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

Something nice

EPIGRAPH

Something pithy

Someone smart

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

PREFACE

Almost nothing is said in the manual about the preface. There is no indication about how it is to be typeset. Given that, one is forced to simply typeset it and hope it is accepted. It is, however, optional and may be omitted.

ACKNOWLEDGEMENTS

I would like to acknowledge Professor Eta Theta for his support as the chair of my committee. Through multiple drafts and many long nights, his guidance has proved to be invaluable.

I would also like to acknowledge the "Smith Clan" of lab 28, without whom my research would have no doubt taken fives times as long. It is their support that helped me in an immeasureable way.

Chapter 2, in full, is a reprint of the material as it appears in Numerical Grid Generational in Computational Fluid Mechanics 2009. Smith, Laura; Smith, Jane D., Pineridge Press, 2009. The dissertation author was the primary investigator and author of this paper.

Chapter 3, in part, has been submitted for publication of the material as it may appear in Education Mechanics, 2009, Smith, Laura; Smith, Jane D., Trailor Press, 2009. The dissertation author was the primary investigator and author of this paper.

Chapter 5, in part is currently being prepared for submission for publication of the material. Smith, Laura; Smith, Jane D. The dissertation author was the primary investigator and author of this material.

VITA

1996	Bachelor of Arts, University of California, Berkeley
1996–2000	U.S. Marines
2000–2002	Teaching Assistant, Department of Mechanical Engineering University of California, San Diego
2002–2006	Research Assistant, University of California, San Diego
2010	Doctor of Philosophy, University of California, San Diego

PUBLICATIONS

"Distributions of Control Points in a System for Analysis of Stress Distribution" IRE Transactions of the I.R.E. Professional Group on Automatic Control, vol. AC-7, pp 272–289, September 2005

FIELDS OF STUDY

Major Field: Engineering (Specialization or Focused Studies)

Studies in Applied Mathematics
Professors Alpha Beta and Gamma Delta

Studies in Mechanices
Professors Epsilon Zeta and Eta Theta

Studies in Electromagnetism
Professors Iota Kappa and Lambda Mu

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

The Abstract begins here. The abstract is limited to 350 words for a doctoral dissertation. It should consist of a short statement of the problem, a brief explanation of the methods and procedures employed in generating the data, and a condensed summary of the findings of the study. The abstract may continute onto a second page if necessary. The text of the abstract must be double spaced.

Introduction

This optional section is barely described in the OGS manual other than saying it is optional and that it appears in the table of contents between the Abstract and the first chapter.

No formatting guidelines appear so presumably, it should be formatted like an ordinary chapter. It should appear after the `\mainmatter` macro because it should start on page 1.

Chapter 1

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

1.1 Introduction

Self-supervised learning (SSL) [26, 27, 100, 7, 21, 46] 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.

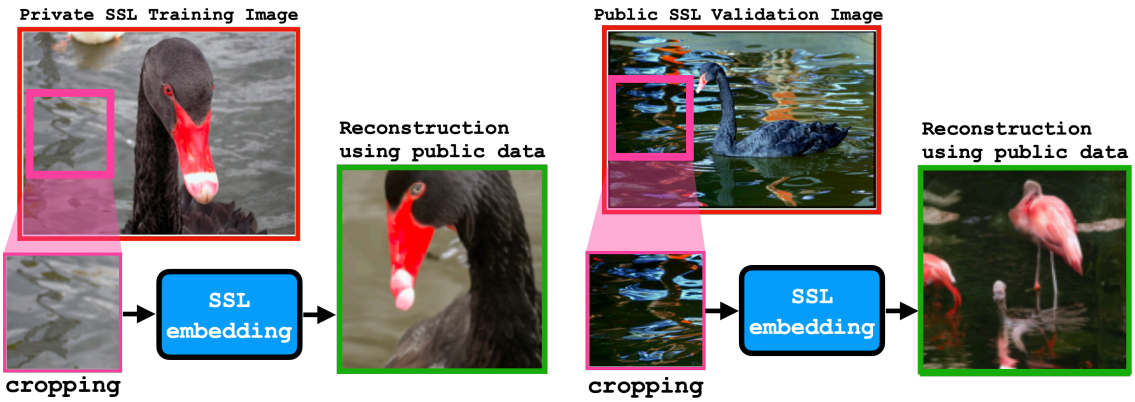


Figure 1.1. Left: Reconstruction of an SSL training image from a crop containing only the background. The SSL model memorizes the association of this *specific* patch of water (pink square) to this *specific* foreground object (a black swan) in its embedding, which we decode to visualize the full training 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 [28]—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 [19, 17], 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 [26], Barlow Twins [100], VICReg [7] and Dino [22]), 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 [11]. It has been shown that SSL representations encode richer visual details about input images than supervised models do [12]. 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 [96, 36]. Privacy attacks in ML range from the simplest and best-studied *membership inference attacks* [78, 76, 74] to *attribute inference* [39, 66, 50] and *data reconstruction* [19, 6, 44] 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 [18, 19]: 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 [17]. 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* [11], 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 [98, 85].

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.

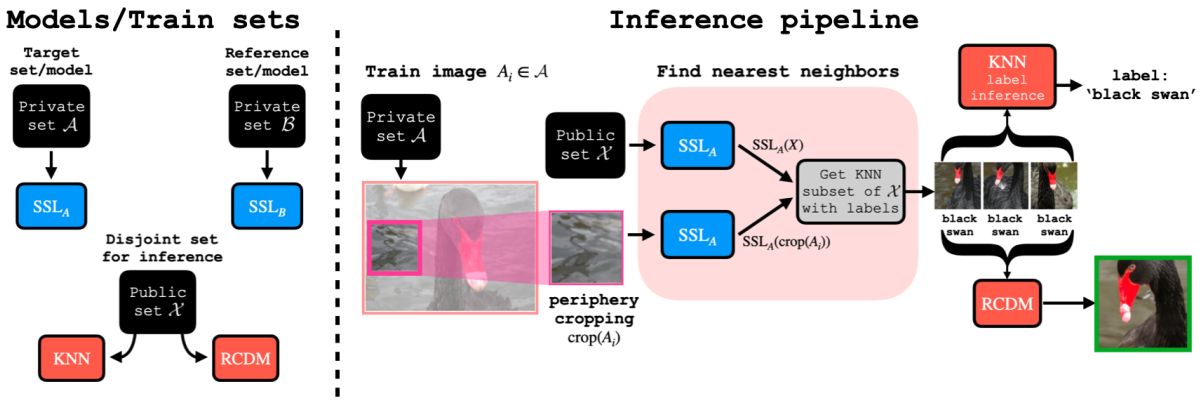


Figure 1.2. Overview of testing methodology. **Left:** Data is split into *target set* $\mathscr{A}$, *reference set* $\mathscr{B}$ and *public set* $\mathscr{X}$ that are pairwise disjoint. $\mathscr{A}$ and $\mathscr{B}$ are used to train two SSL models SSL_A and SSL_B in the same manner. $\mathscr{X}$ 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; [32, 33])—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 [28]. 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?)** [12], 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 [7]. 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 [36, 87, 94]. 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.

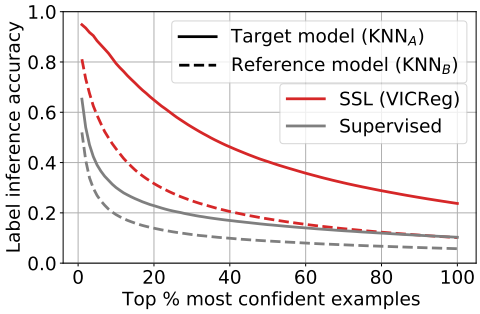


Figure 1.3. Accuracy of label inference using the target model (trained on $\mathscr{A}$) vs. the reference model (trained on $\mathscr{B}$) on the top % most confident examples $A_i \in \mathscr{A}$ using only $\text{crop}(A_i)$. For VICReg, there is a large accuracy gap between the two models, indicating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

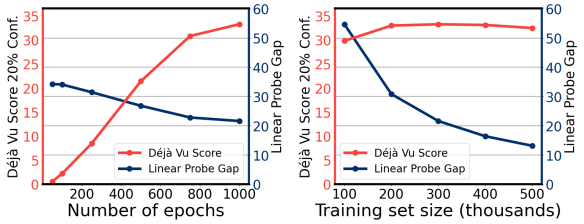
The *déjà vu* score.

