Why do autoencoders work?

Anonymous authors
Paper under double-blind review

Abstract

Deep neural network autoencoders are routinely used computationally for model reduction. They allow recognizing the intrinsic dimension of data that lie in a k-dimensional subset K of an input Euclidean space $\mathbb{R}^n$. The underlying idea is to obtain both an encoding layer that maps $\mathbb{R}^n$ into $\mathbb{R}^k$ (called the bottleneck layer or the space of latent variables) and a decoding layer that maps $\mathbb{R}^k$ back into $\mathbb{R}^n$, in such a way that the input data from the set K is recovered when composing the two maps. This is achieved by adjusting parameters (weights) in the network to minimize the discrepancy between the input and the reconstructed output. Since neural networks (with continuous activation functions) compute continuous maps, the existence of a network that achieves perfect reconstruction would imply that K is homeomorphic to a k-dimensional subset of $\mathbb{R}^k$, so clearly there are topological obstructions to finding such a network. On the other hand, in practice the technique is found to "work" well, which leads one to ask if there is a way to explain this effectiveness. We show that, up to small errors, indeed the method is guaranteed to work. This is done by appealing to certain facts from differential geometry. A computational example is also included to illustrate the ideas.

1 Introduction

Many real-world problems require the analysis of large numbers of data points inhabiting some Euclidean space $\mathbb{R}^n$. The "manifold hypothesis" Fefferman et al. (2016) postulates that these points lie on some k-dimensional submanifold with (or without) boundary $K \subset \mathbb{R}^n$, so can be described locally by $k < n$ parameters. When K is a linear submanifold, classical approaches like principal component analysis and multidimensional scaling are effective ways to learn these parameters. But when K is nonlinear, learning these parameters is the more challenging "manifold learning" problem studied in the rapidly developing literature on "geometric deep learning" Bronstein et al. (2017).

One popular approach to this problem relies on deep neural network **autoencoders** (also called "replicators" Hecht-Nielsen (1995)) of the form $G \circ F$, where the output of the **encoder** $F\colon \mathbb{R}^n \to \mathbb{R}^k$ is the desired $k < n$ parameters, $G\colon \mathbb{R}^k \to \mathbb{R}^n$ is the **decoder**, and F and G are continuous. The goal is to learn F, G to create a perfect autoencoder, one such that $G(F(x)) = x$ for all $x \in K$. Clearly such F, G exist if and only if K is homeomorphic to a k-dimensional submanifold with boundary of $\mathbb{R}^k$, so there are topological obstructions making this goal impossible in general, as observed in Batson et al. (2021).

And yet, the wide practical applicability of the method evidences remarkable empirical success from autoencoders even when K is not homeomorphic to such a subset of $\mathbb{R}^k$. (We give an illustrative numerical experiment in §3.) How can this be?

This apparent paradox is resolved by the following Theorem 1, which asserts that the set of $x \in K$ for which $G(F(x)) \not\approx x$ can be made arbitrarily small with respect to the "intrinsic measure" μ (defined in §2) on K generalizing length and surface area. For the statement, $\mathcal{F}^{\ell,m}$ denotes any set of continuous functions $\mathbb{R}^\ell \to \mathbb{R}^m$ with the "universal approximation" property that any continuous function $H\colon \mathbb{R}^\ell \to \mathbb{R}^m$ can be uniformly approximated arbitrarily closely on any compact set $L \subset \mathbb{R}^\ell$ by some $\tilde{H} \in \mathcal{F}^{\ell,m}$. In particular, $\mathcal{F}^{\ell,m}$ can be the collection of possible functions $\mathbb{R}^\ell \to \mathbb{R}^m$ that can be produced by a suitable class of neural networks.

Theorem 1. Let $k, n \in \mathbb{N}_{\geq 1}$ and $K \subset \mathbb{R}^n$ be a union of finitely many compact smoothly embedded submanifolds with boundary each having dimension less than or equal to k. For each $\varepsilon, \delta > 0$ there is a closed set $K_0 \subset K$ with intrinsic measure $\mu(K_0) < \delta$ and continuous functions $F \in \mathcal{F}^{n,k}$, $G \in \mathcal{F}^{k,n}$ such that

$$\sup_{x \in K \setminus K_0} \|G(F(x)) - x\| < \varepsilon. \tag{1}$$

Moreover, K_0 can be chosen to be disjoint from any finite set S of points in K.

This may be interpreted as a "probably approximately correct (PAC)" theorem for autoencoders complementary to recent PAC theorems obtained in the manifold learning literature Fefferman et al. (2016; 2018; 2023). It asserts that, for any finite training set S of data points in K, there is an autoencoder $G \circ F$ with error smaller than ε on S such that the "generalization" error will also be smaller than ε on any test data in $K \setminus K_0$. (A related idea seems to have appeared in (Hecht-Nielsen, 1995, Fig. 4), but without a general statement or proof.)

The remainder of the paper is organized as follows. Theorem 1 is proved in §2. The numerical experiments are in §3. A result ruling out certain extensions of Theorem 1 is proved in §4. An appendix contains the implementation code for these experiments.

