*, October 2020. arXiv: 2009.05886.

[102] Daniel Kifer. Attacks on privacy and deFinetti's theorem. In *Proceedings of the 2009 ACM SIGMOD International Conference on Management of data*, SIGMOD '09, pages 127–138, Providence, Rhode Island, USA, June 2009. Association for Computing Machinery.

[103] Daniel Kifer and Ashwin Machanavajjhala. No free lunch in data privacy. In *Proceedings of the 2011 ACM SIGMOD International Conference on Management of Data*, SIGMOD '11, page 193–204, New York, NY, USA, 2011. Association for Computing Machinery.

[104] Daniel Kifer and Ashwin Machanavajjhala. No free lunch in data privacy. In *Proceedings of the 2011 ACM SIGMOD International Conference on Management of data*, SIGMOD '11, pages 193–204, Athens, Greece, June 2011. Association for Computing Machinery.

[105] Daniel Kifer and Ashwin Machanavajjhala. Pufferfish: A framework for mathematical privacy definitions. *ACM Trans. Database Syst.*, 39(1), January 2014.

[106] Kihwan Kim, Dongryeol Lee, and Irfan Essa. Gaussian process regression flow for analysis of motion trajectories. In *2011 International Conference on Computer Vision*, pages 1164–1171, November 2011. ISSN: 2380-7504.

[107] Diederik P Kingma and Max Welling. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*, 2013.

[108] Tuomas Kynkäänniemi, Tero Karras, Samuli Laine, Jaakko Lehtinen, and Timo Aila. Improved precision and recall metric for assessing generative models. Advances in Neural Information Processing Systems 33 (NIPS 2019), 2019.

[109] Rachel La Corte. Supreme court: State employee birthdates are public record. https://apnews.com/article/c1ff652f271947b2884dfe1216a11bc2/, 2019.

[110] Ken Lang. Home Page for 20 Newsgroups Data Set, 1995.

[111] Yann LeCun and Corinna Cortes. MNIST handwritten digit database. 2010.

[112] Jaehoon Lee, Yasaman Bahri, Roman Novak, Samuel S. Schoenholz, Jeffrey Pennington, and Jascha Sohl-Dickstein. Deep Neural Networks as Gaussian Processes. *arXiv:1711.00165 [cs, stat]*, March 2018. arXiv: 1711.00165.

[113] Bohan Li, Hao Zhou, Junxian He, Mingxuan Wang, Yiming Yang, and Lei Li. On the sentence embeddings from bert for semantic textual similarity. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages

9119–9130, 2020.

[114] Tao Li and Chris Clifton. Differentially Private Imaging via Latent Space Manipulation. *arXiv:2103.05472 [cs]*, April 2021. arXiv: 2103.05472.

[115] B. Liang and Z.J. Haas. Predictive distance-based mobility management for PCS networks. In *IEEE INFOCOM '99. Conference on Computer Communications. Proceedings. Eighteenth Annual Joint Conference of the IEEE Computer and Communications Societies. The Future is Now (Cat. No.99CH36320)*, volume 3, pages 1377–1384 vol.3, March 1999. ISSN: 0743-166X.

[116] Katrina Ligett, Charlotte Peale, and Omer Reingold. Bounded-Leakage Differential Privacy. In Aaron Roth, editor, *1st Symposium on Foundations of Responsible Computing (FORC 2020)*, volume 156 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 10:1–10:20, Dagstuhl, Germany, 2020. Schloss Dagstuhl–Leibniz-Zentrum für Informatik.

[117] Changchang Liu, Supriyo Chakraborty, and Prateek Mittal. Dependence makes you vulnberable: Differential privacy under dependent tuples. In *NDSS*. The Internet Society, 2016.

[118] Tong Liu, P. Bahl, and I. Chlamtac. Mobility modeling, location tracking, and trajectory prediction in wireless ATM networks. *IEEE Journal on Selected Areas in Communications*, 16(6):922–936, August 1998. Conference Name: IEEE Journal on Selected Areas in Communications.

