This ICCV paper is the Open Access version, provided by the Computer Vision Foundation.
Except for this watermark, it is identical to the accepted version;
the final published version of the proceedings is available on IEEE Xplore.

Riemannian-Geometric Fingerprints of Generative Models

Hae Jin Song * Laurent Itti
USC Department of Computer Science

Abstract

*Recent breakthroughs and rapid integration of generative models (GMs) have sparked interest in the problem of model attribution and their fingerprints. For instance, service providers need reliable methods of authenticating their models to protect their IP, while users and law enforcement seek to verify the source of generated content for accountability and trust. In addition, a growing threat of model collapse is arising, as more model-generated data are being fed back into sources (*e.g.*, YouTube) that are often harvested for training ("regurgitative training"), heightening the need to differentiate synthetic from human data. Yet, a gap still exists in understanding generative models' fingerprints, we believe, stemming from the lack of a formal framework that can define, represent, and analyze the fingerprints in a principled way. To address this gap, we take a geometric approach and propose a new definition of artifact and fingerprint of GMs using Riemannian geometry, which allows us to leverage the rich theory of differential geometry. Our new definition generalizes previous work [60] to non-Euclidean manifolds by learning Riemannian metrics from data and replacing the Euclidean distances and nearest-neighbor search with geodesic distances and kNN-based Riemannian center of mass. We apply our theory to a new gradient-based algorithm for computing the fingerprints in practice. Results show that it is more effective in distinguishing a large array of GMs, spanning across 4 different datasets in 2 different resolutions (64×64, 256×256), 27 model architectures, and 2 modalities (Vision, Vision-Language). Using our proposed definition significantly improves the performance on model attribution, as well as a generalization to unseen datasets, model types, and modalities, suggesting its practical efficacy.*

1. Introduction

In recent years, we have seen a rapid development and integration of generative models (GMs) into our society. Vision generative models like Stable Diffusion [54] and Sora [7] are revolutionizing image and video synthesis, and Large-Language Models (LLMs) [1, 8] are changing how people work in business and science, including software development, entertainment, and scientific discovery. Despite this advancement, there is still a big gap in understanding what makes these models behave as they do and how one model differs from another. In particular, a rapidly arising question regarding GMs is model attribution and fingerprints, *i.e.*, what makes their synthetic data different from natural data (*e.g.*, GAN-generated images vs. real images), as well as from synthetic data generated by different models (*e.g.*, texts generated by GPT-4 [1] vs. by Gemini [62]). While this question has been partially studied in the context of deepfake detection [65, 69], its full extension, *i.e.*, distinguishing *amongst* different methods of data generation (model attribution) remains mostly under-explored. In this paper, we address the problem of **fingerprinting and attributing generative models** in a theoretically-grounded way using Riemannian geometry [39] and manifold learning [3, 4, 23].

The problem of fingerprinting and attributing GMs bears increasingly critical significance in both practice and theory. First, in practice, service providers are looking for a reliable method of authenticating their proprietary models (*e.g.*, Google's Gemini [62], OpenAI's GPT [1]) to protect their IP, while users and law enforcement seek to verify the source of generated content for the trustworthiness of synthetic data and AI regulation [42, 51]. In addition, a growing threat of model collapse [13, 59] is arising as more model-generated data are being fed back into sources (*e.g.*, YouTube) that are often harvested for training ("regurgitative training"), heightening the need to detect and differentiate synthetic from human data. Secondly, on the theoretical front, model attribution and fingerprints provide a formal, analytical way of studying the differences between various GMs and revealing their unique characteristics and limitations, thereby promoting the development of new models that overcome current limitations (*e.g.*, artifact-reduced GMs [10, 21, 30, 58]).

In this paper, we study the problem of fingerprinting GMs based on their samples and provide a formal framework that can define, represent, and analyze the fingerprints to study and compare GMs in a principled way. Recent works on deepfake detection [49, 56] and model biases [58, 65] have suggested that GMs leave distinct traces of computations on their samples that differentiate model-generated data

*Corresponding author: Hae Jin Song <haejinso@usc.edu>

from human data (*e.g.*, GM-generated images vs. images by cameras). Such traces have been observed in both the pixel space (*e.g.*, checkerboard patterns by deconvolution layers [50], semantic inconsistencies like asymmetric eye colors [47]) and the frequency space (*e.g.*, spectral discrepancies [15, 16, 65]). Despite these observations that hint at the existence of artifacts and fingerprints of GMs, an explicit definition of fingerprints themselves remains unclear. A recent work [60] proposes the first definition of fingerprints, but its applicability to real-world data is limited due to its assumption that the embedding space of data is Euclidean, which is often violated in real data like images and videos which follow non-Euclidean geometry [6]. This lack of a proper definition hinders a systematic study of GM fingerprints (that goes beyond showing their mere existence), and development of fingerprinting methods for real-world model attribution.

To this end, the aim of this work is to (i) give a proper definition of GM fingerprints that generalizes to **non-Euclidean** spaces using **Riemannian geometry**, (ii) apply our theory to a new gradient-based algorithm for computing the fingerprints, by learning Riemannian metrics from data and replacing the Euclidean distances and nearest-neighbor search with geodesic distances and kNN-based Riemannian center of mass, and finally (iii) study their efficacy in differentiating a large variety of GMs and generalizing across datasets, model types and modalities. We find that our proposed definition provides a useful feature space for fingerprinting GMs, including state-of-the-art (SoTA) models, and outperforms existing methods on both attribution and generalization.

By providing a formal definition of GM fingerprints, we address another important gap in literature, *i.e.*, the lack of studies on fingerprints across different modalities. While SoTA GMs are being developed on multimodal data (*e.g.*, a combination of images, texts and audios), studies on GM fingerprints have been limited to a single-modality (*e.g.*, images only [60, 65, 69] or texts only [68]). To bridge this gap and encourage research on cross-modal fingerprints, we introduce an extended benchmark dataset (Tab. 2) that includes SoTA multimodal GMs (*i.e.*, text-to-image models). Our contributions can be summarized as following:

- We formalize a definition of fingerprints of GMs that generalizes to non-Euclidean data using Riemannian geometry.
- We propose a practical algorithm to compute the fingerprints from finite samples by learning a latent Riemannian metric as pullback metric from the data space and estimating the fingerprints via a gradient-based algorithm.
- We conduct extensive experiments to show the attributability and generalizability of our fingerprints, outperforming existing attribution methods. In particular, we consider a large array of GMs from all four main families (GAN, VAE, Flow, Score-based), spanning across 4 different datasets (CIFAR-10 [37], CelebA-64 [40], CelebA-HQ [29] and FFHQ [31]) of 2 different resolutions (64×64, 256×256), 27 model architectures, and 2 modalities (Vision, Vision-Language). This, to our knowledge, is the most comprehensive model-attribution study to date.
- Our results show that our generalized definition makes significant improvements on attribution accuracy and generalizability to unseen datasets, model types and modalities, suggesting its efficacy in real-world scenarios.

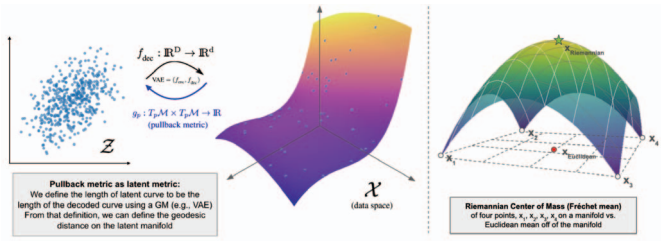

Figure 1. **Learn the latent Riemannian manifold from observed real data.** (Left) We learn the latent geometry of data (real images) by training a VAE with mean and variance estimators and using its decoder to pullback the metric on the data space to the latent space. This pullback metric defines a Riemannian metric [4], based on which we compute geodesic distances and estimate a Riemannian center of mass (RCM) on the latent space in Alg. 3. (Right) An illustration of RCM of four points on a manifold.