2 Proof of Theorem 1

In this section we prove Theorem 1. In this paper, we use the term "manifold with boundary" to encompass manifolds without boundary (those for which $\partial M = \varnothing$) as a special case. When a result is specific to manifolds without boundary, we will explicitly say so. Recall that there is an intrinsic notion of "measure zero" subsets of a smooth manifold with boundary (Lee, 2013, Ch. 6).

Lemma 1. Let M be a k-dimensional connected smooth manifold with boundary. There exists a closed measure zero set $C \subset M$ such that $M \setminus C$ is diffeomorphic to a k-dimensional embedded submanifold with boundary of $\mathbb{R}^k$. Moreover, given any finite set $S \subset M$, C can be chosen so that $C \cap S = \varnothing$.

Proof. First assume that the boundary $\partial M = \varnothing$. If $M = \mathbb{R}^0$ or $M = \mathbb{R}$, set $C = \varnothing$. If $M = \mathbb{S}^1$, let C be any point in $M \setminus S$. Since these are the only options for $k = 0$ or $k = 1$ (Lee, 2013, Ex. 15-13), it remains only to consider the case $k \geq 2$. Equipping M with any complete Riemannian metric yields a diffeomorphism (given by the exponential map) $H \colon M \setminus C_0 \approx \mathbb{R}^k$ where $C_0 \subset M$, the cut locus with respect to the metric and an arbitrary $p \in M$, has measure zero (Sakai, 1996, Lem. III.4.4). There exists a diffeomorphism $J \colon M \to M$ such that $J(S) \cap C_0 = \varnothing$ Michor & Vizman (1994). Hence $C \coloneqq J^{-1}(C_0)$ is also measure zero and disjoint from S, and $H \circ J \colon M \setminus C \approx \mathbb{R}^k$ is a diffeomorphism.

Next assume that $\partial M \neq \varnothing$. Identifying M with one of its two copies in the double DM of M (Lee, 2013, Ex. 9.32), the previous case furnishes a measure zero subset $C_0 \subset DM$ disjoint from S and a diffeomorphism $DM \setminus C_0 \approx \mathbb{R}^k$. This diffeomorphism restricts to one from $M \setminus C$ onto its image, where $C \coloneqq C_0 \cap M$. □

Recall that any union K of smooth submanifolds of a Euclidean space has an **intrinsic measure** μ given by the Riemannian density (Lee, 2013, p. 428) of the restriction of the Euclidean metric to the submanifolds. Any measure zero subset C of K in the sense of (Lee, 2013, Ch. 6) has intrinsic measure $\mu(C) = 0$.

Lemma 2. Let $k, n \in \mathbb{N}_{\geq 1}$ and $K \subset \mathbb{R}^n$ be a closed set equal to a union of smoothly embedded submanifolds with boundary each having dimension less than or equal to k. Assume that each connected component M of K is uniformly separated from $K \setminus M$ by a positive distance. For each $\delta > 0$ there is a closed set $K_0 \subset K$ with intrinsic measure $\mu(K_0) < \delta$ and smooth functions $F \colon \mathbb{R}^n \to \mathbb{R}^k$, $G \colon \mathbb{R}^k \to \mathbb{R}^n$ such that $G \circ F|_{K \setminus K_0} = \mathrm{id}_{K \setminus K_0}$. Moreover, given any finite set $S \subset M$, K_0 can be chosen so that $K_0 \cap S = \varnothing$.

Proof. The assumptions imply that K has at most countably many components and that each such component is a connected smooth manifold with boundary of dimension less than or equal to k. Lemma 1 thus implies that, after deleting a closed measure zero subset $C \subset K$ disjoint from S, each such component is

diffeomorphic to an embedded submanifold with boundary of $\mathbb{R}^k$. Compressing the images of these diffeomorphisms into arbitrarily small disjoint disks by post-composing each with a suitable diffeomorphism of $\mathbb{R}^k$ produces a diffeomorphism F_0 of $K \setminus C$ onto a union of embedded submanifolds with boundary of $\mathbb{R}^k$, with each submanifold uniformly separated from the union of the others.

Outer regularity of the intrinsic measure and Sard's theorem imply the existence of a union $K_0 \subset K$ of properly embedded codimension-0 smooth submanifolds with boundary disjoint from S such that $K_0 \supset C$ and $\mu(K_0) < \delta$. Let $\operatorname{int}(K_0)$ denote the manifold interior and define $F\colon \mathbb{R}^n \to \mathbb{R}^k$ to be any smooth extension (Lee, 2013, Lem. 2.26) of $F|_{K\setminus\operatorname{int}(K_0)}$. Then $F|_{K\setminus\operatorname{int}(K_0)} = F_0|_{K\setminus\operatorname{int}(K_0)}$ is a diffeomorphism onto a union N of properly embedded submanifolds with boundary of $\mathbb{R}^k$, with each submanifold uniformly separated from the union of the others. Defining $G\colon \mathbb{R}^k \to \mathbb{R}^n$ to be any smooth extension (Lee, 2013, Lem. 2.26) of the inverse diffeomorphism $(F|_{K\setminus\operatorname{int}(K_0)})^{-1}\colon N \to K \setminus \operatorname{int}(K_0) \subset \mathbb{R}^n$ yields the desired equality

$$G \circ F|_{K\setminus K_0} = (F|_{K\setminus\operatorname{int}(K_0)})^{-1} \circ F|_{K\setminus K_0} = \mathrm{id}_{K\setminus K_0}.$$