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.

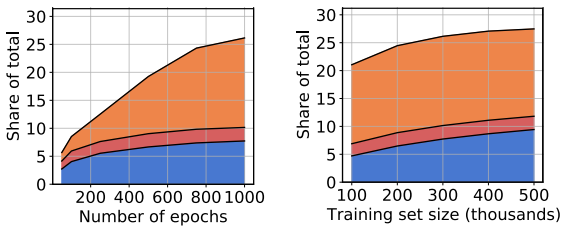
A standard method for measuring SSL performance is to train a linear classifier—what we call a 'linear probe'—on its embeddings and compute its performance on a held out test set. From a learning theory standpoint, one might expect the linear probe's train-test accuracy gap to be indicative of memorization: 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

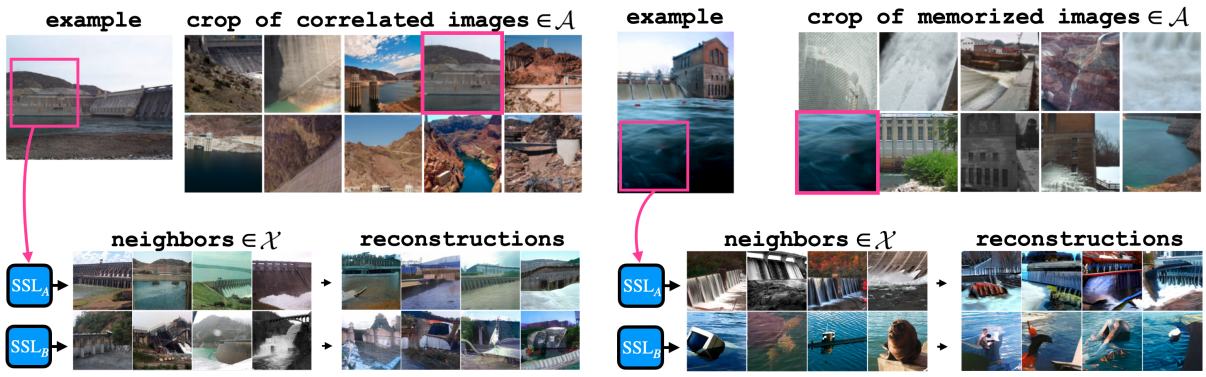
embedding of SSL_A does. To measure the pervasiveness of *déjà vu* memorization, we compare the size of the memorized and misrepresented sets. Figure 1.5 shows how the four categories of examples change with number of training epochs and training set size. The unassociated set is not shown since the 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 [12] 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.

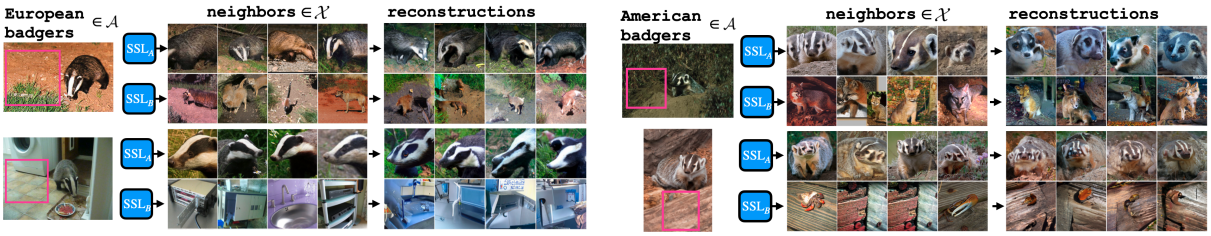
RCDM is a conditional generative model that is trained on the *backbone embedding* of images $X_i \in \mathscr{X}$ to generate an image that resembles X_i. All training images are first face-blurred for privacy purposes. **(author?)** [12] showed that the backbone embedding of SSL models contains more 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.

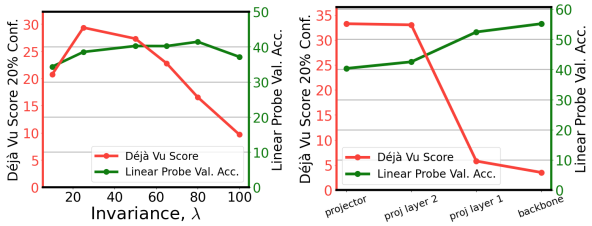
Visualizing Memorization Beyond Class Label.

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

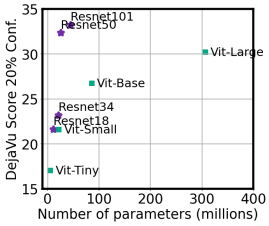
We cannot yet make claims on why *déjà vu* occurs so strongly for some SSL training settings and not for others. To gain some intuition for future work, we present additional

[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 [11]—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[42]	8.0	54.3/59.4
SimCLR[26]	10.0	44.2/54.1
Dino[22]	14.5	26.3/55.7
Barlow T.[100]	30.5	33.7/54.4
VICReg[7]	**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[11] (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

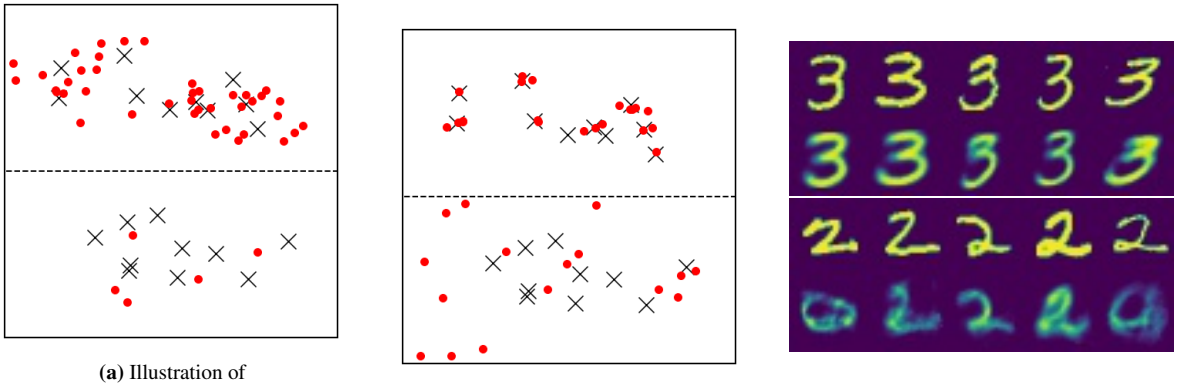
Overfitting is a basic stumbling block of any learning process. While it has been studied in great detail in the context of supervised learning, it has received much less attention in the unsupervised setting, despite being just as much of a problem.

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) [54] and Generative Adversarial Networks (GANs) [41]. 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 [77, 75, 89, 47]; 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 [77, 62, 72, 75, 92, 89] . 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) [47], Precision and Recall test [75], and extensions thereof [55, 25] 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 [72] also compares aggregate training and generated samples, but without the embedding step. The parametric Kernel MMD method proposed by [81] 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 [62]. 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 [92] – 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 [92] (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.

