Hardness of Learning Neural Networks under the Manifold Hypothesis

Bobak T. Kiani* Jason Wang† Melanie Weber‡

Abstract

The *manifold hypothesis* presumes that high-dimensional data lies on or near a low-dimensional manifold. While the utility of encoding geometric structure has been demonstrated empirically, rigorous analysis of its impact on the learnability of neural networks is largely missing. Several recent results have established hardness results for learning feedforward and equivariant neural networks under i.i.d. Gaussian or uniform Boolean data distributions. In this paper, we investigate the hardness of learning under the manifold hypothesis. We ask which minimal assumptions on the curvature and regularity of the manifold, if any, render the learning problem efficiently learnable. We prove that learning is hard under input manifolds of bounded curvature by extending proofs of hardness in the SQ and cryptographic settings for Boolean data inputs to the geometric setting. On the other hand, we show that additional assumptions on the volume of the data manifold alleviate these fundamental limitations and guarantee learnability via a simple interpolation argument. Notable instances of this regime are manifolds which can be reliably reconstructed via manifold learning. Looking forward, we comment on and empirically explore intermediate regimes of manifolds, which have heterogeneous features commonly found in real world data.[4]

1 Introduction

High-dimensional data is often thought to have low-dimensional structure, which may stem, for instance, from symmetries in the underlying system. This observation has given rise to the *manifold hypothesis*, which presumes that high-dimensional data lies on or near a low-dimensional manifold. Empirical studies have confirmed this hypothesis across domains [20, 77, 37, 65]. A plethora of algorithms for geometric data analysis [44, 8, 80, 21] and, more recently, machine learning [30, 15, 23] seek to leverage such structure. While these methods have shown empirical promise, few formal results on the benefits of encoding geometric structure have been established. Here, we study the impact of the manifold hypothesis on the computational complexity of learning algorithms for neural networks. Specifically, we investigate, under which geometric assumptions feedforward neural networks are efficiently learnable.

The computational hardness of learning shallow, fully-connected neural networks under i.i.d. Gaussian data distributions has been established by [36, 26, 48, 32], who show that the complexity of learning even single hidden layer neural networks grows at least exponentially with the input size. Some of this analysis utilizes the correlational statistical query (CSQ) framework, of which learnability via gradient decent is a notable instance. Recently, [57] have shown similar hardness results

*John A. Paulson School of Engineering and Applied Sciences, Harvard University; e-mail: bkiani@g.harvard.edu.

†Harvard College, Harvard University; e-mail: jasonwang1@college.harvard.edu.

‡John A. Paulson School of Engineering and Applied Sciences, Harvard University; e-mail: mweber@g.harvard.edu

[4]Code is made public at https://github.com/Weber-GeoML/manifold-learning-complexity

38th Conference on Neural Information Processing Systems (NeurIPS 2024).

for equivariant neural networks, a class of geometric architectures that explicitly encode symmetries. Both lines of work indicate that additional assumptions on the neural network architecture or the data geometry are needed to establish learnability. Here, we focus on the latter.

A separate body of literature investigates nonlinear, low-dimensional structure in data. Approaches for estimating intrinsic dimension [44, 63, 87] and curvature [1, 86, 46] from data seek to characterize the geometry of low-dimensional manifolds. *Manifold Learning* aims to identify low-dimensional structure by reconstructing low-dimensional manifolds from data. Methods such as Multi-Dimensional Scaling [62], Isomap [89] or Diffusion Maps [31] have shown empirical success in learning low-dimensional data manifolds. For some of these approaches, formal guarantees on their reliability with respect to the geometric characteristics and sample size of the data are known [9]. More generally, the manifold learning problem and its complexity have been formally studied in [43, 42, 3]. However, the impact of data geometry on the complexity of learning neural networks in downstream tasks remains open.

In this paper, we ask: *Which assumptions on the data geometry guarantee learnability of neural networks?* We show that the manifold hypothesis on its own does not guarantee learnability. In particular, we give hardness results for a class of low-dimensional manifolds in the statistical query (SQ) model or under cryptographic hardness assumptions. Our work follows an established line of proof techniques in the SQ literature [32, 48, 26], extending hardness results for neural network training in the Boolean and Gaussian input models to more general geometries. We further show that additional assumptions on the volume and curvature of the data manifold alleviate the fundamental limitations and guarantee learnability through a rather simple interpolation argument. In particular, manifolds which can be reliably reconstructed via manifold learning are in this regime. We further discuss geometric regimes in which our results do not directly apply, and in which provable learnability remains an open question. We illustrate our learnability results through computational experiments on neural network training in the learnable and provably hard regimes. We further complement our theoretical analysis with a brief computational study of the intrinsic dimension of image data manifolds, with the goal of testing the geometric assumptions in our framework.

2 Background

2.1 Basic Notation

We denote scalars, vectors, and matrices as $v, \boldsymbol{v}$, and $\boldsymbol{V}$ respectively. We consider ambient spaces $\mathbb{R}^n$ equipped with the usual inner product $\langle \cdot, \cdot \rangle$ and associated ℓ_2 norm $\|\cdot\|$. Submanifolds $\mathcal{M} \subset \mathbb{R}^n$ considered in this study have intrinsic dimension $d \leq n$ and are compact and connected (unless otherwise stated). The tangent space $T_{\boldsymbol{p}}\mathcal{M}$ at a point $\boldsymbol{p} \in \mathcal{M}$ denotes the d-dimensional linear subspace of $\mathbb{R}^n$ spanned by velocity vectors of smooth curves incident at $\boldsymbol{p}$. Given a subset $S \subset \mathbb{R}^n$, we denote by $d(\boldsymbol{z}, S) = \inf_{\boldsymbol{p} \in S} \|\boldsymbol{z} - \boldsymbol{p}\|$ the distance of a point $\boldsymbol{z}$ to S. We denote Vol_d as the d-dimensional volume measure inherited from the d-dimensional Hausdorff measure and denote $\omega_d(r)$ and $\sigma_d(r)$ as the volume of the d-dimensional ball and d-dimensional sphere of radius r respectively. For a point $\boldsymbol{p} \in \mathcal{M}$ in a given manifold $\mathcal{M}$, $\mathrm{Vol}_{\mathcal{M}}(\boldsymbol{p}, r)$ denotes the volume of the ball of radius r around the point $\boldsymbol{p}$ with respect to the Riemannian distance metric.

2.2 Learning Setting

We consider the task of learning feedforward neural networks $f : \mathbb{R}^n \to \mathbb{R}$ composed of L hidden layers $f^{(\ell)} : \mathbb{R}^{d_\ell} \to \mathbb{R}^{d_{\ell-1}}$ taking the form

$$\begin{aligned} f(\boldsymbol{x}) &= \boldsymbol{v}^\top f^{(L)} \left(f^{(L-1)} \left(\cdots \left(f^{(1)} \left(\boldsymbol{x} \right) \right) \cdots \right) \right), \\ f^{(\ell)}(\boldsymbol{h}) &= \mathrm{ReLU}\left(\boldsymbol{W}_\ell \boldsymbol{h} \right) + \boldsymbol{b}_\ell, \end{aligned} \tag{1}$$

where $\boldsymbol{v} \in \mathbb{R}^{d_L}$, $\boldsymbol{W}_\ell \in \mathbb{R}^{d_\ell \times d_{\ell-1}}$ and $\boldsymbol{b} \in \mathbb{R}^{d_\ell}$ are trainable weights. The input dimension d_0 is set to the ambient dimension n and the output is scalar-valued. Throughout we will consider the setting where the weight entries and hidden widths are bounded by $O(\mathrm{poly}(n))$. When guaranteeing that a class of networks is learnable, we will assume that the number of layers $L = O(1)$ is constant with respect to the input dimension. Our formal hardness results will apply for single hidden layer networks ($L = 1$).

We consider learnability in the distribution-specific probably approximately correct (PAC) setting where the goal is to produce an algorithm which can learn a target function given polynomial time and samples [68].

Definition 2.1 (Efficiently PAC Learnable)**.** A concept class $\mathcal{C}$ consisting of functions $c : \mathcal{X} \to \mathcal{Y}$ is *efficiently PAC-learnable* over distribution $\mathcal{D}$ on $\mathcal{X} \subseteq \mathbb{R}^n$, if there exists an algorithm such that for any $\epsilon > 0$ and $\delta > 0$, and for any target concept $c^* \in \mathcal{C}$, the algorithm takes at most $m = \text{poly}(1/\epsilon, 1/\delta, n)$ samples drawn i.i.d. from $(\boldsymbol{x}, c^*(\boldsymbol{x}))$ with $\boldsymbol{x} \sim \mathcal{D}$, and returns a function f satisfying $\|f - c^*\|_{\mathcal{D}} \coloneqq \sqrt{\mathbb{E}_{\boldsymbol{x}\in\mathcal{D}}[(f(\boldsymbol{x}) - c^*(\boldsymbol{x}))^2]} \leq \epsilon$ with probability at least $1 - \delta$ in time at most $\text{poly}(1/\epsilon, 1/\delta, n)$.

Note, that the above is a distribution specific instance of PAC learning as we require the algorithm to work only for a given distribution and not all distributions. In Section 3.2, we will also show a hard class of functions which is likely not efficiently PAC learnable. Hardness results are proven in the restricted **statistical query (SQ)** setting, a query complexity based model for proving hardness capturing most algorithms in practice [55, 78]. Given a joint distribution $\mathcal{D}$ on input/output space $\mathcal{X} \times \mathcal{Y}$, any SQ algorithm is composed of a set of queries. Each query takes as input a function $g : \mathcal{X} \times \mathcal{Y} \to [-1, 1]$ and tolerance parameter $\tau > 0$, and returns a value $\text{SQ}(g, \tau)$ in the range:

$$\left|\mathbb{E}_{(\boldsymbol{x},y)\sim\mathcal{D}}\left[g(\boldsymbol{x}, y)\right] - \text{SQ}(g, \tau)\right| \leq \tau. \tag{2}$$

Gradient based algorithms can be queried by, for example, setting $g(\boldsymbol{x}, y) = C\frac{\partial}{\partial\theta}\left(\text{NN}_\theta(\boldsymbol{x}) - y\right)^2$ to estimate the gradient of a neural network NN_θ with respect to parameter θ for the MSE loss (constant C chosen so that outputs of g are bounded in $[-1, 1]$ forming a valid query). Hardness is quantified in the number of queries needed to learn a function c^* drawn from function class $\mathcal{C}$.

Manifold smoothness restrictions. To conform to practical settings where input features are normalized (e.g. image pixel values in the range $[0, 1]$), input distributions are assumed to be supported on smooth d-dimensional sub-manifolds $\mathcal{M} \subset [0, 1]^n$ of the n-dimensional hypercube. For any given manifold $\mathcal{M}$, we will assume that data distributions $\mathcal{D}_{\mathcal{M}}$ supported on that manifold have a smooth density f with respect to the volume measure and are appropriately bounded such that $\rho_{\max} \coloneqq \frac{\max_{\boldsymbol{x}\in\mathcal{M}} f(\boldsymbol{x})}{\min_{\boldsymbol{x}\in\mathcal{M}} f(\boldsymbol{x})} = O(\text{poly}(n))$.

We will also place curvature restrictions on the manifold by bounding its reach, a global smoothness quantity introduced by [41] and commonly studied in the manifold learning community [1, 3, 43, 47]. Informally, it is the largest number D, such that any point in ambient space at distance less than D has a unique nearest neighbor in the manifold $\mathcal{M}$. It is defined more formally from descriptions of the medial axis (Figure 1).

Definition 2.2 (Reach from medial axis)**.** Given a closed subset $S \subset \mathbb{R}^n$, the medial axis $\text{Med}(S)$ of S consists of the set of points with no unique nearest neighbor:

$$\text{Med}(S) = \{\boldsymbol{z} \in \mathbb{R}^n : \exists \boldsymbol{p} \neq \boldsymbol{q} \in S, \|\boldsymbol{p} - \boldsymbol{z}\| = \|\boldsymbol{q} - \boldsymbol{z}\| = d(\boldsymbol{z}, S)\}. \tag{3}$$

The reach $\text{Rch}\,(S)$ of S is the minimal distance from S to $\text{Med}(S)$:

$$\text{Rch}\,(S) = \inf_{\boldsymbol{z}\in\text{Med}(S)} \text{dist}(\boldsymbol{z}, S). \tag{4}$$

Bounds on the reach imply corresponding bounds on the radius of curvature (related to the geodesic and normal curvature) at any point in the manifold since $\text{Rch}\,(\mathcal{M})^{-1} \geq \|\gamma''_{\gamma(t),\gamma'(t)}(t)\|$ for any unit-speed geodesic $\gamma : \mathbb{R} \to \mathcal{M}$ where $\|\gamma'_{\gamma(t)}(t)\| = 1$ for all $t \in \mathbb{R}$ [3]. We will encounter a second classical curvature notion, *Ricci curvature*, which is a local, intrinsic curvature notion that characterizes the volume growth of geodesic balls (see sec. B.1 for a more formal definition). Positive lower bounds on the Ricci curvature imply that the manifold has a bounded diameter, a fact that we will use below.

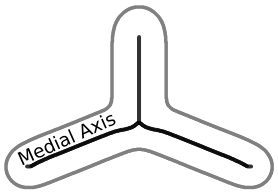


Figure 1: Example of a one-dimensional manifold and its medial axis. Its reach is given by the minimum distance of the medial axis to the manifold.

2.3 Related works

Here, we briefly summarize the most relevant prior work to our study and reserve Appendix A for a more detailed discussion. Our study lies at the intersection of research in manifold learning complexity and neural network learnability. In manifold learning, various works have analyzed the complexity of learning tasks over input manifolds. In one line of work, estimation of smooth manifolds in Hausdorff loss has been shown to require sample complexity of $O(\epsilon^{-d/2}\log(1/\epsilon))$, which is independent of the ambient dimension n [14, 47, 58]; later works also provided algorithms that run in time linear in n [4, 38]. In a learning setting similar to our work, [3] provide upper and lower bounds on the complexity of manifold reconstruction in SQ settings. In a separate context, [71, 43] show that the sample complexity for determining whether a dataset is within a class of manifolds of specified intrinsic dimension, curvature, and volume bounds grows exponentially with the intrinsic dimension and polynomially in the reach and volume. [71] also categorizes the sample complexity for binary classification over smooth cuts on a data manifold where smoothness is defined by the condition number of the classification boundary of the manifold (a quantity closely related to reach).

The hardness of learning neural networks has a long history that we detail further in Appendix A. Under the i.i.d. Gaussian input model, superpolynomial [48] and exponential [36] lower bounds for learning single hidden layer networks in CSQ settings have been shown. For uniformly random Boolean inputs, [32] reduce the problem of learning single hidden layer neural networks to a cryptographically hard problem, a technique, which we also use in Appendix C.3. They also show how to "Booleanize" Gaussian inputs to show hardness for three hidden layer networks, which was later extended to SQ and cryptographic hardness of learning two hidden layer networks in [26]. [57] used similar techniques to give hardness results for equivariant neural networks. To the best of our knowledge, the hardness of learning feedforward neural networks under more general data geometries, such as under the manifold hypothesis, has not been studied previously. Various works have found efficient learning algorithms under i.i.d. input assumptions when the weights of the networks are restricted in their rank, condition number, positivity, and other criteria [90, 27, 7, 36]. We consider the generic setting where weight matrices are bounded in width and magnitude by $O(\mathrm{poly}(n))$, but otherwise unrestricted.

3 Learnability results

Our main results prove the existence of a learnable and hard to learn class of input data manifolds illustrated in Figure 2. The *learnable* setting is the class of efficiently sampleable manifolds, which form the basis for algorithms that can provably reconstruct manifolds [9, 28, 70, 42]. As expected, we find that a simple interpolation argument guarantees learnability of neural networks over these manifolds. Below and in Appendix B.3, we comment on instances of data geometries commonly assumed in machine learning and data science applications that place manifolds within this regime.

The *provably hard* setting are manifolds without bounds on volume but with bounds on curvature quantified globally by the reach. When the bound on the reach grows no faster than $o(\sqrt{n})$ (n denoting the input size), we construct input data manifolds that feature curvature no larger than the stated bounds but for which learning neural networks is exponentially hard. Our proofs construct curves that cover exponentially many quadrants of the Boolean cube, which allows us to extend classical hardness results for learning Boolean functions expressible by neural networks.

Within the manifold regimes we study, our results are relatively tight with respect to bounds on the reach, since whenever the reach grows as $\omega(\sqrt{n})$, such manifolds fall within the first class of efficiently sampleable manifolds. We leave as an open question the learnability of manifolds whose reach grows exactly as $\Theta(\sqrt{n})$ (see Appendix C.4 for more details). Looking beyond our setting, real-world data manifolds feature heterogeneous properties that may not conform to the global bounds on curvature, intrinsic dimension, etc. that we set in our study. Better characterizing these heteregeneous features is a first step to extending our learnability results to more realistic settings. We conduct some preliminary empirical analysis in analyzing this heterogeneity in our experiments (Section 4.2).