□

Assume given for each $\ell, m \in \mathbb{N}_{\geq 1}$ a collection $\mathcal{F}^{\ell,m}$ of continuous functions $\mathbb{R}^\ell \to \mathbb{R}^m$ with the following "universal approximation" property: for any $\varepsilon > 0$, compact subset $L \subset \mathbb{R}^\ell$, and continuous function $H\colon \mathbb{R}^\ell \to \mathbb{R}^m$, there is $\tilde{H} \in \mathcal{F}^{\ell,m}$ such that $\sup_{x\in L} \|H(x) - \tilde{H}(x)\| < \varepsilon$. Equivalently, $\mathcal{F}^{\ell,m}$ is any collection of continuous functions $\mathbb{R}^\ell \to \mathbb{R}^m$ that is dense in the space of continuous functions $\mathbb{R}^\ell \to \mathbb{R}^m$ with the compact-open topology (Hirsch, 1994, Sec. 2.4). We now restate and prove Theorem 1.

Theorem 1. Let $k, n \in \mathbb{N}_{\geq 1}$ and $K \subset \mathbb{R}^n$ be a union of finitely many compact smoothly embedded submanifolds with boundary each having dimension less than or equal to k. For each $\varepsilon, \delta > 0$ there is a closed set $K_0 \subset K$ with intrinsic measure $\mu(K_0) < \delta$ and continuous functions $F \in \mathcal{F}^{n,k}$, $G \in \mathcal{F}^{k,n}$ such that

$$\sup_{x\in K\setminus K_0} \|G(F(x)) - x\| < \varepsilon. \tag{1}$$

Moreover, K_0 can be chosen to be disjoint from any finite set S of points in K.

Proof. The first sentence of the theorem statement implies that K satisfies the assumptions in the first two sentences of Lemma 2. Thus, Lemma 2 implies the existence of a closed set $K_0 \subset K$ disjoint from S with intrinsic measure $\mu(K_0) < \delta$ and smooth functions $\tilde{F} : \mathbb{R}^n \to \mathbb{R}^k$, $\tilde{G}\colon \mathbb{R}^k \to \mathbb{R}^n$ such that $\tilde{G} \circ \tilde{F}|_{K\setminus K_0} = \mathrm{id}_{K\setminus K_0}$. Density of $\mathcal{F}^{n,k}$, $\mathcal{F}^{k,n}$ and continuity of the composition map $(G, F) \mapsto G \circ F$ in the compact-open topologies (Hirsch, 1994, p. 64, Ex. 10(a)) imply the existence of $F \in \mathcal{F}^{n,k}$, $G \in \mathcal{F}^{k,n}$ such that $G \circ F|_K$ is uniformly ε-close to $\tilde{G} \circ \tilde{F}|_K$, so F, G satisfy (1). □

3 Numerical illustration

We next illustrate the results through the training of a deep neural network autoencoder. In our example, inputs and outputs of the network are three dimensional, and the set K is taken to be the union of two embedded submanifolds of $\mathbb{R}^3$. The first manifold is a unit circle centered at $x = y = 0$ and lying in the plane $z = 0$. The second manifold is a unit circle centered at $x = 1$, $z = 0$ and contained in the plane $y = 0$. See Figure 1.

The choice of suitable neural net architecture "hyperparameters" (number of layers, number of units in each layer, activation function) is a bit of an art, since in theory just single-hidden layer architectures (with enough "hidden units" or "neurons") can approximate arbitrary continuous functions on compacts. After some experimentation, we settled on an architecture with three hidden layers of encoding with 128 units each, and similarly for the decoding layers. The activation functions are ReLU (Rectified Linear Unit) functions, except for the bottleneck and output layers, where we pick simply linear functions. Graphically this is shown in Figure 2. An appendix lists the Python code used for the implementation. We generated 500 points in each of the circles. The resulting decoded vectors are shown in Figure 3. Observe how the circles have been broken to make possible their embedding into $\mathbb{R}^1$. Finally, Figure 4 shows the image of the encoder layer

