Electronic Journal of Statistics
Vol. 18 (2024) 1160–1184
ISSN: 1935-7524
https://doi.org/10.1214/24-EJS2227

Intrinsic and extrinsic deep learning on manifolds

Yihao Fang

Department of Applied and Computational Mathematics and Statistics, University of Notre Dame
e-mail: yfang5@alumni.nd.edu

Ilsang Ohn

Department of Statistics, Inha University
e-mail: ilsang.ohn@inha.ac.kr

Vijay Gupta

School of Electrical and Computer Engineering, Purdue University
e-mail: gupta869@purdue.edu

Lizhen Lin

Department of Mathematics, The University of Maryland
e-mail: lizhen01@umd.edu

Abstract: We propose extrinsic and intrinsic deep neural network architectures as general frameworks for deep learning on manifolds. Specifically, extrinsic deep neural networks (eDNNs) preserve geometric features on manifolds by utilizing an equivariant embedding from the manifold to its image in the Euclidean space. Moreover, intrinsic deep neural networks (iDNNs) incorporate the underlying intrinsic geometry of manifolds via exponential and log maps with respect to a Riemannian structure. Consequently, we prove that the empirical risk of the empirical risk minimizers (ERM) of eDNNs and iDNNs converge in optimal rates. Overall, The eDNNs framework is simple and easy to compute, while the iDNNs framework is accurate and fast converging. To demonstrate the utilities of our framework, various simulation studies, and real data analyses are presented with eDNNs and iDNNs.

Keywords and phrases: Manifolds, deep learning, eDNNs and iDNNs.

Received February 2023.

1. Introduction

The last two decades have witnessed an explosive development in deep learning approaches. These approaches have achieved breakthrough performance in a broad range of learning problems from a variety of applications field such as imaging recognition [32], speech recognition [17], natural language processing [2] and other areas of computer vision [45]. Deep learning has also served as the

main impetus for the advancement of recent artificial intelligence (AI) technologies. This unprecedented success has been made possible due to the increasing computational prowess, availability of large data sets, and the development of efficient computational algorithms for training deep neural networks. There have been increasing efforts to understand the theoretical foundations of deep neural networks, including in the statistics community [42, 28, 3, 25, 30, 12].

Most of these efforts from model and algorithmic development to theoretical understanding, however, have been largely focused on the Euclidean domains. In a wide range of problems arising in computer and machine vision, medical imaging, network science, recommender systems, computer graphics, and so on, one often encounters learning problems concerned with non-Euclidean data, particularly manifold-valued data. For example, in neuroscience, data collected in diffusion tensor imaging (DTI), now a powerful tool in neuroimaging for clinical trials, are represented by the diffusion matrices, which are 3×3 *positive definite matrices* [1]. In engineering and machine learning, pictures or images are often preprocessed or reduced to a collection of *subspaces* with each data point (an image) in the sample data represented by a subspace [18, 43]. In machine vision, a digital image can also be represented by a set of k-landmarks, the collection of which form *landmark-based shape spaces* [27]. One may also encounter data that are stored as *orthonormal frames* [10], *surfaces*, *curves*, and *networks* [31]. The underlying space where these general objects belong falls in the general category of *manifolds* whose geometry is generally well-characterized, which should be utilized and incorporated for learning and inference. Thus, there is a natural need and motivation for developing deep neural network models over manifolds.

This work aims to develop general deep neural network architectures on manifolds and take some steps toward understanding their theoretical foundations. The key challenge lies in incorporating the underlying geometry and structure of manifolds in designing deep neural networks. Although some recent works propose deep neural networks for specific manifolds [46, 16, 23, 24], there is a lack of general frameworks or paradigms that work for arbitrary manifolds. In addition, the theoretical understanding of deep neural networks on manifolds remains largely unexplored. To fill in these gaps, in this work, we make the following contributions: (1) we develop *extrinsic deep neural networks (eDNNs)* on manifolds to generalize the popular feedforward networks in the Euclidean space to manifolds via equivariant embeddings. The extrinsic framework is conceptually simple and computationally easy and works for general manifolds where nice embeddings such as *emquivariant embeddings* are available; (2) we develop *intrinsic deep neural networks (iDNNs)* for deep learning networks on manifolds employing a Riemannian structure of the manifold; (3) we study theoretical properties such as approximation properties and estimation error of both eDNNs and iDNNs, and (4) we implement various deep neural networks over a large class of manifolds under simulations and real datasets, including eDNNs, iDNNs and *tangential deep neural networks (tDNNs)*, which is a special case of iDNNs with only one tangent space.

The rest of the paper is organized as follows. In Section 2, we introduce the eDNNs on manifolds and study their theoretical properties. In Section 3, we

propose the iDNNs on manifolds that take into account the intrinsic geometry of the manifold. The simulation study and the real data analysis are carried out in Section 4. Our work ends with a discussion.

Notation For two real numbers $x, y \in \mathbb{R}$, we write $x \vee y = \max\{x, y\}$ and $x \wedge y = \min\{x, y\}$. For a real number $x \in \mathbb{R}$, we write $\lfloor x \rfloor = \max\{z \in \mathbb{Z} : z \leq x\}$, and $\lceil x \rceil = \min\{z \in \mathbb{Z} : z \geq x\}$. We write $a \lesssim b$ or $b \gtrsim a$ if there exists some universal constant $C > 0$ such that $a \leq Cb$, and write $a \asymp b$ if both $a \lesssim b$ and $a \gtrsim b$ hold. For two real-valued functions f_1 and f_2 defined on a set $\mathcal{X}$, we denote $\|f_1 - f_2\|_{L^\infty(\mathcal{X})} = \sup_{x \in \mathcal{X}} |f_1(x) - f_2(x)|$.

2. Extrinsic deep neural networks (eDNNs) on manifolds

2.1. eDNNs and equivariant embeddings

Let M be a d-dimensional manifold. Let (x_i, y_i), $i = 1, \ldots, n$ be a sample of data from some regression model with input $x_i \in M$ and output $y_i \in \mathbb{R}$, and we propose deep neural networks for learning the underlying function $f : M \to \mathbb{R}$. The output space can be $\{1, \ldots, m\}$ for a classification problem. In this work, we propose to develop two general deep neural network architectures on manifolds based on an extrinsic and an intrinsic framework, respectively. The first framework employs an equivariant embedding of a manifold into the Euclidean space and builds a deep neural network on its image after embedding, which is the focus of this section, while the intrinsic framework utilizes Riemannian or intrinsic geometry of the manifold for designing the deep neural networks (Section 3). Our initial focus will be on proposing appropriate analogs of feed-forward neural networks on manifolds which are popular deep neural networks in the Euclidean space and suitable objects for theoretical analysis. The theoretical properties of the proposed geometric deep neural networks will be studied.

Before describing our proposed frameworks, we introduce our mathematical definition of deep neural networks and related classes. A deep neural network $\tilde{f}$ with depth L and a width vector $(p_0, \cdots, p_{L+1}) \in \mathbb{N}^{L+2}$ is a function of the form

$$\tilde{f}(\tilde{x}) := A_{L+1} \circ \sigma_L \circ A_L \circ \cdots \circ \sigma_1 \circ A_1(\tilde{x}), \tag{1}$$

where $A_l : \mathbb{R}^{p_{l-1}} \to \mathbb{R}^{p_l}$ is an affine linear map defined by $A_l(\tilde{x}) = W_l \tilde{x} + b_l$ for $p_l \times p_{l-1}$ weight matrix W_l and p_l dimensional bias vector b_l, and $\sigma_l : \mathbb{R}^{p_l} \to \mathbb{R}^{p_l}$ is an element-wise nonlinear activation map with the ReLU activation function $\sigma(z) = \max\{0, z\}$ as a popular choice. We referred to the maximum value $\max_{j=1,\ldots,L} p_j$ of the width vector as the width of the deep neural network. For $\theta = \big((W_1, b_1), \ldots, (W_{L+1}, b_{L+1})\big)$, the collection of all weight matrices and bias vectors, we denote by $\|\theta\|_0$ the number of non-zero parameter values (i.e., the sparsity) and by $\|\theta\|_\infty$ the maximum of parameters. We denote by $\mathcal{F}(L, (p_0 \sim P \sim p_{L+1}), S, B)$ the class of deep neural networks with depth L, input dimension p_0, width P, output dimension p_{L+1}, sparsity S and the maximum of

parameters B. For simplicity, if the input and output dimensions are clear in the context, we write $\mathcal{F}(L, P, S, B) = \mathcal{F}(L, (p_0 \sim P \sim p_{L+1}), S, B)$.

Let $J : M \to \mathbb{R}^D$ be an embedding of M into some higher dimensional Euclidean space $\mathbb{R}^D$ ($D \geq d$) and denote the image of the embedding as $\tilde{M} = J(M)$. By definition of an embedding, J is a smooth map such that its differential $dJ : T_xM \to T_{J(x)}\mathbb{R}^D$ at each point $x \in M$ is an injective map from its tangent space T_xM to $T_{J(x)}\mathbb{R}^D$, and J is a homeomorphism between M and its image $\tilde{M}$. Our idea of building an *extrinsic deep neural network (eDNN)* on manifold relies on building a deep neural network on the image of the manifold after the embedding. The geometry of the manifold of M can be well-preserved with a good choice of embedding, such as an equivariant embedding which will be defined rigorously in Remark 2.2 below. The extrinsic framework has been adopted for the estimation of Fréchet means [5], regression on manifolds [34], and construction of Gaussian processes on manifolds [33], which have enjoyed some notable features such as ease of computations and accurate estimations.