2. Background on Riemannian geometry and Riemannian manifold learning

To establish a common language and context for introducing our Riemannian-geometric definition of fingerprints, we start with a brief recap of Riemannian geometry [19, 39, 57].

Riemannian manifold and its metric. A Riemannian manifold is a well-studied metric space that locally resembles a Euclidean space, allowing geometric notions such as lengths, distance and curvature to be (locally) defined on a curved (*i.e.* non-Euclidean) space. A formal definition is as follows:

Definition 2.1. A Riemmanian manifold is a smooth manifold $\mathcal{M}$, whose tangent space $T_p\mathcal{M}$ at each point $p \in \mathcal{M}$ is equipped with an inner product (Riemannian metric) $g_p : T_p\mathcal{M} \times T_p\mathcal{M} \to \mathbb{R}$ in a smooth way.

Riemannian metric. A Riemannian metric g on $\mathcal{M}$ assigns to each p an inner product $g_p : T_p\mathcal{M} \times T_p\mathcal{M} \to \mathbb{R}$, which induces a norm (*i.e.*, length of a vector) $||\cdot||_p : T_p\mathcal{M} \to \mathbb{R}$ defined by $||v||_p = \sqrt{g_p(v, v)}$.

Inituitively, a Riemannian metric functions as a measuring stick on every tangent space of $\mathcal{M}$, dictating how to measure the lengths of vectors and curves (and other derived geometric quantities) on $\mathcal{M}$.

Geometric calculations on Riemannian manifolds.

- Measuring lengths on a Riemannian manifold: The inner product structure on $(\mathcal{M}, g)$ is sufficient for defining the

length of a smooth curve $c : [0, 1] \to \mathcal{M}$ as:

$$\text{Length}(c) := \int_0^1 \sqrt{g_{c(t)}(\dot{c}(t), \dot{c}(t))}\mathrm{d}t \tag{1}$$

where $\dot{c} = \partial_t c$ denotes the curve velocity.

- Measuring distances on a Riemannian manifold: Given $p, q \in \mathcal{M}$, the distance from p to q is defined as:

$$\mathrm{d}(p, q) := \inf\{\text{Length}(c) | c : [0, 1] \to \mathcal{M}, \quad c(0) = p, c(1) = q\} \tag{2}$$

Such a minimum-distance curve from p to q is called a "geodesic" and its distance the "geodesic distance".

Learning Riemannian manifold from data. In the context of this paper we focus on Riemannian manifolds whose structure and Riemannian metrics are *learned* from observed data (*i.e.*, images). In particular, we employ the method of *pulling back* the metric on the observed data space to the latent manifold [3, 4, 23] by training a generative model (*e.g.*, VAE) with its encoder and decoder functions.

More specifically, given the data space $\mathcal{X}$ and a VAE $(f_{\text{enc}}, f_{\text{dec}})$ trained with a latent space $\mathcal{Z}$, where its decoder function f_{dec} models *both* the mean and variance, *i.e.*,

$$f_{\text{dec}}(z) = \mu(z) + \sigma(z) \odot \epsilon, \quad \mu : \mathcal{Z} \to \mathcal{X}, \sigma : \mathcal{Z} \to \mathbb{R}_+, \epsilon \sim \mathcal{N}(0, \mathbb{I}_D) \tag{3}$$

we can construct a Riemmanian metric g on $\mathcal{Z}$ from the metric on $\mathcal{X}$ (which is, conveniently, Euclidean), as [4]:

$$g := \mathrm{J}_\mu^T \mathrm{J}_\mu + \mathrm{J}_\sigma^T \mathrm{J}_\sigma \tag{4}$$

where J_μ and J_σ are the Jacobians of $\mu(.)$ and $\sigma(.)$. These Jacobians can be estimated from the standard backpropagation through the trained decoder [4].

In our approach (Sec. 3), we use this pullback approach to learn a latent (Riemannian) geometric structure of real image datasets, and estimate "artifacts" of a GM on model-generated data using this learned, real data manifold as a reference of a true data distribution. Additionally, the learned metric aids in computing a more geometrically appropriate notion of "projection" to the manifold, using geodesic distances and Riemannian center of mass. We now introduce our new definition and implementation of GM fingerprints.

3. Riemannian-Geometric Fingerprints of GMs

As discussed in Sec. 1, explicit formal definitions of artifacts and fingerprints of GMs remain unclear or constrained to a specific data geometry despite many existing works that observed their existence across various classes of GMs. In particular, the first definition of GM fingerprints proposed in [60] assumes the data manifold to be Euclidean, but such assumption is often violated in the high-dimensional data we encounter in the real world (*e.g.*, images, videos, and texts).

$(\mathcal{X}, d_{\mathcal{X}})$	Observed data space as $(\mathbb{R}^{\mathrm{D}}, d_{\mathrm{L2}})$
$(\mathcal{Z}, d_{\mathcal{Z}})$	Latent space of VAE as $(\mathbb{R}^{\mathrm{d}}, d_{\mathrm{L2}})$
$(\mathcal{M}, d_{\mathcal{M}})$	Latent Riemannian manifold immersed in $\mathcal{X}$ with $d_{\mathcal{M}} := g_{\text{pullback}}$ s.t. $g_p : T_p\mathcal{M} \times T_p\mathcal{M} \to \mathbb{R}$
VAE	Generative model that learns a latent manifold of real dataset; $(f_{\text{enc}}, f_{\text{dec}})$
X_R, X_G	real dataset, and synthetic dataset generated by a generative model G
$x_{\text{NN}}^{(k)}$	kth nearest neighbors of x on the manifold $(\mathcal{M}, d_{\mathcal{M}})$
$k\text{NN}(x; X_R)$	the set of K nearest neighbors of x on $(\mathcal{M}, d_{\mathcal{M}})$
x_{RCM}	Riemannian center of mass of $k\text{NN}(x; X_R)$ on $(\mathcal{M}, d_{\mathcal{M}})$
$\boldsymbol{a(x_G; \mathcal{M})}$	Artifact of x_G with respect to $(\mathcal{M}, d_{\mathcal{M}})$

Table 1. Our notation for data and latent spaces as metric spaces and points involved in estimating GM fingerprints.

Motivated by this limitation, in this section, we propose an improved formal definitions of artifacts and fingerprints of generative models that generalize to non-euclidean data by taking into account their geometry. We then describe a practical algorithm for computing them from observed samples, by (i) learning a Riemannian metric from data as a pullback metric [3, 4], and (ii) estimating the projection of generated samples to this learned manifold as a Riemannian center of mass (RCM) [2] of k-nearest neighbors in geodesics.

3.1. Definitions of GM artifacts and fingerprints

The artifacts and fingerprints of generative models intuitively correspond to the models' imperfections in replicating the true data-generating process, consistently manifested in the samples they produce. Under the manifold learning hypothesis [9, 18], which suggests real-world data such as images and texts lie on a lower-dimensional manifold, we can formalize such deficiencies of GMs as how their generated samples deviate from the true data manifold. Formally, let G be a generative model trained on the dataset X_R of real samples that lie on a lower-dimensional data manifold $\mathcal{M}$, P_G its induced probability distribution, S_G its support, and x_G its sample:

Definition 3.1 (Artifact)**.** An artifact left by a generative model G on its sample x_G (denoted as $\boldsymbol{a(x_G; \mathcal{M})}$) is defined as the difference between x_G and its projection x^* onto the manifold $\mathcal{M}$ of the real data used to train G:

$$x^* := \text{Projection}(x_G, \mathcal{M}) \tag{5}$$

$$\boldsymbol{a(x_G; \mathcal{M})} := x_G - x^* \tag{6}$$

Definition 3.2 (Fingerprint)**.** The fingerprint of a generative model G with respect to the true data manifold $\mathcal{M}$ is defined