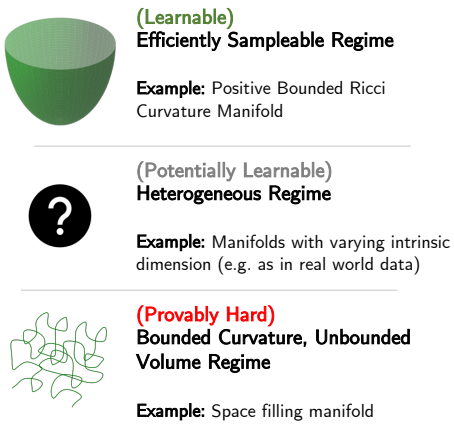


Figure 2: Learnability of neural networks depends on the regularity and smoothness properties of the input data manifold. In the efficiently sampleable regime corresponding to manifolds which can be approximated well with samples, neural networks are learnable via simple interpolation arguments. In the regime where manifolds are bounded solely by their curvature and intrinsic dimension, we show classes of manifolds that obstruct the learnability of algorithms. Real-world data likely lives in an intermediate regime with heterogeneous properties (e.g. manifolds with varying intrinsic dimension; see Section 4.2).

3.1 Sampleable regime

In the manifold reconstruction literature, the goal is to draw samples from a distribution $\mathcal{P}$ (typically supported on a given manifold $\mathcal{M}$) and find a manifold $\mathcal{M}'$ which closely approximates the sample distribution or target manifold in some appropriate error metric such as the Haussdorf distance [89, 9, 2, 43, 28, 70, 42]. Runtime and sample complexity for these algorithms are generally at least linear in the volume of the manifold, polynomial in smoothness parameters such as the reach, and exponential in the intrinsic dimension, though the dependence on the ambient dimension can often be removed [42]. Manifold reconstruction algorithms offer a direct means for learning data from a target neural network f^* as one can apply such algorithms to the graph of the function consisting of points $(\boldsymbol{x}, f^*(\boldsymbol{x})) \in \mathcal{X} \times \mathbb{R}$. As we will show here, guarantees of learning in the manifold reconstruction literature directly imply guarantees for learning neural networks via a simple interpolation argument. However, caution must be taken in assuming this situation holds in practice. For this procedure to be efficient, the volume of the manifold should be at worst polynomial in the intrinsic dimension d and not growing significantly with the ambient dimension n. Some empirical evidence suggests that manifolds of real-world data may not be in this regime [74, 16].

Essential to many of the manifold reconstruction algorithms is the requirement that one can efficiently cover the manifold with samples [28, 43, 9]. This requirement comes in various technical forms and we frame our results here assuming the ability to form an epsilon net with samples.

Definition 3.1 ((ϵ, δ)-net)**.** Given a distribution $\mathcal{D}_{\mathcal{M}}$ over a manifold $\mathcal{M}$, a subset $\mathcal{S} \subset \mathcal{M}$ of points forms an (ϵ, δ)-net if with probability at least $1 - \delta$ over draws $\boldsymbol{x} \sim \mathcal{D}_{\mathcal{M}}$, there exists a point $\boldsymbol{x}' \in \mathcal{S}$ where $\|\boldsymbol{x} - \boldsymbol{x}'\| \leq \epsilon$.

We denote a sequence of manifolds indexed by ambient dimensions as efficiently sampleable if for a fixed intrinsic dimension d, one can draw polynomially many samples in n and error $1/\epsilon$ to form an epsilon net over the manifold $\mathcal{M}$.

Definition 3.2 (Efficiently sampleable manifold)**.** Let $\{\mathcal{M}_n\}$ denote a sequence of manifolds in ambient dimension n with fixed intrinsic dimension $d = O(1)$. Let $\{\mathcal{D}_{\mathcal{M}_n}\}$ be a corresponding sequence of distributions over points on the manifolds. We denote this sequence of manifolds as *efficiently sampleable* if with at most $\tilde{O}(\text{poly}(n, 1/\epsilon))$ samples drawn i.i.d. from $\mathcal{D}_{\mathcal{M}_n}$, one can form an (ϵ, δ)-net of the manifold $\mathcal{M}_n$ for any $\delta = \Omega(\text{poly}(n)^{-1})$.

Note that classical manifold learning algorithms typically implicitly assume efficiently sampleable manifolds as they assume that the algorithms run efficiently or have runtimes that depend polynomially on the manifold's volume $\text{Vol}(\mathcal{M})$ (see Example B.8 for details). With standard regularity assumptions on the manifold, such a volume assumption typically implies that an epsilon net can be formed efficiently with $\tilde{O}(\text{Vol}(\mathcal{M})/\epsilon^d)$ samples by a coupon collector argument (see, e.g., Proposition 3.5).

Remark 3.3. Given a target function f^* with inputs on a manifold $\mathcal{M} \subseteq \mathcal{X}$, we could treat the graph of the function $\mathcal{M}_{f^*} = \{(\boldsymbol{x}, f^*(\boldsymbol{x})) : \boldsymbol{x} \in \mathcal{M}\}$ as a manifold in $\mathcal{X} \times \mathbb{R}$. Then, one can apply manifold reconstruction algorithms directly to the graph $\mathcal{M}_{f^*}$, which lies in an ambient space of

dimension $n+1$. This likely works in practice, though it runs into some technical issues due to discontinuities introduced by ReLU activations. We apply a more direct method here.

Proposition 3.4. *Let n-dimensional inputs be drawn from a sequence of efficiently sampleable manifolds $\mathcal{M}_n$ with intrinsic dimension $d = O(1)$ and distributions $\mathcal{D}_{\mathcal{M}_n}$ over the manifold. Denote $\mathcal{H}_n$ as the function class of constant depth ReLU networks on n inputs with weights bounded in magnitude by $B = O(\text{poly}(n))$ and $O(\text{poly}(n))$ width. Then, one can learn $f^* \in \mathcal{H}_n$ up to error ϵ in runtime and sample complexity $O(\text{poly}(n, B, 1/\epsilon))$.*

Proof. From the efficiently sampleable property of the manifold, we form an (ϵ', δ)-net using $\tilde{O}(\text{poly}(n, 1/\epsilon))$ many samples where for all but a δ-fraction of the manifold any point is within ϵ' of a given point within the net. Any network of L layers and $O(\text{poly}(n))$ bounded width and weight magnitude has the Lipschitz property that [92]

$$\begin{aligned} \|f^*(\boldsymbol{x}) - f^*(\boldsymbol{x}')\| &\leq \|\boldsymbol{x} - \boldsymbol{x}'\| \prod_{\ell=1}^{L} \|W^{(\ell)}\|_2 \\ &\leq \|\boldsymbol{x} - \boldsymbol{x}'\| \prod_{\ell=1}^{L} \|W^{(\ell)}\|_F \\ &\leq O(B' n^L) \|\boldsymbol{x} - \boldsymbol{x}'\|, \end{aligned} \tag{5}$$

where $B' = O(\text{poly}(n, B))$ is a bound on the Frobenius norm of the weight matrices. Therefore, we can interpolate the value of f^* for any $\boldsymbol{x} \in \mathcal{M}_n$ from a sample by setting $\epsilon' = O(\epsilon B'^{-1} n^{-L})$. For the δ-fraction that is not in this net, note that $|f(\boldsymbol{x})| \leq O(\text{poly}(n))$ for all $\boldsymbol{x}$ in the hypercube so setting $\delta = o(|f(\boldsymbol{x})|^{-1})$ will guarantee that the contribution from these points decays. Finally, since $L = O(1)$ and $B = O(\text{poly}(n))$ by assumption, we have the resulting polynomial complexity. □

Restrictions on the curvature or convexity properties of a manifold can often render the manifold efficiently sampleable. To illustrate that many manifolds implicitly fall within the regime of efficiently sampleable manifolds, we give an example below. Additional examples can be found in Appendix B.3.

Proposition 3.5 (Bounded Ricci curvature (Isoperimetric setting, see [52, 17] for motivation))**.** *The set of distributions $\mathcal{P}_{\mathcal{M}}$ over complete manifolds with bounded Ricci curvature* $\text{Ric}(\mathcal{M}) \geq (d-1)K$ *for a constant $K > 0$[5] are efficiently sampleable.*

Proof. The bound on the Ricci curvature guarantees that the diameter of the manifold is at most $2\pi/\sqrt{K}$ and the manifold is contained within a d-dimensional ball of radius $1/\sqrt{K}$ [69] (see also Theorem 6.3.3 of [75]). Thus, we can form an (ϵ, δ)-net over the manifold $\mathcal{M}$ by inducing it from an ϵ-cover on the ball of radius $1/\sqrt{K}$ (see Definition B.1 for definition). To achieve an ϵ-cover of such a ball in d dimensions, it suffices to have $N_\epsilon = O(1/\epsilon^d)$ points (see, e.g., Lemma 5.2 of [91]). We denote the balls in the cover as $b_1, \ldots, b_{N_\epsilon}$. Let τ denote a sampling ratio where we will take $\tau^{-2} = O(\text{poly}(n))$ samples to form the (ϵ, δ)-net. By a coupon-collector argument (see Lemma B.5), τ^{-2} samples suffices to guarantee with high probability that any ball b_i with probability at least $\mathcal{P}_{\mathcal{M}}(b_i) \geq \tau$ has a sample within it. δ then is at most the probability that a randomly drawn sample falls within a ball b_i with probability $\mathcal{P}_{\mathcal{M}}(b_i) < \tau$. Therefore,

$$\delta \leq \sum_{i : \mathcal{P}_{\mathcal{M}}(b_i) < \tau} \mathcal{P}_{\mathcal{M}}(b_i) \leq N_\epsilon \tau. \tag{6}$$

Thus, setting $\tau = \delta / N_\epsilon$ suffices to achieve $\delta = \Omega(\text{poly}(n)^{-1})$.

□

We remark that the efficiently sampleable setting does not necessarily require that the manifold be fully connected. In fact, it is straightforward to extend proofs of learnability such as in Proposition 3.5 to settings where there are multiple such disconnected manifolds as long as the number of disconnected components does not grow superpolynomially in n.

[5] Here, the factor $(n-1)$ accounts for the standard scaling of the Ricci curvature for a sphere of radius $1/\sqrt{K}$ which has Ricci curvature $(n-1)K$.

3.2 Hard regime with bounded curvature

The learnability of the networks in the manifold learning setting relied crucially on the fact that such settings implicitly place bounds on the volume of such manifolds. Here, we lift that restriction and consider a setting where manifolds are smooth and bounded only in their curvature and intrinsic dimension. More formally, we consider studying inputs drawn from manifolds with bounded reach (see Definition 2.2).

In this section, we construct a sequence of manifolds $\{\mathcal{M}_n\}$ with sufficiently bounded reach that are exponentially hard to learn in the SQ model. These manifolds resemble space filling curves which wrap around exponentially many quadrants of the Boolean cube (see Appendix C.1). Given a bound on the reach $\mathrm{Rch}\left(\mathcal{M}_n\}\right) = O(n^\alpha)$ for $\alpha < 0.5$, the space filling curve wraps around $2^{n^{1-2\alpha}} = 2^{n^{\Omega(1)}}$ quadrants allowing one to extend standard hardness results for learning Boolean functions to data supported on this manifold.

Theorem 3.6. *Let $\mathcal{D}_{\mathcal{M}_n}$ denote the uniform distribution over manifolds $\mathcal{M}_n$ constructed in Definition C.4 where $\mathrm{Rch}\left(\mathcal{M}_n\right) = O(n^\alpha)$ for $\alpha < 0.5$. Any SQ algorithm $\mathcal{A}$ capable of learning the class of linear width single hidden layer ReLU neural networks under this sequence of distributions up to mean squared error sufficiently small ($\epsilon/8$ suffices) with queries of tolerance τ must use at least $\Omega(\tau^2 2^{n^{\Omega(1)}})$ queries.*

Proof sketch. We first form manifolds conforming to the stated bounds on the reach and intrinsic dimension while also "looping" around exponentially many quadrants of the hypercube. This is done by forming space-filling curves whose paths follow the indexing of a Gray code [51]. Inputs on this manifold are real-valued, but by carefully selecting a portion of the input dimensions, we obtain inputs that with high probability are approximately distributed as uniform over Boolean inputs $\{0,1\}^{n_b}$. We then extend previous proofs of the hardness of learning networks approximating parity functions under i.i.d. Boolean inputs proven in [32, 26] to our setting. See Appendix C for complete proofs. □

Remark 3.7. In Theorem C.12, we show an equivalent proof of hardness in the cryptographic setting following the techniques in [32]. Namely, in Appendix C.3, we show that the class of functions in Theorem 3.6 is also hard to learn conditional on cryptographic assumptions related to the hardness of learning a class of pseudorandom functions.

The above result is relatively tight with respect to the bound on the reach. Namely, whenever the reach $\mathrm{Rch}\left(\mathcal{M}_n\right) = \omega(n^\alpha)$ is growing faster than $\sqrt{n}$, one can show that the volume of such manifolds is at most $O(\mathrm{poly}(n))$ and fitting within the regime of efficiently sampleable manifolds studied in the previous subsection.

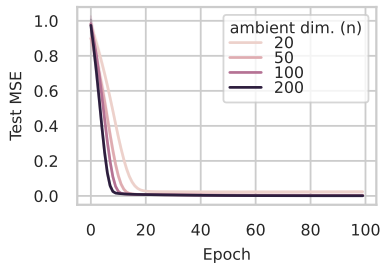
Proposition 3.8. *Given a sequence of manifolds $\{\mathcal{M}_n\}$ of intrinsic dimension d (fixed and independent of n) and reach bounded by $\mathrm{Rch}\left(\mathcal{M}_n\right) = \omega(n^{0.5})$, the volume of the manifolds grows at most $\mathrm{Vol}_d(\mathcal{M}_n) = O(\mathrm{poly}(n))$.*

We refer the reader to Appendix C.4 for further details and formal proofs of this tightness.

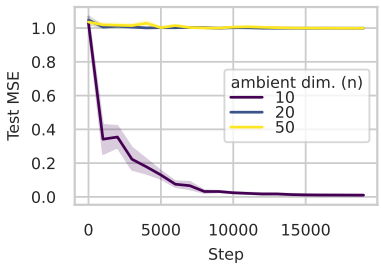
4 Experiments

4.1 Empirical verification of main findings

To verify that neural networks are learnable in the sampleable regime (Section 3.1) and hard to learn in the bounded curvature regime (Section 3.2), we train neural networks over input data manifolds taken from these regimes to confirm the formal theoretical results. We consider target networks, which are single hidden layer neural networks of $O(n)$ width and train overparameterized neural networks of larger width. For the sampleable regime, we draw inputs from a d-dimensional hypersphere supported over d orthogonal dimensions in the n-dimensional ambient space. This is an instance of a complete manifold of bounded curvature as in Proposition 3.5. For the hard to learn regime, we draw inputs from the 1-dimensional manifold constructed in Appendix C.1 with reach $R = 0.5$. Target functions in this hard regime correspond to the parity functions described in our proofs in Appendix C.



(a) Learnable Setting



(b) Hard to Learn Setting

Figure 3: (a) Learning is successful when inputs are drawn from a $d = 10$ intrinsic dimensional hypersphere living in ambient space of dimension n – an instance of the bounded positive curvature model in Proposition 3.5. Target functions are single hidden layer networks taken from the class of hard to learn functions in the Gaussian i.i.d. input model [36], which are no longer hard to learn in the input distribution considered here. (b) When the ambient dimension is large, learning algorithm struggles to learn a single hidden layer neural network drawn from the class of functions in the setting of Theorem 3.6 where the input data manifold has intrinsic dimension $d = 1$ and reach $R = 0.5$. The network trained to learn this target function is over-parameterized with respect to the target. Data is aggregated over five random realizations.

The results in Figure 3 empirically confirm our main findings. When inputs are drawn from the d-dimensional hypersphere (Figure 3a), the trained neural network achieves low test error with a fixed training set of size 1000 for all n. Here, target functions are drawn from those in [36] who provided a class of hard to learn functions in the i.i.d. Gaussian input model. Once the input model is changed to the d-dimensional hypersphere, this class of functions is no longer hard to learn (see Appendix E for similar results over random targets).

In contrast, when inputs are drawn from the hard to learn manifold (Figure 3b), learning is only possible when the ambient dimension is small. Here, target functions are randomly chosen parity functions over the inputs (see Appendix E). At each step of training, we provide the algorithm with a fresh batch of i.i.d. samples. In Figure 8, we replicate these results when training networks overparameterized in depth (i.e. three hidden layers as opposed to one) as well. We refer the reader to Appendix E for further details.