The key idea of proposing an extrinsic feedforward neural network on a manifold M is to build a one-to-one version of its image after the embedding. More specially, we say that f is an eDNN if f is of the form

$$f(x) = \tilde{f}(J(x)), \tag{2}$$

with a deep neural network $\tilde{f}$. We define the eDNN class induced by $\mathcal{F}(L, P, S, B)$ as

$$\mathcal{F}_{eDNN}(L, P, S, B) = \{f = \tilde{f} \circ J : \tilde{f} \in \mathcal{F}(L, P, S, B)\}$$

The extrinsic framework is very general and works for any manifold where a good embedding, such as an equivariant embedding, is available. Under this framework, training algorithms in the Euclidean space, such as the stochastic gradient descent (SGD) with backpropagation algorithms, can be utilized working with the data $(J(x_i), y_i)$, $i = 1, \ldots, n$, with the only additional computation burden potentially induced from working higher-dimensional ambiance space. In our simulation Section 4, the extrinsic deep neural network yields better accuracy than the Naive Bayes classifier, kernel SVM, logistic regression classifier, and the random forester classifier for the planar shape datasets. Due to its simplicity and generality, there is a potential for applying eDNNs in medical imaging and machine vision for broader scientific impacts.

Remark 2.1. *In [41] and [7], a feedforward neural network was used for nonparametric regression on a lower-dimensional submanifold embedded in some higher-dimensional ambient space. It showed that with appropriate conditions on the neural network structures, the convergence rates of the ERM would depend on the dimension of the submanifold d instead of the dimension of the ambient space D. In their framework, they assume the geometry of the submanifold is unknown. From a conceptual point of view, our extrinsic framework can be viewed as a special case of theirs by ignoring the underlying geometry. In this case, the image of the manifold $\tilde{M} = J(M)$ can be viewed as a submanifold in*

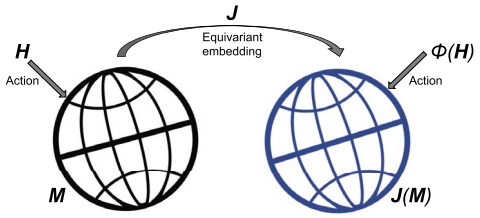


FIG 1. *An simple illustration of equivariant embeddings*

$\mathbb{R}^D$, *so their results follow. On the other hand, our embedding framework allows us to work with very complicated manifolds, such as the quotient manifolds for which no natural ambient coordinates are available. An example is the planar shape which is the quotient of a typically high-dimensional sphere consisting of orbits of equivalent classes, with the submanifold structure only arising after the embedding. And such an embedding is typically not isometric.*

In [7], the charts were constructed by intersecting small balls in $\mathbb{R}^D$ *with the submanifold* M. *In our case, we provide explicit charts of the submanifold based on the knowledge of the geometry of the original manifold* M *and the embedding map* J *that works with the ambient coordinates in* $\mathbb{R}^D$.

Remark 2.2. *One of the essential steps in employing an eDNN is the choice of the embedding* J, *which is generally not unique. It is desirable to have an embedding that preserves as much geometry as possible. An equivariant embedding is one type of embedding that preserves a substantial amount of geometry. Figure 1 provides a visual illustration of equivariant embedding. Suppose* M *admits an action of a (usually 'large') Lie group* H. *Then we say that* J *is an equivariant embedding if we can find a Lie group homomorphism* $\phi : H \to GL(D, \mathbb{R})$ *from* H *to the general linear group* $GL(D, \mathbb{R})$ *of degree* D *acting on* $\tilde{M}$ *such that*

$$J(hp) = \phi(h)J(p)$$

for any $h \in H$ *and* $p \in M$. *For example, in many cases* H *can be an isometry group, which is a Lie group for Riemannian symmetric spaces. In this case, the embedding preserves a lot of symmetries of the underlying manifold. The definition seems technical at first sight. However, the intuition is clear. If a large group* H *acts on manifolds such as by rotation before embedding, such an action can be preserved via* ϕ *on the image* $\tilde{M}$, *thus potentially preserving many of the geometric features of* M, *such as its symmetries. Therefore, the embedding is geometry-preserving in this sense. For the case of the planar shape, which is a collection of shapes consisting of* k*-landmarks modular Euclidean motions such as rotation, scaling, and translation, which is a quotient manifold of a sphere of dimension* S^{2k-3}, *and the embedding can be given by the Veronese-whitnning embedding which is equivariant under the special unitary group. Another example that's less abstract to understand is the manifold of symmetric positive definite*

matrices whose embedding can be given as the log *map (the matrix* log *function) into the space of symmetric matrices, and this embedding is equivariant with respect to the group action of the general linear group via the conjugation group action. See Section 4 for some concrete examples of equivariant embeddings for well-known manifolds, such as the space of the sphere, symmetric positive definite matrices, and planar shapes.*

2.2. Approximation analysis for eDNNs

In this section, we study the ability of the eDNN class in approximating an appropriate smooth class of functions on manifolds. First, we define the ball of β-Hölder functions on a set $U \in \mathbb{R}^D$ with radius $A > 0$ as

$$\mathcal{C}_D^\beta(U, A) = \{f : \|f\|_{\mathcal{C}_D^\beta(U)} \leq A\},$$

where $\|\cdot\|_{\mathcal{C}_D^\beta(U)}$ denotes the β-Hölder norm defined as

$$\|f\|_{\mathcal{C}_D^\beta(U)} = \sum_{m \in \mathbb{N}_0^D : \|m\|_1 \leq \lfloor\beta\rfloor} \|\partial^m f\|_\infty + \sum_{m \in \mathbb{N}_0^D : \|m\|_1 = \lfloor\beta\rfloor} \sup_{x_1, x_2 \in U, x_1 \neq x_2} \frac{|\partial^m f(x_1) - \partial^m f(x_2)|}{\|x_1 - x_2\|_\infty^{\beta - \lfloor\beta\rfloor}}.$$

Here, $\partial^m f$ denotes the partial derivative of f of order m and $\mathbb{N}_0 = \mathbb{N} \cup \{0\}$. Let $\mathcal{C}_D^\beta(U) = \cup_{A>0} \mathcal{C}_D^\beta(U, A)$. To facilitate smooth function approximation on manifolds, following [41], we impose an additional smooth assumption on local coordinates which project inputs in an ambient space to a lower dimensional space.

Definition 1. *We say that a compact d-dimensional manifold $M \subset \mathbb{R}^D$ has smooth local coordinates if there exist charts $(V_1, \psi_1), \ldots, (V_K, \psi_K)$, such that for any $\gamma > 0$, $\psi_k \in \mathcal{C}_D^\gamma(V_k)$ and $\psi_k^{-1} \in \mathcal{C}_d^\gamma(\psi_k(V_k))$ for every $k = 1, \ldots, K$.*

The next theorem reveals the approximation ability of the eDNN architecture.

Theorem 1. *Let $M \subset \mathbb{R}^D$ be a d-dimensional compact manifold and $J : M \to \mathbb{R}^D$ be an embedding map. Assume that $J(M)$ has smooth local coordinates. Then there exist positive constants $C_1, \ldots, C_6$ depending on none of D, β and A such that for any $N \in \mathbb{N} \setminus \{1\}$,*

$$\sup_{f_0 : f_0 \circ J^{-1} \in \mathcal{C}_D^\beta(J(M), A)} \inf_{f \in \mathcal{F}_{eDNN}(L,P,S,B)} \|f - f_0\|_{L^\infty(M)} \leq C_6 A (1 + (\beta \vee 1) D/d)^2 D^{3(\beta \vee 1) D/(2d)} N^{-2\beta/d}$$

with $L = C_1(\beta \vee 1)D/d + 1)^2 + 2D$, $P = C_2((\beta \vee 1)D/d)D^{(\beta \vee 1)D/d+1}N$, $S = C_3((\beta \vee 1)D/d+1)^4 D^{2(\beta \vee 1)D/d+3} N^2$ and $B = C_4((\beta \vee 1)D^2/d)DN^{C_5((\beta \vee 1)/d+1)}$.

In the above theorem, the sparsity is of order $O(N^2)$, so we can write that the approximation error is bounded by $O(S^{-\beta/d})$.

Remark 2.3. *Using the improved mathematical techniques used in a series of papers [25, 12, 35, 38], we do not require the network depth to increase as the desired approximation error goes to 0, unlike [6, 7] and [41].*

2.3. Statistical risk analysis for eDNNs

In this section, we study the statistical risk of the empirical risk minimizer (ERM) based on the eDNN class. We assume the following regression model

$$y_i = f_0(x_i) + \epsilon_i \tag{3}$$

for $i = 1, \ldots, n$, where $x_1, \ldots, x_n \in M$ are i.i.d inputs following a distribution P_x on the manifold and $\epsilon_1, \ldots, \epsilon_n$ are i.i.d. sub-Gaussian errors with mean zero. We consider the ERM over the eDNN class such that

$$\hat{f}_{eDNN} = \underset{f \in \mathcal{F}_{eDNN}(L,P,S,B)}{\operatorname{argmin}} \frac{1}{n} \sum_{i=1}^{n} (y_i - f(x_i))^2. \tag{4}$$

A natural question to ask is whether the ERM type of estimators such as $\hat{f}_n$ defined above achieve minimax optimal estimation of β-Hölder smooth functions on manifolds, in terms of the excess risk

$$R(\hat{f}_{eDNN}, f_0) = E(\hat{f}_{eDNN}(x) - f_0(x))^2$$

where the expectation is taken over the random variable $x \sim P_x$.

Theorem 2. *Assume the model (3) with a d-dimensional compact manifold $M \subset \mathbb{R}^D$ and an embedding map $J : M \to \mathbb{R}^D$. Moreover, assume that $J(M)$ has smooth local coordinates. Then there exist positive constants $C_1, \ldots, C_4$ and $a > 1$ such that the ERM estimator $\hat{f}_{eDNN}$ over the eDNN class $\mathcal{F}_{eDNN}(L, P, S, B)$ in (4) with $L \geq C_1$, $n^a \geq P \geq C_2(n/\log n)^{d/(4\beta+2d)}$, $S = C_3(n/\log n)^{d/(2\beta+d)}$ and $n^a \geq B \geq C_4 n^{(\beta \vee 1)/(4\beta+2d)+1}$ satisfies*

$$\sup_{f_0 : f_0 \circ J^{-1} \in \mathcal{C}_D^\beta(J(M), A)} R(\hat{f}_{eDNN}, f_0) \lesssim \left(\frac{\log n}{n}\right)^{\frac{2\beta}{2\beta+d}}$$

Remark 2.4. *In the above there, we improve the logarithmic factor $\log^3 n$ of the rates of [41] and [7] to $\log^{2\beta/(2\beta+d)} n$. This improvement is from our approximation analysis with constant depth $L \asymp 1$, while those in [41] and [7] require $L \asymp \log n$.*

The following two remarks discuss possible extensions of our result.

