Electronic Journal of Statistics
Vol. 18 (2024) 1248–1292
ISSN: 1935-7524
https://doi.org/10.1214/24-EJS2229

Multiclass classification for multidimensional functional data through deep neural networks

Shuoyang Wang[1] **and Guanqun Cao**[2]
For The Alzheimer's Disease Neuroimaging Initiative

[1]*Department of Bioinformatics and Biostatistics, University of Louisville, U.S.A.*
e-mail: shuoyang.wang@louisville.edu

[2]*Department of Statistics and Probability, Michigan State University, U.S.A.*
e-mail: caoguanq@msu.edu

Abstract: The intrinsically infinite-dimensional features of the functional observations over multidimensional domains render the standard classification methods effectively inapplicable. To address this problem, we introduce a novel multiclass functional deep neural network (mfDNN) classifier as an innovative data mining and classification tool. The architecture incorporates a sparse deep neural network with Rectified Linear Unit (ReLU) activation function, minimizing cross-entropy loss in a multiclass classification framework. This design enables the utilization of modern computational tools. The convergence rates of the misclassification risk functions are also derived for both fully observed and discretely observed multidimensional functional data. The efficacy of mfDNN is demonstrated through simulations and several benchmark datasets from different application domains.

MSC2020 subject classifications: Primary 62G05, 62G08; secondary 62G35.
Keywords and phrases: Functional data analysis, deep neural networks, multiclass classification, rate of convergence, multidimensional functional data.

Received October 2023.

Contents

1. Introduction

Functional data classification has wide applications in many areas, such as machine learning and artificial intelligence [31, 19, 27, 7]. While the majority of the work on functional data classification focuses on one-dimensional functional data cases, for example, temperature data over a certain period [26], and speech recognition data (log-periodogram data) [20, 37], there are few results obtained for multidimensional functional data classification, such as 2D or 3D images classification. In many applications, data are collected in the form of functions such as curves or images. Such data are nowadays commonly referred to as functional data. For instance, in the Alzheimer's Disease Neuroimaging Initiative (ADNI) database, the training samples are PET images from multiple status (e.g., normal, early mild cognitive impairment, Alzheimer's Disease), and the task is to predict the status of a new patient given his/her PET imaging. Classic multivariate analysis techniques, such as logistic regression or discriminant analysis, are not directly applicable, since functional data are intrinsically infinite-dimensional. A common strategy is to adapt the multivariate analysis to functional settings, such as functional discriminant analysis and random forest, among others. Despite their impressive performances, one is often interested in knowing their statistical property, such as the upper bound of the misclassification rate. Unfortunately, no literature can answer the above problem because a multi-class classifier for multi-dimensional functional data has not been established yet. This motivates us to first propose novel and optimal functional classifiers, and then investigate the convergence rate of the proposed classifier.

In this paper, we propose a novel approach, called multi-class functional deep neural network (mfDNN, multiclass-functional+DNN), for multidimensional functional data multiclass classification. We extract the functional principal components scores (FPCs) of the data functions, and train a DNN based classifier on these FPCs and their corresponding class memberships. Given these FPCs as inputs and their corresponding class labels as outputs, we adopt sparse deep ReLU network architecture and minimize the cross-entropy (CE) loss in the multiclass classification setup. CE loss is a standard tool in machine learning classification, but the technique has not been utilized to its fullest potential in the context of functional data classification, especially regarding the study of theoretical properties of DNN classifiers. A recent work [4], has an example of multiclassification by minimizing CE loss in sparse DNN for cross-sectional data setting. While CE loss is a standard tool in machine learning classification, its full potential in the context of functional data classification, particularly in studying the theoretical properties of DNN classifiers, remains underexplored. We apply the same loss function as in [4] but tailor it to the context of multidimensional functional multiclassification. Furthermore, convergence rates and the conditional class probabilities are developed under both fully observed and discretely observed multi-dimensional functional data setting, respectively.

1.1. Related work

This work is related to a series of papers on high-dimensional data and functional data classification. In recent years, many sparse discriminant analysis methods have been proposed for i.i.d. high-dimensional data classification and variable selection. Most of these proposals focus on binary classification and they are not directly applicable to multiclass classification problems. A popular multiclass sparse discriminant analysis proposal is the ℓ_1 penalized Fisher's discriminant [38]. However, [38] does not have theoretical justifications. It is generally unknown how close their estimated discriminant directions are to the true directions, and whether the final classifier will work similarly as the Bayes rule. More recently, [23] proposed a multiclass sparse discriminant analysis method that simultaneously estimates all discriminant directions with theoretical justification for high-dimensional data setting. An efficient way to handle multi-dimensional functional data, such as imaging data, is to incorporate the information that is inherent in order and smoothness of processes over pixels or voxels. We compare our method with both [38] and [23] in terms of numerical performance in Sections 5 and 6.

In addition, a mainstream technique in functional data classification is based on functional principal component analysis (FPCA) such as functional discriminant analysis [29, 10, 9, 11, 13, 8, 3, 25, 1]. They only focus on binary classification for one-dimensional functional data. There is also a lack of literature on multiclass classification for functional data. [20] investigated a Bayesian approach to estimate parameters in multiclass functional models. However, their method is still only suitable for fully observed one-dimensional functional data

classification. To the best of our knowledge, there is only one recent work [36] where binary classification for general non-Gaussian multidimensional functional data was considered. However, their proposed classifier is designed for fully observed functional data and only able to conduct binary classification. In contrast, the target of our framework is both fully observed and sparsely sampled multidimensional functional data cases. Comparing to most classic functional data classification literature, our method offers more flexible applications.

Another related topic is the deep learning and computer vision areas, which have rich references on contemporary DNN architectures commonly used for image classification, for instances, AlexNet [18], GoogLeNet [32], VGG [30], Inception [33], and more modern ones, such as Vision Transformer [12]. These kind of neural network architectures are computationally inefficient with extremely large number of tuning parameters. Furthermore, to the best of our knowledge, there are lacks of theoretical supports for these computer vision methods. In other words, the asymptotic misclassification rates is not available and the interpretability and reliability of these algorithm are still questionable. Last but not least, these computer vision methods require high resolution inputs, i.e., the number of pixels and voxels per subject is relatively large. In contrast, our classifier is designed to work with either sparsely or densely observed multidimensional functional or imaging data.