[14] explores three-sample testing, but for comparing the performance of different models, not for detecting overfitting. [35] uses the three-sample test proposed by [14] 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 [43] 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 [88] 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 [47, 75], 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 [64]. 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 [47, 75] 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 [72], 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:

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

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

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

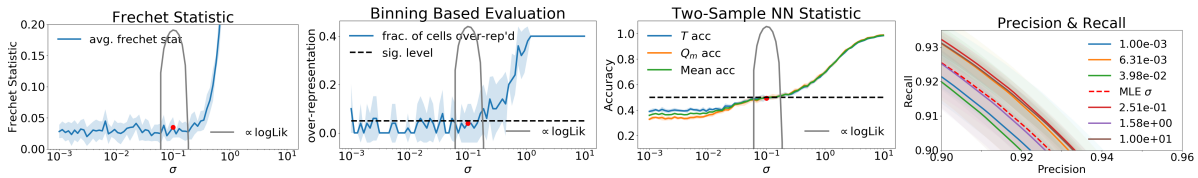
where the posterior probability that a random draw $x \sim q_\sigma(x)$ comes from the Gaussian component centered at training point t is

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

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 [62], FID [47], Binning-Based Evaluation [72], and Precision & Recall [75], 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 σ_{MLE}. This makes sense; as points in Q_m become closer to points in T, the two-sample NN

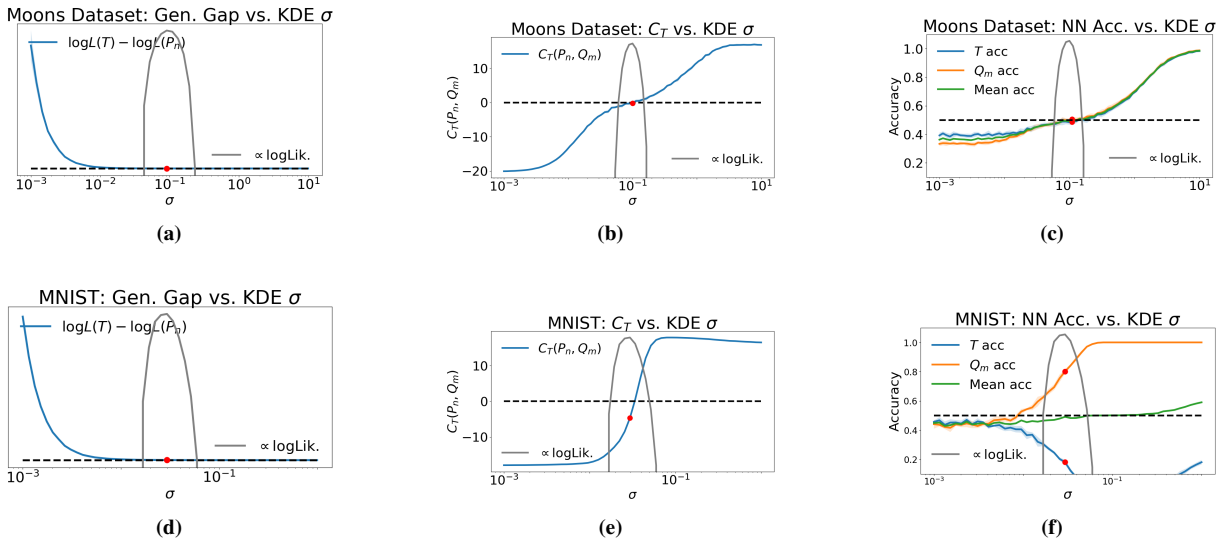


Figure 2.3. $C_T(P_n, Q_m)$ vs. NN baseline and generalization gap on moons and MNIST digits datasets. **(a,b,c)** compare the three methods on the moons dataset. **(d,e,f)** compare the three methods on MNIST. In both data settings, the C_T statistic is far more sensitive to the data-copying regime $\sigma \ll \sigma_{\text{MLE}}$ 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.

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 [82], 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 [57].

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 [102] (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

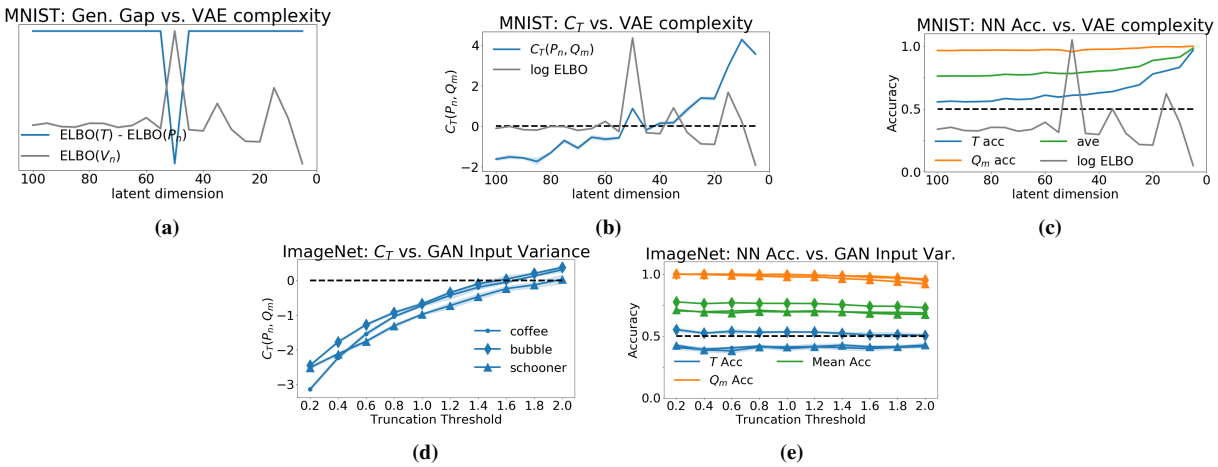
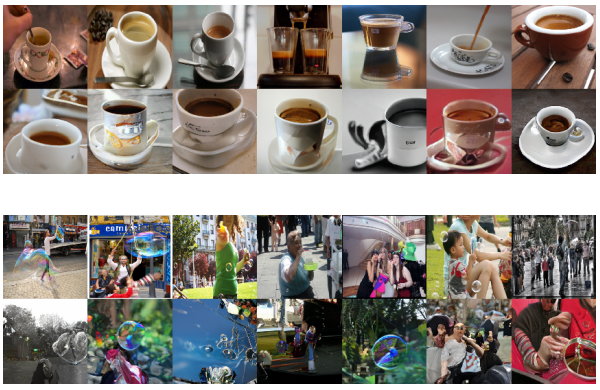


Figure 2.4. Neural model data-copying: figures **(b)** and **(d)** demonstrate the C_T statistic identifying data-copying in an MNIST VAE and ImageNet GAN as they range from heavily over-fit to underfit. **(c)** and **(e)** demonstrate the relative insensitivity of the NN baseline to this overfitting, 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 [15]. 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*' [16], trained on the Imagenet 12 dataset [73]. 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 [92], 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 [29, 61, 3], pushing the state of the art in a variety of tasks [80, 60, 67]. While language models capture meaning and various linguistic properties of text [49, 95], 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 [68, 2, 20]. Reconstruction attacks [90] have even successfully broken through private learning schemes that rely on encryption-type methods [48].

As of now, there is no broad agreement on what constitutes good privacy for natural language [51]. **(author?)** [48] argue that different applications and models require different privacy definitions. Several emerging works propose to apply Metric Differential Privacy [4] at the word level [37, 38, 23, 70, 99, 91] . 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

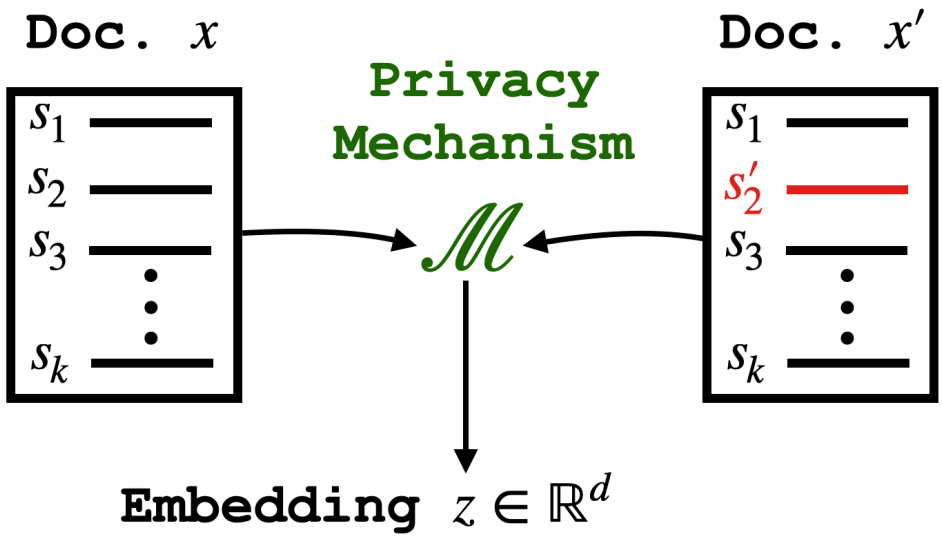


Figure 3.1. x and x' yield $z \in \mathbb{R}^d$ with similar probability.