Remark 2.5. *Our result can be easily extended to a true function f_0 with a hierarchical composition structure which was considered in [42, 30, 12] by using a function approximation result to such a function by neural networks [e.g., Theorem 4 of 12] to each $f_0 \circ \psi_k^{-1}$, the component on the k-th coordinate. With this analysis, we can get a faster convergence rate. For instance, if every $f_0 \circ \psi_k^{-1}$ has an additive structure, i.e., represented as a sum of univariate functions on $\psi_k(V_k)$, the convergence rate becomes $n^{-2\beta/(2\beta+1)}$ up to a logarithmic factor.*

Remark 2.6. *To get the optimal rate, the sparsity of the eDNN estimator in Theorem 2 should be selected in a non-adaptive manner in the sense that it depends on the smoothness β and the dimension d of the manifold, but they are unknown in practice. We may be required to use a data-adaptive method to overcome this limitation. One simple method is to introduce a sparsity-inducing penalty function such as the clipped L_1 penalty, which was proposed by [39] for deep learning and after that used in [12]. In these studies, a penalized deep neural network estimator with the clipped L_1 penalty can attain the optimal convergence rate without knowing any aspect of a true function. We believe that applying a penalization method to our eDNN architecture can lead to adaptive estimation to β and d, but we leave the detailed derivation to future work.*

3. Intrinsic deep neural networks (iDNNs) on manifolds

3.1. The iDNN architectures on a Riemannian manifold

Despite the generality and computational advantage enjoyed by eDNNs on manifolds proposed in the previous section, one potential drawback is that an embedding is not always available on complex manifolds such as some intrinsic structure spatial domains. In this section, we propose a class of *intrinsic deep neural networks (iDNNs)* on manifolds by employing the intrinsic geometry of a manifold to utilize its exponential and log maps with respect to a Riemannian structure. Some works construct a deep neural network on the manifold via mapping the points on the manifold to a *single tangent space* (e.g., with respect to some central points of the data) or proposing deep neural networks on specific manifolds, in particular, matrix manifolds [21, 16]. Using a deep neural network on a single tangent space approximation cannot provide a good approximation of a function on the whole manifold, unless when the manifold can be represented by a global chart [8]. Below we provide a rigorous framework for providing a local approximation of a function on a Riemannian manifold via Riemannian exponential and logarithm maps and thoroughly investigate their theoretical properties.

The key ideas here are to first cover the manifold with images of the subset of tangent spaces $U_1, \ldots, U_K$ under the exponential map, approximate a local function over the tangent space using deep neural networks, which are then patched together via the transition map and a partition of unity on the Riemannian manifold. Specifically, let $\{x_1, \ldots, x_K \in M\}$ be a finite set of points, such that for an open set of subsets $U_k \subset T_{x_k}M$ with $k = 1, \ldots, K$, one has $\bigcup_{k=1}^{K} \exp_{x_k}(U_k) = M$. Namely, one has $\big\{\big(\exp_{x_k}(U_k), \exp_{x_k}\big) : k = 1 \ldots, K\big\}$ as the charts of the manifold M.

For each $k = 1, \ldots, K$ one has orthonormal basis $v_{k1}, \ldots, v_{kd} \in T_{x_k}M$ and respectively the normal coordinates of $x \in \exp_{x_k}(U_k)$

$$v_k^j(x) = \big\langle \log_{x_k} x, v_{kj} \big\rangle \quad \text{for} \quad j = 1, \ldots, d.$$

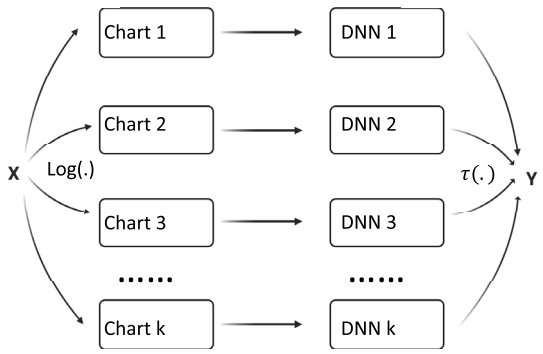


FIG 2. *The iDNN architecture on a Riemannian manifold M. Given the base points $x_1, \ldots, x_K \in M$ and the charts $U_k \subset T_{x_k}M$ for $k = 1 \ldots, K$ on the manifold M, the input data X is mapped to the kth chart U_k after the log map $\log_{x_k}(.)$. Afterward, the transformed data is fed into the deep neural network f_k on each chart k. The final prediction Y is given by the partition of unity $\tau(.)$ as $Y = \sum_{k=1}^{K} \tau_k(x) f_k \left(\log_{x_k}(x)\right)$.*

Thus

$$v_k(x) = \left(v_k^1(x), \ldots, v_k^d(x)\right) = \sum_{j=1}^{d} v_k^j(x) v_{kj} \in T_{x_k}M.$$

The normal coordinate allows one to perform elementwise non-linear activation to tangent vectors easily. For example, any $1 \leq k < l \leq K$ one has the transition map on $\exp_{x_l}(U_l) \cap \exp_{x_k}(U_k)$

$$v_k^j(x) = \left\langle \log_{x_k} x, v_{kj} \right\rangle = \left\langle \log_{x_k} \exp_{x_l} v_l(x), v_{kj} \right\rangle \quad \text{for} \quad j = 1, \ldots, d.$$

A compact manifold M always admits a *finite partition of unity* $\left\{\tau_k : M \to \mathbb{R}_+ : k = 1, \ldots, K\right\}$ such that $\sum_{k=1}^{K} \tau_k(x) = 1$, and for every $x \in M$ there is a neighbourhood of x where all but a finite number of functions are 0 (e.g., Proposition 13.9 of [44]). Therefore, for each function $f : M \to \mathbb{R}$, we can write

$$f(x) = \sum_{k=1}^{K} \tau_k(x) f\Big(\exp_{x_k}\left(\log_{x_k} x\right)\Big) = \sum_{k=1}^{K} \tau_k(x) f_k(\log_{x_k}(x)) \tag{5}$$

with $f_k = f \circ \exp_{x_k} : U_k \to \mathbb{R}$. As a result, one can model the compositions $f_k = f \circ \exp_{x_k}$ instead of f, for which we propose to use deep neural networks. This idea gives rise to our iDNN architecture $f(x) = \sum_{k=1}^{K} \tau_k(x) f_k \left(\log_{x_k}(x)\right)$. Figure 2 illustrates the core ideas of the iDNN architecture. Given a set of points $\{x_1, \ldots, x_K\} \subset M$, we define the iDNN class with depth L, width P, sparsity

S and the maximum of parameters B as

$$\mathcal{F}_{iDNN}(L,P,S,B)=\left\{\sum_{k=1}^{K}\tau_k(x)f_k\left(\log_{x_k}(x)\right):f_k\in\mathcal{F}(L,(d\sim P\sim 1),S,B)\right\}. \tag{6}$$

3.2. Approximation analysis for iDNNs

In this section, we investigate the approximation theory for the iDNN for smooth functions on manifolds.

Theorem 3. *Let $M\subset\mathbb{R}^D$ be a d-dimensional compact manifold. Assume that $\exp_{x_k}\in\mathcal{C}_D^{\gamma}(U_k)$ for $\gamma>\beta$ for every $k=1,\dots,K$. Then there exist positive constants $C_1,\dots,C_6$ depending on none of D, β and A such that for any $N\in\mathbb{N}\setminus\{1\}$,*

$$\sup_{f_0\in\mathcal{C}_D^{\beta}(M,A)}\inf_{f\in\mathcal{F}_{iDNN}(L,P,S,B)}\|f-f_0\|_{L^\infty(M)}\le C_6A(\beta+1)^2d^{3(\beta\vee1)/2}N^{-2\beta/d}$$

with $L=C_1\{(\beta+1)^2+2d\}$, $P=C_2(\beta\vee1)d^{\beta+1}N$, $S=C_3(\beta\vee1)^2\{(\beta+1)^2+2d\}^2d^{2\beta+3}N^2$ and $B=C_4(\beta\vee1)dN^{C_5((\beta\vee1)/d+1)}$

Remark 3.1. *Several existing works [41, 6, 7, 25] propose feedforward neural networks on a manifold that's embedded in a higher-dimensional Euclidean space. They utilize local charts and partition of unities, but due to the unknown geometry of the manifold, they need to use deep neural networks to approximate the local chart ψ_k, the partition of unity function τ_k as well as the mapping $f_0\circ\psi_k^{-1}$ for all $k=1,\dots,K$. Under our iDNN framework, we utilize the Riemannian geometry of the manifold and the* log *map. Further, the partition of unity functions can be constructed so there is no need to approximate them with deep neural networks. From a theoretical point of view, this gives a smaller prefactor of order $O((\beta+1)^2d^{3(\beta\vee1)/2})$ of the approximation error than the state-or-art one of $O(D^{1/2}(\beta+1)^2d^{3\beta/2+1}))$ established in Theorem 6.2 of [25]. Note that our prefactor has no dependence on the input dimension D.*

Remark 3.2. *Our result can be extended to an unknown partition of the unity function. In this case, τ_k can be modeled by a deep neural network. If we assume that τ_k is sufficiently smooth, then τ_k can be well approximated by a deep neural network, which leads to a similar approximation error bound.*

Remark 3.3. *In our theorems, we have assumed that the underlying Riemannian manifold is compact. The assumption on compactness is to ensure the existence of a nice (finite) partition of unity functions, which are feasible for practical algorithms. Compactness guarantees the existence of a partition of unity in which for each point, only a finite number of partition of unity functions are*

non-zero, thus feasible for practical computations. Technically speaking, our theory also works for non-compact manifolds, for example, for manifolds that are locally compact.