1.2. Contributions

The contributions of the present paper are three-fold. First, to the best of our knowledge, we are the first to propose a multi-class classifier for multidimensional functional data within the framework of DNNs. In comparison to much of the state-of-the-art machine learning literature, our classifier does not demand high resolution. This flexible classifier can address a variety of problems with sparsely or densely observed 2D/3D imaging data. Meanwhile, the sparse setting has more practical applications than the fully observed case in certain existing functional data classification work [36]. Second, our work is novel as it establishes the first convergence rate of multi-class classification for multi-dimensional functional data. In comparison with most imaging classification methods in the field of computer vision, our method not only provides solid theoretical guarantees but has also demonstrated computational efficiency in terms of costs. Last but not least, we do not assume the data function follows Gaussian process, which has been required in most classic functional data classification literature [3, 13], even though it is often violated in practice. When data distributions are general non-Gaussian, the resulting decision boundary is often complicated which cannot be accurately recovered by most existing approaches.

The rest of this article is organized as follows. In Section 2, we introduce the K-class functional data classification models and propose the multiclass functional deep neural network classifiers. In Section 3, we establish theoretical properties of mfDNN under suitable technical assumptions. Section 4 provides two progressive examples to demonstrate the validity of these technical assumptions.

In Section 5, performances of mfDNN and its competitors are demonstrated through simulation studies. The corresponding R programs for implementing our method are provided on GitHub. In Section 6, we apply mfDNN to the zip code dataset and Alzeheimer's Disease data. Section 7 summarizes the conclusions. Technical proofs and additional figures are provided in the Appendix.

2. Methodology

2.1. *K*-class functional data classification models

Suppose we observe n i.i.d. functional training samples $\{(X_i(s), \boldsymbol{Y}_i) : 1 \le i \le n, s \in [0,1]^d\}$, which are independent of $X(s)$ to be classified, and class label $\boldsymbol{Y}_i = (Y_{i1}, \ldots, Y_{iK})^\intercal$, such that $Y_{ik} = 1$ if the sample is in the k-th group, $k = 1, 2, \ldots, K$, and 0 otherwise. Throughout the entire paper, we consider the number of class K is a universal constant not depending on sample size n. In multiclass classification with $K \ge 2$ classes, we are interested in grouping a newly observed continuous curves $X(s)$ to one of the K classes given the training data.

If $Y_{ik} = 1$, $X_i(s)$ presumably has some unknown mean function $\mathbb{E}X_i(s) = \mu_k(s)$ and unknown covariance function $\Omega_k(s, s') = \mathbb{E}[(X_i(s) - \mu_k(s))(X_i(s') - \mu_k(s'))]$, for $s, s' \in [0,1]^d$ and $1 \le k \le K$. Suppose the covariance function $\Omega_k(\cdot, \cdot)$ satisfies the Karhunen–Loève decomposition:

$$\Omega_k(s, s') = \sum_{j=1}^{\infty} \lambda_{kj} \psi_{kj}(s) \psi_{kj}(s'), s, s' \in [0,1]^d, \tag{2.1}$$

where ψ_{kj}, $j \ge 1$ is an orthonormal basis of $L^2([0,1]^d)$ with respect to the usual L^2 inner product, and $\lambda_{k1} \ge \lambda_{k2} \ge \cdots > 0$ are nonincreasing positive eigenvalues. For any $X(s) \in L^2([0,1]^d)$ under the k-th class, write $X(s) = \sum_{j=1}^{\infty} \xi_j \psi_{kj}(s)$, where ξ_j's are pairwise uncorrelated random coefficients. By projecting the function space to the vector space, we define the conditional probabilities

$$\pi_k(\boldsymbol{x}) = \mathrm{P}(\boldsymbol{Y} = \boldsymbol{e}_k | \boldsymbol{\xi} = \boldsymbol{x}), \quad k \in \{1, \ldots, K\},$$

where $\boldsymbol{e}_k = (0, \ldots, 0, 1, 0, \ldots, 0)^\intercal$ is a K-dimensional standard basis vector denoting the label of the k-th class is observed and $\boldsymbol{\xi} = (\xi_1, \xi_2, \ldots)^\intercal$ is an infinite dimension random vector. Notably, $\sum_{k=1}^{K} \pi_k(\boldsymbol{x}) = 1$ for any $\boldsymbol{x} \in \mathbb{R}^\infty$.

2.2. Multiclass functional deep neural network classifier

For $k \in \{1, \ldots, K\}$, define sample covariance function

$$\widehat{\Omega}_k(s, s') = \frac{1}{n_k} \sum_{i \in I_k} (X_i(s) - \bar{X}_k(s))(X_i(s') - \bar{X}_k(s')), \quad s, s' \in [0,1]^d,$$

where I_k is the collection of i such that $Y_{ik} = 1$, $n_k = |I_k|$ is the sample size, and $\bar{X}_k(s) = n_k^{-1} \sum_{i \in I_k} X_i(s)$ is the sample mean function of class k. The Karhunen–Loéve decomposition for $\widehat{\Omega}_k$ is

$$\widehat{\Omega}_k(s, s') = \sum_{j=1}^{\infty} \widehat{\lambda}_{kj} \widehat{\psi}_{kj}(s) \widehat{\psi}_{kj}(s'), s, s' \in [0, 1]^d .$$

The sample data function $X_i(s)$, $i = 1, \ldots, n$, under $Y_{ik} = 1$, can be expressed as $X_i(s) = \sum_{j=1}^{\infty} \widehat{\xi}_{ij} \widehat{\psi}_{kj}(s)$. In this representation, $\widehat{\psi}_{kj}(s)$ may either correspond to a set of pre-selected basis functions or be determined by data-driven methods. Pre-selected basis functions typically include Fourier or wavelet bases, whereas data-driven approaches often utilize eigen-functions derived through FPCA. We provide the examples of Fourier basis and FPCA approach in Remark 9 to elaborate on how these varying basis functions influence the theoretical performance of the classifier in Theorem 3.2. It's noteworthy that while $\widehat{\psi}_{kj}(s)$ may differ from $\psi_{kj}(s)$, this dissimilarity does not impact the numerical performance of the proposed classifier. Further details can be found in Section 5. Intuitively, $\widehat{\boldsymbol{\xi}}^{(i)} := (\widehat{\xi}_{i1}, \widehat{\xi}_{i2}, \ldots)$ is an estimator of $\boldsymbol{\xi}^{(i)} := (\xi_{i1}, \xi_{i2}, \ldots)$, in which ξ_{ij}'s are unobservable random coefficients of X_i with respect to the population basis ψ_{kj}. Hence, it is natural to design classifiers based on $\widehat{\boldsymbol{\xi}}^{(i)}$'s.