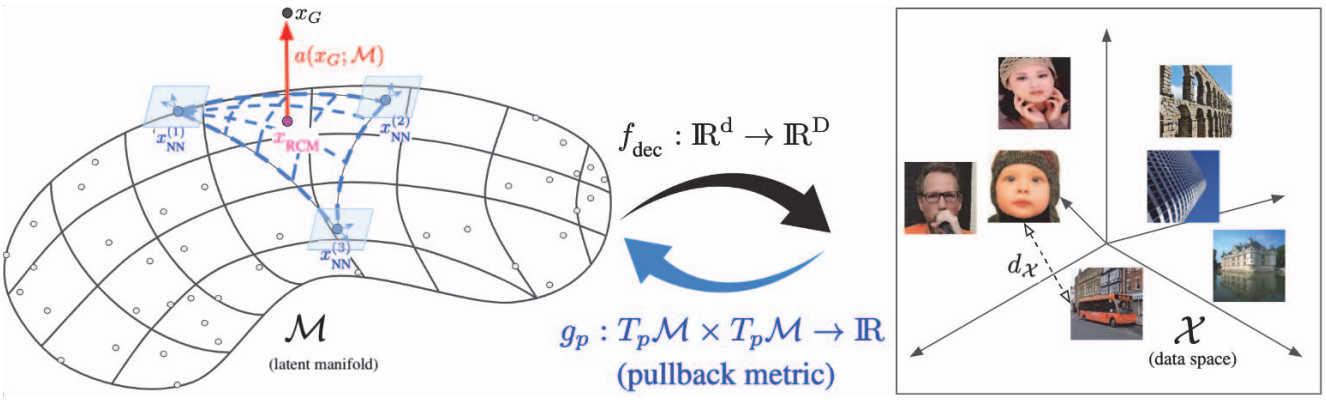

Figure 2. **Overview of our fingerprint estimation.** We learn the latent data manifold $\mathcal{M}$ from the dataset of real images as a Riemannian manifold equipped with the pullback metric G. To pullback the metric of $\mathcal{X}$ to the latent manifold $\mathcal{M}$, we train a VAE (with mean and variance estimation functions $(\theta_\mu, \theta_\sigma)$) on the real dataset, and define the length of a curve on $\mathcal{M}$ to be the length of a decoded curve on $\mathcal{X}$, on which we know how to measure a length (*i.e.*, a Euclidean norm of a vector). From the length of a curve, we define the geodesic distance of two points on $\mathcal{M}$ as the shortest distance of a curve connecting them.

as the set of all its artifacts over its support S_G:

$$\textbf{Fingerprint}(G; \mathcal{M}) = \{\boldsymbol{a(x_G; \mathcal{M})} | x \in S_G\} \quad (7)$$

Figure 2 (left) illustrates our proposed definitions.

3.2. Estimation of Riemannian GM artifacts and fingerprints

Our estimation of GM artifacts and fingerprints consist of two main steps: (i) We first learn the latent data manifold $\mathcal{M}$ from the dataset of real images as a Riemannian manifold equipped with a proper metric tensor g. (ii) We then estimate the artifact on x_G according to this metric, by computing the projection of x_G onto $\mathcal{M}$ as a Riemannian Center of Mass (RCM) of x_G's k-nearest neighbors in $\mathcal{M}$, and the artifact as the difference vector between x_G and its projection onto $\mathcal{M}$. Finally, the fingerprint of G is computed as the set of artifacts, each of which is computed using x_G in its generated sample set X_G. Algorithm 1 describes this workflow of our fingerprint estimation (see notations in Tab. 1).

Step 1. Learning the Riemannian manifold from data. Since we do not have access to the data manifold $\mathcal{M}$ on which the real images lie (*i.e.*, the natural image manifold), we need to estimate it using the available observed samples from $\mathcal{M}$. To this end, we use real images in the training datasets of generative models, and map them to a suitable embedding space to construct a collection of features to be used as an estimated image manifold. Unlike the previous definition in [60], which used either the pixel space, its FFT space, or the embedding spaces of pretrained networks (ResNet and Barlow-Twin pretrained on ImageNet), we *learn* the latent data manifold from the observed real dataset. (In Sec. 4, we experimentally show this improves the generalizability to unseen datasets and model types.)

Algorithm 1 Compute Riemannian GM fingerprints

Input: Set of real images (X_R) and images generated by model G (X_G)

function COMPUTE-FINGERPRINT($X_G \mid X_R$)
  Fingerprint$(\boldsymbol{G; R}) \leftarrow \{\}$
  for $\boldsymbol{x_G} \in \boldsymbol{X_G}$ **do** ▷ Runs in parallel
    $\boldsymbol{a(x_G)} \leftarrow$ COMPUTE-ARTIFACT($x_G \mid X_R$)
    Fingerprint$(\boldsymbol{G; R})$.add($\boldsymbol{a(x_G)}$)
  return **Fingerprint**$(\boldsymbol{G; R})$

function COMPUTE-ARTIFACT($x \mid X_R$)
  $(\boldsymbol{\mathcal{M}, g}) \leftarrow$ LEARN-RIEMANNIAN-MANIFOLD(X_R) ▷ Learned data manifold with metric tensor g
  $x^\star \leftarrow$ PROJECT$(x, \mathcal{M})$ ▷ Project x onto $\mathcal{M}$
  $\boldsymbol{a(x, \mathcal{M})} \leftarrow x - x^\star$ ▷ Artifact as a difference vector
  return $\boldsymbol{a(x, \mathcal{M})}$

The key idea in our approach to learning the latent data manifold from the observed real dataset (*e.g.*, real images) with metric is to pull back the geometry in the observation space (*e.g.*, L2 distances in the image pixel space) to the latent manifold, using generative models. This metric learning based on a pullback is proposed in [3, 4]. In our work, we choose as the generative model a VAE with the mean and variance estimators and train it on the real dataset with the standard VAE loss function. Its learned encoder functions as the embedding map from the data space to the latent space, and its decoder as a vehicle to define the geometry of the latent space (*e.g.*, length of a curve and geodesic distances on the manifold) by pulling back the geometry of the data space. This way, we learn the embedding space and a Riemannian

Algorithm 2 Learn Riemannian manifold from real dataset

Input: Set of real images, X_R

function LEARN-RIEMANNIAN-MANIFOLD(X_R)
  # Train VAE with mean and variance parameters $(\theta_\mu, \theta_\sigma)$ using X_R with standard VAE losses
  VAE $= (f_{\text{enc}}, f_{\text{dec}}) \leftarrow$ TrainVAE(X_R) ▷ $f_{\text{enc}} : \mathcal{X} \to \mathcal{Z}$, $f_{\text{dec}} : \mathcal{Z} \to \mathcal{X}$
  # Pullback metric from the data space $\mathcal{X}$ to $\mathcal{M}$
  $\boldsymbol{g} \leftarrow J_\mu^T J_\mu + J_\sigma^T J_\sigma$ ▷ Riemannian metric on $\mathcal{M}$, pulled back through f_{dec}
  $\boldsymbol{\mathcal{M}} \leftarrow f_{\text{enc}}(X_R)$
  return $(\boldsymbol{\mathcal{M}}, \boldsymbol{g})$, **VAE** $= (f_{\text{enc}}, f_{\text{dec}})$

metric simultaneously:

(i). Train VAE with mean and variance parameters $(\theta_\mu, \theta_\sigma)$ with the standard VAE loss, using the real dataset X_R: Note that *both* the mean and variance parameters are required for proper learning of Riemannian metrics [4, 23], and we train such VAE by (i) first training its inference network with the variance parameter fixed, and (ii) training the variance parameter as in [4] (Sec 4.1). This step provides the embedding map $f_{enc} : \mathcal{X} \to \mathcal{M}$, and $f_{dec} : \mathcal{M} \to \mathcal{X}$.