3.3. Statistical risk analysis for iDNNs

In this section, we study the statistical risk of the ERM over the iDNN class given by

$$\hat{f}_{iDNN} = \underset{f \in \mathcal{F}_{iDNN}(L,P,S,B)}{\operatorname{argmin}} \frac{1}{n} \sum_{i=1}^{n} (y_i - f(x_i))^2. \tag{7}$$

for the nonparametric regression model (3) where the true function f_0 is β-Hölder smooth on a manifold. The following theorem shows that the iDNN estimator attains the optimal rate.

Theorem 4. *Assume the model (3) with a d-dimensional compact manifold M isometrically embedded in $\mathbb{R}^D$. Then there exist positive constants $C_1, \ldots, C_4$ and $a > 1$ such that the ERM estimator $\hat{f}_{iDNN}$ over the iDNN class $\mathcal{F}_{iDNN}(L, P, S, B)$ in (7) with $L \geq C_1$, $n^a \geq P \geq C_2(n/\log n)^{d/(4\beta+2d)}$, $S = C_3(n/\log n)^{d/(2\beta+d)}$ and $n^a \geq B \geq C_4 n^{(\beta \vee 1)/(4\beta+2d)+1}$ satisfies*

$$\sup_{f_0 \in \mathcal{C}_D^\beta(M,A)} R(\hat{f}_{iDNN}, f_0) \lesssim \left(\frac{\log n}{n}\right)^{\frac{2\beta}{2\beta+d}}.$$

The extensions in Remarks 2.5 and 2.6 can be done also for the iDNN architecture.

4. Simulations study and real data analysis

Applications will illustrate the practical impact and utilities of our methods to simulated data sets and some important real data sets, such as in the context of the AFEW database, HDM95 database, the ADHD-200 dataset, an HIV study, and others. The proposed eDNNs, tDNNs, and iDNNs will be applied to learning problems such as regression and classification on various manifolds, including the sphere, the planar shapes, and the manifold of symmetric positive definite matrices, which are the most popular classes of manifolds encountered in medical diagnostics using medical imaging and image classification in digital imaging analysis. For the eDNN models, we list explicit embeddings below and the corresponding lie groups that act on them equivariantly. For the iDNN models, we elaborate on the exponential map and inverse-exponential (log) map on those manifolds. As mentioned before, the tDNN model is the special case of the iDNN model when $K = 1$, which utilizes the exponential map and inverse-exponential map as well.

4.1. Sphere

One of the simplest manifolds of interest is the sphere in particular in directional statistics and spatial statistics [14, 36, 13, 26, 20]. Statistical analysis of data from the two-dimensional sphere $\mathbb{S}^2$, often called directional statistics, has a fairly long history [14, 36, 13]. Modeling data on the sphere has also received recent attention due to applications in spatial statistics, for example, global models for climate or satellite data [26, 20].

To build the eDNN on the sphere, first note that $\mathbb{S}^d$ is a submanifold of $\mathbb{R}^{d+1}$, so that the inclusion map J serves as a natural embedding of $\mathbb{S}^d$ into $\mathbb{R}^{d+1}$. It is easy to check that J is an equivariant embedding with respect to the Lie group $H = SO(d+1)$, the group of $d+1$ by $d+1$ special orthogonal matrices. Intuitively speaking, this embedding preserves a lot of symmetry of the sphere. On the other hand, one can use the geodesics (in this case, the big circles on the sphere) for which the closed-form exponential map and inverse-exponential map are available to construct the iDNN model. Furthermore, given the base points x_k for $k = 1, ..., K$, one has $\tau(x) = \exp(-\frac{1}{1-\|x-x_k\|^2})$ by utilizing the bump function on the sphere.

In this simulation study, we consider the classification problem where Von Mises-Fisher distribution (MF) on the sphere $\mathbb{S}^2$ is considered, which has the following density:

$$f_{\mathrm{MF}}(y; \mu, \kappa) \propto \exp\left(\kappa \mu^T y\right), \tag{8}$$

where κ is a concentration parameter with μ a location parameter. Then we simulate the data from m different classes on the sphere $\mathbb{S}^d$ via a mixture of MF distributions as:

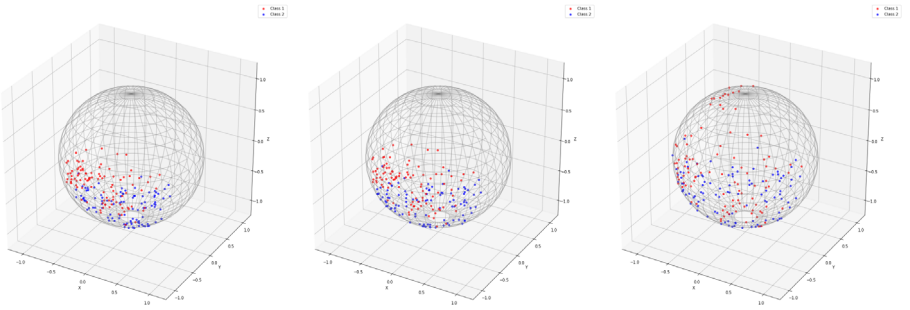
$$u_{j1}, ..., u_{j10} \sim \mathrm{MF}(\mu_j, \kappa_1), \quad j = 1, \cdots, m. \tag{9}$$

$$m_{ij} \sim \mathrm{Unif}(\{u_{j1}, ..., u_{j10}\}), \quad i = 1, \cdots, n,\ j = 1, \cdots, m \tag{10}$$

$$x_{ij} \sim \mathrm{MF}(m_{ij}, \kappa_2), \quad i = 1, \cdots, n, j = 1, \cdots, m, \tag{11}$$

where x_{ij} is the ith sample from jth class, μ_j is the mean for the jth class, and κ_2 is the dispersion for all classes. We first generated 10 means $u_{j1}, ..., u_{j10}$ from the MF distribution for jth class. Then for each class, we generated n observations as follows: for each observation x_{ij}, we randomly picked m_{ij} from $u_{j1}, ..., u_{j10}$ with probability 1/10, and then generated a observation from $\mathrm{MF}(m_{ij}, \kappa_2)$, thus leading to a mixture of MF distributions. Moreover, κ_1 controls the dispersion of the intermediate variable m_{ij} while κ_2 controls the dispersion of observations x_{ij}. Figure 3 shows observations from the mixture model on the sphere under different dispersions.

In the following simulation, we follow the mixture model on the hyper-sphere $\mathbb{S}^2$, $\mathbb{S}^{10}$, $\mathbb{S}^{50}$ with $m = 2$, $n = 2000$, $\kappa_1 = 4$, $\kappa_2 = 20$ and divide the data into 75 percent training set and 25 percent test set. We repeat this split 50 times. Then we compare the eDNN, tDNN, iDNN models to other competing estimators via the classification accuracy on the test set as shown in Table 1.



(a) $\kappa_1 = 10, \kappa_2 = 50$ (b) $\kappa_1 = 8, \kappa_2 = 40$ (c) $\kappa_1 = 4, \kappa_2 = 20$

FIG 3. *Observations for $K = 2$ classes from the mixture of* MF *distribution, $N = 100$. The nonlinear boundary between the two classes becomes harder to see with bare eyes due to the surging variance of the data as the κ_1, κ_2 drop, which makes the classification problem harder.*

TABLE 1

The test accuracy is calculated over 50 *random split. The* 5*-layers network (with* 100 *hidden nodes in each layer) is used for our deep neural network models in all experiments. Our tDNN model achieved the best result when the dimension was low $\mathbb{S}^2$, $\mathbb{S}^{10}$, while our iDNN is the best in high-dimension cases ($\mathbb{S}^{50}, \mathbb{S}^{100}$). Moreover, our tDNN, iDNN models show better accuracy than the classical deep neural network, especially in high-dimensional cases.*

	$\mathbb{S}^2$	$\mathbb{S}^{10}$	$\mathbb{S}^{50}$	$\mathbb{S}^{100}$
DNN	94.12 ± 0.67	96.22 ± 0.63	75.93 ± 1.07	62.53 ± 1.35
tDNN	$\mathbf{94.88 \pm 0.53}$	$\mathbf{97.13 \pm 0.39}$	80.07 ± 0.95	68.26 ± 1.16
iDNN	94.69 ± 0.65	97.11 ± 0.41	$\mathbf{80.72 \pm 0.94}$	$\mathbf{68.43 \pm 1.20}$
kNN	92.16 ± 0.77	94.98 ± 0.60	69.18 ± 1.44	56.24 ± 1.30
LR	92.98 ± 0.76	88.64 ± 0.76	72.38 ± 1.14	66.73 ± 1.37
RF	93.66 ± 0.83	89.93 ± 0.65	70.29 ± 1.48	62.29 ± 1.45
SVM	94.07 ± 0.1	96.85 ± 0.44	79.38 ± 1.15	68.25 ± 1.18

For competitors, we consider the k-nearest neighbors (kNN), the random forest (RF), the logistic regression (LR), and the support vector machine (SVM) with the radial basis function (RBF) kernel. The tuning parameters in each method are selected by evaluation on a validation data set whose size is 25% of the training set.