Let $\widehat{\boldsymbol{\xi}}_J^{(i)} = (\widehat{\xi}_{i1}, \ldots, \widehat{\xi}_{iJ})^\intercal$ be the J-dimensional truncation of $\widehat{\boldsymbol{\xi}}^{(i)}$ for $i = 1, \ldots, n$. Denote $h_k(\cdot)$ as the conditional probability densities of $\boldsymbol{\xi}$ given $\boldsymbol{Y} = \boldsymbol{e}_k$, such that $\pi_k(\boldsymbol{x}) = \frac{h_k \mathrm{P}(\boldsymbol{Y} = \boldsymbol{e}_k)}{\sum_{\ell=1}^K h_\ell \mathrm{P}(\boldsymbol{Y} = \boldsymbol{e}_\ell)}$. When X_i's are some random processes with complex structures, such as non-Gaussian processes, one major challenge is the underlying complicated form of $\{h_k\}_{k=1}^K$ so that estimation of $\{\pi_k\}_{k=1}^K$ is typically difficult. In this section, inspired by the rich approximation power of DNN, we propose a new classifier so-called mfDNN, which can still approach Bayes classifiers closely even when h_k's are non-Gaussian and complicated.

Let $\sigma(x) = (x)_+$ be the ReLU activation function, and K-dimensional softmax activation function $\boldsymbol{\sigma}^*(\boldsymbol{y}) = \left(\frac{\exp(y_1)}{\sum_{k=1}^K \exp(y_k)}, \ldots, \frac{\exp(y_K)}{\sum_{k=1}^K \exp(y_k)} \right)$ for $\boldsymbol{y} = (y_1, \ldots, y_K) \in \mathbb{R}^K$. For any real vectors $\boldsymbol{V} = (v_1, \ldots, v_w)^\intercal$ and $\boldsymbol{z} = (z_1, \ldots, z_w)^\intercal$, define the shift activation function $\sigma_{\boldsymbol{V}}(\boldsymbol{z}) = (\sigma(z_1 - v_1), \ldots, \sigma(z_w - v_w))^\intercal$. For $L \geq 1$, $\boldsymbol{p} = (p_1, \ldots, p_L) \in \mathbb{N}^L$, let $\mathcal{F}(L, J, \boldsymbol{p})$ denote the class of fully connected feedforward DNN with J inputs, L hidden layers and, for $l = 1, \ldots, L$, p_l nodes on the l-th hidden layer. Equivalently, any $f \in \mathcal{F}(L, J, \boldsymbol{p})$ has an expression

$$f(\boldsymbol{x}) = \boldsymbol{\sigma}^* \mathbf{W}_L \sigma_{\boldsymbol{V}_L} \mathbf{W}_{L-1} \sigma_{\boldsymbol{V}_{L-1}} \ldots \mathbf{W}_1 \sigma_{\boldsymbol{V}_1} \mathbf{W}_0 \boldsymbol{x}, \quad \boldsymbol{x} \in \mathbb{R}^J, \tag{2.2}$$

where $\mathbf{W}_l \in \mathbb{R}^{p_{l+1} \times p_l}$, for $l = 0, \ldots, L$, are weight matrices, $\boldsymbol{V}_l \in \mathbb{R}^{p_l}$, for $l = 1, \ldots, L$, are shift vectors. Here, we adopt the convention that $p_0 = J$ and $p_{L+1} = 1$.

Due to the large capacity of the fully connected DNN class and to avoid overparameterization, we consider the following sparse DNN class:

$$\mathcal{F}(L, J, \boldsymbol{p}, s) = \tag{2.3}$$

$$\left\{ f \in \mathcal{F}(L, J, \boldsymbol{p}) : \sum_{l=0}^{L} \|\mathbf{W}_l\|_0 + \|\mathbf{V}_l\|_0 \le s, \max_{0 \le l \le L} \|\mathbf{W}_l\|_\infty \vee \|\mathbf{V}_l\|_\infty \le 1 \right\},$$

where $\|\cdot\|_\infty$ denotes the maximum-entry norm of a matrix or a vector, which is controlled by 1, $\|\cdot\|_0$ represents the number of non-zero elements of a matrix or a vector, and s controls the total number of active neurons.

Given the training data $(\boldsymbol{\xi}_J^{(1)}, \boldsymbol{Y}_1), \ldots, (\boldsymbol{\xi}_J^{(n)}, \boldsymbol{Y}_n)$, let

$$\widehat{\boldsymbol{f}}_\phi(\cdot) = \left(\widehat{f}_\phi^{(1)}, \ldots, \widehat{f}_\phi^{(K)}\right)^\intercal = \arg \min_{\boldsymbol{f} \in \mathcal{F}(L, J, \boldsymbol{p}, s)} -\frac{1}{n} \sum_{i=1}^{n} \phi(\boldsymbol{Y}_i, \boldsymbol{f}(\widehat{\boldsymbol{\xi}}_J^{(i)})), \tag{2.4}$$

where $\phi(\boldsymbol{x}_1, \boldsymbol{x}_2) = \boldsymbol{x}_1^\intercal \log(\boldsymbol{x}_2)$ denotes the CE loss function.

Remark 1. *In the optimization step, each element of the estimated probability vector $\widehat{\boldsymbol{f}}_\phi$ should be within 0 and 1. This is ensured by the definition of our neural network class, where $\widehat{\boldsymbol{f}}_\phi \in \mathcal{F}$, which is defined by Equation (2.2).*

We then propose the following mfDNN classifier: for $X \in L^2([0,1]^d)$,

$$\widehat{G}^{mfDNN}(X) = k, \quad \text{if } \widehat{f}_\phi^{(k)}(\boldsymbol{\xi}_J) \ge \widehat{f}_\phi^{(k')}(\boldsymbol{\xi}_J), \quad \text{for all } \ 1 \le k' \ne k \le K. \tag{2.5}$$

The detailed implementation for the proposed mfDNN classifier is explained in Algorithm 1. The tuning parameters include $J, L, \boldsymbol{p}$, and s, which theoretically contribute to the performance of classification. We discuss the related theories in Section 3. In practice, we suggest the following data-splitting approach to select $(J, L, \boldsymbol{p}, s)$. Note that the sparsity is not directly applicable by assigning the number of active neurons. Instead, we use dropout technique to achieve sparse network.