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'

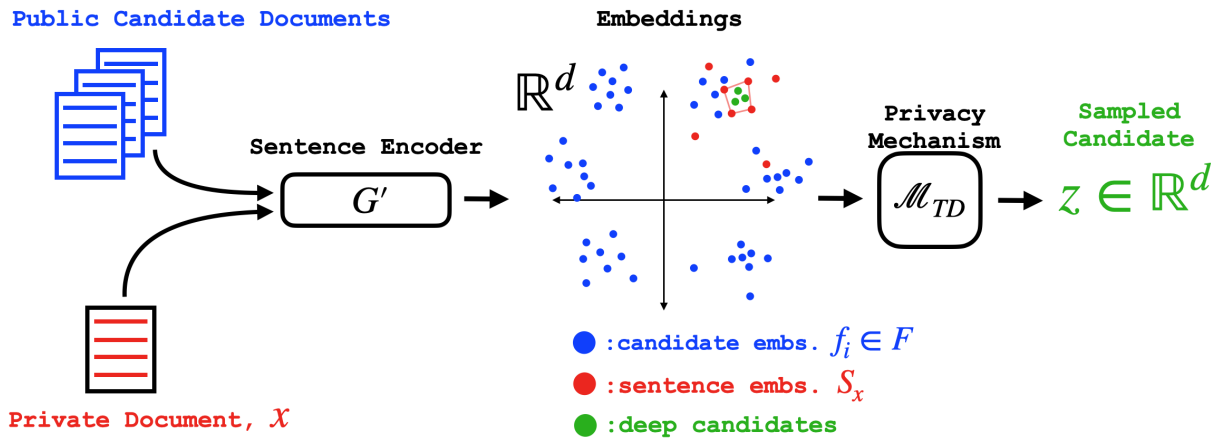


Figure 3.2. DeepCandidate generates a private embedding z of document x by selecting from a set F of public, non-private document embeddings. Sentences from x are encoded by G'. The privacy mechanism $\mathscr{M}_{\text{TD}}$, then privately samples from F, with a preference for candidates with high 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) [30]. 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?)** 34 for more details.

The *exponential mechanism* [65] 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 [20, 2, 68]. 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 [53, 5]. 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 [31], 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 [4] at the word level [37, 38, 23, 70, 99, 91]. 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 [79]. **(author?)** [59] attempt to generate locally private embeddings by bounding the embedding space, and we compare with this method in our experiments. **(author?)** [52] 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?)** [8] 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 [31], 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 [93, 83, 9], we decide to use

averaging as it is a simple and established embedding technique [10, 45, 58].

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?)** [84]:

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 [24], 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 [52, 38, 103] . For instance, adding Laplace noise directly to $\overline{g}(x)$, as done to satisfy some privacy definitions [37, 4], 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 [59, 1]. 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 [104].

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

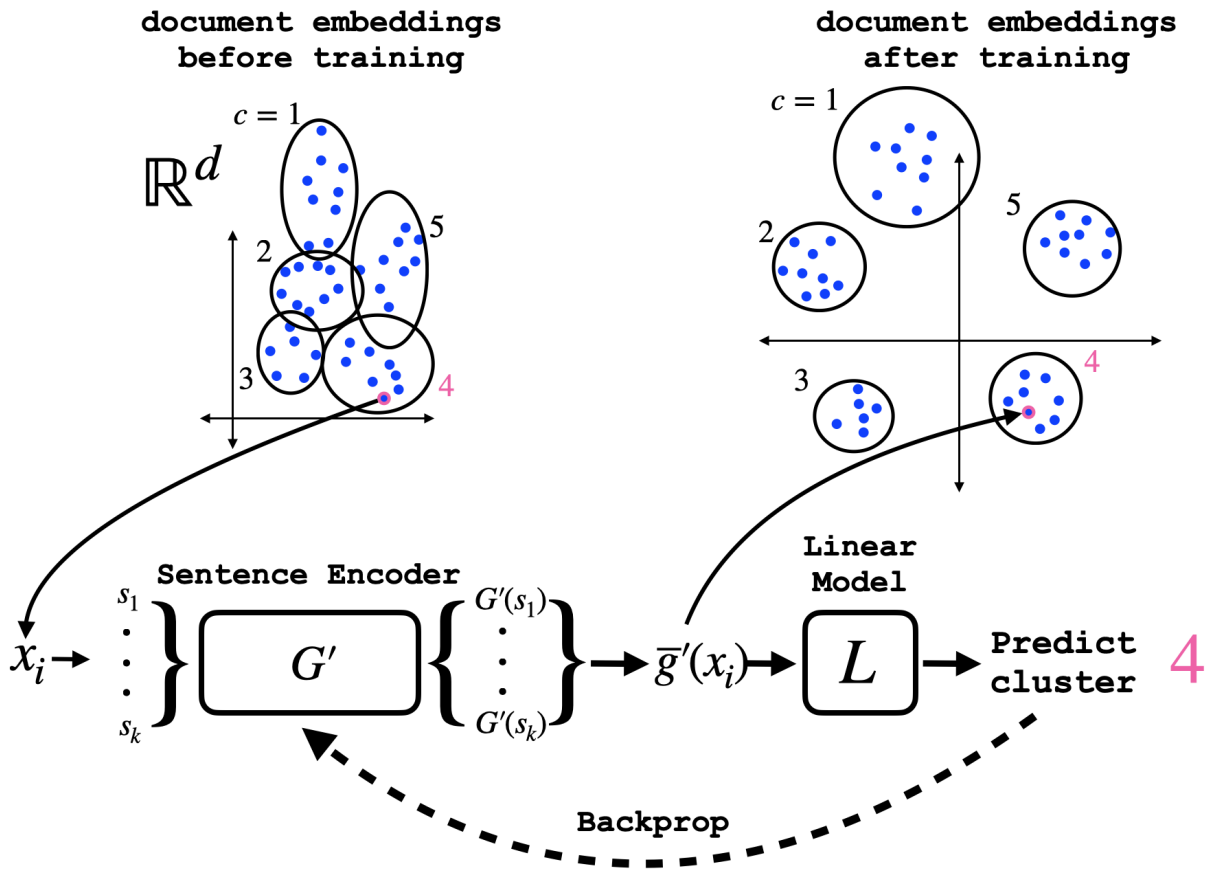


Figure 3.3. G' is trained to encourage similar documents to embed close together and different documents to embed far apart. We first compute embeddings of all (public, non-private) training set documents T with pretrained encoder G, $T_G = \{t_i = \overline{g}(x_i) : x_i \in T\}$ (blue dots). We run 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.

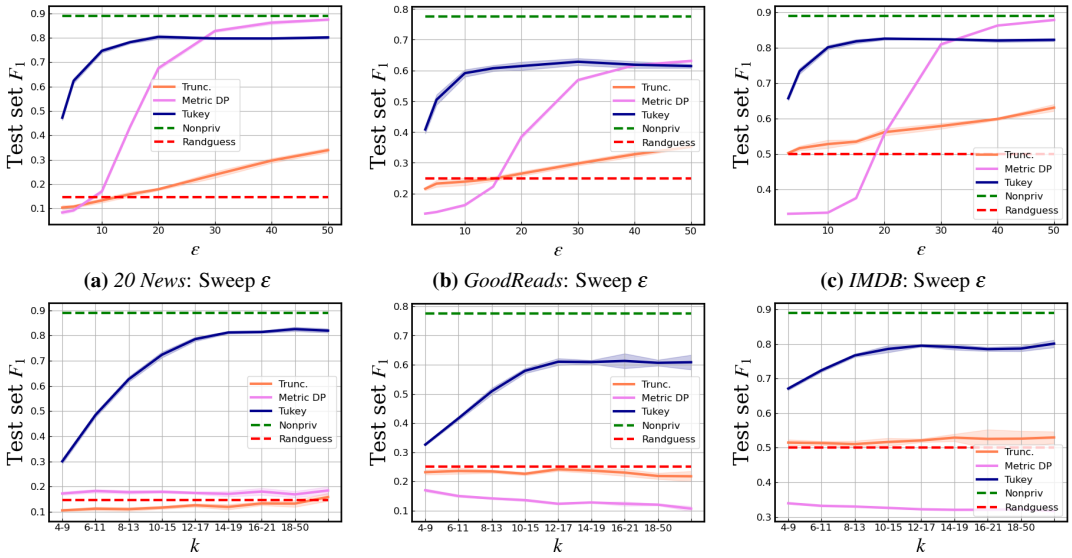
For low $\varepsilon < 10$ (high privacy), about 1% of candidates need to have high depth (≥ 3) in order to be reliably sampled. Note that this is only possible for documents with ≥ 6 sentences. For higher $\varepsilon \geq 10$, $\mathscr{M}_{\text{TD}}$ will reliably sample low depth candidates even 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 [59] and word-level Metric DP [37] 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 [40]. 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 [86] 60k book reviews from four categories: fantasy, history, romance, and childrens literature. Train-48k — Val-8k — Test-4k

20 News Groups [56] 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 [63] 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 [71], 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?)** 59 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?)** 37 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 [37, 38] 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.