For all deep neural network models, we apply a network architecture of 5 hidden layers with the numbers of widths $(100, 100, 100, 100, 100)$. The deep neural network model is the same as the eDNN model on Euclidean since the embedding map from the sphere to the higher Euclidean space is the identity map. In the tDNN model, we consider the Fréchet mean of the training set as the base point and transform all data in the batch to tangent vectors before feeding to the neural network. In the iDNN model, we consider the north and south poles $(\pm 1, 0, .., 0)$ as base points and use the neural network with the same structure for all tangent spaces. All models are trained with Adam optimizer [29]. As shown in Table 1, our tDNN model and iDNN model outperform other competing estimators. Specifically, our tDNN models achieve the best accuracy 94.88 ± 0.53 and 97.13 ± 0.39 in the low dimensional cases. Our iDNN models

obtained the best result 80.72 ± 0.94 and 68.43 ± 1.20 in the high dimensional spaces.

4.2. The planar shape

Let $z = (z_1, \ldots, z_k)$, with $z_1, \ldots, z_k \in \mathbb{R}^2$, be a set of k landmarks. The planar shape Σ_2^k is the collection of z's modulo under the Euclidean motions, including translation, scaling, and rotation. One has $\Sigma_2^k = \mathbb{S}^{2k-3}/SO(2)$, the quotient of sphere by the action of $SO(2)$ (or the rotation), the group of 2×2 special orthogonal matrices. A point in Σ_2^k can be identified as the orbit of some $u \in \mathbb{S}^{2k-3}$, which we denote as $\sigma(z)$. Viewing z as a vector of complex numbers, one can embed Σ_2^k into $S(k, \mathbb{C})$, the space of $k \times k$ complex Hermitian matrices, via the Veronese-Whitney embedding (see, e.g., [4]):

$$J(\sigma(z)) = uu^* = ((u_i \bar{u}_j))_{1 \leq, i, j \leq k}. \tag{12}$$

One can verify that J is equivariant (see [27]) with respect to the Lie group

$$H = SU(k) = \{A \in GL(k, \mathbb{C}) : AA^* = I, \det(A) = I\},$$

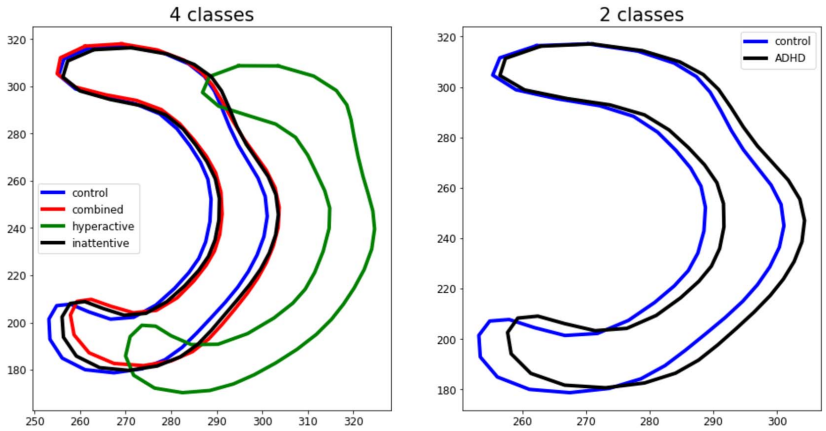
with its action on Σ_2^k induced by left multiplication.

We consider a planar shape data set, which involves measurements of a group of typically developing children and a group of children suffering the ADHD (Attention deficit hyperactivity disorder). ADHD is one of the most common psychiatric disorders for children that can continue through adolescence and adulthood. Symptoms include difficulty staying focused and paying attention, difficulty controlling behavior, and hyperactivity (over-activity). In general, ADHD has three subtypes: (1) ADHD hyperactive-impulsive, (2) ADHD-inattentive, (3) Combined hyperactive-impulsive and inattentive (ADHD-combined). ADHD-200 Dataset (http://fcon_1000.projects.nitrc.org/indi/adhd200/) is a data set that records both anatomical and resting-state functional MRI data of 776 labeled subjects across 8 independent imaging sites, 491 of which were obtained from typically developing individuals and 285 in children and adolescents with ADHD (ages: 7-21 years old). The planar Corpus Callosum shape data are extracted, with 50 landmarks on the contour of the Corpus Callosum of each subject (see [19]). See Figure 4 for a plot of the raw landmarks of a normal developing child and an ADHD child) After quality control, 647 CC shape data out of 776 subjects were obtained, which included 404 (n_1) typically developing children, 150 (n_2) diagnosed with ADHD-Combined, 8 (n_3) diagnosed with ADHD-Hyperactive-Impulsive, and 85 (n_4) diagnosed with ADHD-Inattentive. Therefore, the data lie in the space Σ_2^{50}, which has a high dimension of $2 \times 50 - 4 = 96$.

As shown in the table 2, we consider the classification problem with 4 different classes. We also divided the dataset into a 75 percent training set and a 25 percent test set and evaluated the classification accuracy in the test set compared to other learning methods. Since the sample size is unbalanced, the total number

TABLE 2
Demographic information about processed ADHD-200 CC shape dataset, including disease status, age, and gender.

Disease status	Num.	Range of age in years(mean)	Gender(female/male)
Typically Developing Children	404	7.09 – 21.83(12.43)	179/225
ADHD-Combined	150	7.17 – 20.15(10.96)	39/111
ADHD-Hyperactive/Impulsive	8	9.22 – 20.89(14.69)	1/7
ADHD-Inattentive	85	7.43 – 17.61(12.23)	18/67
All data	647	7.09 – 21.83(12.09)	237/410



(a) Mean shapes of different classes

FIG 4. *CC shapes*

of some classes is too small, i.e., ADHD-Hyperactive case. We also considered the classification with two classes by combing those ADHD samples into one class shown in the right figure in Figure 4.

Similar to the sphere case, we select the k-nearest neighbors (kNN), the random forest (RF), the logistic regression (LR), and the support vector machine (SVM) with the radial basis function (RBF) kernel as competing estimators. The tuning parameters in each method are selected by evaluation on a validation data set whose size is 25% of the training set. For all deep neural network models, we utilize the same network architecture of 5 hidden layers with the numbers of width $(100, 100, 100, 100, 100)$. The deep neural network model is applied to the raw data, while the eDNN model is applied to the embedded data by Veronese-Whitney embedding. And the preshape data (normalized raw data) lying in the hyperspere $\mathbb{S}^{100}$ is used for the tDNN model and iDNN model. In the iDNN model, we chose the north pole and south pole $(\pm 1, 0, .., 0)$ as base points and utilized the geometry of the hypersphere as before. In the tDNN model, we pick the Fréchet mean of the training set as the base point and transform all data in a batch to tangent vectors before feeding to the neural network. All models are trained with Adam optimizer. The competition results can be observed in Table 3. Our tDNN model achieves the best accuracy at 65.84 ± 3.10

TABLE 3

The average accuracy on the test dataset is calculated over 50 *random splits. The* 5*-layers network (with* 100 *hidden nodes in each layer) is used for our deep neural network models in all experiments. Consequently, our tDNN model obtains the best accuracy in the 2 classes case while our iDNN model achieves the best accuracy in the 4 classes case. Furthermore, all our eDNN, tDNN and iDNN models outperform the classical deep neural network model, indicating the advantages of our frameworks.*

	4 Classes	2 Classes
DNN	56.40 ± 10.83	61.09 ± 8.44
eDNN	62.98 ± 3.91	63.81 ± 3.72
tDNN	63.20 ± 3.70	$\mathbf{65.84 \pm 3.10}$
iDNN	$\mathbf{63.55 \pm 3.80}$	65.42 ± 3.41
kNN	57.62 ± 3.37	61.26 ± 3.84
LR	61.35 ± 3.54	59.58 ± 3.44
RF	61.38 ± 3.50	63.20 ± 3.13
SVM	61.80 ± 3.92	64.89 ± 3.64

among 50 splits in the 2 classes case. Also, our iDNN model showed the best result of 63.55 ± 3.80 in the 4 classes case.

4.3. Symmetric semi-positive definite matrix (SPD)

The space $\mathrm{SPD}(d)$ of all $d \times d$ positive definite matrices belongs to an important class of manifolds that possesses particular geometric structures, which should be taken into account for building the deep neural networks. [15] investigates its Riemannian structure and provides somewhat concrete forms of all its geometric quantities. [11] studies different notions of means and averages in SPD(3) with respect to different distance metrics and considers applications to DTI data and covariance matrices.

Under the Riemannian framework of tensor computing [40], several metrics play an important role in machine learning on SPD matrices. Generally, the Riemannian distance $d(P_1, P_2)$ between two points P_1 and P_2 on the manifold is defined as the length of the geodesic $\gamma_{P_1 \to P_2}$, i.e., the shortest parameterized curve connecting them. In the SPD manifold, the distance under the affine metric could be computed as [40]:

$$d\left(Q_1, Q_2\right) = \frac{1}{2} \left\| \log\left(Q_1^{-\frac{1}{2}} Q_2 Q_1^{-\frac{1}{2}} \right) \right\|_F,$$

where $\| \cdot \|$ denotes the Frobenius norm.

Other important natural mappings to and from the manifold and its tangent bundle are the logarithmic mapping Log_{Q_0} and the exponential mapping Exp_{Q_0} at the point Q_0. Under the affine metric, those two mappings are known in closed form:

$$\mathrm{Exp}_{Q_0}(W) = Q_0^{\frac{1}{2}} \exp\left(Q_0^{-\frac{1}{2}} S Q_0^{-\frac{1}{2}} \right) P_0^{\frac{1}{2}} \in \mathrm{SPD}(d)$$

for every $W \in \mathcal{T}_{Q_0}$, and

$$\mathrm{Log}_{Q_0}(Q) = Q_0^{\frac{1}{2}} \log\left(Q_0^{-\frac{1}{2}} Q Q_0^{-\frac{1}{2}}\right) Q_0^{\frac{1}{2}} \in \mathcal{T}_{Q_0},$$

for every $Q \in \mathrm{SPD}(d)$, where $\mathcal{T}_{Q_0}$ denotes the tangent space at Q_0. Furthermore, we consider the log map on the matrix as the embedding J, mapping $\mathrm{SPD}(d)$ to $\mathrm{Sym}(d)$, the space of the symmetric matrix. For example, let $Q \in \mathrm{SPD}(d)$ with a spectral decomposition $Q = U\Sigma U^T$, we have the log-map of Q as $\log(Q) = U \log(\Sigma) U^T$ where $\log(\Sigma)$ denotes the diagonal matrix whose diagonal entries are the logarithms of the diagonal entries of Σ. Moreover, the embedding J is a diffeomorphism, equivariant with respect to the actions of $GL(d, \mathbb{R})$, the d by d general linear group. That is, for $H \in GL(d, \mathbb{R})$, we have $\log(HQH^T) = H \log(Q) H^{-1}$.