Algorithm 1: Training algorithm for mfDNN

Input: Collection of samples $\left\{\{X_i(s_j)\}_{j=1}^m, \boldsymbol{Y}_i\right\}_{i=1}^n$, training index set $\mathcal{I}_1$, testing index set $\mathcal{I}_2$, $|\mathcal{I}_1| = \lfloor 0.7n \rfloor$, $|\mathcal{I}_2| = \lfloor 0.3n \rfloor$, candidates for number of projection scores $\{J_0, \ldots, J_{N_1}\}$, candidates for depth $\{L_0, \ldots, L_{N_2}\}$, candidates for width $\{p_0, \ldots, p_{N_3}\}$, candidates for dropout rates $\{s_0, \ldots, s_{N_4}\}$, number of epochs, batch size, learning rate

Output: $\left\{\widehat{G}^{mfDNN}(X_i)\right\}_{i=1}^n$

Calculate $\widehat{\xi}_j^{(i)} = \int_{[0,1]^d} X_i(s)\psi_j(s)ds$ with pre-selected basis functions $\psi_j(s)$ (e.g., Fourier basis functions), and $\widehat{\boldsymbol{\xi}}_J^{(i)} = (\widehat{\xi}_{i1}, \ldots, \widehat{\xi}_{iJ})^\intercal$ with $J = \max\{J_0, \ldots, J_{N_1}\}$

For all candidates,

(i) Train $\widehat{\boldsymbol{f}}_{J_{\ell_1}, L_{\ell_2}, p_{\ell_3}, s_{\ell_4}}$ by Equation (2.4) with $\left(\{X_i(s_j)\}_{j=1}^m, \boldsymbol{Y}_i\right)_{i \in \mathcal{I}_1}$.

(ii) Compute $\text{err}(\ell_1, \ell_2, \ell_3, \ell_4) := \frac{1}{|\mathcal{I}_2|} \sum_{i \in \mathcal{I}_2} I(\widehat{\boldsymbol{f}}_{J_{\ell_1}, L_{\ell_2}, p_{\ell_3}, s_{\ell_4}} \ne \boldsymbol{Y}_i)$

Obtain $\left(J_{\ell_1^*}, L_{\ell_2^*}, p_{\ell_3^*}, s_{\ell_4^*}\right) = \arg\min_{\ell_1, \ell_2, \ell_3, \ell_4} \text{err}(\ell_1, \ell_2, \ell_3, \ell_4)$

Train $\widehat{\boldsymbol{f}}_{J_{\ell_1^*}, L_{\ell_2^*}, p_{\ell_3^*}, s_{\ell_4^*}}$ by Equation (2.4) with $\left(\{X_i(s_j)\}_{j=1}^m, \boldsymbol{Y}_i\right)_{i=1}^n$

3. Theoretical properties

3.1. Function class for the conditional probability π_k

To obtain the approximation rates, we assume that the underlying true conditional class probability function $\pi_k(\cdot)$ belongs to the class of Hölder-smooth functions defined as below. Without further notice, c, C, C_1, C_2, ..., represent some positive constants and can vary from line to line.

For $t \geq 1$, a measurable subset $D \subset \mathbb{R}^t$, and constants $\beta > 0$, $C > 0$, define

$$\begin{aligned}\mathcal{C}^{\beta}(D, C) \quad = \quad & \Bigg\{ f : D \mapsto \mathbb{R} \Big| \sum_{\boldsymbol{\alpha}:|\boldsymbol{\alpha}|<\beta} \|\partial^{\boldsymbol{\alpha}} f\|_{\infty} f \\ & + \sum_{\boldsymbol{\alpha}:|\boldsymbol{\alpha}|=\lfloor\beta\rfloor} \sup_{\boldsymbol{z},\boldsymbol{z}' \in D, \boldsymbol{z} \neq \boldsymbol{z}'} \frac{|\partial^{\boldsymbol{\alpha}} f(\boldsymbol{z}) - \partial^{\alpha} f(\boldsymbol{z}')|}{\|\boldsymbol{z} - \boldsymbol{z}'\|_{\infty}^{\beta - \lfloor\beta\rfloor}} \leq C \Bigg\},\end{aligned}$$

where $\partial^{\boldsymbol{\alpha}} = \partial^{\alpha_1} \dots \partial^{\alpha_t}$ denotes the partial differential operator with multi-index $\boldsymbol{\alpha} = (\alpha_1, \dots, \alpha_t) \in \mathbb{N}^t$, $|\boldsymbol{\alpha}| = \alpha_1 + \cdots + \alpha_t$. Equivalently, $\mathcal{C}^{\beta}(D, C)$ is the ball of β-Hölder smooth functions on D with radius R. A function $f : \mathbb{R}^t \to \mathbb{R}$ is said to be locally β-Hölder smooth if for any $a, b \in \mathbb{R}$, there exists a constant C (possibly depending on a, b) such that $f \in \mathcal{C}^{\beta}([a, b]^t, C)$.

For $q \geq 0, J \geq 1$, let $d_0 = J$ and $d_{q+1} = 1$. For $\boldsymbol{d} = (d_1, \dots, d_q) \in \mathbb{N}_+^q$, $\boldsymbol{t} = (t_0, \dots, t_q) \in \mathbb{N}_+^{q+1}$ with $t_u \leq d_u$ for $u = 0, \dots, q$, $\boldsymbol{\beta} := (\beta_0, \dots, \beta_q) \in \mathbb{R}_+^{q+1}$, let $\mathcal{G}(q, J, \boldsymbol{d}, \boldsymbol{t}, \boldsymbol{\beta})$ be the class of functions g satisfying a modular expression

$$g(\boldsymbol{z}) = g_q \circ \cdots \circ g_0(\boldsymbol{z}) \in (0, 1)\,, \ \forall \boldsymbol{z} \in \mathbb{R}^{d_0}, \tag{3.1}$$

where $g_u = (g_{u1}, \dots, g_{ud_{u+1}}) : \mathbb{R}^{d_u} \mapsto \mathbb{R}^{d_{u+1}}$ and $g_{uv} : \mathbb{R}^{t_u} \mapsto \mathbb{R}$ are locally β_u-Hölder smooth. The d_u arguments of g_u are locally connected in the sense that each component g_{uv} only relies on $t_u (\leq d_u)$ arguments. Similar structures have been considered by [28, 2, 22, 35, 4, 17, 16] in multivariate regression or classification to overcome high-dimensionality. Generalized additive model [15] and tensor product space ANOVA model [21] are special cases; see [22].