[119] Xiaodong Liu, Pengcheng He, Weizhu Chen, and Jianfeng Gao. Multi-task deep neural networks for natural language understanding. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4487–4496, 2019.

[120] Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*, 2019.

[121] David Lopez-Paz and Maxime Oquab. Revisiting classifier two-sample tests. *arXiv preprint arXiv:1610.06545*, 2016.

[122] Andrew L. Maas, Raymond E. Daly, Peter T. Pham, Dan Huang, Andrew Y. Ng, and Christopher Potts. Learning word vectors for sentiment analysis. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 142–150, Portland, Oregon, USA, June 2011. Association for Computational Linguistics.

[123] Ashwin Machanavajjhala, Daniel Kifer, Johannes Gehrke, and Muthuramakrishnan Venki-

tasubramaniam. L-diversity: Privacy beyond k-anonymity. *ACM Transactions on Knowledge Discovery from Data*, 1(1):3–es, March 2007.

[124] C. L. MALLOWS. NON-NULL RANKING MODELS. I. *Biometrika*, 44(1-2):114–130, 06 1957.

[125] Henry B Mann and Donald R Whitney. On a test of whether one of two random variables is stochastically larger than the other. *The annals of mathematical statistics*, pages 50–60, 1947.

[126] Frank McSherry and Kunal Talwar. Mechanism Design via Differential Privacy. October 2007.

[127] Shagufta Mehnaz, Sayanton V Dibbo, Ehsanul Kabir, Ninghui Li, and Elisa Bertino. Are your sensitive attributes private? novel model inversion attribute inference attacks on classification models. *arXiv preprint arXiv:2201.09370*, 2022.

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

[129] Khalil Mrini, Emilia Farcas, and Ndapa Nakashole. Recursive tree-structured self-attention for answer sentence selection. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 4651–4661, Online, August 2021. Association for Computational Linguistics.

[130] Tobias Nef, Prabitha Urwyler, Marcel Büchler, Ioannis Tarnanas, Reto Stucki, Dario Cazzoli, René Müri, and Urs Mosimann. Evaluation of three state-of-the-art classifiers for recognition of activities of daily living from smart home ambient data. *Sensors*, 15(5):11725–11740, 2015.

[131] Alexandru Niculescu-Mizil and Rich Caruana. Predicting good probabilities with supervised learning. In *Proceedings of the 22nd international conference on Machine learning*, pages 625–632, 2005.

[132] Xudong Pan, Mi Zhang, Shouling Ji, and Min Yang. Privacy risks of general-purpose language models. In *2020 IEEE Symposium on Security and Privacy (SP)*, pages 1314–1331. IEEE, 2020.

[133] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Köpf, Edward Yang, Zach DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. *PyTorch: An Imperative Style, High-Performance Deep Learning Library*. Curran Associates Inc., Red

Hook, NY, USA, 2019.

[134] Chen Qu, Weize Kong, Liu Yang, Mingyang Zhang, Michael Bendersky, and Marc Najork. Privacy-Adaptive BERT for Natural Language Understanding. *arXiv:2104.07504 [cs]*, April 2021. arXiv: 2104.07504.

[135] Nils Reimers and Iryna Gurevych. Sentence-BERT: Sentence Embeddings using Siamese BERT-Networks. *arXiv:1908.10084 [cs]*, August 2019. arXiv: 1908.10084.

[136] Eitan Richardson and Yair Weiss. On gans and gmms. In *Advances in Neural Information Processing Systems*, pages 5847–5858, 2018.

[137] Benedek Rozemberczki, Carl Allen, and Rik Sarkar. Multi-scale attributed node embedding. *arXiv preprint arXiv:1909.13021*, 2019.