In the context of deep neural networks on SPD, our models build on the SPDNet introduced by [22], which mimicked the classical neural networks with the stage of computing an invariant representation of the input data points and a second stage devoted to performing the final classification. The SPDNet exploited the geometry based on threefold layers as described below.

- The BiMap (bilinear transformation) layer, analogous to the usual dense layer; the induced dimension reduction eases the computational burden often found in learning algorithms on SPD data:

$$X^{(l)} = {W^{(l)}}^T Q^{(l-1)} W^{(l)} \text{ with } W^{(l)} \text{ semi-orthogonal.}$$

- The ReEig (rectified eigenvalues activation) layer, analogous to the ReLU activation, can also be seen as an Eigen-regularization, protecting the matrices from degeneracy:

$$X^{(l)} = U^{(l)} \max\left(\Sigma^{(l)}, \epsilon I_n\right) {U^{(l)}}^T, \text{ with } Q^{(l)} = U^{(l)} \Sigma^{(l)} {U^{(l)}}^T.$$

- The LogEig (log eigenvalues Euclidean projection) layer:

$$X^{(l)} = \mathrm{vec}\left(U^{(l)} \log\left(\Sigma^{(l)}\right) {U^{(l)}}^T\right)$$

  with again $U^{(l)}$ the eigenspace of $Q^{(l)}$.

Under our framework, the SPDNet is both an eDNN and a tDNN model. The LogEig layer applies the logarithmic mapping $\log_I(Q^{(l)}) = \mathrm{vec}\left(U^{(l)} \log\left(\Sigma^{(l)}\right) {U^{(l)}}^T\right)$, which is identical to the transformation in the LogEig layer. Thus, SPDNet can also be viewed as a tDNN model. In our experiments, we only consider tDNN models as one tangent space from the base point is sufficient to cover the entire manifold. Our eDNN models on $\mathrm{SPD}(p)$ consist of 3 BiMap layers, 3 ReEig layers, one LogEig layer (for embedding), and

TABLE 4
The accuracy of the test set was reported. We follow the setup and protocols in [22] and our tDNN models outperform the eDNN (SPDNet) under both log and affine metrics.

Data	AFEW	HDM05
(n, d)	$(2135, 400^2)$	$(2086, 93^2)$
eDNN(SPDNet)	34.23 ± 1.44	61.35 ± 1.12
tDNN-Log	$\mathbf{35.85 \pm 1.49}$	$\mathbf{62.59 \pm 1.35}$
tDNN-Affine	35.31 ± 1.68	62.23 ± 1.43

a 5-layer deep neural network with 100 hidden nodes per layer. In tDNN models, we replace the LogEig layer with the intrinsic logarithmic mapping under different metrics.

In our experiments, we evaluate the performance of tDNN and eDNN models on the AFEW and HDM05 datasets using the same setup and protocol as in [22]. The AFEW dataset [9] includes 600 video clips with per-frame annotations of valence and arousal levels and 68 facial landmarks, depicting 7 classes of emotions. The HDM05 dataset [37] contains over three hours of motion capture data in C3D and ASF/AMC formats, covering more than 70 motion classes across multiple actors. We divide the data into a 75-25 percent training-test split, with 10 repetitions, and use the validation set (25 percent of training data) to tune hyperparameters. We implement tDNN models on both affine metrics and log-Euclidean metrics, using the Frechet mean of the batch as the base point. As shown in Table 4, our tDNN model under the Log-Euclidean metric achieves the best results on both datasets, with a 35.85 ± 1.49 accuracy on the AFEW dataset and 62.59 ± 1.35 accuracy on the HDM05 dataset.

5. Discussion

In this work, we develop intrinsic and extrinsic deep neural network architectures on manifolds and characterize their theoretical properties in terms of approximation error and statistical error of the ERM based estimator. The neural networks explore the underlying geometry of the manifolds for learning and inference. Future work will be focused on developing convolutional neural networks in manifolds for image classifications of manifold-values images, which have abundant applications in medical imaging and computer vision.

Appendix A: Proofs

A.1. Approximation of smooth functions on a manifold

The aim of this section is to develop a function approximation result by deep neural networks for smooth functions on a manifold, which is used in the analysis of both the eDNN and iDNN architectures. In this section, we let $\bar{\mathcal{F}}(L, P, B) = \cup_S \mathcal{F}(L, P, S, B)$ be the set of non-sparse deep neural networks with depth L and width P.

Lemma A.1 (Theorem 4 of [12]). *Assume that $f_0 \in \mathcal{C}_D^\beta([0,1]^D, A)$. Then there exist universal positive constants $C_1, \ldots, C_5$ depending only on D, β, A such that for any $N \in \mathbb{N} \setminus \{1\}$, there exists a deep neural network $f \in \bar{\mathcal{F}}(L, P, B)$ with $L = C_1$, $P = C_2 N$ and $B = C_3 N^{C_4}$ such that*

$$\|f - f_0\|_{L^\infty([0,1]^D)} \leq C_5 N^{-2\beta/D}.$$

By inspecting the proof of Theorem 4 of [12] as well as Theorem 3.3 of [25], we can determine the dependence of the universal constants $C_1, \ldots, C_5$ on the input dimension D and the smoothness β explicitly. To this end, we can set

$$\begin{aligned}
C_1 &\geq C_1'\{(\beta+1)^2 + 2D\} \\
C_2 &\geq C_2'(\beta \vee 1) D^{\beta+1} \\
C_3 &\geq C_3'(\beta \vee 1) D \\
C_4 &\geq C_4'((\beta \vee 1)/D + 1) \\
C_5 &\geq C_5' A (\beta+1)^2 D^{\beta + (\beta \vee 1)/2}
\end{aligned}$$

for some positive constants $C_1', \cdots, C_5'$ depending on none of D, β and A without changing the conclusion.

The following lemma is a generalization of Lemma A.1 to an arbitrary compact domain, where we give a detailed specification of the absolute constants as we stated above.

Lemma A.2. *Assume that $f_0 \in \mathcal{C}_D^\beta(U, A)$ where $U \in \mathbb{R}^D$ is a bounded subset. Let $R = 1 \vee \max_{\mathbf{x} \in U} \|\mathbf{x}\|_\infty$. Then there exist universal positive constants $C_1', \ldots, C_5'$ depending on none of D, β and A such that for any $N \in \mathbb{N} \setminus \{1\}$, there exists a deep neural network $f \in \bar{\mathcal{F}}(L, P, B)$ with $L = C_1'\{(\beta+1)^2 + 2D\}$, $P = C_2'(\beta \vee 1) D^{\beta+1} N$ and $B = C_3'(\beta \vee 1) D N^{C_4'((\beta \vee 1)/D + 1)}$ such that*

$$\|f - f_0\|_{L^\infty([0,1]^D)} \leq C_5' A R^\beta (\beta+1)^2 D^{\beta + (\beta \vee 1)/2} N^{-2\beta/D}.$$

Proof. Let T be an affine transfromation such that $Tx = R^{-1}x + (1/2, \ldots, 1/2)^\top$. Then we have $T(U) = [1/4, 3/4]^D$ and $f_0 \circ T^{-1} \in \mathcal{C}_D^\beta(T(U), R^\beta K)$. Applying Lemma A.1 concludes the proof. □

We here give our approximation theorem.

Theorem 5. *Let $M \subset \mathbb{R}^D$ be a d-dimensional compact manifold with smooth local coordinates. Assume that $f_0 \in \mathcal{C}_D^\beta(M, A)$ Then there exists universal positive constants $C_1'', \ldots, C_5''$ depending on none of D, β and A such that for any $N \in \mathbb{N} \setminus \{1\}$, there exists a deep neural network $f \in \bar{\mathcal{F}}(L, P, B)$ with $L = C_1''\{(\beta \vee 1)D/d + 1)^2 + 2D\}$, $P = C_2''((\beta \vee 1)D/d) D^{(\beta \vee 1)D/d + 1} N$ and $B = C_3''((\beta \vee 1) D^2/d) D N^{C_4''((\beta \vee 1)/d + 1)}$ such that*

$$\|f - f_0\|_{L^\infty([0,1]^D)} \leq C_5' A (1 + (\beta \vee 1)D/d)^2 D^{3/2(\beta \vee 1)D/d} N^{-2\beta/d}.$$

Proof. In the proof, the inequality notations $\gtrsim$ and $\lesssim$ hide absolute constants depending on none of D, β and A.