(ii). Define the metric on $\mathcal{M}$ by pulling back the metric on $\mathcal{X}$ (*i.e.*, L2 in the standard generative modeling setup [35]), to $\mathcal{M}$: To do so, we first define the "length" of a curve in M to be the "length" of its decoded curve on X, *i.e.*,

$$\text{length}_{\mathcal{M}}(c) := \text{length}_{\mathcal{X}}(f_{\text{dec}}(c)) \tag{8}$$

and define the pullback metric as the metric tensor g on $\mathcal{M}$:

$$g := J_\mu^T J_\mu + J_\sigma^T J_\sigma \text{ [4]} \tag{9}$$

Step 2. Estimating the artifact of G in its sample. The artifact of G in its sample x_G is computed in two steps:

1. Estimate the projection $x^\star$ as the Riemannian center of mass (RCM) of x_G's k-nearest neighbors on the data manifold:
   - Compute $k\text{NN}_{d_{\mathcal{Z}}}(x_G, \mathcal{M})$, the set of k-nearest neighbors (kNNs) on the data manifold, according to the distance metric $d_{\mathcal{Z}}$ of $\mathcal{Z}$
   - Estimate $x^\star$ as RCM($k\text{NN}_{d_{\mathcal{Z}}}(x_G, \mathcal{M})$: compute the Riemannian center of mass of the kNNs using the metric on the latent Riemannian manifold, learned in Step.1 (see Alg. 2).
2. Compute the artifact as a difference vector between x_G and x^*: $\boldsymbol{a(x_G; \mathcal{M})} = x_G - x^*$

Algorithm 3 describes how to compute the projection of x_G onto $\mathcal{M}$ using its k-nearest neighbors of x_G in $\mathcal{M}$ and the Riemannian center of mass of the kNNs.

Computing Riemannian center of mass (RCM) on the latent manifold. We first define the RCM as following [2]:

Algorithm 3 Compute projection of x_G onto $\mathcal{M}$ as RCM

Input: A model-generated sample x_G, and Riemannian manifold $\mathcal{M}$

function PROJECT($x_G, \mathcal{M}$)
  # Compute the projection of x_G onto $\mathcal{M}$ as an RCM of its kNNs in $\mathcal{M}$
  $\boldsymbol{Q}_K \leftarrow k\text{NN}_{\boldsymbol{dz}}(x_G, X_R)$ ▷ k-nearest neighbors of x_G on $\mathcal{M}$ using the metric $d_{\mathcal{Z}}$
  RCM$(\boldsymbol{Q}_K) \leftarrow$ COMPUTE-RCM(Q_K, G) ▷ Riemannian center of mass of kNNs on $\mathcal{M}$
  return RCM$(\boldsymbol{Q}_K)$

Definition 3.3. Given the dataset $\{x_i\}_{i=1}^N \subset \mathcal{M}$ and L^p-distance d^p on $\mathcal{M}$, its Riemannian L^p center of mass (a.k.a. Fréchet mean) with respect to weights $0 \le w_i \le 1$ ($\sum_{i=1}^N w_i = 1$) is defined as the minimizer(s) of

$$f_p(x) = \begin{cases} \frac{1}{p}\sum_{i=1}^N w_i d^p(x, x_i) & 1 \le p < \infty \\ \max_i d(x, x_i) & p = \infty \end{cases} \text{ [2]} \tag{10}$$

in $\mathcal{M}$. By convention, the weights are assumed to be equally distributed unless specified [2]. Our method adopts an iterative, gradient-based algorithm [2, 38, 43] for computing this Riemannian center of mass, using the metric g learned in Step 1 (Eqn. 9). Alg. 4 describes how we compute a local RCM given a set of query points $Q \subset \mathcal{M}$ on our learned data manifold $\mathcal{M}$, using the pullback metric g as the distance d in Eq. 10. In particular, we follow until convergence the gradient of f_p using the k-nearest neighbors of x_G on X_R:

$$\nabla f_p(x) = -\sum_{i=1}^K w_i d^{p-2}(x, x_i) \exp_x^{-1} x_i \text{ [2]} \tag{11}$$

This gradient is valid for any $x \in \mathcal{M}$ as long as it is not in the small open neighborhood of other data points [2]. For our purposes, we assume that this condition is achieved due to the sparsity of our observed sample, *i.e.*, the dimensionality of our data manifold (*i.e.*, image manifold) is much higher than our sample size (*i.e.*, number of observed images).

Algorithm 4 Compute RCM: Gradient Descent for finding the Riemannian L^p center of mass of $\{x_i\}_{i=1}^N$

Input: $\{x_i\}_{i=1}^N \subset \mathcal{M}$, $\{w_i\}_{i=1}^N$ and choose $x^0 \in \mathcal{M}$

1: `if` $\nabla f_p(x^k) = 0$ `then` stop, `else set`

$$x^{k+1} \leftarrow \exp_{x^k}(-t_k \nabla f_p(x^k)) \tag{12}$$

where $t_k > 0$ is an "appropriate" step-size and $\nabla f_p(\cdot)$ is defined in (11).

2: `repeat` step 1

In Algo. 4, we initialize x^0 with a random data point in X_R, and set t_k for each k as in [38].

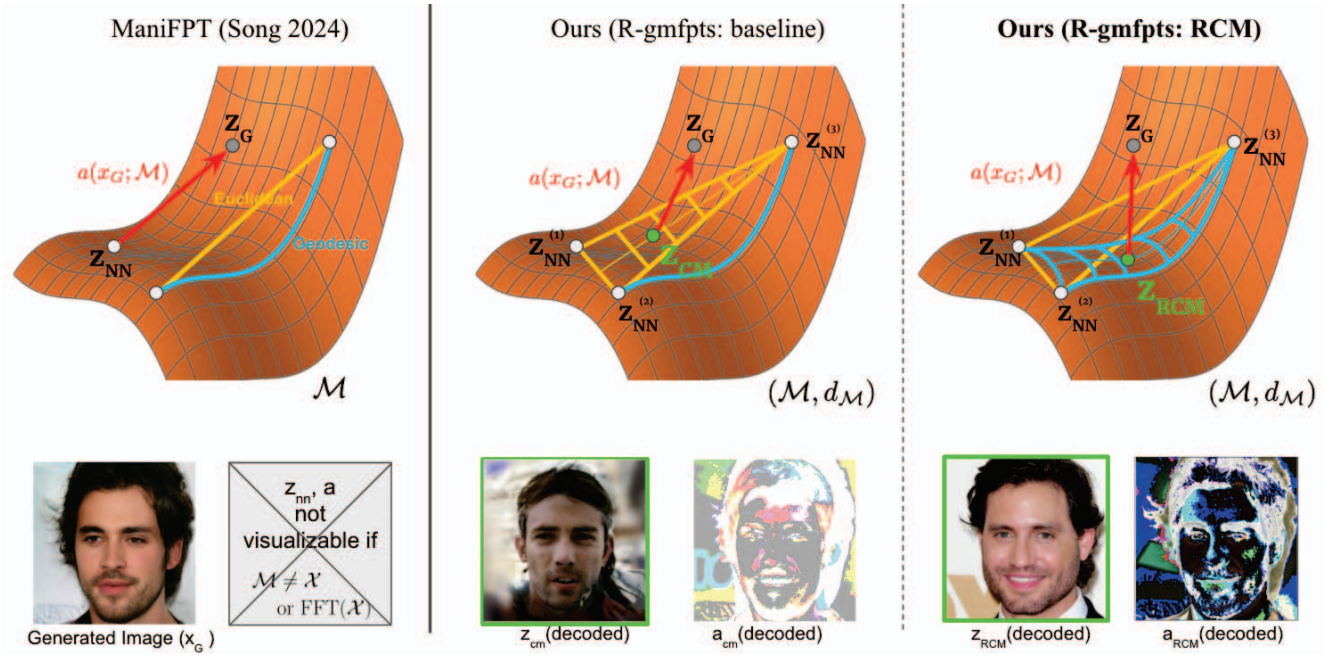

Figure 3. **Estimating the projection of x_G onto the manifold $\mathcal{M}$ as Riemannian center of mass of k-nearest neighbors.** *(left)* Definition and fingerprint estimation proposed in [60]. Here, the projection of z_G is estimated the nearest-neighbor (1-NN) in the observed real dataset, based on the standard Euclidean distance. *(middle)* Baseline method using k-nearest neighbors (k>1; k=3 in this figure): we estimate the artifact of z_G by finding the k-nearest neighbors of z_G in the real dataset using the Euclidean distance, and computing their center of mass (also in L2). *(right)* Our proposed method (**R-gmftps: RCM**) that estimates the artifact $a(x_G, \mathcal{M})$ using a Riemannian center of mass of k-nearest neighbors, based on the geodesic distances learned from data (Sec. 3.2). Note z_{cm} does *not* lie on the manifold $\mathcal{M}$ (which is manifested on the synthetic artifacts in z_{cm} (decoded) in the center), while the projection z_{RCM} estimated as Riemannian center of mass (right column) *does* lie on $\mathcal{M}$, thus corresponding to an actual real image. (Background manifold image modified with permission [23])

Step 3. Computing the fingerprint of a generative model. Given a set of model-generated samples $X_G = \{x_i\}_{i=1}^N$ where $x_i \sim P_G$, we estimate its fingerprint w.r.t. real data X_R by computing an artifact of each sample in X_G, *i.e.* **Fingerprint**$(G; X_R) = \{a(x; X_R) \mid x \in X_G\}$.