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 [69] with FFCV-SSL [13]. 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 [97] 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 [7]. 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:

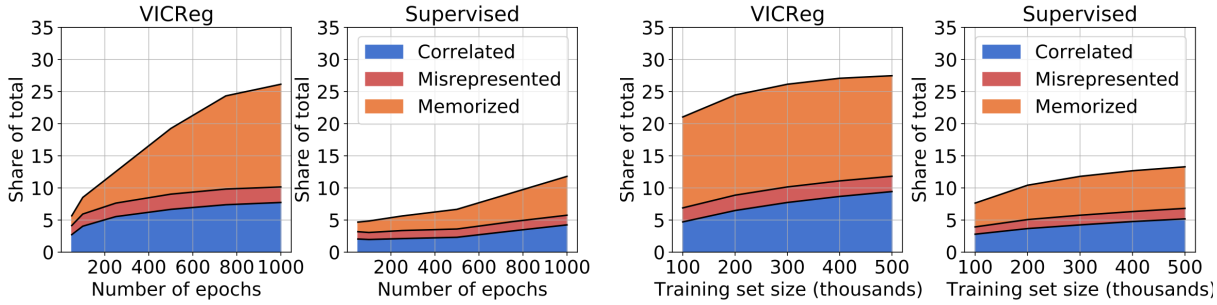
The RCDM is trained on a face-blurred version of ImageNet [28] and is used to decode the SSL backbone embedding of an image back into an approximation of the original image. All RCDMs are trained on the public set of images $\mathscr{X}$ used for the KNN. A separate RCDM must be trained for each SSL 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

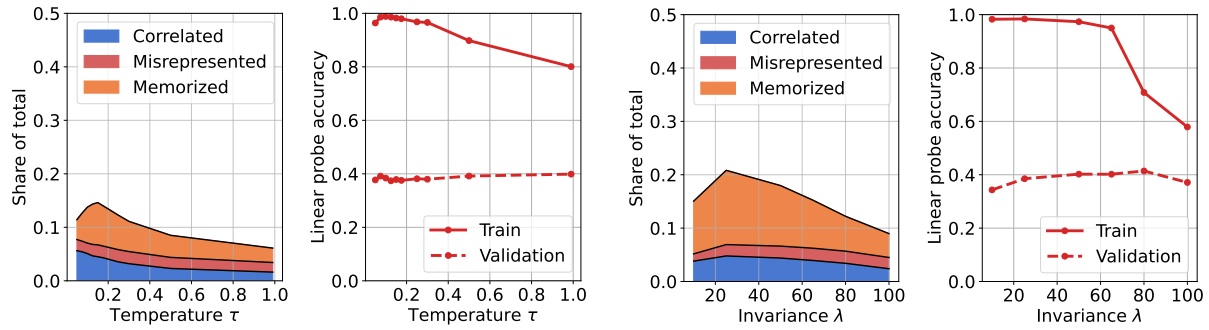
Sample-level memorization



(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

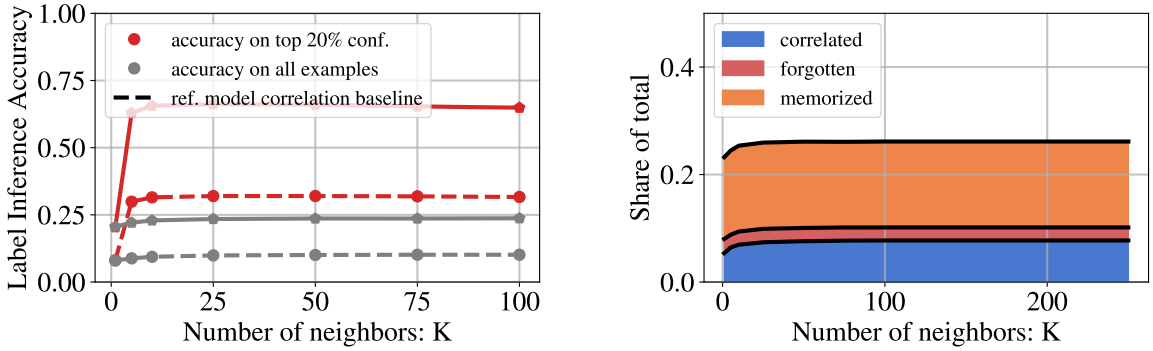
Figure A.2. Effect of SSL hyperparameter on *déjà vu* memorization. The left plot of Figures A.8a and A.8b show the size of the memorized set as a function of the temperature parameter for SimCLR and invariance parameter for VICReg, respectively. *Déjà vu* memorization is the highest within a narrow 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[7], Barlow-Twins[100], Dino[22], Byol[42], SimCLR[101] 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.

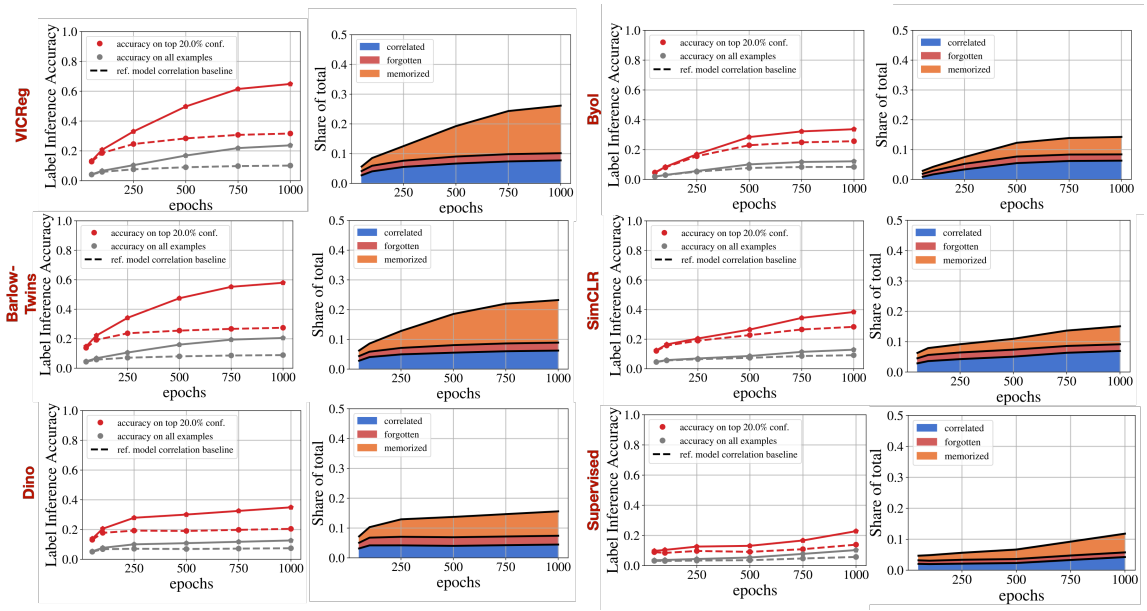


Figure A.4. Comparison of *déjà vu* memorization for VICReg, Barlow Twins, Dino, Byol, SimCLR, and a supervised model. All tests are described in Section 1.4. We are showing *déjà vu* vs. number of training epochs. We see that SimCLR (center row) shows less *déjà vu* than VICReg, yet marginally more 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.