Most of the proof resembles the proof of Theorem 2 of [41]. Since M has smooth local coordinates, there exist charts $(V_1, \psi_1), \ldots, (V_r, \psi_r)$, such that for any $\gamma > 0$, $\psi_j \in \mathcal{C}_D^\gamma(V_j)$. Morover, by Lemma 3 of [41], there exist $\underline{\delta} > 0$ and a partition of unity on the manifold $\tau_k : M \to \mathbb{R}$ for $k = 1, \ldots, K$, such that for any $\gamma > 0$, and any $x \in M$, we have $\{y \in M : \tau_k(y) > 0\} \subseteq V_k^{-\underline{\delta}} = \{y \in V_k : \|y - (M \setminus V_k)\|_\infty \geq \underline{\delta}\}$, $\tau_k \in \mathcal{C}_D^\gamma(M)$, and $\sum_{k=1}^K \tau_k(x) = 1$. Using these, we decompose the target function f_0 as

$$f_0(x) = \sum_{k=1}^K \tau_j(x) \times (f_0 \circ \psi_j^{-1}) \circ \psi_k(x)$$

We will construct deep neural networks $\hat{\tau}_k$, $\hat{g}_k$ and $\hat{\psi}_k$ that approximate τ_k, $g_k = f_0 \circ \psi_k^{-1}$ and ψ_k, respectively, for $k = 1, \ldots, K$ and combine them as

$$f = \sum_{k=1}^K \hat{\times}(\tau_k, \hat{g}_k \circ \hat{\psi}_k),$$

where $\hat{\times}$ is a deep neural network that approximates the multiplication operation. Then following the last argument of the proof of Theorem 2 of [41], for a deep neural network f constructed as above, we have

$$\begin{aligned}\|f - f_0\|_{L^\infty(M)} \lesssim \sum_{k=1}^K \Big[&\|\hat{\times}(\hat{\tau}_k, \hat{g}_k \circ \hat{\psi}_k) - \hat{\tau}_k(\hat{g}_k \circ \hat{\psi}_k)\|_{L^\infty(M)} + \|\hat{\tau}_k - \tau_k\|_{L^\infty(M)} \\ &+ \|\hat{g}_k - g_k\|_{L^\infty(H_k)} + \|\|\hat{\psi}_k - \psi_k\|_\infty\|_{L^\infty(V_k)}^{\beta \wedge 1} \Big]\end{aligned}$$

where $H_k = \psi_k(V_k^{-\underline{\delta}})^{\delta'} = \{y \in \mathbb{R}^d : \|y - \psi_j(V_k^{-\underline{\delta}})\|_\infty \leq \delta'\}$ for some $\delta' > 0$.

We set $\gamma = (\beta \vee 1)D/d \geq 1$. Then by applying Lemma A.2 to each component of ψ_j and parallelizing the approximating networks, there exists a d-dimensional network $\hat{\psi}_j : \mathbb{R}^D \to \mathbb{R}^d$ with depth $L_1 \gtrsim (\gamma+1)^2 + 2D$ and width $P_1 \gtrsim \gamma D^{\gamma+1} N$ such that

$$\|\|\hat{\psi}_k - \psi_k\|_\infty\|_{L^\infty(V_k)} \lesssim (1 + (\beta \vee 1)D/d)^2 D^{3/2(\beta \vee 1)D/d} N^{-2(\beta \vee 1)/d}.$$

for any $N \in \mathbb{N}$. Next, by Lemma A.2, there exists a network $\hat{g}_k$ with depth $L_2 \gtrsim (\beta+1)^2 + 2d$ and width $P_2 \gtrsim (\beta \vee 1) d^{\beta+1} N$ such that

$$\|\hat{g}_k - g_k\|_{L^\infty(H_k)} \lesssim A(\beta+1)^2 d^{3(\beta \vee 1)/2} N^{-2\beta/d} \tag{13}$$

and a network $\hat{\tau}_k$ with depth $L_3 \gtrsim (\gamma+1)^2 + 2D$ and width $P_3 \gtrsim \gamma D^{\gamma+1} N$

$$\|\hat{\tau}_j - \tau_j\|_{L^\infty(M)} \lesssim (1 + (\beta \vee 1)D/d)^2 D^{3(\beta \vee 1)D/(2d)} N^{-2\beta/d}.$$

for any $N \in \mathbb{N}$. Lastly, by Lemma 4.2 of [35], for any $N \in \mathbb{N}$ there is a network $\hat{\times}$ with depth $L_4 \gtrsim \beta/d$ and width $P_4 = 9N + 1$ such that $|\hat{\times}(x, y) - xy| \leq 6(b-a)^2 N^{-2\beta/d}$ for any $x, y \in [a, b]$. Combining these approximation results, we get the desired result. □

A.2. Proofs for Section 2

The following lemma translates approximation results by non-sparse deep neural networks to sparse ones.

Lemma A.3. *$\bar{\mathcal{F}}(L,P,B) \subset \mathcal{F}(L,\bar{P},S,B)$ for any $\bar{P} \geq P$ and $S \geq P(D+1)+(P^2+P)(L-1)+P+1$.*

Proof. This is trivial since the number of parameters of a depth L and width P is given by $P(D+1)+(P^2+P)(L-1)+P+1$. □

Proof of Theorem 1. Let $\tilde{f}_0 = f_0 \circ J^{-1}$, then $\tilde{f}_0$ is a function on the d-dimensional manifold $\tilde{M} = J(M) \subset \mathbb{R}^D$. Since $\tilde{M}$ has smooth local coordinates, we can apply Theorem 5 to construct a deep neural network $\tilde{f} \in \bar{\mathcal{F}}(L,P,B)$ that approximates $\tilde{f}_0$. The approximation error of $\tilde{f}$ to $\tilde{f}_0$ is the same as that of $f = \tilde{f} \circ J$ to f_0:

$$\|f - f_0\|_{L^\infty(M)} = \|\tilde{f} \circ J - \tilde{f}_0 \circ J\|_{L^\infty(M)} = \|\tilde{f} - \tilde{f}_0\|_{L^\infty(\tilde{M})}.$$

But in view of Lemma A.3, $\tilde{f} \in \mathcal{F}(L,\bar{P},S,B)$ with $\bar{P} \geq P$ and $S = P(P+1)LD$, which completes the proof. □

Proof of Theorem 2. For any $\tilde{f}_1, \tilde{f}_2 \in \mathcal{F}(L,P,S,B)$, we have $\|\tilde{f}_2 \circ J - \tilde{f}_2 \circ J\|_{L^\infty(M)} = \|\tilde{f}_2 - \tilde{f}_2\|_{L^\infty(\tilde{M})} \leq \|\tilde{f}_2 - \tilde{f}_2\|_{L^\infty(\mathbb{R}^D)}$. Hence the entropy of the eDNN class $\mathcal{F}_{eDNN}(L,P,S,B)$ is bounded by that of $\mathcal{F}(L,P,S,B)$. Thus, by Lemmas 4 and 5 of [42] together with our approximation analysis in Theorem 1, we have

$$\begin{aligned} R\left(\hat{f}_{eDNN}, f_0\right) &\lesssim \inf_{f \in \mathcal{F}_{eDNN}(L,P,S,B)} \|f - f_0\|^2_{L^\infty(M)} \\ &\quad + \frac{(S+1)\log\left(2n(L+1)P^{2L}(D+1)^2\right)+1}{n} \\ &\lesssim S^{-2\beta/d} + \frac{S\log n}{n}. \end{aligned}$$

Then if we take $S \asymp (n/\log n)^{d/(2\beta+d)}$, we get the desired result. □

A.3. Proofs for Section 3

Proof of Theorem 3. We construct a deep neural network approximating $f_{0k} = f_0 \circ \exp_{x_k}$ for each $k = 1,\ldots,K$. Note that f_{0k} is β-Hölder smooth by assumption. Therefore, by Lemma A.2, there exists a network f_k with depth $L_2 \gtrsim (\beta+1)^2 + 2d$ and width $P_2 \gtrsim (\beta \vee 1)d^{\beta+1}N$ such that

$$\|f_k - f_{0k}\|_{L^\infty(U_k)} \lesssim A(\beta+1)^2 d^{3(\beta\vee 1)/2} N^{-2\beta/d} \tag{14}$$

Now, let $f = \sum_{k=1}^K \tau_k(x) f_k(\log_{x_k}(x)) \in \mathcal{F}_{iDNN}(L,P,S,B)$. Then

$$\|f - f_0\|_{L^\infty(M)} = \sup_{x \in M} \left| \sum_{k=1}^K \tau_k(x) f_k(\log_{x_k}(x)) - \sum_{k=1}^K \tau_k(x) f_{0k}(\log_{x_k}(x)) \right|$$

$$\leq \sup_{x\in M} \sum_{k=1}^{K} \tau_k(x) \left| f_k(\log_{x_k}(x)) - f_{0k}(\log_{x_k}(x)) \right|$$
$$\leq \max_{1\leq k\leq K} \|f_k - f_{0k}\|_{L^\infty(U_k)} .$$

But in view of Lemma A.3, $f \in \mathcal{F}(L, \bar{P}, S, B)$ with $\bar{P} \geq P$ and $S = P(P+1)LD$, which completes the proof. □

Proof of Theorem 4. For any two iDNNs $f(\cdot) = \sum_{k=1}^{K} \tau_k(\cdot) f_k(\log_{x_k}(\cdot))$ and $f'(\cdot) = \sum_{k=1}^{K} \tau_k(\cdot) f'_k(\log_{x_k}(\cdot))$ in $\mathcal{F}_{iDNN}(L, P, S, B)$, we have

$$\|f - f'\|_{L^\infty(M)} \leq \sup_{x\in M} \sum_{k=1}^{K} \tau_k(x) \left| f_k(\log_{x_k}(x)) - f'_k(\log_{x_k}(x)) \right|$$
$$\leq \max_{1\leq k\leq K} \left\| f_k - f'_k \right\|_{L^\infty(U_k)} .$$

Therefore, the entropy of $\mathcal{F}_{iDNN}(L, P, S, B)$ is bounded by the K-times of the entropy of the class $\mathcal{F}(L, P, S, B)$. So by the same argument as in the proof of Theorem 2, we get the desired result. □

Acknowledgments

We would like to thank Dong Quan Nguyen, Steve Rosenberg, and Bayan Saparbayeva for very helpful discussions.

Funding

LL and YF are supported by grants DMS CAREER 1654579 and DMS 2113642. IO was supported by the National Research Foundation of Korea(NRF) grant funded by the Korea government(MSIT) (NRF-2022R1F1A1069695) and Inha University Research Grant.