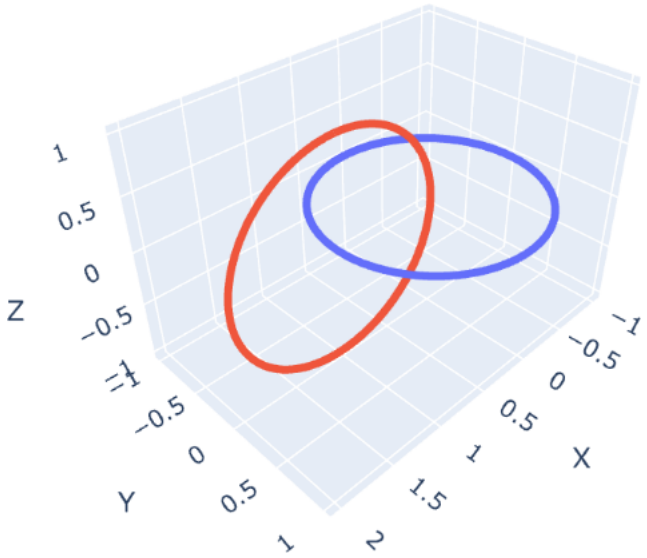

Figure 1: Two interlaced unit circles, one centered at $x = y = 0$ in the plane $z = 0$ (blue), and another centered at $x = 1$, $z = 0$ in the plane $y = 0$ (red).

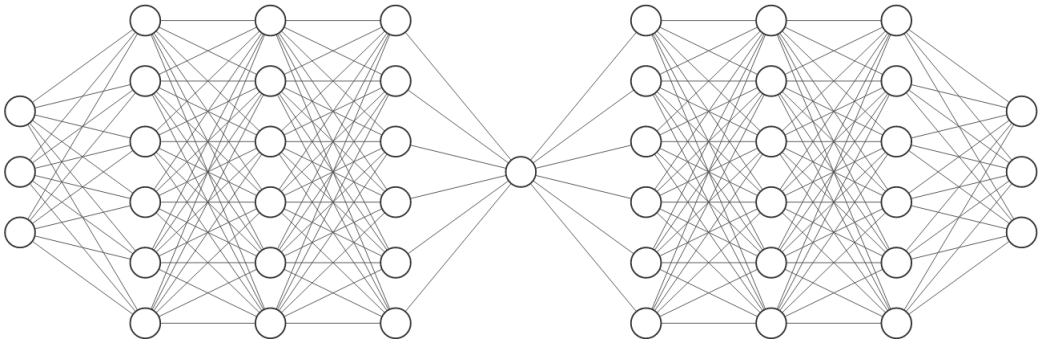

Figure 2: The architecture used in the computational example. For clarity in the illustration, only 6 units are depicted in each layer of the encoder and decoder, but the number used was 128.

mapping as a subset of $\mathbb{R}^1$.

It is important to observe that most neural net training algorithms, including the one that we employ (TensorFlow), are stochastic, and different training runs might give different results or simply not converge. As an illustration of how results may differ, see Figure 5 and Figure 6 for the decoded and bottleneck data in another training run (with the same data). It is a little difficult to see the break point for the blue circle, so we have rotated the image in Figure 7 (decoded) in order to appreciate the topology better.

4 Optimality of Theorem 1

Theorem 1 asserts that arbitrarily accurate autoencoding is always possible on the complement of a closed subset $K_0 \subset K$ having arbitrarily small positive intrinsic measure. This leads one to ask whether that result can be improved by imposing further "smallness" conditions on K_0. For example, rather than small positive measure, can one require that K_0 has measure zero? Alternatively, can one require that K_0 is small in the Baire sense (meagre)? In either case, the complement of K_0 in K would be dense, so the ability to arbitrarily accurately autoencode $K \setminus K_0$ as in Theorem 1 would imply the same for all of K (by continuity).

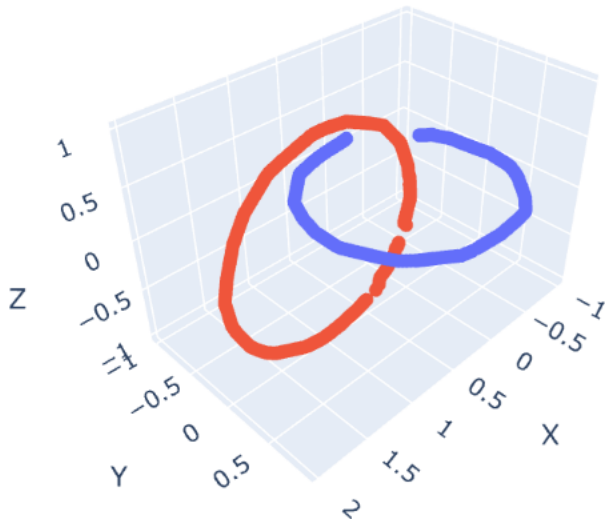

Figure 3: The output of the autoencoder for the two interlaced unit circles, one centered at $x = y = 0$ in the plane $z = 0$ (blue), and another centered at $x = 1$, $z = 0$ in the plane $y = 0$ (red). The network training algorithm automatically picked the points at which the circles should be "opened up" to avoid the topological obstruction.