3.3. Attribution network

Given an observed image, our goal is to predict its source generative model. Our attribution network takes as input our artifact representation of an image (computed in Step 2) and predicts the identity of its source generative model. To do so, we first represent the input image as an artifact feature by computing its deviation from the learned data manifold (as in Sec. 3.2). Note that we do not learn the Riemannian manifold (and its metric) at the inference time, since this step is done only once per real dataset provided during training or data-preprocessing steps. Next, the artifact feature is fed into our ResNet-based classifier to perform multi-class classification over the generative models. To train this classifier, we use the pretrained ResNet50 [25] as the backbone and fine-tune it using cross-entropy loss computed from the classification of images in the training split of our dataset in Tab. 2.

4. Experiments

We evaluate our proposed fingerprint definitions and attribution method on model attribution and generalization to unseen datasets, generative models and modalities. Sec. 4.1 outlines our experimental setup. Sec. 4.2 evaluates the attributability of GM fingerprints on a large array of generative models via (multi-class) model attribution and feature space analysis. Sec. 4.3 studies the generalizability of our fingerprints across dataset and model type, and Sec. 4.4 evaluates their robustness against post-processing operations.

4.1. Experimental setup

Datasets. To evaluate the performance of different fingerprints on model attribution for a multi-class classification, we use the GM datasets from [60] – GM-CIFAR10, GM-CelebA, GM-CHQ and GM-FFHQ – constructed from real datasets and generative models trained on CIFAR-10 [37], CelebA-64 [40], CelebA-HQ (256) [29] and FFHQ (256) [31], respectively. This dataset includes 100k images from each generative model, collectively covering GAN, VAE, Flow, and diffusion families, and state-of-the-art generative models (*e.g.*, NAVE, NCSN++, LSGM) that have not

Family	GM-CIFAR10	GM-CelebA	GM-CHQ	GM-FFHQ
Real	CIFAR-10 [37]	CelebA [40]	CelebA-HQ (256) [29]	FFHQ (256) [31]
GAN	BigGAN-Deep [5]	plain GAN [20]	BigGAN-Deep [5]	BigGAN-Deep [5]
	StyleGAN2 [32]	DCGAN [53]	StyleGAN2 [32]	StyleGAN2 [32]
		LSGAN [44]	StyleGAN3 [30]	StyleGAN3 [30]
	WGAN-gp [22]	WGAN-gp/lp [22]	VQ-GAN [17]	VQ-GAN [17]
		DRAGAN-gp/lp [36]	StyleSwin [70]	
	DDGAN [67]		DDGAN [67]	
VAE		β-VAE [26]		
		DFC-VAE [28]	StyleALAE [52]	
	NVAE [63]	NVAE [63]	NVAE [63]	NVAE [63]
	VAE-BM [66]	VAE-BM [66]	VAE-BM [66]	
	Eff-VDVAE [24]	Eff-VDVAE [24]	Eff-VDVAE [24]	Eff-VDVAE [24]
Flow	GLOW [34]	GLOW [34]		
	MaCow [41]		MaCow [41]	
			Residual Flow [11]	
Score-Based	DDPM [27]	DDPM [27]	DDPM [27]	
			NCSN++ [61]	NCSN++ [61]
	RVE [33]	RVE [33]	RVE [33]	
	LSGM [64]		LSGM [64]	
			LDM [55]	LDM [55]
Vision-Language	Flux.1-dev	Stable-Diffusion-3.5	Dall-E-3	Openjourney

Table 2. **Our experimental dataset of generative models.** We introduce an extended benchmark dataset for model attribution that includes SoTA multimodal GMs (*i.e.*, text-to-image models) in addition to the large array of SoTA GMs trained on 4 different datasets (CIFAR10, CelebA, CelebA-HQ, FFHQ) in 2 different resolutions (64×64, 256×256) from [60]. We evaluate the model fingerprints on their attributability and cross-data/model/modality generalization. **Real**: training datasets of the generative models.

been studied before. In addition, we extend this dataset to include SoTA multimodal GMs (vision-language models) to evaluate fingerprints' generalization across a wider variety of modalities. Tab. 2 summarizes our datasets, organized in column by the training datasets, with the last row indicating the vision-language models. See Appendix A for details on how we collected the images from the listed GMs, in particular the new vision-language models. We emphasize that each dataset in the column consists exclusively of images from models trained on the same training dataset. This consistency is crucial for evaluating the effects of model architectures and datasets on attribution and generalization independently, as discussed in Sec. 4.2 and Sec. 4.3.

Baselines We consider the three main categories of existing model attribution methods: color-based, frequency-based and supervised-learning methods. We evaluate representative methods from each group and compare them with our proposed method. Furthermore, we include ManiFPT [60], a SoTA fingerprinting method based on a Euclidean data manifold, in our comparison.

- Color-based methods: Histogram of saturated and under-exposed pixels [47], Color co-occurrence matrix [48]
- Frequency-based methods: 1-dim power spectrum via azimuthal integration on DCT [15], high-frequency decay parameters fitted to normalized reduced spectra [12, 16]
- Supervised learning methods: InceptionNet-v3 [45], XceptionNet [45], Yu et al. [69], Wang et al. [65]
- Manifold-based methods (Euclidean): $\text{ManiFPT}_{\text{RGB}}$, $\text{ManiFPT}_{\text{FREQ}}$ [60] which use RGB and frequency (FFT) spaces, respectively, as their embedding spaces

For comparison, we consider two variants of our attribution method, **R-GMfpts** which computes the RCM on $\mathcal{M}$ using Euclidean distances, and $\textbf{R-GMfpts}_{\textbf{RCM}}$ which uses the learned geodesic distances.

4.2. Attribution of generative models

We test the attributability of fingerprints by training a classifier for model attribution based solely on our computed artifacts. For other baselines, the inputs to the classifier may be a full image or frequency spectrum, depending on their expected representations. A high test accuracy implies the classifier is able to predict source models from the fingerprints, thereby supporting the existence of their fingerprints.

Metrics. We evaluate the attributability of the baseline methods presented in Sec. 4.1, along with our proposed methods, using our GM datasets (Tab. 2). Performance is measured in classification accuracy (%). We additionally measure the separability of fingerprint representations using the ratio of inter-class and intra-class Fréchet Distance (FDR) [14], as done in [60]. A higher FDR indicates greater attribution strength, suggesting the fingerprints are more distinct and traceable to their respective generative models.

Evaluation Protocol. Each dataset in Tab. 2 consists of real images and synthetic images from GMs trained on those real images. Each image is labeled with the ID of its source model (*e.g.*, 0 for Real, 1 for G_1, ..., M for G_M). We split the data into train, val, test in ratio of 7:2:1, train the methods on the train split and measure the accuracies on the test split.

Results. Tab. 3 shows the result of model attribution. First, we observe that our attribution methods (**R-GMfpts**, $\textbf{R-GMfpts}_{\textbf{RCM}}$) outperform all compared methods on all datasets with significant margins. In particular, our method achieves higher accuracies across all dataset than the Euclidean-based method (ManiFPT [60]), supporting that taking the Riemannian geometry structure of data when estimating the GM fingerprints improves their estimation.