We define the class of $\boldsymbol{h} = \{h_1, \dots, h_K\}$ as

$$\begin{aligned}\mathcal{H} \equiv & \mathcal{H}\Big(K, \{q^{(k)}\}_{k=1}^K, \{\boldsymbol{d}^{(k)}\}_{k=1}^K, \{\boldsymbol{t}^{(k)}\}_{k=1}^K, \{\boldsymbol{\beta}^{(k)}\}_{k=1}^K, \{\alpha_k\}_{k=1}^K, \{\pi_k(\cdot)\}_{k=1}^K, \\ & \zeta(\cdot), \Gamma(\cdot), C, \rho, \epsilon\Big)\,,\end{aligned}$$

such that for any $J \geq 1$, $\pi_k^{(J)} \in \mathcal{G}\left(q^{(k)}, J, \boldsymbol{d}^{(k)}, \boldsymbol{t}^{(k)}, \boldsymbol{\beta}^{(k)}\right)$, where $\boldsymbol{d}^{(k)} = (d_1^{(k)}, \dots, d_{q^{(k)}}^{(k)}) \in \mathbb{N}_+^{q^{(k)}}$, $\boldsymbol{t}^{(k)} = (t_0^{(k)}, \dots, t_{q^{(k)}}^{(k)}) \in \mathbb{N}_+^{q^{(k)}+1}$ with $t_u^{(k)} \leq d_u^{(k)}$ for $u = 0, \dots, q^{(k)}$, $\boldsymbol{\beta}^{(k)} := (\beta_0, \dots, \beta_{q^{(k)}}^{(k)}) \in \mathbb{R}_+^{q^{(k)}+1}$. Note that this density class $\mathcal{H}$ includes many popular models studied in literature, both Gaussian and non-Gaussian; see Section 4.

Throughout the paper, we explore π_k in some complicated $\mathcal{G}$ with group-specific parameters $q^{(k)}, \boldsymbol{d}^{(k)}, \boldsymbol{t}^{(k)}$ and $\boldsymbol{\beta}^{(k)}$. The selection range of the truncation parameter J is based on the asymptotic order provided in Assumptions 3 and 4 in the next section. Although $\pi_k^{(J)}$ has J arguments, it involves at most $t_0^{(k)} d_1^{(k)}$ effective arguments, implying that the two population densities differ by a small number of variables. Relevant conditions are necessary for high-dimensional classification. For instance, in high-dimensional Gaussian data classification, [6, 5] show that, to consistently estimate Bayes classifier, it is necessary that the mean vectors differ at a small number of components. The modular structure holds for arbitrary J, which may be viewed as a extension of [28] in the functional data analysis setting.

3.2. Approximation and boundary conditions for the conditional probability $\boldsymbol{\pi_k}$

One of the unique phenomenons of the functional data is the intrinsically infinite dimension. Hence, the finite approximation condition for $\boldsymbol{\pi}$ is necessary. For any positive integer J, define $\boldsymbol{\xi}_J = (\xi_1, \ldots, \xi_J)^\intercal$ and $\boldsymbol{\pi}_J(\boldsymbol{x}_J) = (\pi_1^{(J)}(\boldsymbol{x}_J), \ldots, \pi_K^{(J)}(\boldsymbol{x}_J))^\intercal$, where $\pi_k^{(J)}(\boldsymbol{x}_J) = \mathrm{P}(\boldsymbol{Y}_k = \boldsymbol{e}_k | \boldsymbol{\xi}_J = \boldsymbol{x}_J)$, $k = 1, \ldots, K$.

Assumption 1. *(Approximation error of $\pi_k^{(J)}$) There exist a constant $J_0 \geq 1$ and decreasing functions $\zeta(\cdot) : [1, \infty) \to \mathbb{R}_+$ and $\Gamma(\cdot) : [0, \infty) \to \mathbb{R}_+$, with $\sup_{J \geq 1} J^\rho \zeta(J) < \infty$ for some $\rho > 0$ and $\int_0^\infty \Gamma(x) dx < \infty$, such that for any $J \geq J_0$, $k = 1 \ldots, K$ and $x > 0$,*

$$\mathrm{P}\left(|\pi_k(\boldsymbol{\xi}) - \pi_k^{(J)}(\boldsymbol{\xi}_J)| \geq x \right) \leq \zeta(J)\Gamma(x). \tag{3.2}$$

Remark 2. *Any finite number of projection scores $\boldsymbol{\xi}_J$ can not recover the information of $\boldsymbol{\xi}$ totally. However, in practice, it is infeasible to utilize an infinity number of project scores. Hence, Assumption 1 provides a uniform upper bound on the probability of π_k differing from $\pi_k^{(J)}$ by at least x, which approaches zero if either J or x goes to infinity, implying that when dimension is reduced, there exists some large enough J, such that $\pi_k^{(J)}$ is an accurate approximation of π_k with high probability. ρ controls the decay rate of function $\zeta(J)$, and larger ρ implies a faster shrinkage of the desired probability.*

Assumption 2. *(Boundary condition)*

(a) *There exists a relatively small $\epsilon > 0$, such that for all $k = 1, \ldots, K$, one has $\mathrm{P}(\pi_k(\boldsymbol{\xi}) > \epsilon) = 1$;*

(b) *There exist an absolute constant $C > 0$ and some positive vector $\boldsymbol{\alpha} = (\alpha_1, \ldots, \alpha_K)^\intercal$ such that*

$$\mathrm{P}(\pi_k(\boldsymbol{\xi}) \leq x) \leq C x^{\alpha_k}, \qquad \forall x \in (0, 1], \forall k \in \{1, \ldots, K\}.$$

Remark 3. *Assumption 2 (a) provides a uniform lower bound of π_k, indicating that π_k is bounded away from zero with probability one. In another word, for*

some k, $\pi_k \equiv 0$ indicates there only exist $K-1$ classes. Assumption 2 (b) describes the behaviour of π_k around zero. Trivially, when $\alpha_k = 0$ for $k = 1, \ldots, K$, the condition holds universally only if $C \geq 1$. Specifically, Assumption 2 controls the decay rate of the probability measure of $\{\pi_k(\boldsymbol{\xi}) : 0 \leq \pi_k(\boldsymbol{\xi}) \leq x\}$. Similar conditions can be found for multi-class classification for multivariate data in [4] and binary classification in [24] and [34]. It can be seen that Assumption 2 (a) and (b) are closely related in certain circumstances. For example, given some $\epsilon > 0$, Assumption 2(a) implies Assumption 2(b) for arbitrary $\boldsymbol{\alpha}$ and $C = \max_k \epsilon^{-\alpha_k}$. However, they are not equivalent in most scenarios when C, $\{\boldsymbol{\alpha}_k\}_{k=1}^K$, and ϵ are all universally positive constants.

Both Assumptions 1 and 2 can be verified by two concrete examples in Section 4.