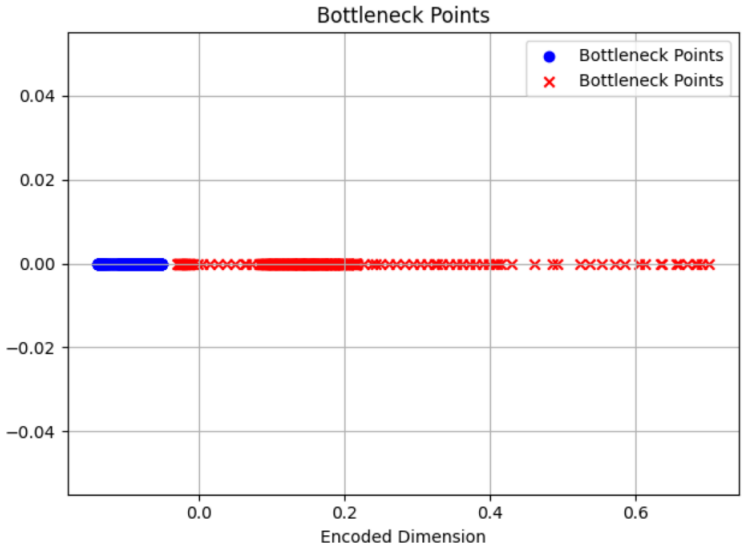

Figure 4: The bottleneck layer, showing the images of the blue and red circles.

The following Theorem 2 eliminates the possibility of such extensions by showing that, for a broad class of K, the maximal autoencoder error on K is bounded below by the **reach** $r_K \geq 0$ of K, a constant depending only on K. Here r_K is defined to be the largest number such that any $x \in \mathbb{R}^n$ satisfying $\operatorname{dist}(x, K) < r_K$ has a unique nearest point on K Federer (1959); Aamari et al. (2019); Berenfeld et al. (2022); Fefferman et al. (2016; 2018). Figure 8 illustrates this concept.

Remark 1. The example $K \coloneqq \{0\} \cup \{1/n \colon n \in \mathbb{N}_{\geq 1}\} \subset \mathbb{R}$ shows that a compact subset of a Euclidean space need not have a positive reach $r_K \geq 0$. However, $r_K > 0$ if K is a compact smoothly embedded submanifold (cf. (2) below).

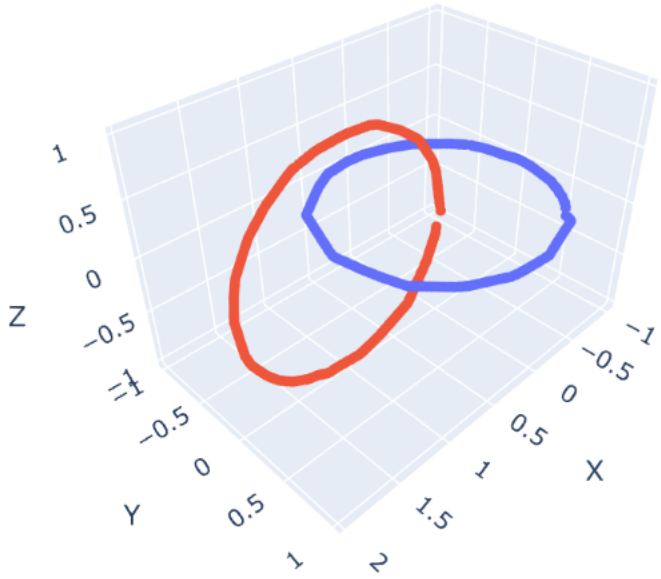

Figure 5: Another training run gave this output of the autoencoder for the two interlaced unit circles, one centered at $x = y = 0$ in the plane $z = 0$ (blue), and another centered at $x = 1$, $z = 0$ in the plane $y = 0$ (red). The network training algorithm automatically picked the points at which the circles should be "opened up" to avoid the topological obstruction.

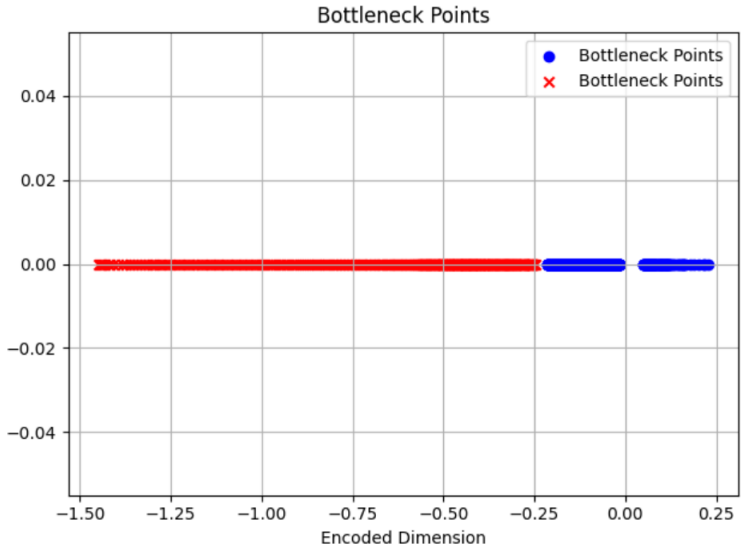

Figure 6: The bottleneck layer in another training run, showing the images of the blue and red circles. Note that the images were transposed compared to the first run.