Secondly, we note that our method that takes full advantage of the learned Riemannian metric on the data manifold ($\textbf{R-GMfpts}_{\textbf{RCM}}$) outperforms the one that does not and uses a standard Euclidean metric instead (**R-GMfpts**). This result suggests that computing the projection of x_G as a Riemannian center of mass using the learned geodesics contributes the estimation to lie closer to the data manifold, minimizing potential errors in the artifact computation. We visualize this point in Figure 3 (center vs. right) where z_{CM} (center) lies off the manifold whereas z_{RCM} (right) on the manifold.

4.3. Cross-dataset generalization

We evaluate the generalizability of fingerprinting methods across datasets. Since new models are developed constantly by training or fine-turning on new datasets, this cross-dataset generalizability is crucial in practice. We consider two sce-

Methods	GM-CIFAR10		GM-CelebA		GM-CHQ		GM-FFHQ	
	Acc.(%)↑	FDR↑	Acc.(%)↑	FDR↑	Acc.(%)↑	FDR↑	Acc.(%)↑	FDR↑
McCloskey et al. [47]	40.22 ± 1.10	32.4	62.6 ± 0.31	70.2	57.4 ± 0.81	36.3	50.8 ± 0.34	26.3
Nataraj et al. [48]	46.29 ± 1.43	36.7	61.1 ± 0.91	74.0	56.3 ± 0.32	37.9	51.3 ± 0.58	35.3
Durall et al. [15]	57.29 ± 0.93	46.5	62.2 ± 0.24	75.5	59.1 ± 0.80	38.8	60.9 ± 0.25	37.9
Dzanic et al. [16]	56.12 ± 1.21	43.1	61.6 ± 1.02	88.1	56.9 ± 1.21	38.2	55.7 ± 0.32	30.3
Corvi et al. [12]	60.12 ± 0.14	47.2	59.2 ± 0.59	46.5	60.5 ± 0.76	48.2	59.3 ± 0.53	49.2
Wang et al. [65]	62.23 ± 0.84	53.6	62.2 ± 1.20	89.8	59.5 ± 1.25	30.3	64.2 ± 0.31	37.9
Marra et al. (MIPR) [45]	55.94 ± 1.09	41.2	63.1 ± 1.10	83.4	51.3 ± 1.28	20.5	53.2 ± 0.21	30.4
Marra et al. (WIFS) [46]	60.71 ± 1.24	47.2	61.1 ± 1.72	101.4	59.1 ± 0.75	34.9	51.8 ± 0.23	30.9
Yu et al. [69]	62.01 ± 0.79	50.1	60.6 ± 1.10	111.4	61.1 ± 1.12	73.3	60.5 ± 0.10	35.1
ManiFPT$_{\text{RGB}}$ [60]	69.48 ± 1.08	55.2	70.5 ± 1.56	115.3	63.7 ± 0.63	64.2	63.3 ± 0.12	50.1
ManiFPT$_{\text{FREQ}}$ [60]	70.19 ± 0.96	57.2	72.8 ± 1.32	120.9	54.8 ± 0.32	70.1	63.8 ± 0.20	43.8
R-GMfpts (ours)	$\underline{72.01} \pm 0.92$	$\underline{58.9}$	$\underline{73.6} \pm 0.70$	**168.0**	$\underline{64.3} \pm 0.72$	$\underline{77.2}$	$\underline{65.2} \pm 0.30$	$\underline{58.8}$
R-GMfpts$_{\text{RCM}}$ (ours)	**78.17** ± 0.53	**60.1**	**74.7** ± 0.62	$\underline{125.9}$	**65.8** ± 0.75	**74.5**	**67.1** ± 0.20	**60.8**

Table 3. **Model attribution results.** We evaluate different artifact features on predicting the source generative model of a generated sample. Separability of the feature spaces are measured in Fréchet distance ratio (FDR). Higher FDR means better separability. Our methods based on the proposed definition of artifacts outperform all baseline methods on all datasets (results from [60] and our evaluations).

Methods	C10→CA	CA→C10	CHQ→FFHQ	FFHQ→CHQ
McCloskey et al. [47]	52.3 ± 0.14	43.2 ± 0.08	34.2 ± 0.13	31.2 ± 0.10
Nataraj et al. [48]	56.2 ± 0.11	46.1 ± 0.19	42.1 ± 0.07	40.4 ± 0.19
Durall et al. [15]	60.1 ± 0.14	53.5 ± 0.12	51.9 ± 0.09	42.6 ± 0.12
Dzanic et al. [16]	56.9 ± 0.13	54.7 ± 0.11	45.2 ± 0.22	42.5 ± 0.17
Corvi et al. [12]	59.2 ± 0.21	59.3 ± 0.14	48.1 ± 0.35	46.6 ± 0.14
Wang et al. [65]	62.5 ± 0.15	60.1 ± 0.10	61.4 ± 0.13	53.4 ± 0.12
Marra et al. (MIPR) [45]	57.0 ± 0.14	58.4 ± 0.33	50.2 ± 0.17	35.9 ± 0.27
Marra et al. (WIFS) [46]	61.0 ± 0.15	58.6 ± 0.23	54.3 ± 0.12	30.3 ± 0.17
Yu et al. [69]	60.5 ± 0.13	60.4 ± 0.20	55.2 ± 0.17	50.3 ± 0.13
ManiFPT$_{\text{RGB}}$ [60]	62.7 ± 0.14	60.2 ± 0.15	58.1 ± 0.22	53.2 ± 0.11
ManiFPT$_{\text{FREQ}}$ [60]	65.8 ± 0.12	62.1 ± 0.11	57.6 ± 0.20	53.5 ± 0.18
R-GMfpts (ours)	**68.2** ± 0.11	$\underline{62.3} \pm 0.16$	$\underline{63.5} \pm 0.14$	**56.9** ± 0.28
R-GMfpts$_{\text{RCM}}$ (ours)	$\underline{67.8} \pm 0.21$	**65.0** ± 0.12	**67.3** ± 0.24	$\underline{54.3} \pm 0.32$

Table 4. **Generalization of model attribution across datasets.** We evaluate how well baselines and our fingerprints generalize across training datasets. We consider two scenarios: (i) generalization across GM-CIFAR10 and GM-CelebA, and (ii) generalization across GM-CHQ and GM-FFHQ. For each case, we train attribution methods on one set of generative models (*e.g.*, GM-CIFAR10) and test on a different set of models (*e.g.*, GM-CelebA). Our artifact-based attribution method outperforms all baseline methods in both scenarios. **C10**: CIFAR-10. **CA**: CelebA. **CHQ**: CelebA-HQ.

narios whose data semantics are meaningfully different: generalization (i) between GM-CIFAR10 and GM-CelebA, and (ii) between GM-CHQ and GM-FFHQ.

Evaluation. In each case, we train attribution methods on the training dataset (*e.g.*, GM-CelebA) and test their accuracies on the other unseen dataset (*e.g.*, GM-CIFAR10).

Results. Tab. 4 shows the result of cross-dataset generalizations between GM-CIFAR10 and GM-CelebA, and between GM-CHQ and GM-FFHQ. First, our attribution methods based on Riemannian fingerprints (**R-GMfpts$_{\text{RCM}}$**, **R-GMfpts**) outperform all compared methods in both cases of generalization. In particular, given that CIFAR-10 and CelebA contain images from different semantic domains (CIFAR-10: objects and animals vs. CelebA: faces), the high generalization between these datasets indicate our new fingerprints improve generalization not only across datasets of similar semantics (CHQ $\leftrightarrow$ FFHQ), but also across different ones (CIFAR-10 $\leftrightarrow$ CelebA). Overall, these higher accuracies show that our methods are more generalizable to the change of datasets, supporting their improved efficacy in practice where attribution needs to address new models fine-tuned by various entities using their own datasets.