4.2 Empirical study of geometry of data manifolds

We empirically investigate the intrinsic dimension of real-world data manifolds as a first step towards testing the manifold regularity assumptions of our framework on real data. Largely, our results corroborate findings in other works highlighting the heterogeneous nature of real-world data manifolds [77, 87, 16].

Experimental Setup. We estimate intrinsic dimension using samples generated by a diffusion model, closely following the approach of [87]. The use of a diffusion model allows us to generate arbitrarily dense samples in a neighborhood of a given point from the data manifold. Given these samples, we perform PCA on the collection of score vectors $\{\boldsymbol{s}_i\}$ of the diffusion model at each of these samples, which point towards the direction of de-noising and hence towards the manifold itself. Estimating the rank of such a collection of score vectors recovers an estimate of the normal and intrinsic dimensions. We defer a detailed description of our approach, its implementation, and hyperparameter choices to Appendix E.1.

Data set	**#0**	**#1**	**#2**	**#3**	**#4**	**#5**	**#6**	**#7**	**#8**	**#9**
MNIST (28 x 28)	102	66	120	109	73	101	109	72	114	104
KMNIST (28 x 28)	180	199	134	169	135	128	149	191	205	199
FMNIST (28 x 28)	429	177	596	275	225	233	312	125	201	418

Table 1: Estimated intrinsic dimension shown for each of the ten classes in MNIST, KMNIST, FMNIST.

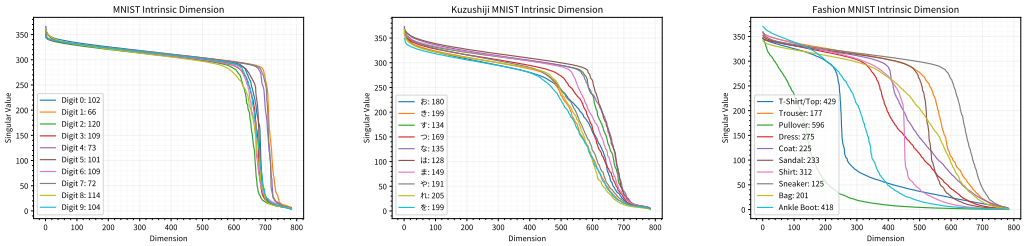


Figure 4: Sample spectrum of the singular values of a collection of score vectors given by a trained diffusion model pointing towards the direction of de-noising an image.

Results. Estimated intrinsic dimensions for three image data manifolds are given in Table 3. Score spectra for the three manifolds are shown in Figure 4 confirming that the score vectors do indeed cluster in a linear subspace spanning the normal dimensions. Furthermore, our approach gives a similar estimate for the intrinsic dimension of MNIST as reported in [87]. To the best of our knowledge, FMNIST and KMNIST were not investigated in prior work. Comparing estimated intrinsic dimension across classes indicates significant heterogeneity of the real data manifolds, which would place them in the *potentially learnable* regime in our proposed framework. This may seem surprising as images drawn from different classes can have different properties and symmetries. However, it corroborates our initial hypothesis that real-world data may not conform to global bounds on intrinsic dimension. Furthermore, we find that upon re-sizing the images to a smaller scale, the ambient dimension decreases faster in comparison to the intrinsic dimension for some datasets (see Table 3 in Appendix) lending some limited evidence to the underlying assumption in the manifold hypothesis that bounds on intrinsic dimension do not grow in tandem with ambient dimension. Additional experimental results and further discussion can be found in Appendix E.3.

5 Discussion

In this paper we have investigated whether regularity assumptions on the data geometry can render feedforward neural networks efficiently learnable. We show that bounding curvature in low-dimensional data manifolds, a common assumption in high-dimensional learning, does not suffice to alleviate the fundamental hardness of learning such architectures. However, we establish learning guarantees under geometric assumptions common in the manifold reconstruction literature that allow manifolds to be efficiently covered with $O(\mathrm{poly}(n))$ samples. Our results contribute to a recent body of literature that seeks to establish learnability guarantees and provable hardness results for neural network architectures [12, 26, 36, 57, 48, 90].

While our results establish guarantees for a wide range of data manifolds, including manifolds that can be provably reconstructed via manifold learning, there are geometric regimes in which the question of learnability remains open. We hypothesize that the heterogeneity of real data manifolds might place them in this more uncertain regime and find some evidence of this heterogeneity in our experiments. A further theoretical and empirical investigation of this class of data manifolds is an important direction for future work.

The experimental results on estimating the intrinsic dimension of data manifolds are merely a starting point for understanding realistic assumption on data geometry in machine learning architectures. In future work we hope to look at a wider range of synthetic and real data sets, as well as a wider range of geometric characteristics and estimation techniques.

Broadly, our results indicate that assumptions beyond global smoothness of data manifolds are needed to guarantee learnability of neural networks in real-world settings. To circumvent hardness results, the particular form of the networks and their weights can be changed to for example incorporate symmetries via equivariant architectures or place restrictions on weights such as bounds on condition number or rank [27, 7]. Furthermore, some datasets live in a discrete input space (e.g., language data) which does not conform directly to Riemannian analysis. In these settings, adaptations of the manifold hypothesis to analysis via graphs and other combinatorial objects presents an interesting direction for future work [81, 15].

Acknowledgement

The authors thank Andrew Cheng and Adityanarayanan Radhakrishnan for insightful discussions. BTK and MW were supported by the Harvard Data Science Initiative Competitive Research Fund and NSF award 2112085. JW acknowledges support from the Harvard College Research Program (HCRP).

References

[1] E. Aamari, J. Kim, F. Chazal, B. Michel, A. Rinaldo, and L. Wasserman. Estimating the reach of a manifold. 2019.

[2] E. Aamari and A. Knop. Statistical query complexity of manifold estimation. In *Proceedings of the 53rd Annual ACM SIGACT Symposium on Theory of Computing*, pages 116–122, 2021.

[3] E. Aamari and A. Knop. Adversarial manifold estimation. *Foundations of Computational Mathematics*, pages 1–97, 2022.

[4] E. Aamari and C. Levrard. Stability and minimax optimality of tangential delaunay complexes for manifold reconstruction. *Discrete & Computational Geometry*, 59:923–971, 2018.

[5] E. Abbe, E. B. Adsera, and T. Misiakiewicz. Sgd learning on neural networks: leap complexity and saddle-to-saddle dynamics. In *The Thirty Sixth Annual Conference on Learning Theory*, pages 2552–2623. PMLR, 2023.

[6] Y. Bahri, E. Dyer, J. Kaplan, J. Lee, and U. Sharma. Explaining neural scaling laws. *arXiv preprint arXiv:2102.06701*, 2021.

[7] A. Bakshi, R. Jayaram, and D. P. Woodruff. Learning two layer rectified neural networks in polynomial time. In *Conference on Learning Theory*, pages 195–268. PMLR, 2019.

[8] M. Belkin and P. Niyogi. Laplacian eigenmaps for dimensionality reduction and data representation. *Neural computation*, 15(6):1373–1396, 2003.

[9] M. Bernstein, V. De Silva, J. C. Langford, and J. B. Tenenbaum. Graph approximations to geodesics on embedded manifolds. Technical report, Citeseer, 2000.

[10] R. L. Bishop. A relation between volume, mean curvature and diameter. In *Euclidean Quantum Gravity*, pages 161–161. World Scientific, 1964.

[11] A. Blum, M. Furst, J. Jackson, M. Kearns, Y. Mansour, and S. Rudich. Weakly learning dnf and characterizing statistical query learning using fourier analysis. In *Proceedings of the twenty-sixth annual ACM symposium on Theory of computing*, pages 253–262, 1994.

[12] A. Blum and R. Rivest. Training a 3-node neural network is np-complete. *Advances in neural information processing systems*, 1, 1988.

[13] A. Bogdanov and A. Rosen. Pseudorandom functions: Three decades later. In *Tutorials on the Foundations of Cryptography: Dedicated to Oded Goldreich*, pages 79–158. Springer, 2017.

[14] J.-D. Boissonnat and A. Ghosh. Manifold reconstruction using tangential delaunay complexes. In *Proceedings of the twenty-sixth annual symposium on Computational geometry*, pages 324–333, 2010.

[15] M. M. Bronstein, J. Bruna, Y. LeCun, A. Szlam, and P. Vandergheynst. Geometric deep learning: going beyond euclidean data. *IEEE Signal Processing Magazine*, 34(4):18–42, 2017.

[16] B. C. Brown, A. L. Caterini, B. L. Ross, J. C. Cresswell, and G. Loaiza-Ganem. Verifying the union of manifolds hypothesis for image data. In *The Eleventh International Conference on Learning Representations*, 2022.

[17] S. Bubeck and M. Sellke. A universal law of robustness via isoperimetry. *Journal of the ACM*, 70(2):1–18, 2023.

[18] S. Buchanan, D. Gilboa, and J. Wright. Deep networks and the multiple manifold problem. *arXiv preprint arXiv:2008.11245*, 2020.

[19] F. Camastra and A. Vinciarelli. Estimating the intrinsic dimension of data with a fractal-based method. *IEEE Transactions on pattern analysis and machine intelligence*, 24(10):1404–1407, 2002.

[20] G. Carlsson. Topology and data. *Bulletin of the American Mathematical Society*, 46(2):255–308, 2009.

[21] L. Cayton et al. *Algorithms for manifold learning*. eScholarship, University of California, 2008.

[22] M. Chen, K. Huang, T. Zhao, and M. Wang. Score approximation, estimation and distribution recovery of diffusion models on low-dimensional data. *arXiv preprint arXiv:2302.07194*, 2023.

[23] R. T. Chen and Y. Lipman. Riemannian flow matching on general geometries. *arXiv preprint arXiv:2302.03660*, 2023.

[24] S. Chen, S. Chewi, J. Li, Y. Li, A. Salim, and A. Zhang. Sampling is as easy as learning the score: theory for diffusion models with minimal data assumptions. In *The Eleventh International Conference on Learning Representations*, 2023.

[25] S. Chen, Z. Dou, S. Goel, A. R. Klivans, and R. Meka. Learning narrow one-hidden-layer relu networks. *arXiv preprint arXiv:2304.10524*, 2023.

[26] S. Chen, A. Gollakota, A. Klivans, and R. Meka. Hardness of noise-free learning for two-hidden-layer neural networks. *Advances in Neural Information Processing Systems*, 35:10709–10724, 2022.

[27] S. Chen and S. Narayanan. A faster and simpler algorithm for learning shallow networks. *arXiv preprint arXiv:2307.12496*, 2023.

[28] S.-W. Cheng, T. K. Dey, and E. A. Ramos. Manifold reconstruction from point samples. In *SODA*, volume 5, pages 1018–1027, 2005.

[29] T. Clanuwat, M. Bober-Irizar, A. Kitamoto, A. Lamb, K. Yamamoto, and D. Ha. Deep learning for classical japanese literature, 2018.

[30] T. Cohen and M. Welling. Group equivariant convolutional networks. In *ICML*, 2016.

[31] R. R. Coifman and S. Lafon. Diffusion maps. *Applied and Computational Harmonic Analysis*, 21(1):5–30, 2006. Special Issue: Diffusion Maps and Wavelets.

[32] A. Daniely and G. Vardi. From local pseudorandom generators to hardness of learning. In *Conference on Learning Theory*, pages 1358–1394. PMLR, 2021.

[33] S. Dasgupta and Y. Freund. Random projection trees and low dimensional manifolds. In *Proceedings of the fortieth annual ACM symposium on Theory of computing*, pages 537–546, 2008.

[34] V. De Bortoli. Convergence of denoising diffusion models under the manifold hypothesis. *arXiv preprint arXiv:2208.05314*, 2022.

[35] L. Deng. The mnist database of handwritten digit images for machine learning research. *IEEE Signal Processing Magazine*, 29(6):141–142, 2012.

[36] I. Diakonikolas, D. M. Kane, V. Kontonis, and N. Zarifis. Algorithms and sq lower bounds for pac learning one-hidden-layer relu networks. In *Conference on Learning Theory*, pages 1514–1539. PMLR, 2020.

[37] J. J. DiCarlo, D. Zoccolan, and N. C. Rust. How does the brain solve visual object recognition? *Neuron*, 73(3):415–434, 2012.

[38] V. Divol. Minimax adaptive estimation in manifold inference. *Electronic Journal of Statistics*, 15(2):5888–5932, 2021.

[39] P. Erdős and A. Rényi. On a classical problem of probability theory. *Magyar Tud. Akad. Mat. Kutató Int. Közl*, 6(1):215–220, 1961.

[40] E. Facco, M. d'Errico, A. Rodriguez, and A. Laio. Estimating the intrinsic dimension of datasets by a minimal neighborhood information. *Scientific reports*, 7(1):12140, 2017.

[41] H. Federer. Curvature measures. *Transactions of the American Mathematical Society*, 93(3):418–491, 1959.

[42] C. Fefferman, S. Ivanov, Y. Kurylev, M. Lassas, and H. Narayanan. Fitting a putative manifold to noisy data. In *Conference On Learning Theory*, pages 688–720. PMLR, 2018.

[43] C. Fefferman, S. Mitter, and H. Narayanan. Testing the manifold hypothesis. *Journal of the American Mathematical Society*, 29(4):983–1049, 2016.

[44] K. Fukunaga and D. R. Olsen. An algorithm for finding intrinsic dimensionality of data. *IEEE Transactions on computers*, 100(2):176–183, 1971.

[45] O. Ganea, G. Bécigneul, and T. Hofmann. Hyperbolic neural networks. *Advances in neural information processing systems*, 31, 2018.

[46] N. Garcia Trillos and M. Weber. Continuum limits of Ollivier's Ricci curvature on data clouds: pointwise consistency and global lower bounds. *arXiv preprint arXiv:2307.02378*, 2023.

[47] C. R. Genovese, M. Perone Pacifico, V. Isabella, L. Wasserman, et al. Minimax manifold estimation. *Journal of machine learning research*, 13:1263–1291, 2012.

[48] S. Goel, A. Gollakota, Z. Jin, S. Karmalkar, and A. Klivans. Superpolynomial lower bounds for learning one-layer neural networks using gradient descent. In *International Conference on Machine Learning*, pages 3587–3596. PMLR, 2020.

[49] O. Goldreich. Candidate one-way functions based on expander graphs. *Studies in Complexity and Cryptography. Miscellanea on the Interplay between Randomness and Computation: In Collaboration with Lidor Avigad, Mihir Bellare, Zvika Brakerski, Shafi Goldwasser, Shai Halevi, Tali Kaufman, Leonid Levin, Noam Nisan, Dana Ron, Madhu Sudan, Luca Trevisan, Salil Vadhan, Avi Wigderson, David Zuckerman*, pages 76–87, 2011.

[50] V. V. Goncharov and G. E. Ivanov. Strong and weak convexity of closed sets in a hilbert space. *Operations Research, Engineering, and Cyber Security: Trends in Applied Mathematics and Technology*, pages 259–297, 2017.

[51] F. Gray. Pulse code communication. *United States Patent Number 2632058*, 1953.

[52] M. Gromov. Isoperimetric inequalities in riemannian manifolds. *Asymptotic Theory of Finite Dimensional Spaces*, 1200:114–129, 1986.

[53] J. S. Judd. Learning in networks is hard. In *Proc. of 1st International Conference on Neural Networks, San Diego, California, June 1987*. IEEE, 1987.

[54] N. Kambhatla and T. Leen. Fast non-linear dimension reduction. *Advances in neural information processing systems*, 6, 1993.

[55] M. Kearns. Efficient noise-tolerant learning from statistical queries. *Journal of the ACM (JACM)*, 45(6):983–1006, 1998.

[56] B. Kégl. Intrinsic dimension estimation using packing numbers. *Advances in neural information processing systems*, 15, 2002.

[57] B. Kiani, T. Le, H. Lawrence, S. Jegelka, and M. Weber. On the hardness of learning under symmetries. In *The Twelfth International Conference on Learning Representations*, 2024.

[58] A. K. Kim and H. H. Zhou. Tight minimax rates for manifold estimation under hausdorff loss. 2015.

[59] T. N. Kipf and M. Welling. Semi-supervised classification with graph convolutional networks. *arXiv preprint arXiv:1609.02907*, 2016.

[60] D. E. Knuth. *The art of computer programming*, volume 3. Pearson Education, 1997.

[61] M. Kossaczká and J. Vybíral. Entropy numbers of finite-dimensional embeddings. *Expositiones Mathematicae*, 38(3):319–336, 2020.

[62] J. B. Kruskal. Multidimensional scaling by optimizing goodness of fit to a nonmetric hypothesis. *Psychometrika*, 29(1):1–27, 1964.