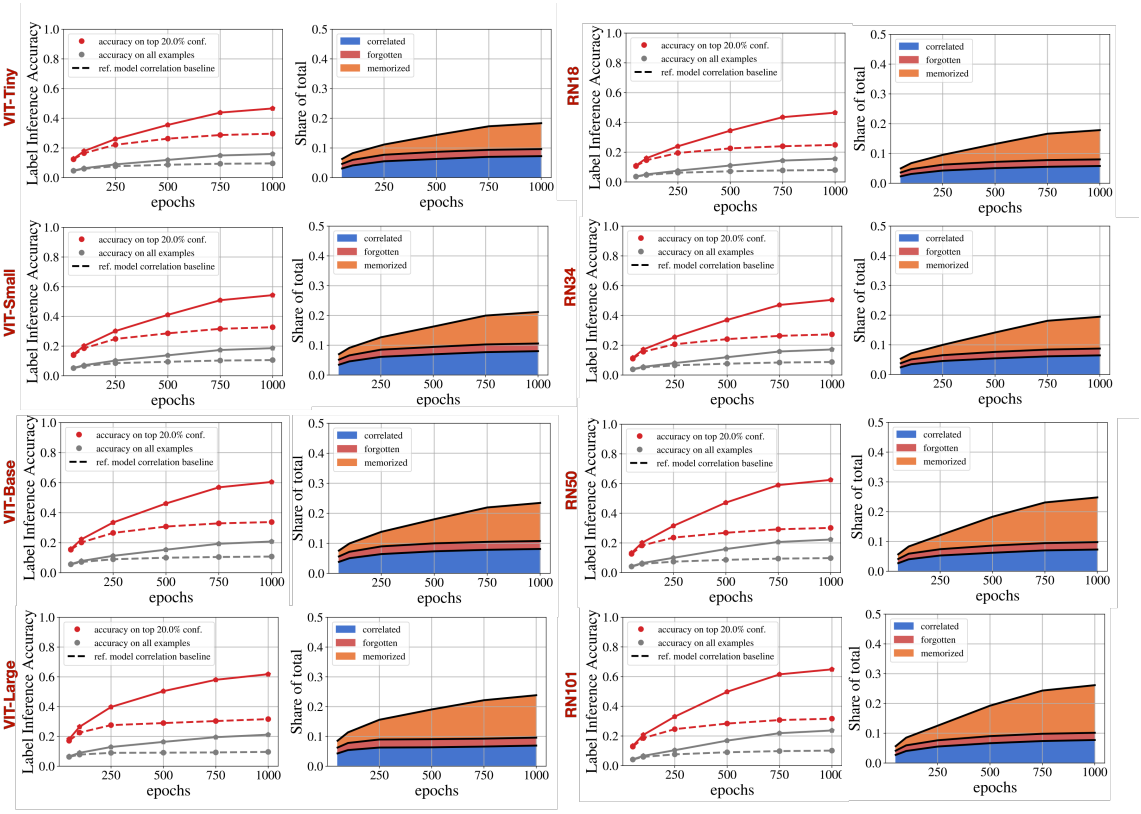
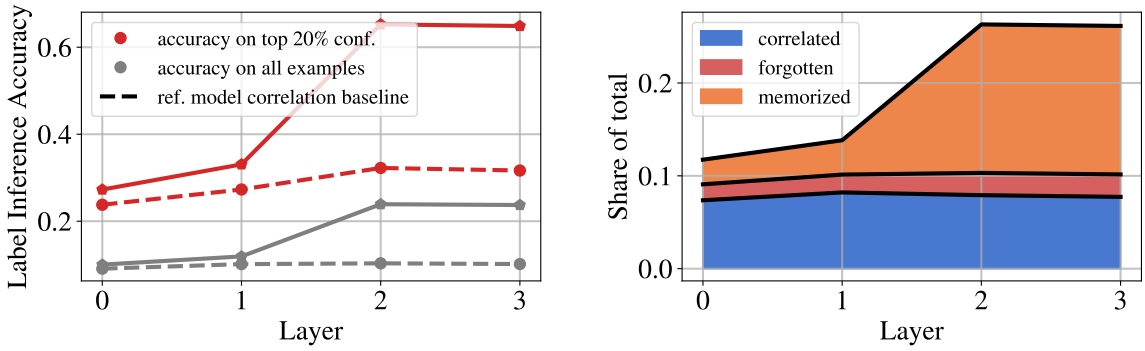


Figure A.5. Comparison of VICReg *déjà vu* memorization for different architectures and model sizes. On the left, we present deja vu memorization using VIT architectures (from vit-tiny in the first row to vit-base in the last row). On the right, we use Resnet based architectures (from resnet18 in the 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?)** [11]. 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.

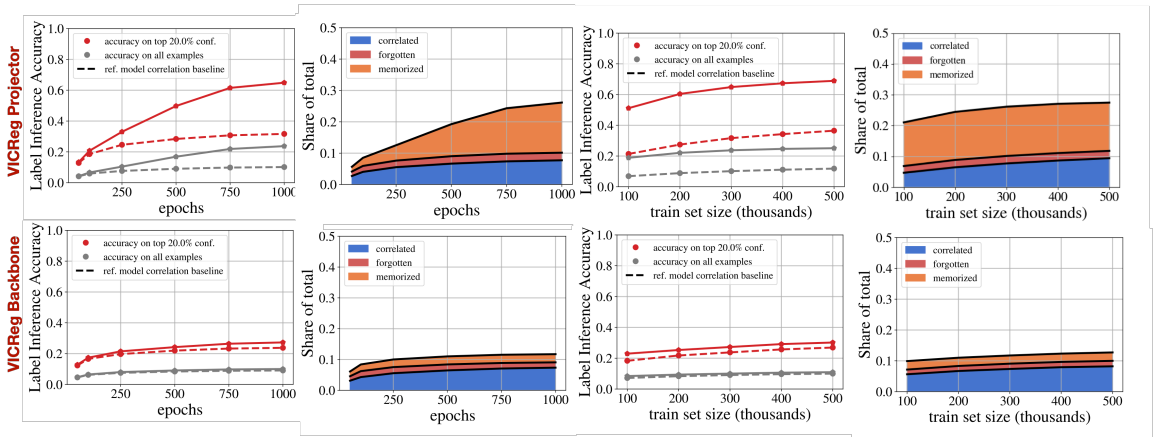
Figure A.6 shows how guillotine regularization significantly reduces the degree of memorization in VICReg. The vast majority of VICReg's *déjà vu* appears to occur in the final two layers of the projector (2,3): in earlier layers (0,1), the label inference accuracy of the target model and reference model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

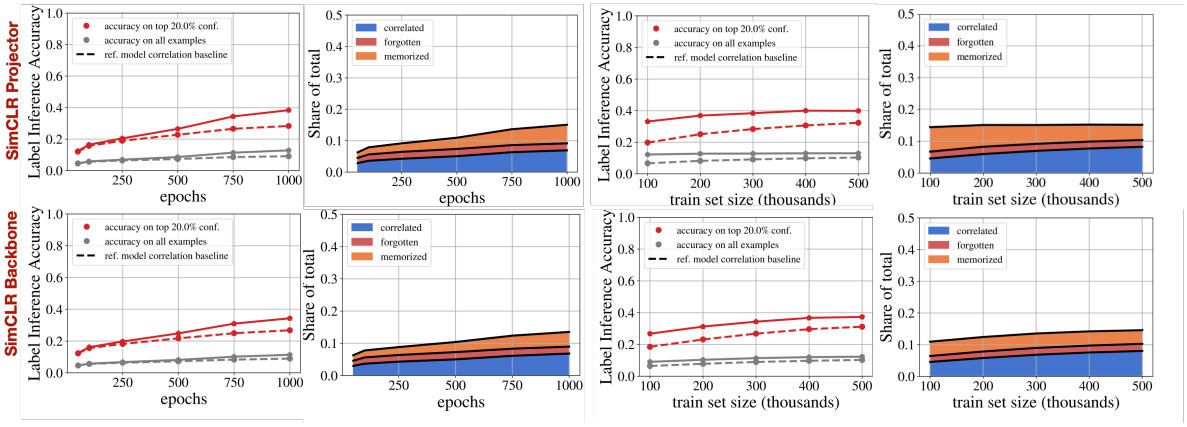
Comparison of *déjà vu* in projector and backbone vs. epochs and dataset size