3.3. Convergence of Kullback-Leibler divergence

3.3.1. Kullback-Leibler divergence

For the true probability distribution $\boldsymbol{\pi}(\boldsymbol{x}) = (\pi_1(\boldsymbol{x}), \ldots, \pi_K(\boldsymbol{x}))^\intercal$ and any generic estimation $\widehat{\boldsymbol{\pi}}(\boldsymbol{x}) = (\widehat{\pi}_1(\boldsymbol{x}), \ldots, \widehat{\pi}_K(\boldsymbol{x}))^\intercal$, define the corresponding discrete version Kullback-Leibler (KL) divergence

$$KL\left(\boldsymbol{\pi}(\boldsymbol{x}), \widehat{\boldsymbol{\pi}}(\boldsymbol{x})\right) = \sum_{k=1}^{K} \pi_k(\boldsymbol{x}) \log\left(\frac{\pi_k(\boldsymbol{x})}{\widehat{\pi}_k(\boldsymbol{x})}\right).$$

For any $\widehat{\boldsymbol{\pi}}$ trained with $\{(X_i(s), \boldsymbol{Y}_i)\}_{i=1}^n$, we evaluate its performance by the log-likelihood ratio

$$\mathbb{E}\left[\sum_{k=1}^{K} Y_k \log\left(\frac{\pi_k(\boldsymbol{\xi})}{\widehat{\pi}_k(\boldsymbol{\xi})}\right)\right] = \mathbb{E}\left[KL\left(\boldsymbol{\pi}(\boldsymbol{\xi}), \widehat{\boldsymbol{\pi}}(\boldsymbol{\xi})\right)\right].$$

Note that the estimation risk is associate with the well-known CE loss in Section 2.2.

To formulate our classification task, we first introduce two mild assumptions for $\pi_k(\cdot)$. Without further notice, c, C, C_1, C_2, ..., represent some positive constants and can vary from line to line.

Different from the popular least square loss, the CE loss is the expectation with respect to the input distribution of the KL divergence of the conditional class probabilities. If anyone of the conditional class probabilities has zero estimation while the underlying conditional class probability is positive, the risk can even become infinite. To avoid the infinite risk, we truncate the CE loss function and derive convergence rates without assuming either the true conditional class probabilities or the estimators away from zero. Instead, our misclassification risks depend on an index quantifying the behaviour of the conditional class probabilities near zero.

Given an absolute constant $C_0 \geq 2$, for any classifier $\widehat{\boldsymbol{\pi}}$, define the truncated KL risk for some density $\boldsymbol{h}$ as

$$R_{\boldsymbol{h},C_0}(\widehat{\boldsymbol{\pi}}) = \mathrm{E}_{\boldsymbol{h}} \left\{ \sum_{k=1}^{K} \pi_k(\boldsymbol{\xi}) \left[C_0 \wedge \log\left(\frac{\pi_k(\boldsymbol{\xi})}{\widehat{\pi}_k(\boldsymbol{\xi})} \right) \right] \right\}. \tag{3.3}$$

Remark 4. *C_0 is commonly introduced to avoid infinity value of the ordinary KL risk. It is trivial to show that C_0 can only be abandoned when $\widehat{\pi}_k$ is lower bounded by $\exp(-C_0)$ for all k. When $\widehat{\pi}_k$ has a large deviation from π_k for some k, the KL risk explodes to infinity, which makes it is infeasible to evaluate the performance of $\widehat{\boldsymbol{\pi}}$.*

3.3.2. Convergence rate for fully observed functional data

In this section, we provide the non-asymptotic KL risk of mfDNN classifier. Let

$$(\widehat{k}, \widehat{u}) = \arg \max_{\substack{(k,u),k=1\ldots,K,\\ u=0,\ldots,q^{(k)}}} \frac{t_u^{(k)}}{\tilde{\beta}_u^{(k)}}, \tag{3.4}$$

where $\tilde{\beta}_u^{(k)} = \beta_u^{(k)} \prod_{\ell=u+1}^{q^{(k)}} \beta_\ell^{(k)} \wedge 1$, and

$$(\widehat{l}, \widehat{v}) = \arg \max_{\substack{(k,u),k=1\ldots,K,\\ u=0,\ldots,q^{(k)}}} \frac{t_u^{(k)}}{\tilde{\beta}_u^{(k)}(1+\alpha_k)}.$$

Let $\theta = \frac{(1+\alpha_{\widehat{l}})\tilde{\beta}_{\widehat{v}}^{(\widehat{l})}}{(1+\alpha_{\widehat{l}})\tilde{\beta}_{\widehat{v}}^{(\widehat{l})} + t_{\widehat{v}}^{(\widehat{l})}}$ and $\nu = \frac{\theta t_{\widehat{u}}^{(\widehat{k})}}{\tilde{\beta}_{\widehat{u}}^{(\widehat{k})}(1+\widetilde{\alpha})}$, where $\widetilde{\alpha} = \min_k \alpha_k \wedge 1$.

To facilitate a clearer understanding of the aforementioned notations, we offer the following intuitive explanations.

Remark 5. *Firstly, $\widehat{k}$ and $\widehat{u}$ serve as indices for the most complex function across all classes and each of their compositions. In consideration of the margin condition, $\widehat{l}$ denotes the class that is the most challenging to classify, and $\widehat{v}$ represents the most intricate composition function within class $\widehat{l}$. The margin parameter $\widetilde{\alpha}$ is specifically assigned to the most formidable class in the classification process, spanning all classes. The smoothness parameter $\widetilde{\beta}_u^{(k)}$ signifies the smoothness order at the u-th layer for the k-th class, taking into account information from all preceding layers. The parameter θ denotes a typical convergence rate in nonparametric classification problems (see [34, 4]), tailored to the composition functions in the multi-class classification setting.*

Assumption 3. *There exist some constants C_1, C_2', C_2, C_3, only depending on $\mathcal{H}$ and C_0, such that the DNN class $\mathcal{F}(L, J, \boldsymbol{p}, s)$ satisfies*

(a) $L \leq C_1 \log n$;
(b) $C_2' n^{\theta/\rho} \leq J \leq C_2 n^{\nu}$;

(c) $\max_{1\le\ell\le L} p_\ell \le C_2 n^\nu$;
(d) $s \le C_3 n^\nu \log n$.

Remark 6. *Assumption 3 provides exact orders on* $(L, \boldsymbol{p}, s)$ *for network, respectively. Assumption 3(b) provides the precise range on* J*. It is worth mentioning this condition implies* $\rho \ge \theta\nu^{-1}$*, meaning the function* $\zeta(J)$ *converges to zero in a relatively fast rate when* $J \to \infty$*. This is because the KL risk associate with a classifier constructed using finite components through dimension reduction, and the discrepancies between these finite components and their infinite counterparts. We set a minimum threshold for* ρ *to prevent the discrepancies from overshadowing the error bound, thereby highlighting the benefits of employing DNN for the classification of complex multi-class relationships.*

In the following, we provide the convergence rate in the ideal case when the entire functional curve is fully observed.