[63] E. Levina and P. Bickel. Maximum likelihood estimation of intrinsic dimension. *Advances in neural information processing systems*, 17, 2004.

[64] H. Liu, M. Chen, T. Zhao, and W. Liao. Besov function approximation and binary classification on low-dimensional manifolds using convolutional residual networks. In *International Conference on Machine Learning*, pages 6770–6780. PMLR, 2021.

[65] Y. Ma and Y. Fu. *Manifold learning theory and applications*. CRC press, 2011.

[66] A. Maloney, D. A. Roberts, and J. Sully. A solvable model of neural scaling laws. *arXiv preprint arXiv:2210.16859*, 2022.

[67] T. Minka. Automatic choice of dimensionality for pca. *Advances in neural information processing systems*, 13, 2000.

[68] M. Mohri, A. Rostamizadeh, and A. Talwalkar. *Foundations of machine learning*. MIT press, 2018.

[69] S. B. Myers. Riemannian manifolds with positive mean curvature. 1941.

[70] H. Narayanan and S. Mitter. Sample complexity of testing the manifold hypothesis. *Advances in neural information processing systems*, 23, 2010.

[71] H. Narayanan and P. Niyogi. On the sample complexity of learning smooth cuts on a manifold. In *COLT*, 2009.

[72] A. Q. Nichol and P. Dhariwal. Improved denoising diffusion probabilistic models. In *International Conference on Machine Learning*, pages 8162–8171. PMLR, 2021.

[73] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga, A. Desmaison, A. Kopf, E. Yang, Z. DeVito, M. Raison, A. Tejani, S. Chilamkurthy, B. Steiner, L. Fang, J. Bai, and S. Chintala. Pytorch: An imperative style, high-performance deep learning library. In *Advances in Neural Information Processing Systems 32*, pages 8024–8035. 2019.

[74] W. Peng, T. Varanka, A. Mostafa, H. Shi, and G. Zhao. Hyperbolic deep neural networks: A survey. *IEEE Transactions on pattern analysis and machine intelligence*, 44(12):10023–10044, 2021.

[75] P. Petersen. *Riemannian geometry*, volume 171. Springer, 2006.

[76] J. Pidstrigach. Score-based generative models detect manifolds. *Advances in Neural Information Processing Systems*, 35:35852–35865, 2022.

[77] P. Pope, C. Zhu, A. Abdelkader, M. Goldblum, and T. Goldstein. The intrinsic dimension of images and its impact on learning. *arXiv preprint arXiv:2104.08894*, 2021.

[78] L. Reyzin. Statistical queries and statistical algorithms: Foundations and applications. *arXiv preprint arXiv:2004.00557*, 2020.

[79] O. Ronneberger, P. Fischer, and T. Brox. U-net: Convolutional networks for biomedical image segmentation. In *Medical Image Computing and Computer-Assisted Intervention–MICCAI 2015: 18th International Conference, Munich, Germany, October 5-9, 2015, Proceedings, Part III 18*, pages 234–241. Springer, 2015.

[80] S. T. Roweis and L. K. Saul. Nonlinear dimensionality reduction by locally linear embedding. *science*, 290(5500):2323–2326, 2000.

[81] P. Rubin-Delanchy. Manifold structure in graph embeddings. *Advances in neural information processing systems*, 33:11687–11699, 2020.

[82] D. Scieur, T. Kerdreux, A. d'Aspremont, S. Pokutta, et al. Strong convexity of sets in riemannian manifolds. *arXiv e-prints*, pages arXiv–2312, 2023.

[83] U. Sharma and J. Kaplan. A neural scaling law from the dimension of the data manifold. *arXiv preprint arXiv:2004.10802*, 2020.

[84] J. Skilling. Programming the hilbert curve. In *AIP Conference Proceedings*, volume 707, pages 381–387. American Institute of Physics, 2004.

[85] L. Song, S. Vempala, J. Wilmes, and B. Xie. On the complexity of learning neural networks. *Advances in neural information processing systems*, 30, 2017.

[86] D. Sritharan, S. Wang, and S. Hormoz. Computing the riemannian curvature of image patch and single-cell rna sequencing data manifolds using extrinsic differential geometry. *Proceedings of the National Academy of Sciences*, 118(29):e2100473118, 2021.

[87] J. Stanczuk, G. Batzolis, T. Deveney, and C.-B. Schönlieb. Your diffusion model secretly knows the dimension of the data manifold. *arXiv preprint arXiv:2212.12611*, 2022.

[88] B. Tahmasebi and S. Jegelka. The exact sample complexity gain from invariances for kernel regression, 2023.

[89] J. B. Tenenbaum, V. d. Silva, and J. C. Langford. A global geometric framework for nonlinear dimensionality reduction. *science*, 290(5500):2319–2323, 2000.

[90] S. Vempala and J. Wilmes. Gradient descent for one-hidden-layer neural networks: Polynomial convergence and sq lower bounds. In *Conference on Learning Theory*, pages 3115–3117. PMLR, 2019.

[91] R. Vershynin. Introduction to the non-asymptotic analysis of random matrices. *arXiv preprint arXiv:1011.3027*, 2010.

[92] A. Virmaux and K. Scaman. Lipschitz regularity of deep neural networks: analysis and efficient estimation. *Advances in Neural Information Processing Systems*, 31, 2018.

[93] M. Weber, M. Zaheer, A. S. Rawat, A. Menon, and S. Kumar. Robust large-margin learning in hyperbolic space. In *Advances in Neural Information Processing Systems 34*, 2020.

[94] N. Whiteley, A. Gray, and P. Rubin-Delanchy. Statistical exploration of the manifold hypothesis, 2023.

[95] H. Xiao, K. Rasul, and R. Vollgraf. Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms. *arXiv preprint arXiv:1708.07747*, 2017.

[96] M. Zaheer, S. Kottur, S. Ravanbakhsh, B. Poczos, R. R. Salakhutdinov, and A. J. Smola. Deep sets. *Advances in neural information processing systems*, 30, 2017.

A Extended Related Works

Hardness of learning neural networks. The first results showing the hardness of learning neural networks are those of [53, 12] proving that proper learning of neural networks is an NP complete task. As mentioned earlier, a number of works prove hardness results for learning feedforward ReLU networks under the Gaussian i.i.d. data model [48, 36, 85, 26, 32]. Many of these works prove hardness results in the statistical query model [78, 55]. Similar hardness results, also in the SQ model, have also been shown for classes of symmetric or equivariant neural networks [57]. These results show that additional assumptions on the network class are needed to prove learnability results. Such assumptions which lead to provable learnability include those on the condition number or positivity of weights [7, 36], polynomial approximation guarantees [90], or bounds on width [27, 25]. When the target is not necessarily a feedforward network, [5] show that single hidden layer ReLU networks can efficiently learn functions with low so-called leap complexity categorized by the growth in the degrees of polynomials over a linear subspace for i.i.d. Gaussian or uniform Boolean inputs.

Intrinsic dimension estimation. There are many algorithms, dating back decades, which estimate the intrinsic dimension of a given set of data points including approaches based on PCA [44, 67], nearest neighbor methods [63], and others [19, 56]. In the context of modern large image datasets such as Imagenet, [77] perform estimates of the intrinsic dimension which are noticeably smaller than the input dimension. Various works study scaling laws arguing that the intrinsic dimension may be a useful indicator of the complexity of a dataset correlating to the rate at which neural network performance scale with increased data or training [83, 6]. These works estimate the intrinsic dimension via a maximum likelihood estimator calculating the intrinsic dimension in relation to the distances of a point to its nearest neighbors [63, 40]. In later work which estimates the intrinsic dimension using diffusion models, [87] empirically show that this MLE estimator can underestimate the intrinsic dimension for various toy datasets where the intrinsic dimension is known. For this reason, our experiments follow the approach in [87].

Learning algorithms over data manifolds. Among the earliest algorithms for learning over data manifolds are those for dimensionality reduction and representation learning [44, 8, 54, 33, 80]. As mentioned earlier, a number of works provide algorithms and statistical analysis for manifold reconstruction or learning where the goal is to construct a manifold that closely fits a target [21, 3, 42]. Perhaps closest to our work are studies of the hardness of learning functions defined over input manifolds. [71] categorize the sample complexity for binary classification over smooth cuts on a data manifold where smoothness is defined by the condition number of the classification boundary of the manifold (a quantity closely related to the reach). When no restrictions are placed on the probability distribution of data on the manifold, they show that the VC dimension is unbounded when this condition number exceeds the reach of the manifold. For nicer distributions where there is an upper bound on the density of the distribution on the manifold, a covering number argument shows that the sample complexity is dependent on the intrinsic dimension, reach, and condition number but independent of the ambient dimension. [64] show that for a certain class of convolutional networks trained on data lying in a low-dimensional manifold, the sample complexity depends weakly on the ambient dimension. [88] study the sample complexity of learning a target dataset over inputs taken from manifolds in kernel settings where the target function is invariant over some group operation acting on the input data manifold. The goal of their work was to quantify the gains in sample complexity from enforcing invariance and their sample complexity bounds depend on the volume of the manifold and dimension of the quotient space.

In recent years, various works have empirically tested and analyzed various aspects and implications of the manifold hypothesis. [16] argue that manifolds of data may have many connected components of potentially varying intrinsic dimension and provide some empirical estimates of intrinsic dimension by image class supporting this argument. Our empirical analysis of the intrinsic dimension also supports this point. [18] consider the task of learning a binary classification task where the two classes are two separate disconnected one dimensional manifolds (curves) on the sphere. Under regularity conditions on the extrinsic curvature and Riemannian distance properties of the curves, they prove that sufficiently wide and deep feedforward networks can learn to separate the manifolds. Their proofs are based on a neural tangent kernel approach. [66] theoretically and empirically study kernel models trained under settings where the latent dimension of the data can be varied. They

show that the rate of decay in the spectrum of the kernel is more strongly related to the scaling of the loss as opposed to the intrinsic dimension of the data with some discussion of potential connections between the two measures. [94] propose a statistical model of real-world data where low dimensional features and manifold structures can naturally emerge. [71] categorize the sample complexity for binary classification over smooth cuts on a data manifold where smoothness is defined by the condition number of the classification boundary of the manifold (a quantity closely related to reach). When no restrictions are placed on the probability distribution of data on the manifold, they show that the VC dimension is unbounded when this condition number exceeds the reach of the manifold. For nicer distributions where there is an upper bound on the density of the distribution on the manifold, a covering number argument shows that the sample complexity is dependent on the intrinsic dimension, reach, and condition number but independent of the ambient dimension.

Generative modeling. From the perspective of generative modeling, various works have recently considered the task of sampling from low-dimensional manifolds embedded in high dimensional space. [76] consider distributions supported over a data manifold of low intrinsic dimension. They provide sufficient conditions for score-based generative models to identify the support of the learned distribution matching that of the target manifold. [34] prove convergence results in Wasserstein distance between score-based models and target distributions which are supported on a manifold under the assumption that the score estimator is accurate in the L^∞ norm. These results were subsequently improved in [24] which only required a score estimator is accurate in the L^2 norm, among other improvements. For data supported on a low-dimensional linear subspace, [22] provide an end-to-end sampling guarantee avoiding the curse of dimensionality and showing that the score function can be efficiently estimated. The resulting sampling algorithm in their work thus approaches the true distribution in total variation distance and recovers the linear subspace appropriately.

Geometric Machine Learning. We briefly mention a related body of work, which studies machine learning algorithms, which directly leverage geometric structure in data. Such methods have shown empirical promise in a variety of domains [15]. Examples of such architectures are translation- and rotation-equivariant neural networks [30], graph neural networks [59] and DeepSets for permutation-invariant inputs [96], as well as hyperbolic machine learning algorithms [45, 93], which assume that data lies on a manifold of constant negative curvature.

B Deferred Proofs

B.1 Auxiliary statements

For completeness, we recall several classical notions from metric geometry.

Definition B.1 (Packing and covering numbers)**.** Given a set $S \subseteq \mathbb{R}^n$, the covering number $\mathrm{cv}_S(\epsilon)$ is the minimum number of balls of size ϵ needed to cover every point in S:

$$\mathrm{cv}_S(\epsilon) \coloneqq \min\left\{k > 0 \mid \exists\, \boldsymbol{x}_1, \ldots, \boldsymbol{x}_k \in S \text{ such that } \forall \boldsymbol{x} \in S, \exists\, i \in [k] \text{ such that } \|\boldsymbol{x}_i - \boldsymbol{x}\| \leq \epsilon\right\}. \tag{7}$$

The packing number $\mathrm{pk}_S(\epsilon)$ is the maximum number of disjoint ϵ-balls that can be contained in S:

$$\mathrm{pk}_S(\epsilon) \coloneqq \max\left\{k > 0 \mid \exists\, \boldsymbol{x}_1, \ldots, \boldsymbol{x}_k \in S \text{ such that } \forall i \neq j : \|\boldsymbol{x}_i - \boldsymbol{x}_j\| > 2\epsilon\right\}. \tag{8}$$

Lemma B.2 (Packing and covering duality)**.** *For any compact set $S \subset \mathbb{R}^n$ and $\epsilon > 0$:*

$$\mathrm{pk}_S(2\epsilon) \leq \mathrm{cv}_S(2\epsilon) \leq \mathrm{pk}_S(\epsilon). \tag{9}$$

Proof. For the first inequality, assume by contradiction that $\mathrm{cv}_S(2\epsilon) < \mathrm{pk}_S(2\epsilon)$. Then, there must exist two points $\boldsymbol{x}_i, \boldsymbol{x}_j$ in the maximal packing, which are within the same 2ϵ-ball in the cover and thus have distance $\|\boldsymbol{x}_i - \boldsymbol{x}_j\| \leq 2\epsilon$ contradicting the assumption.

For the second inequality, note that any maximal packing with balls of radius ϵ is a 2ϵ-covering, because otherwise, there would exist a point $\boldsymbol{x} \in S$ which is at least 2ϵ distance away from all the packing points contradicting the definition of a packing. □

Before stating the next result, we give a brief definition of *Ricci curvature*, which locally characterizes the curvature of $\mathcal{M}$ in a neighborhood of a point $x \in \mathcal{M}$.

Definition B.3 (Ricci Curvature)**.** Let $v \in T_x\mathcal{M}$ denote a unit vector and $\{u_1, \ldots, u_{m-1}, v\}$ an orthonormal basis of $T_x\mathcal{M}$; g_x denotes the inner product on $T_x\mathcal{M}$. Then the *Ricci curvature* at x along the direction v is defined as

$$\mathrm{Ric}_x(v) := \frac{1}{m-1} \sum_{i=1}^{m-1} g_x(R(v, u_i)v, u_i) \ , \tag{10}$$

where $R(u, v)w := \nabla_u \nabla_v w - \nabla_v \nabla_u w - \nabla_{[u,v]} w$ is the Riemann curvature tensor, with $[u, v]$ denoting the Lie Bracket between u and v.

Theorem B.4 (Corollary of Bishop-Gromov Theorem [10] (see [75] Lemma 7.1.3))**.** *Let $\mathrm{Vol}_{\mathcal{M}}(\boldsymbol{p}, r)$ denote the volume of the ball of radius r around the point $\boldsymbol{p}$ with respect to the Riemannian distance metric. If a manifold $\mathcal{M}$ of intrinsic dimension d has bounded Ricci curvature $\mathrm{Ric}(\mathcal{M}) \geq (d-1)K$, then for all $r > 0$ and $\boldsymbol{p} \in \mathcal{M}$, $\mathrm{Vol}_{\mathcal{M}}(\boldsymbol{p}, r) \leq \mathrm{Vol}_{S_d^K}(r)$ where $\mathrm{Vol}_{S_d^K}(r)$ denotes the volume of a ball of radius r around an arbitrary point $\boldsymbol{p} \in S_d^K$ in the space S_d^K with constant sectional curvature K (d-sphere if $K > 0$ or d-dimensional Hyperbolic space if $K < 0$).*

Lemma B.5 (Coupon collector bound [39])**.** *Let $\xi_1, \ldots, \xi_N$ be i.i.d. random variables taking values in $[n]$ with uniform probability: $\mathbb{P}[\xi_i = k] = \frac{1}{n}$ for all $k \in [n]$. Let T_n be the minimum value T such that for all $k \in [n]$, there exists at least one $i \in [T]$ where $\xi_i = k$ (i.e. every value in $[n]$ is covered). Then, $\mathbb{E}[T_n] = \Theta(n \log(n))$ and*

$$\lim_{n\to\infty} \mathbb{P}\left[T_n < n\log(n) + cn\right] = \exp(-\exp(-c)). \tag{11}$$

B.2 SQ lower bounds

To show lower bounds in the statistical query model, we follow the technique employed in [26, 13], which constructs hardness from statistical independence properties of a function class. Since our setting differs slightly from those in [26, 13], we provide an adapted version of their constructions below.