References

[1] Alexander, A., Lee, J. E., Lazar, M. and Field, A. S. (2007). Diffusion Tensor Imaging of the Brain. *Neurotherapeutics* **4(3)** 316–329.

[2] Bahdanau, D., Cho, K. and Bengio, Y. (2015). Neural Machine Translation by Jointly Learning to Align and Translate. *Proceedings of the 4th International Conference on Learning Representations* **abs/1409.0473**. MR4390194

[3] Bauer, B. and Kohler, M. (2019). On deep learning as a remedy for the curse of dimensionality in nonparametric regression. *The Annals of Statistics* **47** 2261–2285. MR3953451

[4] Bhattacharya, A. and Bhattacharya, R. N. (2012). *Nonparametric Inference on Manifolds: with Applications to Shape Spaces*. Cambridge University Press IMS monographs #2. MR2934285

[5] Bhattacharya, R. and Lin, L. (2017). Omnibus CLTs for Fréchet means and nonparametric inference on non-Euclidean spaces. *The Proceedings of the American Mathematical Society* **145** 413-428. MR3565392

[6] Chen, M., Jiang, H., Liao, W. and Zhao, T. (2019). Efficient approximation of deep relu networks for functions on low dimensional manifolds. *Advances in neural information processing systems* **32**.

[7] Chen, M., Jiang, H., Liao, W. and Zhao, T. (2022). Nonparametric regression on low-dimensional manifolds using deep ReLU networks: Function approximation and statistical recovery. *Information and Inference: A Journal of the IMA* **11** 1203–1253. MR4526322

[8] Chevallier, E., Li, D., Lu, Y. and Dunson, D. (2022). Exponential-Wrapped Distributions on Symmetric Spaces. *SIAM Journal on Mathematics of Data Science* **4** 1347-1368. MR4522875

[9] Dhall, A., Goecke, R., Lucey, S. and Gedeon, T. (2011). Static facial expression analysis in tough conditions: Data, evaluation protocol and benchmark. In *2011 IEEE International Conference on Computer Vision Workshops (ICCV Workshops)* 2106–2112. IEEE.

[10] Downs, T., Liebman, J. and Mackay, W. (1971). Statistical methods for vectorcardiogram orientations. *In Vectorcardiography 2: Proc. XIth International Symposium on Vectorcardiography (I. Hoffman, R.I. Hamby and E. Glassman, Eds.)* 216-222. North-Holland, Amsterdam.

[11] Dryden, I. L., Koloydenko, A. and Zhou, D. (2009). Non-Euclidean statistics for covariance matrices, with applications to diffusion tensor imaging. *The Annals of Applied Statistics* **3** 1102–1123. MR2750388

[12] Fan, J. and Gu, Y. (2022). Factor augmented sparse throughput deep relu neural networks for high dimensional regression. *arXiv preprint arXiv: 2210.02002*.

[13] Fisher, N. I., Lewis, T. and Embleton, B. J. J. (1987). *Statistical Analysis of Spherical Data*. Cambridge Uni. Press, Cambridge. MR0899958

[14] Fisher, R. A. (1953). Dispersion on a sphere. *Proceedings of the Royal Society A* **217** 295-305. MR0056866

[15] Fletcher, P. T. and Joshi, S. (2007). Riemannian geometry for the statistical analysis of diffusion tensor data. *Signal Processing* **87** 250 – 262. Tensor Signal Processing. http://dx.doi.org/10.1016/j.sigpro.2005.12.018

[16] Harandi, M. and Fernando, B. (2016). Generalized BackPropagation, Étude De Cas: Orthogonality. *arXiv e-prints arXiv:1611.05927*.

[17] Hinton, G., Deng, L., Yu, D., Dahl, G. E., Mohamed, A., Jaitly, N., Senior, A., Vanhoucke, V., Nguyen, P., Sainath, T. N. and Kingsbury, B. (2012). Deep Neural Networks for Acoustic Modeling in Speech Recognition: The Shared Views of Four Research Groups. *IEEE Signal Processing Magazine* **29** 82-97. https://doi.org/10.1109/MSP.2012.2205597

[18] Ho, J., Lee, K.-C., Yang, M.-H. and Kriegman, D. (2004). Visual tracking using learned linear subspaces. In *Computer Vision and Pattern Recognition, 2004. CVPR 2004. Proceedings of the 2004 IEEE Computer Society Conference on* **1** I-782-I-789 Vol.1. https://doi.org/10.1109/CVPR.2004.1315111

[19] Huang, C., Styner, M. and Zhu, H. (2015). Clustering High-Dimensional Landmark-Based Two-Dimensional Shape Data. *Journal of the American Statistical Association* **110** 946-961. https://doi.org/10.1080/01621459.2015.1034802 MR3420675

[20] Huang, C., Zhang, H. and Robeson, S. (2011). On the Validity of Commonly Used Covariance and Variogram Functions on the Sphere. *Mathematical Geosciences* **43** 721-733. https://doi.org/10.1007/s11004-011-9344-7 MR2824128

[21] Huang, Z. and Gool, L. V. (2017). A Riemannian Network for SPD Matrix Learning. In *Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence. AAAI'17* 2036–2042. AAAI Press.

[22] Huang, Z. and Van Gool, L. (2017). A Riemannian network for spd matrix learning. In *Proceedings of the AAAI Conference on Artificial Intelligence* **31**.

[23] Huang, Z., Wan, C., Probst, T. and Van Gool, L. (2017). Deep Learning on Lie Groups for Skeleton-Based Action Recognition. In *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* 1243-1252. https://doi.org/10.1109/CVPR.2017.137

[24] Huang, Z., Wu, J. and Van Gool, L. (2016). Building Deep Networks on Grassmann Manifolds. *arXiv preprint arXiv:1611.05742*.

[25] Jiao, Y., Shen, G., Lin, Y. and Huang, J. (2023). Deep nonparametric regression on approximate manifolds: Nonasymptotic error bounds with polynomial prefactors. *The Annals of Statistics* **51** 691–716. MR4600998

[26] Jun, M. and Stein, M. L. (2008). Nonstationary covariance models for global data. *The Annals of Applied Statistics* **2** 1271–1289. https://doi.org/10.1214/08-AOAS183 MR2655659

[27] Kendall, D. G. (1984). Shape Manifolds, Procrustean Metrics, and Complex Projective Spaces. *Bull. of the London Math. Soc.* **16** 81-121. MR0737237

[28] Kim, Y., Ohn, I. and Kim, D. (2021). Fast convergence rates of deep neural networks for classification. *Neural Networks* **138** 179–197.

[29] Kingma, D. P. and Ba, J. (2014). Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

[30] Kohler, M. and Langer, S. (2021). On the rate of convergence of fully connected deep neural network regression estimates. *The Annals of Statistics* **49** 2231–2249. MR4319248

[31] Kolaczyk, E. D., Lin, L., Rosenberg, S., Walters, J. and Xu, J. (2020). Averages of unlabeled networks: Geometric characterization and asymptotic behavior. *Ann. Statist.* **48** 514–538. https://doi.org/10.1214/19-AOS1820 MR4065172

[32] Krizhevsky, A., Sutskever, I. and Hinton, G. E. (2012). Imagenet

classification with deep convolutional neural networks. In *Advances in neural information processing systems* 1097–1105.

[33] Lin, L., Mu, N., Cheung, P. and Dunson, D. (2019). Extrinsic Gaussian Processes for Regression and Classification on Manifolds. *Bayesian Anal.* **14** 887–906. https://doi.org/10.1214/18-BA1135 MR3960775

[34] Lin, L., Thomas, B. S., Zhu, H. and Dunson, D. B. (2017). Extrinsic Local Regression on Manifold-Valued Data. *Journal of the American Statistical Association* **112** 1261-1273. https://doi.org/10.1080/01621459.2016.1208615 MR3735375

[35] Lu, J., Shen, Z., Yang, H. and Zhang, S. (2021). Deep network approximation for smooth functions. *SIAM Journal on Mathematical Analysis* **53** 5465–5506. MR4319100

[36] Mardia, K. V. and Jupp, P. E. (2000). *Directional Statistics*. Wiley, New York. MR1828667

[37] Müller, M., Röder, T., Clausen, M., Eberhardt, B., Krüger, B. and Weber, A. (2007). Mocap database hdm05. *Institut für Informatik II, Universität Bonn* **2**.

[38] Nakada, R. and Imaizumi, M. (2020). Adaptive approximation and generalization of deep neural network with intrinsic dimensionality. *The Journal of Machine Learning Research* **21** 7018–7055. MR4209460

[39] Ohn, I. and Kim, Y. (2022). Nonconvex sparse regularization for deep neural networks and its optimality. *Neural computation* **34** 476–517. MR4381798

[40] Pennec, X., Fillard, P. and Ayache, N. (2006). A Riemannian framework for tensor computing. *International Journal of computer vision* **66** 41–66.

[41] Schmidt-Hieber, J. (2019). Deep ReLU network approximation of functions on a manifold. *arXiv preprint arXiv:1908.00695*. MR2659223

[42] Schmidt-Hieber, J. (2020). Nonparametric regression using deep neural networks with ReLU activation function. *The Annals of Statistics* **48** 1875–1897. https://doi.org/10.1214/19-AOS1875 MR4134774

[43] Teja, G. P. and Ravi, S. (2012). Face recognition using subspaces techniques. In *Recent Trends In Information Technology (ICRTIT), 2012 International Conference on* 103-107. https://doi.org/10.1109/ICRTIT.201

[44] Tu, L. W. (2011). *An introduction to manifolds*. Springer. MR2723362

[45] Voulodimos, A., Doulamis, N., Doulamis, A. and Protopapadakis, E. (2018). Deep Learning for Computer Vision: A Brief Review. *Computational Intelligence and Neuroscience* **2018**.

[46] Zhang, J., Zhu, G., Heath, R. and Huang, K. (2018). Grassmannian Learning: Embedding Geometry Awareness in Shallow and Deep Learning. *arXiv preprint arXiv:1808.02229*.