Theorem 3.1. *There exist a positive constant* ω_1*, only depending on* $\mathcal{H}$ *and* C_0*, such that*

$$\sup_{\boldsymbol{h}\in\mathcal{H}} R_{\boldsymbol{h},C_0}\left(\widehat{\boldsymbol{f}}_\phi\right) \le \omega_1 n^{-\theta}\log^3 n,$$

where network classifier $\widehat{\boldsymbol{f}}_\phi$ *belongs to* $\mathcal{F}(L, J, \boldsymbol{p}, s)$ *in Assumption 3.*

Remark 7. *Theorem 3.1 provides the upper bound of the KL misclassification risk of the proposed mfDNN. When* $K = 2$*, the multiclass classification downgrades to the binary classification problem. Compared with the minimax excess misclassification risk derived in [36] for the binary classification, the upper bound rate in Theorem 3.1 provides a slightly larger order, Specifically, the leading term of the upper bound rate in Theorem 3.1 is* $n^{-\theta}$ *with* $\theta = \frac{(1+\alpha)\beta}{(1+\alpha)\beta+t}$*, while the leading terms in Theorem 1 in [36] is* n^{-S_0} *with* $S_0 = \frac{(\alpha+1)\beta}{(\alpha+2)\beta+t}$*. Hence, when* β*'s, i.e. the degrees of Hölder smooth functions in the class* $\mathcal{H}$*, are large enough, the discrepancy between these two bounds are rarely negligible. Hence, to reduce potential slightly larger risks, we recommend [36] for the binary classification problems and when the distribution functions are not smooth enough. Meanwhile, we recommend the mfDNN classifier for multiclassification problems regardless the smoothness of the conditional distribution functions.*

We assume the number of classes K *is a constant. The decay order* θ *does not depend on* K *and is instead determined by the class which presents the most significant classification challenge. In another word, even with arising number of classes, as long as the difficulty of classification remains the same, the asymptotic convergence rate does not change. Also, the constant* ω_1 *indeed absorbs the information of* K*, but it does not affect the asymptotic convergence rate for* K*.*

In Appendix B, Figure 3 describes a numerical comparison between the theoretical KL misclassification risk and the empirical risk. It shows that the theoretical convergence rate is tight enough to bound the empirical risk.

3.3.3. Convergence rate for discretely observed functional data

Practically, it is usually unrealistic to observe the full trajectory of each individual, thus the rate in Theorem 3.1 can only be reached if sampling frequency is dense enough. Hence, it is interesting to discuss the upper bound of the risk of mfDNN classifier when functional data are discretely observed at m occasions for each subject. Let $\widetilde{\beta} = \max_{k=1\ldots,K}\left(\widetilde{\beta}_0^{(k)} \wedge 1\right)$, $\theta' = \tau\widetilde{\beta}$, and $\nu' = \frac{\theta' t_{\widehat{u}}^{(\widehat{k})}}{\widetilde{\beta}_{\widehat{u}}^{(\widehat{k})}(1+\widetilde{\alpha})}$, where $\widehat{k}$ is defined in (3.4) and τ is a positive universal constant.

Assumption 4. *There exist some constants C_1, C_2', C_2, C_3, $\widetilde{C}_1, \widetilde{C}_2', \widetilde{C}_2$, and $\widetilde{C}_3$ only depending on $\mathcal{H}$, C_0 and τ, and a phase transition point $m^* \in \mathbb{N}^+$, such that the DNN class $\mathcal{F}(L, J, \boldsymbol{p}, s)$ satisfies*

(a) $L \le C_1 \log n \mathbb{I}(m \ge m^*) + \widetilde{C}_1 \log m \mathbb{I}(m < m^*)$;
(b) $C_2' n^{\theta/\rho}\mathbb{I}(m \ge m^*) + \widetilde{C}_2' m^{\theta'/\rho}\mathbb{I}(m < m^*) \le J \le C_2 n^{\nu}\mathbb{I}(m \ge m^*) + \widetilde{C}_2 m^{\nu'}\mathbb{I}(m < m^*)$;
(c) $\max_{1\le\ell\le L} p_\ell \le C_2 n^{\nu}\mathbb{I}(m \ge m^*) + \widetilde{C}_2 m^{\nu'}\mathbb{I}(m < m^*)$;
(d) $s \le C_3 n^{\nu} \log n \mathbb{I}(m \ge m^*) + \widetilde{C}_3 m^{\nu'} \log m \mathbb{I}(m < m^*)$.

Similar to Assumption 3, Assumption 4 provides exact orders on $L, \boldsymbol{p}, s$, and range of J when sampling frequency m is involved. When $m \ge m^*$, Assumption 4 coincides with Assumption 3 for dense functional data.

The following theorem provides the phase transition rate when functional data are discretely observed on m locations at a certain rate with respect to τ.

Theorem 3.2. *When $E|\xi_j - \widehat{\xi}_j| \lesssim m^{-\tau}$ for all $j = 1, \ldots, J$, there exists positive constants ω_1, ω_2 and ω_3 only depending on $\mathcal{H}$, C_0 and τ, such that*

$$\sup_{\boldsymbol{h}\in\mathcal{H}} R_{\boldsymbol{h},C_0}\left(\widehat{\boldsymbol{f}}_\phi\right) \le \omega_1 n^{-\theta}\log^3 n \mathbb{I}(m \ge m^*) + \omega_2 m^{-\theta'}\mathbb{I}(m < m^*),$$

where $m^* = \lfloor \left(\omega_3 n^{\theta}/\log^3 n\right)^{1/\theta'} \rfloor$, and $\widehat{\boldsymbol{f}}_\phi \in \mathcal{F}(L, J, \boldsymbol{p}, s)$ defined in Assumption 4.