Definition B.6 ($(1-\eta)$-pairwise independent; see also Definition C.1 of [26])**.** Let $\mathcal{C}$ be a function class consisting of functions $f : \mathcal{X} \to \mathcal{Y}$. Let $\mathcal{D}$ be a distribution on $\mathcal{X}$. $\mathcal{C}$ is $(1-\eta)$-pairwise independent if there exists a finite subset $\widehat{\mathcal{Y}} \subseteq \mathcal{Y}$ such that with probability $(1-\eta)$ over $\boldsymbol{x}, \boldsymbol{x}' \in \mathcal{X}$ drawn independently from $\mathcal{D}$, the distribution of $(f(\boldsymbol{x}), f(\boldsymbol{x}'))$ for $f \sim \mathrm{Unif}(\mathcal{C})$ drawn uniformly at random from $\mathcal{C}$ is the product distribution $\mathrm{Unif}(\widehat{\mathcal{Y}}) \otimes \mathrm{Unif}(\widehat{\mathcal{Y}})$.

The above agrees with Definition C.1 of [26] apart from the extension that here we define the independence with respect to a finite subset $\widehat{\mathcal{Y}} \subseteq \mathcal{Y}$ of the output space.

Theorem B.7 (Theorem C.4 of [26])**.** *Let the function class $\mathcal{C}$ of functions $f : \mathcal{X} \to \mathcal{Y}$ be a $(1-\eta)$-pairwise independent function family with respect to a distribution $\mathcal{D}$ on $\mathcal{X}$. For any $f \in \mathcal{C}$, let $\mathcal{D}_f$ denote the distribution of $(\boldsymbol{x}, f(\boldsymbol{x}))$ where $\boldsymbol{x} \sim \mathcal{D}$. Let $\mathcal{D}_{\mathrm{Unif}(\mathcal{C})}$ denote the distribution of $(\boldsymbol{x}, y)$ where $\boldsymbol{x} \sim \mathcal{D}$ and $y = f(\boldsymbol{x})$ for $f \sim \mathrm{Unif}(\mathcal{C})$. Any SQ learner able to distinguish the labeled distribution $\mathcal{D}_{f^*}$ for an unknown $f^* \in \mathcal{C}$ from the randomly labeled distribution $\mathcal{D}_{\mathrm{Unif}(\mathcal{C})}$ using bounded queries of tolerance τ requires at least $\frac{\tau^2}{2\eta}$ such queries.*

Proof. We follow the proof in [26]. Let $\phi : \mathcal{X} \times \mathcal{Y} \to [-1, 1]$ be any query made by the learner and set $\phi[f] = \mathbb{E}_{\boldsymbol{x}\sim\mathcal{D}}[\phi(\boldsymbol{x}, f(\boldsymbol{x}))]$. Then,

$$\begin{aligned}
\mathrm{Var}_{f\sim\mathrm{Unif}(\mathcal{C})}\left[\phi[f]\right] &= \mathbb{E}_f\left[\phi[f]\phi[f]\right] - \mathbb{E}_f\left[\phi[f]\right]\mathbb{E}_{f'}\left[\phi[f']\right] \\
&= \mathbb{E}_{f,f'}\left[\mathbb{E}_{\boldsymbol{x}}[\phi(\boldsymbol{x}, f(\boldsymbol{x}))]\mathbb{E}_{\boldsymbol{x}'}[\phi(\boldsymbol{x}', f(\boldsymbol{x}'))] - \mathbb{E}_{\boldsymbol{x}}[\phi(\boldsymbol{x}, f(\boldsymbol{x}))]\mathbb{E}_{\boldsymbol{x}'}[\phi(\boldsymbol{x}', f'(\boldsymbol{x}'))]\right] \\
&= \mathbb{E}_{\boldsymbol{x},\boldsymbol{x}'}\mathbb{E}_{f,f'}\left[\phi(\boldsymbol{x}, f(\boldsymbol{x}))\phi(\boldsymbol{x}', f(\boldsymbol{x}')) - \phi(\boldsymbol{x}, f(\boldsymbol{x}))\phi(\boldsymbol{x}', f'(\boldsymbol{x}'))\right] \\
&\leq 2\eta.
\end{aligned} \tag{12}$$

In the last line, we use the fact that for any $(1-\eta)$-pairwise independent class $\mathcal{C}$, the inner expectation is zero with probability at least $1-\eta$ over the choice of $\boldsymbol{x}, \boldsymbol{x}' \sim \mathcal{D}$ and at most 2 otherwise.

Since any SQ algorithm must work with any given value of the noise within the stated tolerance, consider the adversarial strategy where the SQ oracle responds to a query with $\overline{\phi} = \mathbb{E}_{f\sim\mathrm{Unif}(\mathcal{C})}[\phi[f]]$ whenever possible. By Chebyshev's inequality,

$$\mathbb{P}_{f\sim\mathrm{Unif}(\mathcal{C})}\left[\left|\phi[f]-\overline{\phi}\right|>\tau\right] \leq \frac{\mathrm{Var}_{f\sim\mathrm{Unif}(\mathcal{C})}\left[\phi[f]\right]}{\tau^2} \leq \frac{2\eta}{\tau^2}. \tag{13}$$

So each such query only allows the learner to rule out at most a $\frac{2\eta}{\tau^2}$ fraction of $\mathcal{C}$. Thus to distinguish $\mathcal{D}_{f^*}$ from $\mathcal{D}_{\mathrm{Unif}(\mathcal{C})}$, the learner requires at least $\frac{\tau^2}{2\eta}$ queries. □

B.3 Additional examples of learnable manifolds

Here, we provide additional examples of manifolds and their properties that render such manifolds learnable as detailed in Section 3.1. There, learnability of the distribution of the manifold corresponded to the property that one can efficiently approximate the manifold with an epsilon net.

As mentioned in the main text, many algorithms in the manifold reconstruction literature implicitly assume that the manifold is efficiently sampleable. We give one such example below.

Example B.8 (Manifold Learning Setting [9])**.** [9] provide provable guarantees for the Isomap algorithm in the sense of recovering geodesic distances on a manifold $\mathcal{M}$ via shortest-path distance on a similarity graph G under the following assumptions (listed in Main Theorem A in [9]), which informally require among other things: (1) the algorithm is given a set of samples $\{\boldsymbol{x}_i\}$, which form a δ-net over the manifold $\mathcal{M}$; (2) $\mathcal{M}$ is geodesically convex; (3) the reach (equivalently encoded as bounds on the radius of curvature and branch separation) is bounded as $\mathrm{Rch}\,(\mathcal{M}) = O(\delta)$. Assumption (1) suffices to guarantee learnability in our setting.

We also extend Proposition 3.5 to incorporate manifolds with negative curvature. Negatively curved manifolds have volume that can grow exponentially with respect to the radius around a given point. However, as long as the radius of a sequence of manifolds does not grow with the ambient dimension, the Bishop-Gromov inequality bounds the volume of such manifolds rendering them efficiently sampleable. As far as we are aware, estimating the diameter of real-world manifolds is a challenging task and it is unknown whether real-world data manifolds can fit within this regime.

Example B.9 (Manifolds with bounded curvature and radius)**.** Any sequence of distributions $\{\mathcal{P}_{\mathcal{M}_n}\}$ supported over manifolds $\{\mathcal{M}_n\}$ where each manifold $\mathcal{M}_n$ has bounded Ricci curvature $\mathrm{Ric}\,(\mathcal{M}_n) \geq (d-1)K$ for an arbitrary constant K and is contained within a ball of radius $r = O_n(\log(n))$ around a point $\boldsymbol{p}_n \in \mathcal{M}_n$ is efficiently sampleable.

Proof. By the Bishop-Gromov theorem (Theorem B.4), the volume of any such manifold is bounded by $\mathrm{Vol}_{S_d^K}(r)$ which is the volume of a ball of radius r in the d-dimensional Hyperbolic space of constant curvature K. This means that

$$\mathrm{Vol}_d(\mathcal{M}_n) \leq \mathrm{Vol}_{S_d^K}(r) = \sigma_{d-1}\int_{t=0}^{r}\sinh^{d-1}(t)\,dt \leq \sigma_{d-1}r\left(\frac{\exp(r)}{2}\right)^{d-1}. \tag{14}$$

Thus, whenever $r = O_n(\log(n))$, this allows for an ϵ-cover of size $O_n(\mathrm{poly}(n))$. The rest of the proof follows directly from that of Proposition 3.5. □

Another example of a class of manifolds that fall in the learnable regime are those which are α-strongly convex [82], a property that can be leveraged in instances of Riemannian optimization. This property is extended from the standard notion of strong convexity on Hilbert spaces.

Definition B.10 (α-strong convexity of a Hilbert space [50])**.** Given a real Hilbert space H with inner product $\langle\cdot,\cdot\rangle$ and corresponding induced norm $\|\cdot\|$, denote by $B_R(c)$ the closed ball of radius R centered around $c \in H$. A subset $A \subset H$ is α-strongly convex if there exists a set $C \subset H$ such that

$$A = \bigcap_{c\in C} B_R(c), \tag{15}$$

where $R = \frac{1}{2\alpha}$.

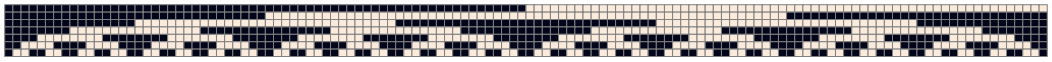


Figure 5: Enumeration of Gray code for $n = 7$ bits. Each column corresponds to a bitstring where black square is equal to 0 and tan square is 1. Note that the variation takes place largely in the last entries (bottom-most).

The above definition captures the equivalent notion in the Euclidean space $\mathbb{R}^d$ that an α-strongly convex set $A \subset \mathbb{R}^d$ is one where for any $\boldsymbol{x}, \boldsymbol{y} \in A$ and unit norm $\boldsymbol{z} \in \mathbb{R}^d$ such that $\|\boldsymbol{z}\| = 1$, we have that [82]

$$(1-t)\boldsymbol{x} + t\boldsymbol{y} + \alpha(1-t)t\|\boldsymbol{x} - \boldsymbol{y}\|^2\boldsymbol{z} \in A. \tag{16}$$

Riemannian manifolds are α-strongly convex if the image of the exponential map over inputs in a set A living in the tangent space are α-strongly convex in the Euclidean sense.

Definition B.11 (α-strong convexity of a Riemannian manifold [82])**.** Let $\mathcal{M}$ be a Riemannian manifold that is uniquely geodesic (i.e. any two points $\boldsymbol{x}, \boldsymbol{y} \in \mathcal{M}$ can be connected by a unique geodesic). Then, $\mathcal{M}$ is Riemannian α-strongly convex if for any $\boldsymbol{x} \in \mathcal{M}$, the set

$$\operatorname{Exp}_{\boldsymbol{x}}^{-1}(\mathcal{M}) \coloneqq \{\boldsymbol{y} \in T_{\boldsymbol{x}}\mathcal{M} : \boldsymbol{z} = \operatorname{Exp}_{\boldsymbol{x}}(\boldsymbol{y}), \boldsymbol{z} \in \mathcal{M}\} \tag{17}$$

is α-strongly convex with respect to the inner product $\|\cdot\|_{\boldsymbol{x}}$ on $T_{\boldsymbol{x}}\mathcal{M}$ in the sense of Definition B.10.

The above definition places restrictions on the diameter of manifolds, essentially placing them in the setting of bounded radius manifolds studied in Example B.9.

Example B.12 (Strongly convex manifolds)**.** As stated in Definition B.10 for any α-strongly convex manifold $\mathcal{M}$, the diameter of $\operatorname{Exp}_{\boldsymbol{x}}^{-1}(\mathcal{M})$ is at most $2r = \alpha^{-1}$. For a sequence of such Riemannian α-strongly convex manifolds $\{\mathcal{M}_n\}$, as long as the metric over the manifold in the normal coordinates at $T_{\boldsymbol{x}}\mathcal{M}$ is bounded everywhere (or say the curvature is bounded), then such sequences of manifolds are efficiently sampleable similar to the setting of Example B.9.

C Proofs for Section 3.2

C.1 Space-filling manifold

To construct a manifold which achieves the desired lower bound, we form a customized manifold which acts as a space-filling curve touching exponentially many quadrants of the n-dimensional hypercube. This space-filling curve is constructed by maneuvering around a Gray code which enumerates Boolean strings in such a fashion that successive strings differ by Hamming distance one. This procedure is motivated by and reminiscent of techniques used in constructing Hilbert curves from Gray codes [84].

Lemma C.1 (Gray code [51])**.** *For every integer $k > 0$, there exists a bijective function $G : [2^k] \to \{0,1\}^k$ enumerating length k bitstrings such that successive bitstrings $G(i)$ and $G(i+1)$ differ in only one coordinate, i.e., $|G(i) - G(i+1)|_H = 1$ where $|\cdot|_H$ denotes the Hamming distance.*

As a simple example, for $k = 3$, the sequence $[000, 001, 011, 010, 110, 111, 101, 100]$ is one such Gray code which can be constructed recursively as a binary-reflected Gray code [60]. We also visualize the $k = 7$ bit Gray code in Figure 5.

Given an intrinsic dimension d, we will construct manifolds $\mathcal{M}$ as products of $P_d = \{\boldsymbol{x} \in [0,1]^{d-1}\}$, the $(d-1)$-dimensional hypercube, and a one-dimensional submanifold over the remaining dimensions, which resembles a one-dimensional space-filling curve. The submanifold is constructed to touch the corners of the hypercube following a Gray code.

One-dimensional space-filling curve construction. Our goal is to construct a one-dimensional sub-manifold $\mathcal{M} \subset [0,1]^{n-d+1}$ that covers as many quadrants of the hypercube as possible. Given a radius of curvature R, which we will later set to be within the bounds of the stated reach, let $n_R = \lfloor (n-d+1)/\delta_R \rfloor$ and $\delta_R = \lceil 4R^2 \rceil$. In our construction, we will use the Gray code on n_R bits and map these to bitstrings in dimension n by copying the bitstring of the Gray code δ_R times.

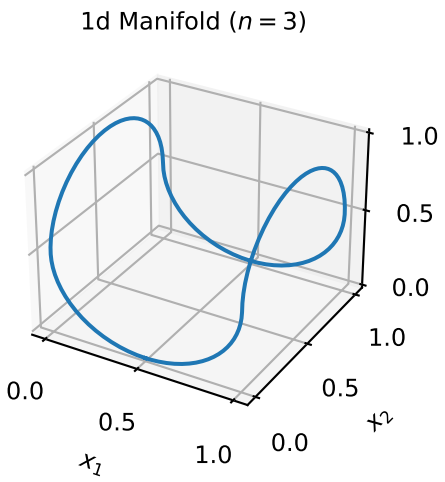


Figure 6: Shape of constructed manifold $\mathcal{M}_3$.

This construction will allow the manifold to have radius of curvature conforming to the given bound on the reach.

Specifically, let $G : [2^{n_R}] \to \{0,1\}^{n_R}$ be a Gray code over n_R bits, which maps indices $1, \ldots, 2^{n_R}$ to the corresponding bitstring in the Gray code (see Lemma C.1). Then, for $i \in [2^{n_R}]$ let $\boldsymbol{b}_i = G(i)^{\oplus \delta_R} \oplus \mathbf{0}_{n-n_R}$ where $G(i)^{\oplus \delta_R}$ is the bitstring $G(i)$ repeated δ_R times and $\mathbf{0}_k$ is the bitstring of length k with each entry equal to 0. $\mathbf{0}_k$ is concatenated at the end to ensure the input is of dimension n. Then, the space-filling sub-manifold takes the form:

$$\mathcal{M}_{n_R} = \bigcup_{k=1}^{2^{n_R}} \left\{ \frac{\boldsymbol{b}_{k-1} + \boldsymbol{b}_{k+1}}{2} + \left(\frac{\boldsymbol{b}_k - \boldsymbol{b}_{k+1}}{2} \right) \cos(t) + \left(\frac{\boldsymbol{b}_k - \boldsymbol{b}_{k-1}}{2} \right) \sin(t) : t \in [0, \pi/2] \right\}. \tag{18}$$

As an example, the manifold $\mathcal{M}_3$ is visualized in Figure 6. Indexing of the bitstrings $\boldsymbol{b}_k$ is assumed circular and taken $k \mod 2^n$.

Lemma C.2. *The manifold $\mathcal{M}_{n_R}$ defined in Equation (18) has $\operatorname{Rch}(\mathcal{M}_{n_R}) = R$. Furthermore, defining*

$$\operatorname{Round}(\mathcal{M}_{n_R}) = \left\{ \frac{1}{2} \left(\operatorname{sign}(\boldsymbol{x} - \mathbf{1}_{n_R}/2) + \mathbf{1}_{n_R} \right) : \boldsymbol{x} \in \mathcal{M}_{n_R} \right\} \tag{19}$$

as the operation which projects each point onto the nearest corner corresponding to bitstrings of the hypercube, we have that $|\operatorname{Round}(\mathcal{M}_{n_R})| = 2^{n_R}$.