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

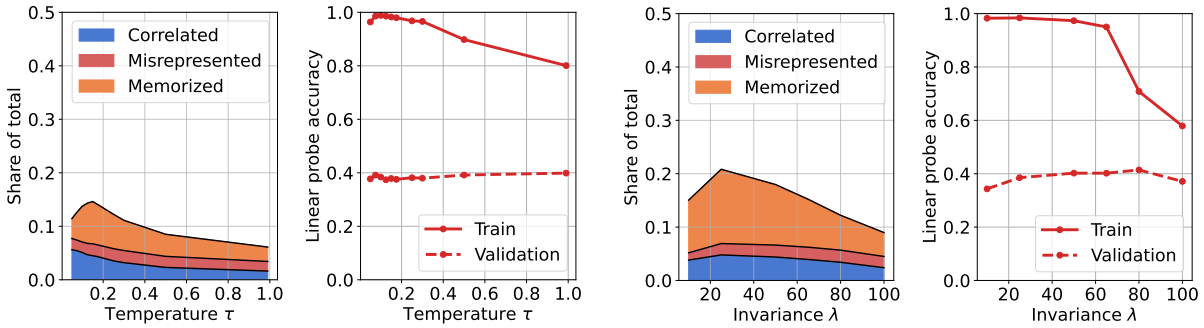


(b) SimCLR

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 [11], 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.

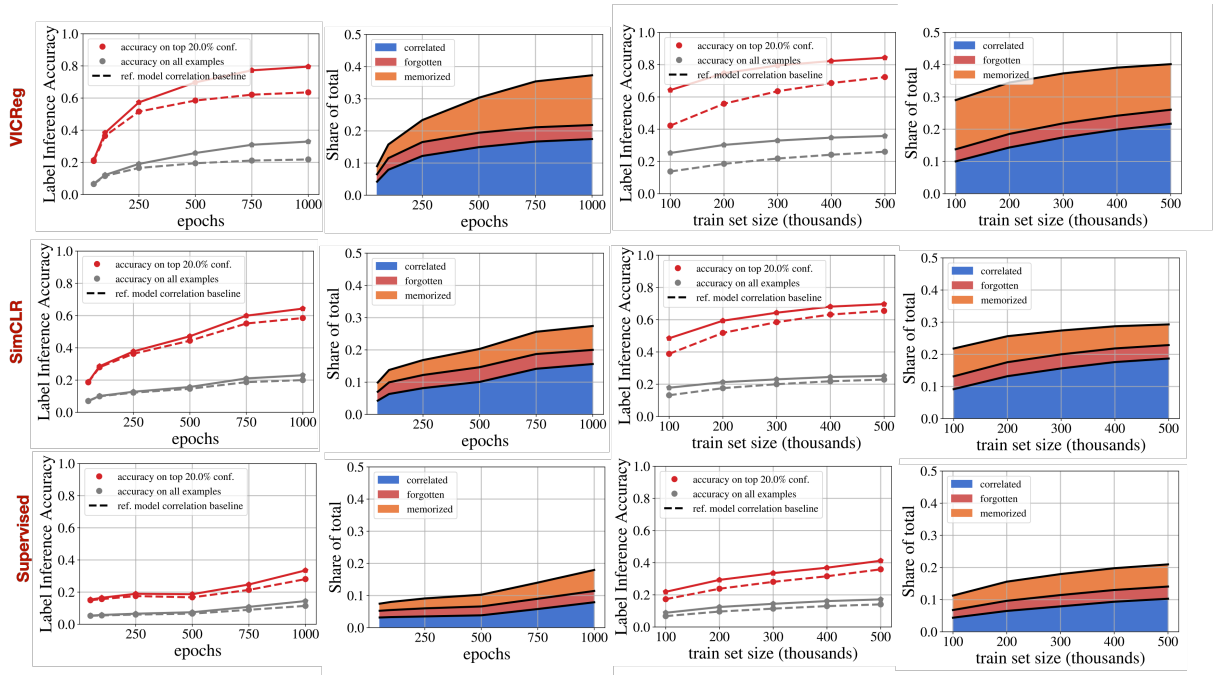


Figure A.10. Deja Vu Memorization using a simple corner crop instead of the periphery crop extracted using bounding box annotations. While the heuristic overall underestimates the degree of *déjà vu* , it roughly follows the same trends versus dataset size and training epochs. This is crucial, since 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 4. *(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 4 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

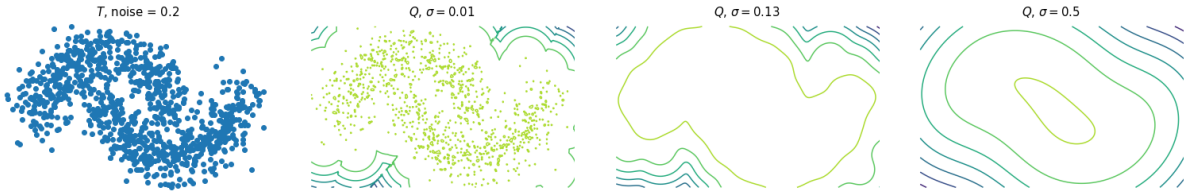
is chosen uniformly at random from T, we have

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

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)** [47]: 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** [72]: 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** [62]: In this test — our primary baseline — we report the three LOO NN values discussed in [92]. 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** [75]: 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

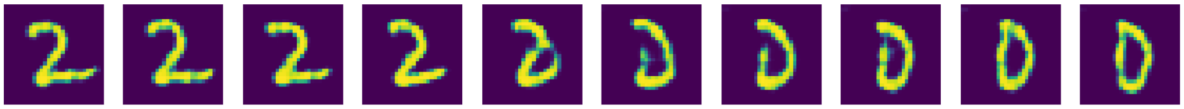
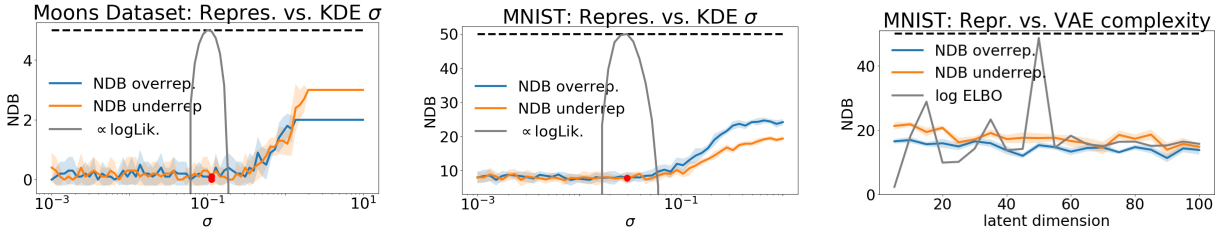


Figure B.2. Interpolating between two points in the latent space to demonstrate L_2 perceptual significance



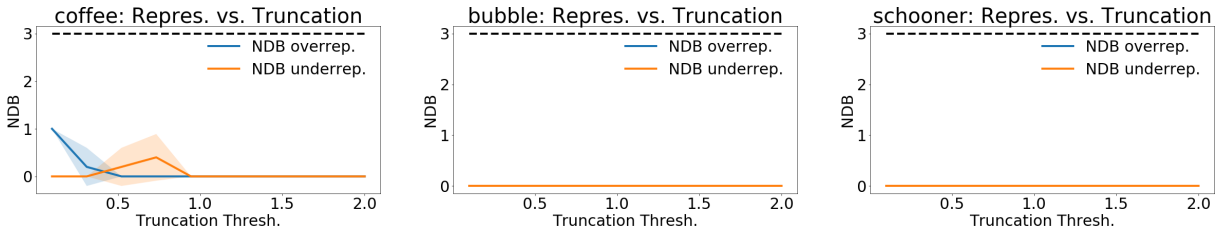
(a) **(b)** **(c)**

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 [57]. 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 [102]. 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 encoder compresses to $z \in \mathbb{R}^{64}$, and decoder expands back up to $\widehat{t} \in \mathbb{R}^{784}$. Our loss is then

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



(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.