Theorem 2. Let $k, n \in \mathbb{N}_{\geq 1}$ and $K \subset \mathbb{R}^n$ be a k-dimensional compact smoothly embedded submanifold without boundary. For any continuous functions $F\colon \mathbb{R}^n \to \mathbb{R}^k$ and $G\colon \mathbb{R}^k \to \mathbb{R}^n$,

$$\sup_{x \in K} \|G(F(x)) - x\| \geq r_K > 0. \tag{2}$$

Remark 2. The ability to make K_0 small in Theorem 1 relies on an autoencoder's ability to produce functions $G \circ F$ that change rapidly over small regions. E.g., if $G \circ F$ is Lipschitz then Theorem 2 implies a lower bound on the size of K_0 in terms of r_K and the Lipschitz constant.

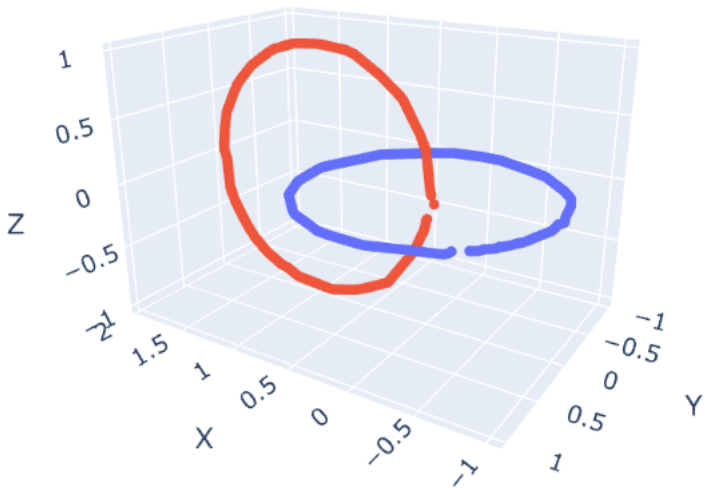

Figure 7: A different view of the data in Figure 5.

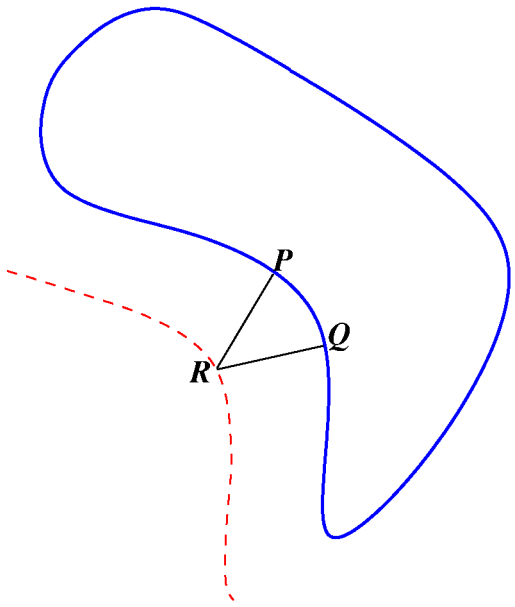

Figure 8: Illustration of reach. A one-dimensional manifold without boundary K in $\mathbb{R}^2$ is shown in blue. Two segments are drawn normal to K, starting at points P and Q in a non-convex high-curvature region. These segments intersect at a point R and have length r_K. If perturbations of P and Q lead to R, then there is no way to recover P and Q unambiguously as the unique point nearest to R. The dotted line represents points at distance r_K from K.

To prove Theorem 2 we instead prove the following more general Theorem 3, because the proof is the same. Here $\check{H}^k(S;\Gamma)$ denotes the k-th Čech cohomology of a topological space S with coefficients in an abelian group Γ (Dold, 1995, Def. VIII.6.1), (Bredon, 1997, p. 349). The statement implies Theorem 2 since $\check{H}^k(K;\Gamma) \neq 0$ when K is a compact orientable manifold and $\Gamma = \mathbb{Z}$, or when K is a compact nonorientable manifold and $\Gamma = \mathbb{Z}/2\mathbb{Z}$ (Dold, 1995, Ex. VIII.6.25), (Bredon, 1997, Cor. VI.8.4). Recall that r_K denotes the reach of $K \subset \mathbb{R}^n$.

Theorem 3. Let $k, n \in \mathbb{N}_{\geq 1}$ and $K \subset \mathbb{R}^n$ be a compact subset such that $\check{H}^k(K;\Gamma) \neq 0$ for some abelian group Γ. For any continuous functions $F\colon \mathbb{R}^n \to \mathbb{R}^k$ and $G\colon \mathbb{R}^k \to \mathbb{R}^n$,

$$\sup_{x\in K} \|G(F(x)) - x\| \geq r_K. \tag{3}$$

Proof. Since (3) holds automatically if $r_K = 0$, assume $r_K > 0$ and suppose, to obtain a contradiction, that the theorem does not hold. Then there are F, G as in its statement with

$$\sup_{x\in K} \|G(F(x)) - x\| < r_K,$$

which implies that

$$G(F(K)) \subset N_{r_K}(K) \coloneqq \{x \in \mathbb{R}^n \colon \operatorname{dist}(x, K) < r_K\}.$$