[138] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, Alexander C. Berg, and Li Fei-Fei. ImageNet Large Scale Visual Recognition Challenge. *International Journal of Computer Vision (IJCV)*, 115(3):211–252, 2015.

[139] Alexandre Sablayrolles, Matthijs Douze, Cordelia Schmid, Yann Ollivier, and Hervé Jégou. White-box vs black-box: Bayes optimal strategies for membership inference. In *International Conference on Machine Learning*, pages 5558–5567. PMLR, 2019.

[140] Mehdi S. M. Sajjadi, Olivier Bachem, Mario Lucic, Olivier Bousquet, and Sylvain Gelly. Assessing generative models via precision and recall. Advances in Neural Information Processing Systems 31 (NIPS 2017), 2018.

[141] Ahmed Salem, Yang Zhang, Mathias Humbert, Pascal Berrang, Mario Fritz, and Michael Backes. Ml-leaks: Model and data independent membership inference attacks and defenses on machine learning models. *arXiv preprint arXiv:1806.01246*, 2018.

[142] Tim Salimans, Ian Goodfellow, Wojciech Zaremba, Vicki Cheung, Alec Radford, and Xi Chen. Improved techniques for training gans. In *Advances in neural information processing systems*, pages 2234–2242, 2016.

[143] Reza Shokri, Marco Stronati, Congzheng Song, and Vitaly Shmatikov. Membership inference attacks against machine learning models. In *2017 IEEE symposium on security and privacy (SP)*, pages 3–18. IEEE, 2017.

[144] Congzheng Song and Ananth Raghunathan. Information Leakage in Embedding Models. *arXiv:2004.00053 [cs, stat]*, August 2020. arXiv: 2004.00053.

[145] Shuang Song, Yizhen Wang, and Kamalika Chaudhuri. Pufferfish Privacy Mechanisms

for Correlated Data. In *Proceedings of the 2017 ACM International Conference on Management of Data*, SIGMOD '17, pages 1291–1306, Chicago, Illinois, USA, May 2017. Association for Computing Machinery.

[146] Shuang Song, Yizhen Wang, and Kamalika Chaudhuri. Pufferfish privacy mechanisms for correlated data. In *Proceedings of the 2017 ACM International Conference on Management of Data*, SIGMOD '17, page 1291–1306, New York, NY, USA, 2017. Association for Computing Machinery.

[147] Emma Strubell, Patrick Verga, Daniel Andor, David Weiss, and Andrew McCallum. Linguistically-informed self-attention for semantic role labeling. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 5027–5038, 2018.

[148] Dougal J. Sutherland, Hsiao-Yu Fish Tung, Heiko Strathmann, Soumyajit De, Aaditya Ramdas, Alexander J. Smola, and Arthur Gretton. Generative models and model criticism via optimized maximum mean discrepancy. *CoRR*, abs/1611.04488, 2016.

[149] Latanya Sweeney. Achieving k-anonymity privacy protection using generalization and suppression. *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems*, 10(5):571–588, October 2002.

[150] Tamir Tassa, Arnon Mazza, and Aristides Gionis. k-concealment: An alternative model of k-type anonymity. *Trans. Data Priv.*, 5(1):189–222, 2012.

[151] Lucas Theis, Aäron van den Oord, and Matthias Bethge. A note on the evaluation of generative models. In *4th International Conference on Learning Representations, ICLR 2016, San Juan, Puerto Rico, May 2-4, 2016, Conference Track Proceedings*, 2016.

[152] Tan Thongtan and Tanasanee Phienthrakul. Sentiment classification using document embeddings trained with cosine similarity. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics: Student Research Workshop*, pages 407–414, 2019.

[153] M. C. Tschantz, S. Sen, and A. Datta. Sok: Differential privacy as a causal property. In *2020 IEEE Symposium on Security and Privacy (SP)*, pages 354–371, 2020.