4.4. Robustness against post-processing

We conduct additional experiments to evaluate robustness against two common post-processing perturbations: gaussian blurring (with increasing standard deviations) and JPEG compression (with decreasing quality factors). We apply these perturbations at test time and evaluate attribution accuracies on all four datasets, using all 12 baseline methods and our methods. Our results in Figure 4 show that our methods (colored purple) consistently outperform all baselines, under both perturbation types and across all datasets.

5. Conclusion

Our work addresses an increasingly critical problem of attributing and fingerprinting GMs, by proposing fingerprints that generalize to non-Euclidean data using Riemannian geometry. Our experiments on an extended SoTA dataset showed that our method is more effective in distinguishing a wider variety of GMs (4 datasets in 2 resolutions, 27 model architectures, 2 modalities). Using our definition significantly improves the performance on model attribution and generalization. We believe this generalized definition of GM fingerprints can help address gaps in the theory and practical understanding of GMs, by providing a geometric framework to systematically analyze the characteristics of GMs.

References

[1] Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, et al. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774*, 2023. 1

[2] Bijan Afsari, Roberto Tron, and René Vidal. On the convergence of gradient descent for finding the riemannian center of mass. *SIAM Journal on Control and Optimization*, 51(3):2230–2260, 2013. 3, 5

[3] Georgios Arvanitidis, Miguel González-Duque, Alison Pouplin, Dimitris Kalatzis, and Søren Hauberg. Pulling back information geometry. *arXiv preprint arXiv:2106.05367*, 2021. 1, 3, 4

[4] Georgios Arvanitidis, Lars Kai Hansen, and Søren Hauberg. Latent space oddity: on the curvature of deep generative models. *arXiv preprint arXiv:1710.11379*, 2017. 1, 2, 3, 4, 5

[5] Andrew Brock, Jeff Donahue, and Karen Simonyan. Large Scale GAN Training for High Fidelity Natural Image Synthesis. In *International Conference on Learning Representations*, Jan. 2023. 7

[6] Michael M Bronstein, Joan Bruna, Yann LeCun, Arthur Szlam, and Pierre Vandergheynst. Geometric deep learning: going beyond euclidean data. *IEEE Signal Processing Magazine*, 34(4):18–42, 2017. 2

[7] Tim Brooks, Bill Peebles, Connor Holmes, Will DePue, Yufei Guo, Li Jing, David Schnurr, Joe Taylor, Troy Luhman, Eric Luhman, Clarence Ng, Ricky Wang, and Aditya Ramesh. Video generation models as world simulators. 2024. 1

[8] Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901, 2020. 1

[9] Lawrence Cayton. Algorithms for manifold learning. *Univ. of California at San Diego Tech. Rep*, 12(1-17):1, 2005. 3

[10] Keshigeyan Chandrasegaran, Ngoc-Trung Tran, and Ngai-Man Cheung. A Closer Look at Fourier Spectrum Discrepancies for CNN-generated Images Detection, Mar. 2021. arXiv:2103.17195 [cs, eess]. 1

[11] Ricky T. Q. Chen, Jens Behrmann, David K Duvenaud, and Joern-Henrik Jacobsen. Residual Flows for Invertible Generative Modeling. In *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc., 2019. 7

[12] Riccardo Corvi, Davide Cozzolino, Giovanni Poggi, Koki Nagano, and Luisa Verdoliva. Intriguing properties of synthetic images: From generative adversarial networks to diffusion models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) Workshops*, pages 973–982, June 2023. 7, 8

[13] Elvis Dohmatob, Yunzhen Feng, Pu Yang, Francois Charton, and Julia Kempe. A tale of tails: Model collapse as a change of scaling laws. *arXiv preprint arXiv:2402.07043*, 2024. 1

[14] D.C. Dowson and B.V. Landau. The fréchet distance between multivariate normal distributions. *Journal of multivariate analysis*, 12(3):450–455, 1982. 7

[15] R. Durall, Margret Keuper, and J. Keuper. Watch Your Up-Convolution: CNN Based Generative Deep Neural Networks Are Failing to Reproduce Spectral Distributions. *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020. 2, 7, 8

[16] Tarik Dzanic, Karan Shah, and Freddie Witherden. Fourier Spectrum Discrepancies in Deep Network Generated Images. In *Advances in Neural Information Processing Systems*, volume 33, pages 3022–3032. Curran Associates, Inc., 2020. 2, 7, 8

[17] Patrick Esser, Robin Rombach, and Bjorn Ommer. Taming Transformers for High-Resolution Image Synthesis. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 12873–12883, 2021. 7

[18] Charles Fefferman, Sanjoy Mitter, and Hariharan Narayanan. Testing the manifold hypothesis. *Journal of the American Mathematical Society*, 29(4):983–1049, 2016. 3

[19] Sylvestre Gallot, Dominique Hulin, Jacques Lafontaine, Sylvestre Gallot, Dominique Hulin, and Jacques Lafontaine. Riemannian metrics. *Riemannian Geometry*, pages 51–127, 2004. 2

[20] Ian J. Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron C. Courville, and Yoshua Bengio. Generative adversarial nets. In *NIPS*, 2014. 7

[21] Luca Guarnera, Oliver Giudice, and Sebastiano Battiato. DeepFake Detection by Analyzing Convolutional Traces. Apr. 2020. 1

[22] Ishaan Gulrajani, Faruk Ahmed, Martín Arjovsky, Vincent Dumoulin, and Aaron C. Courville. Improved training of wasserstein gans. In *NIPS*, 2017. 7

[23] Søren Hauberg. Only bayes should learn a manifold (on the estimation of differential geometric structure from data). *arXiv preprint arXiv:1806.04994*, 2018. 1, 3, 5, 6

[24] Louay Hazami, Rayhane Mama, and Ragavan Thurairatnam. Efficient-VDVAE: Less is more, Apr. 2022. 7

[25] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016. 6

[26] Irina Higgins, Loïc Matthey, Arka Pal, Christopher P. Burgess, Xavier Glorot, Matthew M. Botvinick, Shakir Mohamed, and Alexander Lerchner. beta-vae: Learning basic visual concepts with a constrained variational framework. In *ICLR*, 2017. 7

[27] Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising Diffusion Probabilistic Models. In *Advances in Neural Information Processing Systems*, volume 33, pages 6840–6851. Curran Associates, Inc., 2020. 7

[28] Xianxu Hou, L. Shen, Ke Sun, and Guoping Qiu. Deep feature consistent variational autoencoder. *2017 IEEE Winter Conference on Applications of Computer Vision (WACV)*, pages 1133–1141, 2017. 7

[29] Tero Karras, Timo Aila, Samuli Laine, and Jaakko Lehtinen. Progressive growing of gans for improved quality, stability, and variation. *ArXiv*, abs/1710.10196, 2018. 2, 6, 7

[30] Tero Karras, Miika Aittala, Samuli Laine, Erik Harkonen, Janne Hellsten, Jaakko Lehtinen, and Timo Aila. Alias-free generative adversarial networks. In *NeurIPS*, 2021. 1, 7

[31] Tero Karras, Samuli Laine, and Timo Aila. A style-based generator architecture for generative adversarial networks. *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 4396–4405, 2019. 2, 6, 7

[32] Tero Karras, Samuli Laine, Miika Aittala, Janne Hellsten,

Jaakko Lehtinen, and Timo Aila. Analyzing and Improving the Image Quality of StyleGAN. In *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 8107–8116, Seattle, WA, USA, June 2020. IEEE. 7

[33] Dongjun Kim, Seungjae Shin, Kyungwoo Song, Wanmo Kang, and Il-Chul Moon. Soft Truncation: A Universal Training Technique of Score-based Diffusion Model for High Precision Score Estimation. In *Proceedings of the 39th International Conference on Machine Learning*, pages 11201–11228. PMLR, June 2022. 7

[34] Durk P Kingma and Prafulla Dhariwal. Glow: Generative Flow with Invertible 1x1 Convolutions. In *Advances in Neural Information Processing Systems*, volume 31. Curran Associates, Inc., 2018. 7