Since for each $x \in N_{r_K}(K)$ the optimization problem $\min_{y\in K} \operatorname{dist}(x, y)$ has a unique minimizer $y_* = \rho(x)$, $\rho\colon N_{r_K}(K) \to K$ is a continuous retraction ($\rho|_K = \mathrm{id}_K$). The line segment from $x \in K$ to $G(F(x))$ is contained in $N_{r_K}(K)$, since for $t \in [0, 1]$

$$\begin{aligned}\operatorname{dist}(tG(F(x)) + (1-t)x, K) &\leq \|tG(F(x)) + (1-t)x - x\| \\ &\leq \|G(F(x)) - x\| \\ &< r_K.\end{aligned}$$

Thus,

$$(t, x) \mapsto \rho\left(tG(F(x)) + (1-t)x\right)$$

defines a homotopy $[0, 1] \times K \to K$ from id_K to $(\rho \circ G \circ F)|_K\colon K \to K$, so by homotopy invariance (Dold, 1995, Prop. VIII.6.6) the induced map (Dold, 1995, Def. VIII.6.3)

$$(\rho \circ G \circ F)|_K^* = F^* \circ (G|_{F(K)})^* \circ \rho^* \colon \check{H}^k(K;\Gamma) \to \check{H}^k(K;\Gamma)$$

is equal to the identity map $(\mathrm{id}_K)^*$ induced by id_K. But this contradicts the fact that

$$(G|_{F(K)})^* \colon \check{H}^k(N_{r_K}(K)) \to \check{H}^k(F(K))$$

is the zero map, since $F(K)$ is compact and $\check{H}^k(C) = 0$ for any compact subset $C \subset \mathbb{R}^k$ (Bredon, 1997, Cor. VI.8.5). □

References

E Aamari, J Kim, F Chazal, B Michel, A Rinaldo, and L Wasserman. Estimating the reach of a manifold. *Electron. J. Stat.*, 13(1):1359–1399, 2019. ISSN 1935-7524. doi: 10.1214/19-ejs1551. URL `https://doi.org/10.1214/19-ejs1551`.

J. Batson, C G Haaf, Y Kahn, and D A Roberts. Topological obstructions to autoencoding. *Journal of High Energy Physics*, 2021(4):280, Apr 2021. ISSN 1029-8479. doi: 10.1007/JHEP04(2021)280. URL `https://doi.org/10.1007/JHEP04(2021)280`.

C Berenfeld, J Harvey, M Hoffmann, and K Shankar. Estimating the reach of a manifold via its convexity defect function. *Discrete Comput. Geom.*, 67(2):403–438, 2022. ISSN 0179-5376,1432-0444. doi: 10.1007/s00454-021-00290-8. URL `https://doi.org/10.1007/s00454-021-00290-8`.

G E Bredon. *Topology and geometry*, volume 139 of *Graduate Texts in Mathematics*. Springer-Verlag, New York, 1997. ISBN 0-387-97926-3. Corrected third printing of the 1993 original.

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.

A Dold. *Lectures on algebraic topology*. Classics in Mathematics. Springer-Verlag, Berlin, 1995. ISBN 3-540-58660-1. doi: 10.1007/978-3-642-67821-9. URL https://doi.org/10.1007/978-3-642-67821-9. Reprint of the 1972 edition.

H Federer. Curvature measures. *Trans. Amer. Math. Soc.*, 93:418–491, 1959. ISSN 0002-9947,1088-6850. doi: 10.2307/1993504. URL https://doi.org/10.2307/1993504.

C Fefferman, S Mitter, and H Narayanan. Testing the manifold hypothesis. *J. Amer. Math. Soc.*, 29(4): 983–1049, 2016. ISSN 0894-0347,1088-6834. doi: 10.1090/jams/852. URL https://doi.org/10.1090/jams/852.

C Fefferman, S Ivanov, Y Kurylev, M Lassas, and H Narayanan. Fitting a putative manifold to noisy data. In Sébastien Bubeck, Vianney Perchet, and Philippe Rigollet (eds.), *Proceedings of the 31st Conference On Learning Theory*, volume 75 of *Proceedings of Machine Learning Research*, pp. 688–720. PMLR, 06–09 Jul 2018. URL https://proceedings.mlr.press/v75/fefferman18a.html.

C Fefferman, S Ivanov, M Lassas, and H Narayanan. Fitting a manifold of large reach to noisy data. *Journal of Topology and Analysis*, 2023. doi: 10.1142/S1793525323500012.

R Hecht-Nielsen. Replicator neural networks for universal optimal source coding. *Science*, 269(5232):1860–1863, 1995. doi: 10.1126/science.269.5232.1860. URL https://www.science.org/doi/abs/10.1126/science.269.5232.1860.

M W Hirsch. *Differential topology*, volume 33 of *Graduate Texts in Mathematics*. Springer-Verlag, New York, 1994. ISBN 0-387-90148-5. Corrected reprint of the 1976 original.