[154] John W Tukey. Mathematics and the picturing of data. In *Proceedings of the International Congress of Mathematicians, Vancouver, 1975*, volume 2, pages 523–531, 1975.

[155] Dmitry Ulyanov, Andrea Vedaldi, and Victor Lempitsky. Deep image prior. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 9446–9454, 2018.

[156] Natasha; Keller Michael; Krolik Aaron Valentino-DeVryes, Jennifer; Singer. Your apps know where you were last night, and they're not keeping it secret. *The New York Times*, 2018.

[157] Lieven Vandenberghe. The cvxopt linear and quadratic cone program solvers. *Online: http://cvxopt. org/documentation/coneprog. pdf*, 2010.

[158] Lieven Vandenberghe and Stephen Boyd. Semidefinite Programming. *SIAM Review*, 38(1):49–95, March 1996.

[159] Mengting Wan and Julian J. McAuley. Item recommendation on monotonic behavior chains. In Sole Pera, Michael D. Ekstrand, Xavier Amatriain, and John O'Donovan, editors, *Proceedings of the 12th ACM Conference on Recommender Systems, RecSys 2018, Vancouver, BC, Canada, October 2-7, 2018*, pages 86–94. ACM, 2018.

[160] Stanley L Warner. Randomized response: A survey technique for eliminating evasive answer bias. *Journal of the American Statistical Association*, 60 60, no. 309:63–69, 1965.

[161] Lauren Watson, Chuan Guo, Graham Cormode, and Alex Sablayrolles. On the importance of difficulty calibration in membership inference attacks. *arXiv preprint arXiv:2111.08440*, 2021.

[162] Ryan Webster, Julien Rabin, Loïc Simon, and Frédéric Jurie. Detecting overfitting of deep generative networks via latent recovery. In *IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2019, Long Beach, CA, USA, June 16-20, 2019*, pages 11273–11282. Computer Vision Foundation / IEEE, 2019.

[163] Wai Kit Wong, Nikos Mamoulis, and David Wai Lok Cheung. Non-homogeneous generalization in privacy preserving data publishing. In *Proceedings of the 2010 ACM SIGMOD International Conference on Management of data*, pages 747–758, 2010.

[164] Yuhuai Wu, Yuri Burda, Ruslan Salakhutdinov, and Roger B. Grosse. On the quantitative analysis of decoder-based generative models. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*, 2017.

[165] Xiaokui Xiao and Yufei Tao. Anatomy: Privacy and Correlation Preserving Publication. January 2006.

[166] Yonghui Xiao and Li Xiong. Protecting locations with differential privacy under temporal correlations. In *Proceedings of the 22nd ACM SIGSAC Conference on Computer and Communications Security*, pages 1298–1309. ACM, 2015.

[167] Shangyu Xie and Yuan Hong. Reconstruction attack on instance encoding for language

understanding. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 2038–2044, 2021.

[168] Nan Xu, Oluwaseyi Feyisetan, Abhinav Aggarwal, Zekun Xu, and Nathanael Teissier. Density-Aware Differentially Private Textual Perturbations Using Truncated Gumbel Noise. *The International FLAIRS Conference Proceedings*, 34(1), April 2021.

[169] Qiantong Xu, Gao Huang, Yang Yuan, Chuan Guo, Yu Sun, Felix Wu, and Kilian Weinberger. An empirical study on evaluation metrics of generative adversarial networks. *arXiv preprint arXiv:1806.07755*, 2018.

[170] Mingqiang Xue, Panagiotis Karras, Chedy Raïssi, Jaideep Vaidya, and Kian-Lee Tan. Anonymizing set-valued data by nonreciprocal recoding. In *Proceedings of the 18th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 1050–1058, 2012.

[171] Bin Yang, Issei Sato, and Hiroshi Nakagawa. Bayesian Differential Privacy on Correlated Data. In *Proceedings of the 2015 ACM SIGMOD International Conference on Management of Data*, SIGMOD '15, pages 747–762, Melbourne, Victoria, Australia, May 2015. Association for Computing Machinery.