Proof. Each segment of the manifold in Equation (18) is an arc of a circle centered at $\frac{\boldsymbol{b}_{k-1}+\boldsymbol{b}_{k+1}}{2}$ and of radius $\sqrt{\delta_R}/2$. For two segments indexed by k' such that $|k - k'| \geq 2$, the segments differ maximally in at least δ_R locations so the medial axis between these segments is at least $\sqrt{\delta_R}/2 \geq R$ distance away.

Between neighboring segments indexed by k and $k+1$, we note that we have a curve which consists of circular arcs from $[\mathbf{0}_{\delta_R}, \mathbf{1}_{\delta_R}/2, \mathbf{1}_{\delta_R}] \oplus \boldsymbol{s}$ to $[\mathbf{0}_{\delta_R}, \mathbf{1}_{\delta_R}, \mathbf{1}_{\delta_R}/2] \oplus \boldsymbol{s}$ and then to $[\mathbf{1}_{\delta_R}/2, \mathbf{1}_{\delta_R}, \mathbf{0}_{\delta_R}] \oplus \boldsymbol{s}$ for some bitstring $\boldsymbol{s}$ which is shared between all of the points in the curve. Here, for simplicity, we have permuted the distinct elements in the neighboring segments to the beginning of the bitstrings. Consider now the medial axis of this curve and set $\delta_R = 1$ for simplicity as the general case will directly follow. Here, the medial axis takes the form of points $[a, 1-b, c] \oplus \boldsymbol{s}$ where $0 \leq a, b \leq 1/2$ and $0 \leq c \leq 1$. For any such point where $c > 0.5$ or $c < 0.5$, the nearest point in the manifold will be a point in the first or second segment respectively. For such points, the medial axis will thus be the center of the corresponding circle which the arc spans. When $c = 0.5$, if $a = b < 0.5$, then the unique nearest point is always $[0, 1, 1/2] \oplus \boldsymbol{s}$. Otherwise, if $c = 0.5$ and $a < b$ or $b < a$, the nearest point in the manifold will be a point in the first or second segment respectively as in the previous case. Putting this all together, the reach of this manifold is $\sqrt{\delta_R}/2 \geq R$.

Finally, to count the number of elements in $\mathrm{Round}(\mathcal{M}_{n_R})$, note that setting $t = \pi/4$ for segment k in Equation (18) obtains a point of the form

$$(\boldsymbol{b}_{k-1} + \boldsymbol{b}_{k+1}) \frac{1}{2} \left(1 - \frac{\sqrt{2}}{2}\right) + \frac{\sqrt{2}}{2} \boldsymbol{b}_k. \tag{20}$$

The above is rounded to $\boldsymbol{b}_k$. Applying this procedure for all k gives 2^{n_R} total rounded points. □

Corollary C.3. *For a sequence of manifolds $\{\mathcal{M}_n\}$ of intrinsic dimension $d = O(1)$ and reach $\mathrm{Rch}(\mathcal{M}_n) = O(n^\alpha)$ for $\alpha < 0.5$ taking the form of Lemma C.2, the manifold covers exponentially many quadrants of the Boolean cube, i.e. $|\mathrm{Round}(\mathcal{M}_n)| = 2^{\Omega(n)}$.*

In our proofs, we will consider the setting where we have a sequence of manifolds of increasing intrinsic dimension n and bounded reach. We choose the parameters of this sequence as follows.

Definition C.4 ((α, d)-sequence of manifolds)**.** For given intrinsic dimension d and bound on the reach $R = O(n^\alpha)$ for some $\alpha < 0.5$, set $n = \lceil 4R^2 \rceil n_b + d - 1$. We construct a sequence of manifolds $\{\mathcal{M}_n\}$ of the form given in Equation (18) with the given values of R, n, d where n is incremented by increasing values n_b.

C.2 SQ hardness

We present here the hardness results in the statistical query (SQ) model by mapping a class of hard to learn Boolean functions to real-valued neural network functions, which approximate those Boolean functions well. This follows a commonly used set of proof techniques employed in [36, 32, 26, 48]. We follow most closely the line of reasoning in [26] for showing hardness results for learning two hidden layer ReLU networks though the nature of the distribution in our case will allow us to show SQ hardness results for single hidden layer networks. Similar hardness results based on reductions to cryptographically hard problems as shown in the work of [32] are also provided in Appendix C.3.

Hardness in our setting is based on classic results of the SQ hardness of learning parity functions over Boolean inputs [55, 11]. A parity function $\chi_S : \{0,1\}^d \to \{0,1\}$ is a Boolean function taking the form

$$\chi_S(\boldsymbol{x}_b) = \sum_{i \in S} [\boldsymbol{x}_b]_i \mod 2, \tag{21}$$

which determines whether the summation of the input Boolean string over indices in the set $S \subseteq [d]$ is even or odd. The class of such parity functions indexed by subsets $S \subseteq [d]$ is exponentially hard to learn in the SQ model.

Theorem C.5 (SQ hardness of parities [11, 55], see also Theorem 4.3 of [26])**.** *Any SQ algorithm given SQ access to the distribution of labeled pairs $(\boldsymbol{x}_b, y)$ where $\boldsymbol{x}_b \sim \mathrm{Unif}(\{0,1\}^d)$ and $y = \chi_S(\boldsymbol{x}_b)$ for an unknown $S \subseteq [d]$ capable of learning χ_S up to classification error ϵ sufficiently small with queries of tolerance τ requires $\Omega(\tau^2 2^d)$ queries.*

For Boolean inputs $\boldsymbol{x}_b \in \{0,1\}^d$, the parity function $\chi_S(\boldsymbol{x}_b)$ can be constructed as a single hidden layer neural network with $O(d)$ nodes. Ideally, one would want to replicate this over real-valued inputs, but since ReLU networks are continuous, this construction cannot be made exactly. [32] show that, in practice, one can form a network which is equivalent to the sign function for all but a small percentage of inputs. Combined with an indicator function that zeros out the output wherever such inputs cannot be rounded exactly, they show that equivalent hardness results can be obtained in real-valued settings. This construction was extended to SQ settings by [26].

In our setting where the underlying distribution is uniformly drawn from the space-filling manifold described in Appendix C.1, we can resort to a simpler construction given that most input values are either 0 or 1 at any point in the space-filling manifold. This lets us identify approximate parity functions which for all but an exponentially small fraction of inputs is equivalent to the Boolean parity function.

To achieve this goal, we need to map Boolean parity functions $\chi_S : \{0,1\}^{n_b} \to \{0,1\}$ to real-valued neural network functions $f : \mathbb{R}^n \to \mathbb{R}$ that approximate the parity functions appropriately. In this construction, for given intrinsic dimension d and bound on the reach $R = O(n^\alpha)$ for some $\alpha < 0.5$, we set $n = \lceil 4R^2 \rceil n_b + d - 1$ in the construction given in Appendix C.1 (n chosen so

that the number of bits in the Gray code n_R is equal to n_b). Each input entry is repeated $\lceil 4R^2 \rceil$ times in this construction, so for a given input $\boldsymbol{x} \in \mathbb{R}^n$, we define $\mathcal{P} : \mathbb{R}^n \to \mathbb{R}^{n_b}$ as an operation, which selects one of the $\lceil 4R^2 \rceil$ many equivalent inputs for each of the n_b potentially unique values. That is, for segments of the manifold taking the form of Equation (18), each segment is a sum of elements $\boldsymbol{b}_i = G(i)^{\oplus \delta_b} \oplus \mathbf{0}_{n-n_b}$ where $G : [2^{n_b}] \to \{0,1\}^{n_b}$ is a Gray code over n_b bits. Setting $\mathcal{P}\boldsymbol{x} = [\boldsymbol{x}]_{:n_b}$ to capture the first n_b elements then suffices to perform such an operation.

The intuition for our construction is as follows: We have to show that at most points on the manifold, inputs are with high probability equal to some Boolean string. If we choose the marginal distribution over the first k input locations of $\mathcal{P}\boldsymbol{x}$ with $k < n_b$ sufficiently smaller than n_b, then with high probability, these will always be constantly set to either value 0 or 1 in the segments of the manifold (Equation (18)). This is because the reflected Gray code is formed by changing the right-most bits first whenever possible so the left-most bits are constant along most segments. We formalize this below.

Lemma C.6. *Let $\mathcal{D}_{\mathcal{M}_n}$ denote the uniform distribution over the (α, d)-sequence of manifolds $\mathcal{M}_n$ constructed in Definition C.4 for given bounds on the reach $R = O(n^\alpha)$ and intrinsic dimension d. Given $\boldsymbol{x} \in \mathcal{M}_n$ constructed over the binary code over n_b bits, let $[\mathcal{P}\boldsymbol{x}]_{:n_b-t}$ denote the vector consisting of the first $n_b - t$ entries of $\mathcal{P}\boldsymbol{x}$ for $t \in [n_b]$. Then, with probability $1 - O(2^{-t})$, a draw of $\boldsymbol{x} \sim \mathcal{D}_{\mathcal{M}_n}$ will have the property that $[\mathcal{P}\boldsymbol{x}]_{:n_b-t} \in \{0,1\}^{n_b-t}$. Additionally, with probability $2^{t-n_b} + O(2^{-t})$ over independent draws $\boldsymbol{x}, \boldsymbol{x}' \sim \mathcal{D}_{\mathcal{M}_n}$, $[\mathcal{P}\boldsymbol{x}]_{:n_b-t} \neq [\mathcal{P}\boldsymbol{x}']_{:n_b-t}$ and the resulting values will differ in at least one location.*

Proof. In the binary reflected Gray code $G : [2^{n_b}] \to \{0,1\}^{n_b}$ over n_b bits, at any location $i \in [2^{n_b}]$, there exists a span of integers $[a, b)$ containing i of length $b - a = 2^t$ such that $[G(j)]_{n_b-t} = [G(j')]_{n_b-t}$ is equal for all $j, j' \in [a, b)$. Furthermore, by removing the endpoints where for any $j, j' \in [a+1, b-1)$, any entry of $[G(j)]_k$ for $k \in [n_b - t]$ will also have the property that $[G(j)]_k = [G(j')]_k$ (see Figure 5 for example for a visualization of this property). The value at a given point in a segment of the manifolds in Equation (18) is not equal to zero or one only when bitstring indices are differing in three adjacent points in the Gray code. Given a random point $\boldsymbol{x} \sim \mathcal{D}_{\mathcal{M}_n}$, it will fall within a given segment of the Gray code spanned over elements $i-1, i, i+1$ of the Gray code for $i \sim \mathrm{Unif}([2^{n_b}])$. Thus, the probability that $[\mathcal{P}\boldsymbol{x}]_{:n_b-t} \in \{0,1\}^{n_b-t}$ is at least $1 - O(2^{-t})$.

To prove the additional fact, note that for any $\boldsymbol{z} \in \{0,1\}^{n_b-t}$, we have that

$$\mathbb{P}[[\mathcal{P}\boldsymbol{x}]_{:n_b-t} = \boldsymbol{z}] \geq \frac{2^t - 8}{2^{n_b}}. \tag{22}$$

This follows from the previously stated property that at any location $i \in [2^{n_b}]$ in the gray code, there exists a span of integers $[a, b)$ containing i of length $b - a = 2^t$ such that $[G(j)]_k = [G(j')]_k$ is equal for all $j, j' \in [a+1, b-1)$ and all $k \in [n_b - t]$. From this span $[a, b)$ we remove the first and last three indices to get the span $[a+4, b-4)$ over which $[G(i)]_{n_b-t} = [G(i+k)]_{n_b-t}$ for all $i \in [a+4, b-4)$ and $k \in [-2, 2]$. I.e. this span is chosen to ensure that $\boldsymbol{x}_{n_b-t}$ is constant in Equation (18) over those segments of the Gray code. From here, we have that over independent draws $\boldsymbol{x}, \boldsymbol{x}' \sim \mathcal{D}_{\mathcal{M}_n}$:

$$\begin{aligned}
\mathbb{P}_{\boldsymbol{x},\boldsymbol{x}' \sim \mathcal{D}_{\mathcal{M}_n}} \left[[\mathcal{P}\boldsymbol{x}]_{:n_b-t} = [\mathcal{P}\boldsymbol{x}']_{:n_b-t}, [\mathcal{P}\boldsymbol{x}]_{:n_b-t} \in \{0,1\}^{n_b-t} \right] &= \sum_{\boldsymbol{z} \in \{0,1\}^{n_b-t}} \mathbb{P}\left[[\mathcal{P}\boldsymbol{x}]_{:n_b-t} = \boldsymbol{z} \right]^2 \\
&\leq \max_{\boldsymbol{z} \in \{0,1\}^{n_b-t}} \mathbb{P}\left[[\mathcal{P}\boldsymbol{x}]_{:n_b-t} = \boldsymbol{z} \right] \\
&\leq 1 - (2^{n_b-t} - 1)\frac{2^t - 8}{2^{n_b}} \\
&\leq 2^{t-n_b} + 8(2^{-t}).
\end{aligned} \tag{23}$$

The second line above follows from Hölder's inequality. The third line above applies Equation (22). Noting that the last term is $2^{t-n_b} + O(2^{-t})$ completes the proof. □

The above shows that with exponentially high probability, samples drawn from the distribution over the space-filling manifold will be equivalent to those drawn uniformly over Boolean inputs. This lets us complete the proof of hardness.

Theorem C.7. *Let $\mathcal{D}_{\mathcal{M}_n}$ denote the uniform distribution over the (α, d) sequence of manifolds $\mathcal{M}_n$ constructed in Definition C.4 where $\mathrm{Rch}(\mathcal{M}_n) = O(n^\alpha)$ for $\alpha < 0.5$. Any SQ algorithm $\mathcal{A}$ capable of learning the class of linear width single hidden layer ReLU neural networks under this sequence of distributions up to mean squared error sufficiently small ($\epsilon/8$ suffices) with queries of tolerance τ must use at least $\Omega(\tau^2 2^{n^{\Omega(1)}})$ queries.*

Proof. Given the sequence of manifolds $\{\mathcal{M}_n\}$ from the construction in Definition C.4, we have $n_b = \Omega(n^{1-2\alpha})$ which is chosen to fit the bounds on the reach of the manifold.

Given a parity function $\chi_S : \{0,1\}^d \to \{0,1\}$ taking the form

$$\chi_S(\boldsymbol{x}_b) = \sum_{i \in S} [\boldsymbol{x}_b]_i \mod 2, \tag{24}$$

consider its continuous approximation $\widetilde{\chi}_S : [0,1]^d \to \{0,1\}$ defined as

$$\widetilde{\chi}_S(\boldsymbol{x}) = \begin{cases} \sum_{i \in S} \boldsymbol{x}_i - k & \sum_{i \in S} \boldsymbol{x}_i \in [k, k+1], k \in \{0, 2, 4, ..., d\} \\ k + 1 - \sum_{i \in S} \boldsymbol{x}_i & \sum_{i \in S} \boldsymbol{x}_i \in [k, k+1], k \in \{1, 3, 5, ..., d-1\} \end{cases}. \tag{25}$$

Note, that the above is valid for d even and a similar equation can be obtained for d odd. The above is also piecewise linear and can be constructed as a single hidden layer ReLU network with $O(d)$ width. We consider learning the class $\mathcal{C}$ of single hidden layer networks consisting of

$$\mathcal{C} = \{f_S : S \subseteq [n_b - t]\} \text{ where } f_S(\boldsymbol{x}) = \widetilde{\chi}_S\left([\mathcal{P}\boldsymbol{x}]_{:n_b - t}\right). \tag{26}$$

Setting $t = n_b/2$ suffices by Lemma C.6 to guarantee that $[\mathcal{P}\boldsymbol{x}]_{:n_b/2}$ is a Boolean string with probability $1 - O(2^{-\Omega(n_b)})$. Furthermore, for $\boldsymbol{x}, \boldsymbol{x}'$ drawn i.i.d. from $\mathcal{D}_{\mathcal{M}_n}$, $[\mathcal{P}\boldsymbol{x}]_{:n_b/2} \neq [\mathcal{P}\boldsymbol{x}']_{:n_b/2}$ with probability $1 - O(2^{-\Omega(n_b)})$. Conditioned on this event, the distribution of $(f_S(\boldsymbol{x}), f_S(\boldsymbol{x}'))$ is equal to $\mathrm{Unif}(\{0,1\}^2)$. Since $n_b = \Omega(n^{1-2\alpha})$, we can apply Theorem B.7 noting that learning the function class up to MSE ϵ sufficiently small suffices to distinguish the distribution task in Theorem B.7. □

C.3 Cryptographic hardness reduction

Here, we will prove hardness results for learning single hidden layer networks under cryptographic hardness assumptions following the methodology in [32]. First, we detail the construction of Goldreich's pseudorandom generator (PRG) [49] which are used as the basis for the hard to learn class of single hidden layer neural networks in [32]. In fact, [32] show that given random Boolean inputs, a neural network with width $\omega(1)$ suffices to construct these hard to learn functions. From here, we show that the manifold constructed in Section 3.1 can produce inputs that are approximately uniformly distributed over the Boolean cube when restricted to a certain set of input locations. This lets us directly apply the results of [32] to this distribution.