J M Lee. *Introduction to smooth manifolds*, volume 218 of *Graduate Texts in Mathematics*. Springer, New York, second edition, 2013. ISBN 978-1-4419-9981-8.

P W Michor and C Vizman. n-transitivity of certain diffeomorphism groups. *Acta Math. Univ. Comenianae*, 63(2):221–225, 1994.

T Sakai. *Riemannian geometry*, volume 149 of *Translations of Mathematical Monographs*. American Mathematical Society, Providence, RI, 1996. ISBN 0-8218-0284-4. doi: 10.1090/mmono/149. URL https://doi.org/10.1090/mmono/149. Translated from the 1992 Japanese original by the author.

A Code used for implementation

```
import matplotlib.pyplot as plt
import pandas as pd
import numpy as np
import scipy as sp
import numpy as np
import tensorflow as tf
from tensorflow.keras.layers import Input, Dense
from tensorflow.keras.models import Model
import matplotlib.pyplot as plt
import plotly.graph_objects as go

# Define the parametric equations for the circles
def circle_xy(t, h, k, r):
    x = h + r * np.cos(t)
    y = k + r * np.sin(t)
    z = 0 * np.ones_like(t)
    return x, y, z

def circle_yz(t, h, k, r):
```

```
    x = h + r * np.sin(t)
    y = 0 * np.ones_like(t)
    z = k + r * np.cos(t)
    return x, y, z

howmany_points = 500
t = np.linspace(0, 2 * np.pi, howmany_points)

x1, y1, z1 = circle_xy(t, 0, 0, 1)
x2, y2, z2 = circle_yz(t, 1, 0, 1)

input_data = np.vstack((np.column_stack((x1, y1, z1)), np.column_stack((x2, y2, z2))))
# Build the autoencoder architecture with a bottleneck layer of dimension 1
input_dim = 3

# Encoder model
input_layer = Input(shape=(input_dim,))
encoded = Dense(128, activation='relu')(input_layer)
encoded = Dense(128, activation='relu')(encoded)
encoded = Dense(128, activation='relu')(encoded)
encoded = Dense(1, activation='linear')(encoded)  # Bottleneck layer with dimension 1
encoder = Model(inputs=input_layer, outputs=encoded)

# Decoder model
decoded_input = Input(shape=(1,))
decoded = Dense(128, activation='relu')(decoded_input)
decoded = Dense(128, activation='relu')(decoded)
decoded = Dense(128, activation='relu')(decoded)
decoded = Dense(input_dim, activation='linear')(decoded)
decoder = Model(inputs=decoded_input, outputs=decoded)

# Autoencoder model
autoencoder = Model(inputs=input_layer, outputs=decoder(encoder(input_layer)))

autoencoder.compile(optimizer='adam', loss='mean_squared_error')

# Train the autoencoder
epochs = 2000
batch_size = 20

autoencoder.fit(input_data, input_data, epochs=epochs, batch_size=batch_size, shuffle=True)

# Test the autoencoder
encoded_vectors = encoder.predict(input_data)
decoded_vectors = decoder.predict(encoded_vectors)

decoded_vectors_1 = decoded_vectors[0:howmany_points,:]
decoded_vectors_2 = decoded_vectors[-howmany_points:,:]
encoded_vectors_1 = encoded_vectors[0:howmany_points,:]
encoded_vectors_2 = encoded_vectors[-howmany_points:,:]

# Create the 3D plot of data vectors in plotly
fig1 = go.Figure()
# Add circles to the plot
fig1.add_trace(go.Scatter3d(x=x1, y=y1, z=z1, mode='lines', line=dict(width=8)))
fig1.add_trace(go.Scatter3d(x=x2, y=y2, z=z2, mode='lines', line=dict(width=8)))
# Setting the axis labels
fig1.update_layout(scene=dict(xaxis_title='X', yaxis_title='Y', zaxis_title='Z'))
fig1.show()
```

```
fig2 = go.Figure()
fig2.add_trace(go.Scatter3d(x=decoded_vectors_1[:, 0], y=decoded_vectors_1[:, 1],
    z=decoded_vectors_1[:,2], mode='markers', marker=dict(size=4)))
fig2.add_trace(go.Scatter3d(x=decoded_vectors_2[:, 0], y=decoded_vectors_2[:, 1],
    z=decoded_vectors_2[:,2], mode='markers', marker=dict(size=4)))
fig2.update_layout(scene=dict(xaxis_title='X', yaxis_title='Y', zaxis_title='Z'))
fig2.show()

# Plot the bottleneck points
plt.scatter(encoded_vectors_1, np.zeros_like(encoded_vectors_1), marker='o',
    label='Bottleneck Points', color='b')
plt.scatter(encoded_vectors_2, np.zeros_like(encoded_vectors_1), marker='x',
    label='Bottleneck Points', color='r')
plt.xlabel('Encoded Dimension')
plt.title('Bottleneck Points')
plt.legend()
plt.grid()

plt.tight_layout()
plt.show()
```