[172] Bin Yang, Issei Sato, and Hiroshi Nakagawa. Bayesian differential privacy on correlated data. In *Proceedings of the 2015 ACM SIGMOD International Conference on Management of Data*, SIGMOD '15, page 747–762, New York, NY, USA, 2015. Association for Computing Machinery.

[173] Zichao Yang, Diyi Yang, Chris Dyer, Xiaodong He, Alex Smola, and Eduard Hovy. Hierarchical attention networks for document classification. In *Proceedings of the 2016 conference of the North American chapter of the association for computational linguistics: human language technologies*, pages 1480–1489, 2016.

[174] Jiayuan Ye, Aadyaa Maddi, Sasi Kumar Murakonda, Vincent Bindschaedler, and Reza Shokri. Enhanced membership inference attacks against machine learning models. *arXiv preprint arXiv:2111.09679*, 2021.

[175] David Yenicelik, Florian Schmidt, and Yannic Kilcher. How does bert capture semantics? a closer look at polysemous words. In *Proceedings of the Third BlackboxNLP Workshop on Analyzing and Interpreting Neural Networks for NLP*, pages 156–162, 2020.

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

[177] Yang You, Igor Gitman, and Boris Ginsburg. Large batch training of convolutional

networks. *arXiv preprint arXiv:1708.03888*, 2017.

[178] Jiahui Yu, Zhe Lin, Jimei Yang, Xiaohui Shen, Xin Lu, and Thomas S Huang. Generative image inpainting with contextual attention. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 5505–5514, 2018.

[179] Xiang Yue, Minxin Du, Tianhao Wang, Yaliang Li, Huan Sun, and Sherman S. M. Chow. Differential Privacy for Text Analytics via Natural Text Sanitization. *arXiv:2106.01221 [cs]*, June 2021. arXiv: 2106.01221.

[180] Fransisco Zamora-Martinez, Pablo Romeu, Pablo Botella-Rocamora, and Juan Pardo. On-line learning of indoor temperature forecasting models towards energy efficiency. *Energy and Buildings*, 83:162–172, 2014.

[181] Jure Zbontar, Li Jing, Ishan Misra, Yann LeCun, and Stéphane Deny. Barlow twins: Self-supervised learning via redundancy reduction. *arXiv preprint arxiv:2103.03230*, 2021.

[182] Hui Zeng and Xiaohui Cui. Simclrt: A simple framework for contrastive learning of rumor tracking. *Eng. Appl. Artif. Intell.*, 110:104757, 2022.

[183] Richard Zhang, Phillip Isola, Alexei A Efros, Eli Shechtman, and Oliver Wang. The unreasonable effectiveness of deep features as a perceptual metric. In *CVPR*, 2018.

[184] Wanrong Zhang, Olga Ohrimenko, and Rachel Cummings. Attribute privacy: Framework and mechanisms, 2020.

[185] Yu Zheng, Xing Xie, Wei-Ying Ma, et al. Geolife: A collaborative social networking service among user, location and trajectory. *IEEE Data Eng. Bull.*, 33(2):32–39, 2010.

[186] Shuheng Zhou, Katrina Ligett, and Larry Wasserman. Differential privacy with compression. In *2009 IEEE International Symposium on Information Theory*, pages 2718–2722, June 2009. ISSN: 2157-8117.

[187] Banghua Zhu, Jiantao Jiao, and Jacob Steinhardt. When does the Tukey Median work? In *2020 IEEE International Symposium on Information Theory (ISIT)*, pages 1201–1206, Los Angeles, CA, USA, June 2020. IEEE.

[188] T. Zhu, P. Xiong, G. Li, and W. Zhou. Correlated differential privacy: Hiding information in non-iid data set. *IEEE Transactions on Information Forensics and Security*, 10(2):229–242, 2015.