Pseudorandomness assumption The pseudorandom functions are constructed over an (n, m, k)-hypergraph with n vertices $[n]$ with m ordered hyperedges $S_1, \ldots, S_m$, each having cardinality k and no repeated entries. Hypergraphs are drawn from the Erdős–Rényi distribution $\mathcal{G}_{n,m,k}$ where each hyperedge is drawn i.i.d. from the set of $n!/(n-k)!$ possible hyperedges (equivalent to ordered sets).

Definition C.8 (Goldreich's pseudorandom generator (PRG) [49])**.** Given constant integer $k > 0$, predicate $P : \{0,1\}^k \to \{0,1\}$, and a (n, m, k)-hypergraph G, Goldreich's pseudorandom generator (PRG) is the function $f_{P,G} : \{0,1\}^n \to \{0,1\}^m$, such that given input $\boldsymbol{x} \in \{0,1\}^n$ returns $f_{P,G}(\boldsymbol{x}) = (P(\boldsymbol{x}_{S_1}), \ldots, P(\boldsymbol{x}_{S_m}))$. The PRG has polynomial stretch if $m = n^a$ for some $a > 1$.

Denoting the collection of functions $f_{P,G}$ over (n, m, k)-hypergraphs as $\mathcal{F}_{P,n,m}$ then $\mathcal{F}_{P,n,m}$ is an ε-pseudorandom generator (ε-PRG) if every polynomial-time probabilistic algorithm $\mathcal{A}$ has advantage at most ϵ in distinguishing pairs $(G, f_{P,G}(\boldsymbol{x}))$ from random pairs $(G, \boldsymbol{y})$ where $\boldsymbol{y}$ is drawn i.i.d. from the uniform distribution on $\{0,1\}^m$. This is formalized below.

Definition C.9 (ε-pseudorandom generator (ε-PRG) [32])**.** $\mathcal{F}_{P,n,m}$ is an ε-pseudorandom generator (ε-PRG) if for every polynomial-time probabilistic algorithm $\mathcal{A}$, it holds that

$$\left|\mathbb{P}_{G \sim \mathcal{G}_{n,m,k}, \boldsymbol{x} \sim \mathcal{B}_n}\left[\mathcal{A}(G, f_{P,G}(\boldsymbol{x})) = 1\right] - \mathbb{P}_{G \sim \mathcal{G}_{n,m,k}, \boldsymbol{y} \sim \mathcal{B}_m}\left[\mathcal{A}(G, \boldsymbol{y}) = 1\right]\right| \leq \epsilon, \tag{27}$$

where $\mathcal{B}_k$ denotes the uniform distribution over bitstrings $\{0,1\}^k$.

The cryptographic hardness assumption posits that there does exist such a class as above for which it is hard to distinguish between the cases where $\boldsymbol{y}$ is completely random or the output of a randomly drawn function $f_{P,G}$.

Assumption C.10 (Existence of ϵ-PRG [32])**.** For every constant $s > 1$, there exists a constant k and a predicate $P : \{0,1\}^k \to \{0,1\}$, such that $\mathcal{F}_{P,n,n^s}$ is $\frac{1}{3}$-PRG.

Hard to learn neural network function class We follow the procedure in the previous subsection to map hard to learn Boolean functions to the task of learning real-valued functions where inputs are drawn from the data manifold. In this procedure, we must show that instances where a given input does not correspond to a Boolean string occur with a vanishingly small probability. Following the previous construction, for given intrinsic dimension d and bound on the reach $R = O(n^\alpha)$ for some $\alpha < 0.5$, we again set $n = \lceil 4R^2 \rceil n_b + d - 1$ in the construction given in Appendix C.1 (n chosen so that the number of bits in the Gray code n_R is equal to n_b). As each input entry is repeated $\lceil 4R^2 \rceil$ times in this construction, we recall the definition of $\mathcal{P} : \mathbb{R}^n \to \mathbb{R}^{n_b}$ selecting one of the $\lceil 4R^2 \rceil$ many equivalent inputs for each of the n_b potentially unique values. The following lemma then guarantees upon that taking the first $n_b - t$ entries, with probability $1 - O(2^{-t})$ the distribution of $[\mathcal{P}\boldsymbol{x}]_{:n_b-t}$ will be equivalent to the uniform distribution over Boolean strings.

Lemma C.11. *Let $\mathcal{D}_{\mathcal{M}_n}$ denote the uniform distribution over the (α, d)-sequence of manifolds $\mathcal{M}_n$ constructed in Definition C.4 for given bounds on the reach $R = O(n^\alpha)$ and intrinsic dimension d. Given $\boldsymbol{x} \in \mathcal{M}_n$ constructed over the binary code over n_b bits, let $[\mathcal{P}\boldsymbol{x}]_{:n_b-t}$ denote the vector consisting of the first $n_b - t$ entries of $\mathcal{P}\boldsymbol{x}$ for $t \in [n_b]$. Denoting $\mathcal{PD}_{\mathcal{M}_n}$ as the distribution of $[\mathcal{P}\boldsymbol{x}]_{:n_b-t}$ for $\boldsymbol{x} \sim \mathcal{D}_{\mathcal{M}_n}$, we have that with probability at least $1 - 2^{3-t}$, $\mathcal{PD}_{\mathcal{M}_n}$ returns a uniformly random Boolean string:*

$$\mathcal{PD}_{\mathcal{M}_n} \sim \begin{cases} \mathcal{B}_{n_b-t} & \text{w.p. } 1 - 2^{3-t}, \\ \mathcal{D}_{\text{rem}} & \text{w.p. } 2^{3-t}, \end{cases} \tag{28}$$

where $\mathcal{B}_k$ denotes the uniform distribution over bitstrings $\{0,1\}^k$ and $\mathcal{D}_{\text{rem}}$ is an arbitrary distribution handling the event where draws from the distribution $\mathcal{PD}_{\mathcal{M}_n}$ are not uniform over the Boolean string.

Proof. It suffices to show that there exists an event E occuring with probability at least $1 - 2^{3-t}$ such that conditioned on this event, for every $\boldsymbol{z} \in \{0,1\}^{n_b-t}$:

$$\mathbb{P}\left[[\mathcal{P}\boldsymbol{x}]_{:n_b-t} = \boldsymbol{z} | E\right] = 2^{t-n_b}, \tag{29}$$

i.e. every bitstring $\boldsymbol{z} \in \{0,1\}^{n_b-t}$ is returned with equal probability. In fact, previously in Lemma C.6 and specifically Equation (22), we showed that

$$\mathbb{P}[[\mathcal{P}\boldsymbol{x}]_{:n_b-t} = \boldsymbol{z}] \geq \frac{2^t - 8}{2^{n_b}}. \tag{30}$$

Therefore, we can construct the event E by taking the union over the bitstrings $\boldsymbol{z}$, each of probability $\frac{2^t-8}{2^{n_b}}$ ignoring a portion of the distribution whenever a given $\mathbb{P}[[\mathcal{P}\boldsymbol{x}]_{:n_b-t} = \boldsymbol{z}]$ exceeds this value. This implies that

$$\mathbb{P}[E] \geq 2^{n_b-t}\left(\frac{2^t - 8}{2^{n_b}}\right) = 1 - 2^{3-t}. \tag{31}$$

□

Theorem C.12 (Cryptographic hardness)**.** *Let $\mathcal{D}_{\mathcal{M}_n}$ denote the uniform distribution over manifolds $\mathcal{M}_n$ constructed in Definition C.4 where $\text{Rch}(\mathcal{M}_n) = O(n^\alpha)$ for $\alpha < 0.5$. Set $a > 0$ to be constant. Then under Assumption C.10, there does not exist any polynomial time algorithm $\mathcal{A}$ that with probability $1 - \delta$ for $0 < \delta < 1/2$ is capable of learning the class of width $O(n^a)$ single hidden layer* ReLU *neural networks under this sequence of distributions up to mean squared error sufficiently small ($\epsilon/40$ suffices).*

Proof. Theorem 8 (part 3) of [32] states that under Assumption C.10, there does not exist an efficient algorithm to learn the class of $O(n^a)$ width neural networks under the uniform input distribution $\mathcal{B}_m$ over bitstrings $\{0,1\}^m$. Namely, they show that there exist a set of neural network functions $f : \mathbb{R}^m \to [0,1]$ that have the property that they recover a hard to learn Boolean DNF formula

$f(\boldsymbol{x}) \in \{0,1\}$ for any $\boldsymbol{x} \in \{0,1\}^m$. They then show that any algorithm which with probability $1-\delta$ returns a hypothesis $h_b : \{0,1\}^m \to \{0,1\}$ achieving classification error $\mathbb{P}_{\boldsymbol{x}\sim\mathcal{B}_m}\left[h_b(\boldsymbol{x}) \neq f(\boldsymbol{x})\right] \leq 1/10$ suffices to contradict Assumption C.10.

Any efficient algorithm can only take $O(\mathrm{poly}(n))$ samples. From Lemma C.11, we can set $m = n_b/2$ with $t = n_b/2$ so that with probability at least $1 - \exp(-n^{\Omega(1)})$ a given sample is drawn from $\mathcal{B}_{n_b/2}$. By union bounding over $O(\mathrm{poly}(n))$ samples, we are guaranteed that all samples are drawn i.i.d. from $\mathcal{B}_{n_b/2}$ with probability $1-\delta'$ for δ' vanishing exponentially in n. Given any input $\boldsymbol{x}_b \in \{0,1\}^m$, we can convert it to a random point on the corresponding quadrant in the manifold $\boldsymbol{x} = \mathcal{T}\boldsymbol{x}_b \in \mathcal{M}_n$ by drawing $t \sim \mathrm{Unif}([0,\pi/2])$ and applying Equation (18). This recovers the distribution $\mathcal{D}_{\mathcal{M}_n}$. Now, assume that the algorithm for learning single hidden layer neural networks under the distribution $\mathcal{D}_{\mathcal{M}_n}$ returns a hypothesis $h : \mathbb{R}^n \to \mathbb{R}$. We can convert this to a boolean-output function $h' : \mathbb{R}^m \to \{0,1\}$ by setting $h'(\boldsymbol{x}) = \mathrm{sign}(h(\boldsymbol{x}) - 1/2)$. Note that (see also Lemma 23 of [32]) whenever $h'(\mathcal{T}\boldsymbol{x}_b) \neq f(\boldsymbol{x}_b)$ then $(h(\mathcal{T}\boldsymbol{x}_b) \neq f(\boldsymbol{x}_b))^2 \geq 1/4$ and

$$\mathbb{P}_{\boldsymbol{x}_b\sim\mathcal{B}_m, t\sim\mathrm{Unif}([0,\pi/2])}\left[h'(\mathcal{T}\boldsymbol{x}_b) \neq f(\boldsymbol{x}_b)\right] \leq 4\mathbb{E}_{\boldsymbol{x}_b\sim\mathcal{B}_m, t\sim\mathrm{Unif}([0,\pi/2])}\left[(h(\mathcal{T}\boldsymbol{x}_b) - f(\boldsymbol{x}_b))^2\right]. \tag{32}$$

The right hand side above contains the mean squared error loss over random Boolean inputs. This is equivalent to the mean squared error loss under the distribution $\mathcal{D}_{\mathcal{M}_n}$ up to an exponentially small correction for the event where $[\mathcal{P}\boldsymbol{x}]_{:n_b-t}$ is not Boolean. Thus, learning the function f up to mean squared error $\epsilon/40$ suffices to output a function h' achieving classification error $\epsilon/10$ (up to these exponentially vanishing corrections). Applying Theorem 8 of [32] finishes the proof. □

C.4 Volume bounds based on reach

The reach of a manifold lower bounds the radius of curvature of a manifold. In our setting where manifolds are contained within the hypercube $[0,1]^n$, restrictions on the reach can intuitively limit the space-filling capacity of a manifold resulting in upper bounds on the volume of the manifold. Namely, as we will show below, for a sequence of manifolds $\{\mathcal{M}_n\}$ of intrinsic dimension d, if the reach is bounded by $\mathrm{Rch}\left(\mathcal{M}_n\right) = \omega(n^{0.5})$, then the volume of the manifold is $\mathrm{Vol}_d(\mathcal{M}_n) = \mathrm{poly}(n)$. This implies that the results proven in the previous subsections are tight up to $\mathrm{Rch}\left(\mathcal{M}_n\right) = \Theta(n^{0.5})$, where similar exponential hardness results may apply though are not immediately obtained from our proofs. We formally detail the tightness of our results in what follows below.

Our volume bounds will follow from Lemma B.1 of [3] which locally bound the volume of a manifold of bounded reach contained within a ball.

Lemma C.13 (Adapted from Lemma B.1 of [3])**.** *Consider a Riemannian manifold $\mathcal{M} \subset \mathbb{R}^n$ of intrinsic dimension d with reach bounded by $R \leq \mathrm{Rch}\left(\mathcal{M}\right)$. Let D be a measure on $\mathbb{R}^n$ supported on $\mathcal{M}$ with density function f with respect to the volume measure Vol_d where f is Lipschitz smooth and bounded by $0 < f_{\min} \leq f(\boldsymbol{x}) \leq f_{\max}$ for all $\boldsymbol{x} \in \mathcal{M}$. Denote $B(\boldsymbol{x},h) = \{\boldsymbol{x}' : \|\boldsymbol{x}-\boldsymbol{x}'\| \leq h\}$ as the ball centered around $\boldsymbol{x}$ of radius h. Then for any $\boldsymbol{x}_0 \in \mathbb{R}^n$ and $h \leq R/8$, it holds that*

$$D(B(\boldsymbol{x},h)) \leq (5/4)^{d/2} 2^d f_{\max}\omega_d \max\left\{\left(h^2 - d(\boldsymbol{x},\mathcal{M})^2\right)^{d/2}, 0\right\}. \tag{33}$$

Lemma C.14. *Denote $B(\boldsymbol{x},h) = \{\boldsymbol{x}' : \|\boldsymbol{x}-\boldsymbol{x}'\| \leq h\}$ as the ball centered around $\boldsymbol{x}$ of radius h. Given a manifold $\mathcal{M} \subset \mathbb{R}^n$ of intrinsic dimension d bounded in reach by $\mathrm{Rch}\left(\mathcal{M}\right) \geq R$, the volume of the intersection of the manifold and a ball $B(\boldsymbol{x},h)$ with $h \leq R/8$ is bounded by*

$$\mathrm{Vol}_d\left(\mathcal{M} \cap B(\boldsymbol{x},h)\right) \leq \omega_d \left(\frac{R}{2}\right)^d, \tag{34}$$

where ω_d is the volume of the d-dimensional unit ball.

Proof. This is a consequence of Lemma B.1 of [3] recalled in Lemma C.13 setting $f_{\min} = f_{\max} = 1$ to recover the standard volume measure, i.e. for a given point $\boldsymbol{x} \in \mathbb{R}^n$ and $h \leq \mathrm{Rch}\left(\mathcal{M}\right)/8$, Lemma C.13 states

$$\mathrm{Vol}_d\left(\mathcal{M} \cap B(\boldsymbol{x},h)\right) \leq (5/4)^{d/2} 2^d \omega_d \left(h^2 - d(\boldsymbol{x},\mathcal{M})^2\right)^{d/2}. \tag{35}$$

Applying the bound $h \leq \mathrm{Rch}\left(\mathcal{M}\right)/8$ and noting that $(5/4)^{d/2}2^d(1/8)^d \leq (1/2)^d$ completes the proof. □

The above lemma provides an immediate bound on the volume as one can note that for large enough n, any manifold $\mathcal{M} \subset [0,1]^n$ is contained within a single ball $B(x,h)$ for $h = \omega(n^{0.5})$.