[35] Diederik P. Kingma and Max Welling. Auto-encoding variational bayes. *CoRR*, abs/1312.6114, 2014. 5

[36] Naveen Kodali, Jacob Abernethy, James Hays, and Zsolt Kira. On convergence and stability of gans. *arXiv preprint arXiv:1705.07215*, 2017. 7

[37] Alex Krizhevsky. Learning multiple layers of features from tiny images. 2009. 2, 6, 7

[38] Huiling Le. Estimation of riemannian barycentres. *LMS Journal of Computation and Mathematics*, 7:193–200, 2004. 5

[39] John M Lee. *Introduction to Riemannian manifolds*, volume 2. Springer, 2018. 1, 2

[40] Ziwei Liu, Ping Luo, Xiaogang Wang, and Xiaoou Tang. Deep learning face attributes in the wild. In *Proceedings of International Conference on Computer Vision (ICCV)*, December 2015. 2, 6, 7, 12

[41] Xuezhe Ma and Eduard H. Hovy. Macow: Masked convolutional generative flow. In *NeurIPS*, 2019. 7

[42] Tambiama Madiega. Artificial intelligence act. *European Parliament: European Parliamentary Research Service*, 2021. 1

[43] Jonathan H Manton. A globally convergent numerical algorithm for computing the centre of mass on compact lie groups. In *ICARCV 2004 8th Control, Automation, Robotics and Vision Conference, 2004.*, volume 3, pages 2211–2216. IEEE, 2004. 5

[44] Xudong Mao, Qing Li, Haoran Xie, Raymond Y. K. Lau, Zhen Wang, and Stephen Paul Smolley. Least squares generative adversarial networks. *2017 IEEE International Conference on Computer Vision (ICCV)*, pages 2813–2821, 2017. 7

[45] Francesco Marra, Diego Gragnaniello, Davide Cozzolino, and Luisa Verdoliva. Detection of GAN-Generated Fake Images over Social Networks. In *2018 IEEE Conference on Multimedia Information Processing and Retrieval (MIPR)*, pages 384–389, Apr. 2018. 7, 8

[46] Francesco Marra, Cristiano Saltori, Giulia Boato, and Luisa Verdoliva. Incremental learning for the detection and classification of GAN-generated images. In *2019 IEEE International Workshop on Information Forensics and Security (WIFS)*, pages 1–6, Dec. 2019. 8

[47] Scott McCloskey and Michael Albright. Detecting gan-generated imagery using saturation cues. In *2019 IEEE international conference on image processing (ICIP)*, pages 4584–4588. IEEE, 2019. 2, 7, 8

[48] Lakshmanan Nataraj, Tajuddin Manhar Mohammed, Shivkumar Chandrasekaran, Arjuna Flenner, Jawadul H. Bappy, Amit K. Roy-Chowdhury, and B. S. Manjunath. Detecting GAN generated Fake Images using Co-occurrence Matrices, Oct. 2019. 7, 8

[49] Thanh Thi Nguyen, Quoc Viet Hung Nguyen, Dung Tien Nguyen, Duc Thanh Nguyen, Thien Huynh-The, Saeid Nahavandi, Thanh Tam Nguyen, Quoc-Viet Pham, and Cuong M Nguyen. Deep learning for deepfakes creation and detection: A survey. *Computer Vision and Image Understanding*, 223:103525, 2022. 1

[50] Augustus Odena, Vincent Dumoulin, and Chris Olah. Deconvolution and checkerboard artifacts. *Distill*, 2016. 2

[51] US Copyright Office. Copyright registration guidance: Works containing material generated by artificial intelligence., 2023. 1

[52] Stanislav Pidhorskyi, Donald A. Adjeroh, and Gianfranco Doretto. Adversarial Latent Autoencoders. In *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 14092–14101, Seattle, WA, USA, June 2020. IEEE. 7

[53] Alec Radford, Luke Metz, and Soumith Chintala. Unsupervised representation learning with deep convolutional generative adversarial networks. *CoRR*, abs/1511.06434, 2016. 7

[54] Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-resolution image synthesis with latent diffusion models, 2022. 1

[55] Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-Resolution Image Synthesis With Latent Diffusion Models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10684–10695, 2022. 7

[56] Andreas Rossler, Davide Cozzolino, Luisa Verdoliva, Christian Riess, Justus Thies, and Matthias Nießner. Faceforensics++: Learning to detect manipulated facial images. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 1–11, 2019. 1

[57] Takashi Sakai. *Riemannian geometry*, volume 149. American Mathematical Soc., 1996. 2

[58] Katja Schwarz, Yiyi Liao, and Andreas Geiger. On the Frequency Bias of Generative Models. In *Advances in Neural Information Processing Systems*, volume 34, pages 18126–18136. Curran Associates, Inc., 2021. 1

[59] Ilia Shumailov, Zakhar Shumaylov, Yiren Zhao, Nicolas Papernot, Ross Anderson, and Yarin Gal. Ai models collapse when trained on recursively generated data. *Nature*, 631(8022):755–759, 2024. 1

[60] Hae Jin Song, Mahyar Khayatkhoei, and Wael AbdAlmageed. Manifpt: Defining and analyzing fingerprints of generative models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10791–10801, 2024. 1, 2, 3, 4, 6, 7, 8, 12

[61] Yang Song, Jascha Sohl-Dickstein, Diederik P. Kingma, Abhishek Kumar, Stefano Ermon, and Ben Poole. Score-Based Generative Modeling through Stochastic Differential Equations. In *International Conference on Learning Representations*, Jan. 2023. 7

[62] Gemini Team, Rohan Anil, Sebastian Borgeaud, Yonghui

Wu, Jean-Baptiste Alayrac, Jiahui Yu, Radu Soricut, Johan Schalkwyk, Andrew M Dai, Anja Hauth, et al. Gemini: a family of highly capable multimodal models. *arXiv preprint arXiv:2312.11805*, 2023. 1

[63] Arash Vahdat and Jan Kautz. NVAE: A Deep Hierarchical Variational Autoencoder. In *Advances in Neural Information Processing Systems*, volume 33, pages 19667–19679. Curran Associates, Inc., 2020. 7

[64] Arash Vahdat, Karsten Kreis, and Jan Kautz. Score-based Generative Modeling in Latent Space. In *Advances in Neural Information Processing Systems*, volume 34, pages 11287–11302. Curran Associates, Inc., 2021. 7

[65] Sheng-Yu Wang, O. Wang, Richard Zhang, Andrew Owens, and Alexei A. Efros. CNN-Generated Images Are Surprisingly Easy to Spot... for Now. *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020. 1, 2, 7, 8

[66] Zhisheng Xiao, Karsten Kreis, Jan Kautz, and Arash Vahdat. VAEBM: A Symbiosis between Variational Autoencoders and Energy-based Models. In *International Conference on Learning Representations*, Feb. 2022. 7

[67] Zhisheng Xiao, Karsten Kreis, and Arash Vahdat. Tackling the generative learning trilemma with denoising diffusion gans. 2022. 7

[68] Jiashu Xu, Fei Wang, Mingyu Derek Ma, Pang Wei Koh, Chaowei Xiao, and Muhao Chen. Instructional fingerprinting of large language models. *arXiv preprint arXiv:2401.12255*, 2024. 2

[69] Ning Yu, Larry Davis, and Mario Fritz. Attributing fake images to gans: Learning and analyzing gan fingerprints. *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 7555–7565, 2019. 1, 2, 7, 8

[70] Bowen Zhang, Shuyang Gu, Bo Zhang, Jianmin Bao, Dong Chen, Fang Wen, Yong Wang, and Baining Guo. StyleSwin: Transformer-based GAN for High-resolution Image Generation. In *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 11294–11304, New Orleans, LA, USA, June 2022. IEEE. 7

Acknowledgements

This work was supported by the National Science Foundation (award 2318101), and a research grant from Sandia National Laboratories. The authors affirm that the views expressed herein are solely their own, and do not represent the views of the United States government or any agency thereof.