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 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

(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.

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

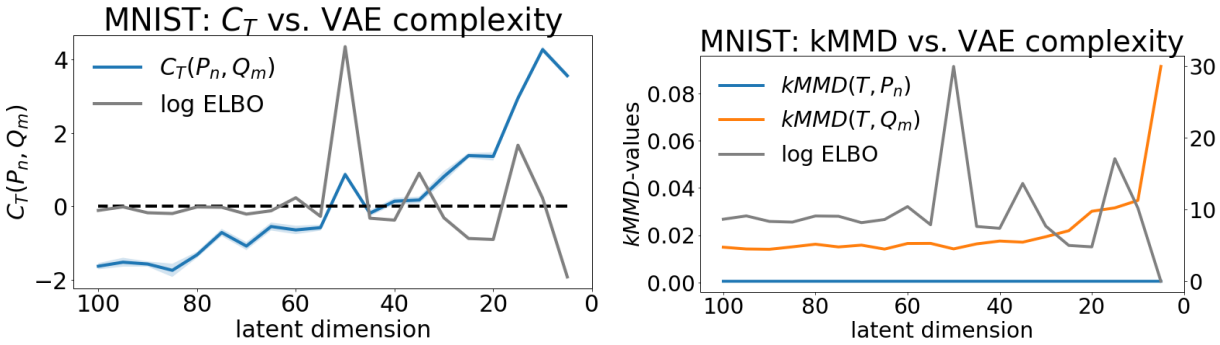
to be acceptable due to low test variance.

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 [14] 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 [35] 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

[14] 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?)** [40].

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 1: $\mathscr{M}_{\text{TD}}$ 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 1.

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

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

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 [71], 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 [59] 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 [37]. 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 [37], 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.

Bibliography

[1] 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.

[2] 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.

[3] 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.

[4] 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.

[5] 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.

[6] Borja Balle, Giovanni Cherubin, and Jamie Hayes. Reconstructing training data with informed adversaries. *arXiv preprint arXiv:2201.04845*, 2022.

[7] 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.

[8] 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.

[9] Federico Bianchi, Silvia Terragni, and Dirk Hovy. Pre-training is a hot topic: Contextual-

ized document embeddings improve topic coherence. *arXiv preprint arXiv:2004.03974*, 2020.

[10] 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.

[11] Florian Bordes, Randall Balestriero, Quentin Garrido, Adrien Bardes, and Pascal Vincent. Guillotine regularization: Improving deep networks generalization by removing their head, 2022.

[12] Florian Bordes, Randall Balestriero, and Pascal Vincent. High fidelity visualization of what your self-supervised representation knows about. *Transactions on Machine Learning Research*, 2022.

[13] Florian Bordes, Randall Balestriero, and Pascal Vincent. Towards democratizing joint-embedding self-supervised learning, 2023.

[14] 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.

[15] Alican Bozkurt, Babak Esmaeili, Dana H Brooks, Jennifer G Dy, and Jan-Willem van de Meent. Can vaes generate novel examples? 2018.

[16] Andrew Brock, Jeff Donahue, and Karen Simonyan. Large scale gan training for high fidelity natural image synthesis, 2018.

[17] 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.

[18] 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.

[19] 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.

[20] 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.

[21] 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.

[22] 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.

[23] Ricardo Silva Carvalho, Theodore Vasiloudis, and Oluwaseyi Feyisetan. Tem: High utility metric differential privacy on text. *arXiv preprint arXiv:2107.07928*, 2021.

[24] Timothy M Chan. An optimal randomized algorithm for maximum tukey depth. In *SODA*, volume 4, pages 430–436, 2004.

[25] Tong Che, Yanran Li, Athul Paul Jacob, Yoshua Bengio, and Wenjie Li. Mode regularized generative adversarial networks. *CoRR*, abs/1612.02136, 2016.

[26] Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey E. Hinton. A simple framework for contrastive learning of visual representations. In *ICML*, 2020.

[27] Xinlei Chen and Kaiming He. Exploring simple siamese representation learning. In *CVPR*, 2020.

[28] 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.

[29] 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.

[30] Cynthia Dwork. *Differential Privacy*, volume 4052. July 2006.

[31] 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.

[32] 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.

[33] Cynthia Dwork and Aaron Roth. The algorithmic foundations of differential privacy. *Theoretical Computer Science*, 9(3-4):211–407, 2013.

[34] Cynthia Dwork, Aaron Roth, et al. The algorithmic foundations of differential privacy. *Found. Trends Theor. Comput. Sci.*, 9(3-4):211–407, 2014.

[35] Cristóbal Esteban, Stephanie L Hyland, and Gunnar Rätsch. Real-valued (medical) time series generation with recurrent conditional gans. *arXiv preprint arXiv:1706.02633*, 2017.

[36] 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.

[37] Oluwaseyi Feyisetan, Borja Balle, Thomas Drake, and Tom Diethe. Privacy- and Utility-Preserving Textual Analysis via Calibrated Multivariate Perturbations. October 2019.

[38] 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.

[39] 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.

[40] Ran Gilad-Bachrach and Chris J. C. Burges. The Median Hypothesis. June 2012.

[41] 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.

[42] 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.

[43] Ishaan Gulrajani, Colin Raffel, and Luke Metz. Towards GAN benchmarks which require generalization. *CoRR*, abs/2001.03653, 2020.

[44] 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.

[45] 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.

[46] 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.

[47] 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.

[48] 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.

[49] 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.

[50] Bargav Jayaraman and David Evans. Are attribute inference attacks just imputation? *arXiv preprint arXiv:2209.01292*, 2022.

[51] 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.

[52] 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.

[53] 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.

[54] Diederik P Kingma and Max Welling. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*, 2013.

[55] 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.

[56] Ken Lang. Home Page for 20 Newsgroups Data Set, 1995.

[57] Yann LeCun and Corinna Cortes. MNIST handwritten digit database. 2010.

[58] 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.

[59] Tao Li and Chris Clifton. Differentially Private Imaging via Latent Space Manipulation. *arXiv:2103.05472 [cs]*, April 2021. arXiv: 2103.05472.

[60] 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.

[61] 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.

[62] David Lopez-Paz and Maxime Oquab. Revisiting classifier two-sample tests. *arXiv preprint arXiv:1610.06545*, 2016.

[63] 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.

[64] 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.

[65] Frank McSherry and Kunal Talwar. Mechanism Design via Differential Privacy. October 2007.

[66] 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.

[67] 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.

[68] 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.

[69] 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.

[70] 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.

[71] Nils Reimers and Iryna Gurevych. Sentence-BERT: Sentence Embeddings using Siamese BERT-Networks. *arXiv:1908.10084 [cs]*, August 2019. arXiv: 1908.10084.

[72] Eitan Richardson and Yair Weiss. On gans and gmms. In *Advances in Neural Information Processing Systems*, pages 5847–5858, 2018.

[73] 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.

[74] 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.

[75] 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.

[76] 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.

[77] 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.

[78] 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.

[79] Congzheng Song and Ananth Raghunathan. Information Leakage in Embedding Models. *arXiv:2004.00053 [cs, stat]*, August 2020. arXiv: 2004.00053.

[80] 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.

[81] 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.

[82] 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.

[83] 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.

[84] 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.

[85] 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.

[86] 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.

[87] 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.

[88] 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.

[89] 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.

[90] 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.

[91] 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.

[92] 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.

[93] 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.

[94] 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.

[95] 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.

[96] 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.

[97] Yang You, Igor Gitman, and Boris Ginsburg. Large batch training of convolutional networks. *arXiv preprint arXiv:1708.03888*, 2017.

[98] 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.

[99] 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.

[100] 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.

[101] Hui Zeng and Xiaohui Cui. Simclrt: A simple framework for contrastive learning of rumor tracking. *Eng. Appl. Artif. Intell.*, 110:104757, 2022.

[102] 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.

[103] 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.

[104] 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.