Proposition C.15. *Given a sequence of manifolds $\{\mathcal{M}_n\}$ of intrinsic dimension d (fixed and independent of n) and reach bounded by $\mathrm{Rch}(\mathcal{M}_n) = \omega(n^{0.5})$, the volume of the manifolds grows at most $\mathrm{Vol}_d(\mathcal{M}_n) = O(\mathrm{poly}(n))$.*

Proof. There exists large enough n such that $[0,1]^n$ is contained within a ball of radius equal to the bound on the reach $\mathrm{Rch}(\mathcal{M}_n) = \omega(n^{0.5})$. Applying Lemma C.14 with $h = \sqrt{n}/2$ and $\boldsymbol{x} = \mathbf{1}/2$ then implies that for large enough n:

$$\mathrm{Vol}_d(\mathcal{M}_n) = \mathrm{Vol}_d(\mathcal{M}_n \cap B(\boldsymbol{x}, h)) \leq \omega_d \left(\frac{\sqrt{n}}{4}\right)^d = \mathrm{poly}(n). \tag{36}$$

□

Remark C.16. The above implies that manifolds with reach bounded as $\mathrm{Rch}(\mathcal{M}_n) = \omega(n^{0.5})$ fall within the class of efficiently sampleable manifolds defined in Definition 3.2. The proof above does not extend to the setting where $\mathrm{Rch}(\mathcal{M}_n) = \Omega(n^{0.5})$. Covering number bounds (e.g. see Theorem 2 of [61]) show that exponentially many (in n) balls of radius $cn^{0.5}$ are needed to cover the space $[0,1]^n$ when c is sufficiently small. Therefore, Proposition C.15 does not guarantee that volume is polynomially bounded in n specifically for the setting where $\mathrm{Rch}(\mathcal{M}_n) = \Theta(n^{0.5})$.

D Experiments confirming learnability results

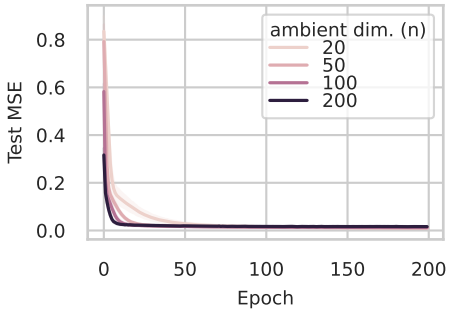


Figure 7: Replication of Figure 3a where the target is a neural network of the same form but with randomly chosen weights. Results are aggregate over five random initializations.

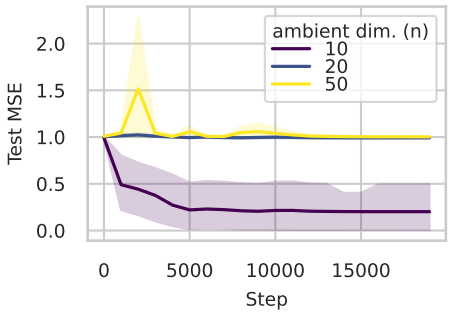


Figure 8: Replication of Figure 3b where the trained neural network is overparameterized in both width (3 hidden layers instead of just 1) and width. Results are consistent with that of Figure 3b further corroborating the hardness of this learning task. Results are aggregated over five random initializations including random tuning of the optimization hyperparameters.

For Figure 3a, the data is generated by drawing random unit norm vectors from a subspace of dimension $d = 10$ with ambient space of dimension n. The random d-dimensional subspace is obtained by drawing a random $n \times n$ orthogonal matrix and taking its first $d = 10$ columns. Target functions for Figure 3a are drawn from the class in Equation (5) in [36], which form a hard class of single hidden layer neural networks to learn under the i.i.d. Gaussian input model. We set the hidden layer width $k = n/4 = O(n)$ for this construction. To corroborate these findings, we also show in Figure 7 that the results are consistent when the objective is to learn a target function which is a single hidden layer ReLU neural network with randomly chosen weights of the same form. To normalize all target functions to have a consistent MSE benchmark, we divide by the norm of the function which is approximated by taking the average norm over a batch of 100 randomly chosen inputs.

We use the Pytorch package for our neural network experiments [73]. Training a single neural network on the tasks shown in Figure 3 took no longer than a few minutes on a single GPU. For the learnable setting in Figure 3a, we use a training set of size 1000 and a neural network of a single hidden layer and width 100 to learn the target function. For the hard setting of Figure 3b, we provide the algorithm with a fresh randomly drawn batch of data each training step. The trained neural network is overparameterized with width $2n$ in these experiments. For the hard setting in Figure 3b, we also attempted to learn the target function with an overparameterized network with three hidden layers and width $2n$. The results, shown in Figure 8, are consistent with those shown in Figure 3b. In all experiments, the learning rate for the optimizer was varied by the default value times a random multiplicative factor $\exp(c)$ where $c \sim \mathrm{Unif}([-2, 1])$; results were largely consistent across all ranges of the tuned values.

E Estimating Intrinsic Dimension

E.1 Details on Experimental Setup

Following [87], to calculate the intrinsic dimension at a given point $\boldsymbol{p} \in \mathcal{M}$, we sample a local neighborhood of points around $\boldsymbol{p}$ by noising $\boldsymbol{p}$ a small amount. Then, from a trained diffusion model, we collect a series of score vectors $\{\boldsymbol{s}_i\}_{i=1}^N$ at the sampled neighborhood which point in the direction of the de-noised sample. We then apply PCA upon this collection of N score vectors (treated as a matrix $\boldsymbol{S} \in \mathbb{R}^{n \times N}$) which gives us a basis for the tangent space $T_{\boldsymbol{p}}\mathcal{M}$ and normal space $N_{\boldsymbol{p}}\mathcal{M}$. Given potential instabilities inherent in training diffusion models and performing statistical estimation on manifolds, we use the stable rank to estimate the rank of $\boldsymbol{S}$ [91]. Given a matrix $\boldsymbol{M}$, the stable rank is given by

$$r_S(\boldsymbol{M}) \coloneqq \frac{\|\boldsymbol{M}\|_F^2}{\|\boldsymbol{M}\|_\infty^2}, \tag{37}$$

where $\|\cdot\|_F$ and $\|\cdot\|_\infty$ denote the Frobenius norm and maximum singular value respectively. Since $\boldsymbol{S}$ is actually high rank, we estimate instead the rank of $s_{\max}\boldsymbol{I} - \boldsymbol{S}$ where $s_{\max}$ is the max singular value of $\boldsymbol{M}$ to convert this into a low rank estimation problem. We find the stable rank above to be a more robust measure of estimated rank in comparison to other such metrics such as the maximum difference in singular values.

We note that this approach can also be taken by estimating the rank over a neighborhood of sampled points as well. That is, we find that collecting difference vectors between $\boldsymbol{p}$ and the denoised points as samples also estimates the dimension of the tangent space d accurately. Such a method may be more suited to higher dimensional inputs. However, to be consistent with prior work, we perform our analysis on collections of score vectors here. [87] have confirmed this overall methodology on synthetic datasets where the intrinsic dimension is known, showing that in general, it outperforms other statistical means to estimate intrinsic dimension.

Diffusion model and training. For Euclidean datasets, we train a fully-connected 5-layer ReLU network of width 2048 on 100000 samples for 30 epochs, batch size 64, and a learning rate of 4×10^{-4}.

For image datasets, we pre-process the images to have pixel values in $[-1, 1]$. We train a DDPM U-Net 2D model [79] on 10000 samples for 30 epochs, batch size 64, and a learning rate of 4×10^{-4}. The architecture is held the same across all three datasets: a ResNet block, a ResNet downsampling

block with spatial self-attention, and a ResNet block with 64, 128, and 256 channels each (and the reverse for upsampling).

We use a noise scheduler in the DDPM paradigm with 1000 timesteps between the image and pure Gaussian noise, with the squared cosine beta schedule introduced by [72]. This training was done on an NVIDIA L4 24GB GPU.

We investigate synthetic unit hyperspheres to sanity check our method and proceed to three real-world image datasets: MNIST [35], Fashion MNIST (FMNIST) [95], and Kuzushiji-MNIST (KMNIST) [29]. For each of the three image datasets, we consider two sizes: the original 28×28 and a downsampled 12×12.

E.2 Results on synthetic data manifolds

Following [87], we verify our intrinsic dimension estimation procedure on unit hyperspheres of different dimension, randomly projected into a higher-dimensional ambient space. We vary intrinsic and ambient dimension and report estimates of intrinsic dimension measurements (Table 2). We observe that, generally, the estimated and true intrinsic dimension align well. Our results further suggest an overall higher accuracy if the intrinsic dimension is small compared to the ambient dimension. Figure 9 shows the singular values of the score vector for different configurations; the curves, resembling a step function, distinguish normal from tangent directions.

Ambient Dim.	Intrinsic Dim.	Estimated Intrinsic Dim.
20	2	3
20	10	10
20	18	17
100	10	12
100	50	47
100	90	82

Table 2: Estimated intrinsic dimension for the hypersphere.

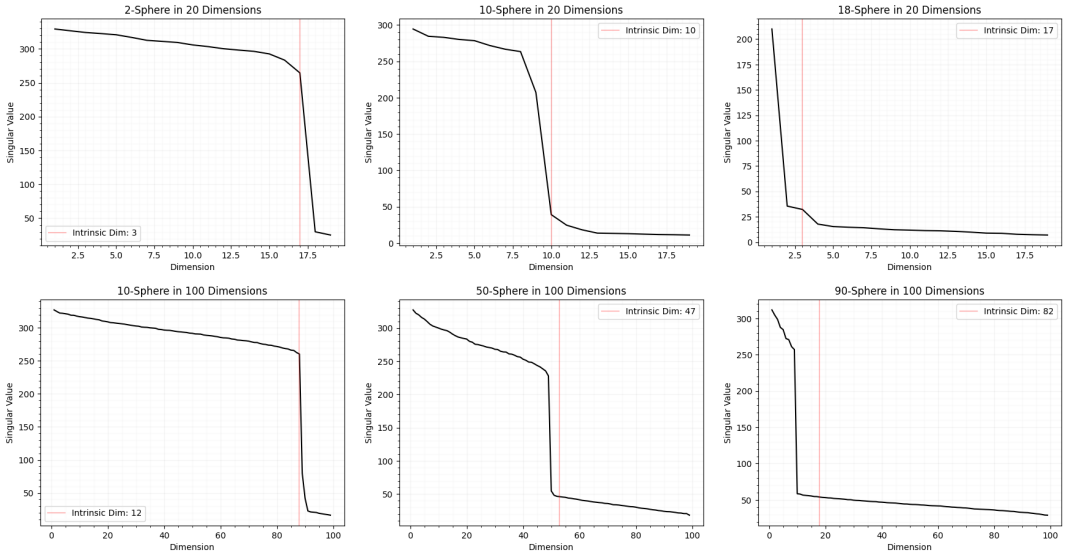


Figure 9: Intrinsic Dimension Estimation on Hyperspheres

E.3 Results on real-world data manifolds

As stated in the main text, we observe that our estimate of MNIST's intrinsic dimension matches that obtained in [87]. We observe slightly lower values, which we believe to be due to different

Dataset	Intrinsic Dimension (Std Dev)
MNIST (12 x 12)	29.8 (7.0)
MNIST (28 x 28)	97.0 (18.4)
KMNIST (12 x 12)	52.9 (9.4)
KMNIST (28 x 28)	168.9 (28.6)
FMNIST (12 x 12)	39.8 (10.5)
FMNIST (28 x 28)	299.1 (135.9)

Table 3: Estimated intrinsic dimension for the data manifolds considered. We report the mean across classes with standard deviation in brackets.

input sizes. In [87], MNIST images were upscaled to size 32×32, which differs from our input sizes (28×28, 12×12). In addition to MNIST, we also analyze FMNIST and KMNIST, which have not been considered in previous studies. Our results suggest that among the three datasets, FMNIST has the largest intrinsic dimension. Images in FMNIST cover part of the image in comparison to MNIST and KMNIST; we believe that much of this increase in intrinsic dimension is due to the larger set of possible perturbations of the image within this area.

We also explore how resizing the image impacts the data manifold's intrinsic dimension in Figure 10 and Table 3. We see that for MNIST and KMNIST, resizing to a smaller ambient dimension can reduce the intrinsic dimension though at a slightly slower rate than that of the ambient dimension. We note that all the images we consider here have relatively intrinsic dimension, where the effects of resizing may be more prominent.

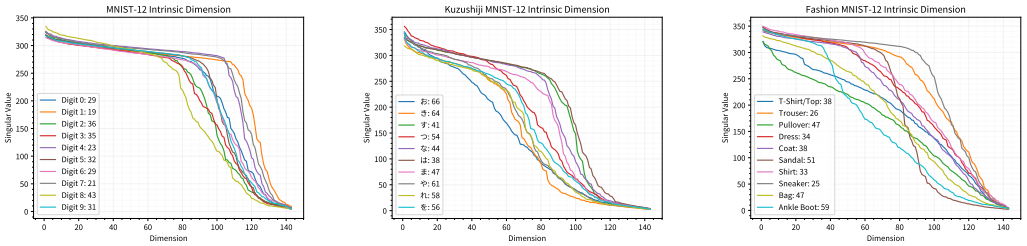


Figure 10: Impact of Resizing on Score Spectrum for Estimating Intrinsic Dimension from Diffusion Models

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: In this work, we support formal claims with proofs and include experimental evidence of the main findings.

   Guidelines:

   - The answer NA means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A No or NA answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: We have described limitations of our work at various points. Given our work describes a theoretical setting where we can prove hardness under data manifold assumptions, many of these limitations relate to the assumptions fo the work. For example, in the discussion section, we mention various places where our formal statements do not apply and mention future areas of work to address these limitations. In our experiments, we also outline areas of differences between practice and theory.

   Guidelines:

   - The answer NA means that the paper has no limitation while the answer No means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory Assumptions and Proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

   Answer: [Yes]

   Justification: All formal statements are proven either in the main text or Appendix.

   Guidelines:

   - The answer NA means that the paper does not include theoretical results.
   - All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
   - All assumptions should be clearly stated or referenced in the statement of any theorems.
   - The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
   - Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
   - Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental Result Reproducibility**

   Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

   Answer: [Yes]

   Justification: Code is supplied which replicates all experiments. We include additional details of experiments in the main text and in appendix.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - If the paper includes experiments, a No answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
   - If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
   - Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
   - While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
     (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
     (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
     (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
     (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility.

In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

   Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

   Answer: [Yes]

   Justification: Code is supplied and details are provided in appendix for experiments.

   Guidelines:

   - The answer NA means that paper does not include experiments requiring code.
   - Please see the NeurIPS code and data submission guidelines (`https://nips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - While we encourage the release of code and data, we understand that this might not be possible, so "No" is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
   - The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://nips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
   - The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
   - At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
   - Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental Setting/Details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer, etc.) necessary to understand the results?

   Answer: [Yes]

   Justification: Details of optimizers, hyperparameters, etc. are provided in Appendices D and E.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
   - The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment Statistical Significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

   Answer: [Yes]

   Justification: For experiments where we train neural networks (Appendix D), we include error bars over random instantiations of the training.

   Guidelines:

   - The answer NA means that the paper does not include experiments.

- The authors should answer "Yes" if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
- The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
- The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
- The assumptions made should be given (e.g., Normally distributed errors).
- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g. negative error rates).
- If error bars are reported in tables or plots, The authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments Compute Resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: Computational resources were relatively limited for this largely theoretical work. All experiments required only a single GPU (see Appendices D and E).

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code Of Ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

   Answer: [Yes]

   Justification:

   Guidelines:

   - The answer NA means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer No, they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader Impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [NA]

Justification: This work fits within the scope of foundational research and is theoretical in nature. We do not introduce any new models or techniques, instead focusing on understanding data manifolds and learning in a more theoretical framework. Although this work can advance our knowledge of machine learning, there are no clear negative societal impacts that directly arise from this work.

Guidelines:

- The answer NA means that there is no societal impact of the work performed.
- If the authors answer NA or No, they should explain why their work has no societal impact or why the paper does not address societal impact.
- Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pretrained language models, image generators, or scraped datasets)?

Answer: [NA]

Justification: Our work is largely theoretical and apart from some small datasets like MNIST, we do not perform any experiments on data that has a high risk for misuse.

Guidelines:

- The answer NA means that the paper poses no such risks.
- Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
- Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
- We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

Answer: [Yes]

Justification: The real-world datasets used in our paper are cited with their respective licenses explicitly named in the README and are properly respected.

Guidelines:

- The answer NA means that the paper does not use existing assets.
- The authors should cite the original paper that produced the code package or dataset.
- The authors should state which version of the asset is used and, if possible, include a URL.
- The name of the license (e.g., CC-BY 4.0) should be included for each asset.
- For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
- If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
- For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New Assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [NA]

    Justification:

    Guidelines:

    - The answer NA means that the paper does not release new assets.
    - Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
    - The paper should discuss whether and how consent was obtained from people whose asset is used.
    - At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and Research with Human Subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [NA]

    Justification:

    Guidelines:

    - The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
    - Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
    - According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional Review Board (IRB) Approvals or Equivalent for Research with Human Subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [NA]

Justification:

Guidelines:

- The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
- Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
- We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
- For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.