Remark 8. *Theorem 3.2 reveals that m^* is a critical sampling frequency for the rate of KL misclassification risk over the parameter space $\mathcal{H}$ at any fixed sample size n. When $m \ge m^*$, the KL risk is of rate $\omega_1 n^{-\theta}\log^3 n$ which is free of m and is consistent with the rate derived in Theorem 3.1. In other words, when $m \ge m^*$, the optimal classifier performs as well as the one based on fully observed data. When $m < m^*$, the KL risk is of rate $m^{-\theta'}$, which is solely dependent on m. Note that m^* is increasing with n, indicating that when sample size is sufficient enough, a higher sampling frequency is required to achieve the convergence rate of fully observed data.*

Remark 9. *When functional curves are discretely observed, Theorem 3.2 provides the convergence rate of truncated KL risk when the biases of projection scores are uniformly bounded by $m^{-\tau}$. This assumption does not hold universally for all scenarios. Nevertheless, it can be satisfied in various examples.*

For any empirical process on $[0,1]$*, if we use Fourier basis with* m *terms to decompose the curve, we can show that*

$$E|\xi_j - \widehat{\xi}_j| \leq O\left(\max_{k=1,\ldots,K} \sqrt{\sum_{j=m+1}^{\infty} \left(\lambda_{kj} + \mu_{kj}^2\right)}\right),$$

where $\mu_{kj} = \mathrm{E}\,\xi_j$ *in the* k*-th group. As* $\{\lambda_{kj}\}_{j=1}^{\infty}$ *and* $\left\{\mu_{kj}^2\right\}_{j=1}^{\infty}$ *are both convergent, when* λ_{kj} *and* μ_{kj}^2 *are decreasing no faster than some polynomial order, the assumption easily follows.*

Another well-known example is the FPCA provided by [14]. According to Theorem 1 in [14], for all $j = 1, \ldots, J$*, the estimators of projection scores satisfy* $\mathrm{E}\,|\xi_j - \widehat{\xi}_j| \leq \max_k \left(\int_{[0,1]^d \times [0,1]^d} \left(\Omega_k - \widehat{\Omega}_k\right)^2 (s,s')dsds'\right)^{1/2}$*. Hence,* $E|\xi_j - \widehat{\xi}_j| \lesssim m^{-\tau}$ *holds easily when* $\max_k \left(\int_{[0,1]^d \times [0,1]^d} \left(\Omega_k - \widehat{\Omega}_k\right)^2 (s,s')dsds'\right)$ *is bounded properly.*

4. Examples

In this section, we provide two examples of exponential families to justify the validation of our model assumptions, and emphasize the necessity of applying DNN approach owing to the complicated structure of data population. For simplicity, we assume the prior probability satisfies $\mathbb{P}(\boldsymbol{Y} = \boldsymbol{e}_k) = k^{-1}$ for all k throughout the section.

4.1. Independent exponential family

We first consider independent projection scores, which are from exponential families. For some collection of unknown parameters $\{\theta_{kj}\}_{k=1,j=1}^{K,\infty}$, and unknown collections of functions $\{\eta_{kj}\}_{k=1,j=1}^{K,\infty}$, $\{U_{kj}\}_{k=1,j=1}^{K,\infty}$, and $\{W_{kj}\}_{k=1,j=1}^{K,\infty}$, we consider the k-th class conditional density, such that

$$\boldsymbol{\xi} | \{\theta_{kj}\}_{j=1}^{\infty}, \boldsymbol{Y} = \boldsymbol{e}_k \sim h_k(\boldsymbol{x}) = \exp\left(\sum_{j=1}^{\infty} \eta_{kj}(\theta_{kj}) U_{kj}(x_j) + W_{kj}(x_j)\right).$$

For all $1 \leq k, k' \leq K$, let $A_{kk'} = \{j : \eta_{kj}(\theta_{kj})U_{kj}(x_j) \neq \eta_{k'j}(\theta_{k'j})U_{k'j}(x_j), \forall x_j \in \mathbb{R}\}$ and $B_{kk'} = \{j : W_{kj} \neq W_{k'j}\}$ be the two sets identifying the difference between h_k and $h_{k'}$. Therefore, we have the pairwise log likelihood

$$\begin{aligned}\log\left(h_k/h_{k'}\right) \quad = \quad & \sum_{j \in (A_{kk'} \bigcup B_{kk'})} \{[\eta_{kj}(\theta_{kj})U_{kj}(x_j) - \eta_{k'j}(\theta_{k'j})U_{k'j}(x_j)] \\ & + [W_{kj}(x_j) - W_{k'j}(x_j)]\}.\end{aligned}$$

Given some universal constant $N_{kk'}$, when $|A_{kk'} \bigcup B_{kk'}| \leq N_{kk'}$, there exists a positive integer $J_{max} = \max \bigcup_{k,k'} (A_{kk'} \bigcup B_{kk'})$, such that $\pi_k = \pi_k^{(J)}$ for all $J \geq J_{max}$.

By definition, Assumption 2(a) holds for $\alpha = 1$. Assumption 2(b) holds when $h_k/h_{k'}$ is bounded for all pairs. Note that it is trivial when $\{h_k\}_{k=1}^K$ share the same $\{U_{kj}\}_{k=1,j=1}^{K,\infty}$ and $\{W_{kj}\}_{k=1,j=1}^{K,\infty}$, such as Gaussian distribution, student's t distribution and exponential distribution, whose density ratio of their kind is always bounded. Assumption 1 holds for $J_0 = J_{max}$, and for arbitrary function $e(\cdot)$ with exponential tails and density $\pi(\cdot)$. Since $\pi_k = \left[1 + \sum_{k' \neq k} \log(h_k/h_{k'})\right]^{-1}$, the smoothness is determined by $\{\eta_{kj}\}_{k=1,j=1}^{K,\infty}$, $\{U_{kj}\}_{k=1,j=1}^{K,\infty}$, and $\{W_{kj}\}_{k=1,j=1}^{K,\infty}$, thus $\boldsymbol{h}$ is trivially in some $\mathcal{H}$.

4.2. Exponential family with in-block interaction

In this example, we consider ξ_j are dependent with each other in a block, but independent across blocks, which is an extension of the example in Section 4.1. Given a sequence of positive integers $\{\ell_p\}_{j=1}^{\infty}$, such that $0 = \ell_1 < \ell_2 < \ldots$, we define the p-th group index set $\mathcal{E}_p = \{\ell_p + 1, \ldots, \ell_{p+1}\}$, such that the cardinality $|\mathcal{E}_p| = \ell_{p+1} - \ell_p$, therein grouping are based on adjacent members for simplicity. For a collection of unknown parameters $\{\boldsymbol{\theta}_{kp}\}_{k=1,p=1}^{K,\infty}$, and unknown collections of functions $\{\widetilde{\eta}_{kp}\}_{k=1,p=1}^{K,\infty}$, $\left\{\widetilde{U}_{kp}\right\}_{k=1,p=1}^{K,\infty}$, and $\left\{\widetilde{W}_{kp}\right\}_{k=1,p=1}^{K,\infty}$, where $\widetilde{U}_{kp}$ and $\widetilde{W}_{kp}$ are functions from $\mathbb{R}^{|\mathcal{E}_p|}$ to $\mathbb{R}$.

Consider the joint conditional density

$$h_k(\boldsymbol{x}) = \exp\left(\sum_{p=1}^{\infty} \eta_{kp}(\boldsymbol{\theta}_{kp})\widetilde{U}_{kp}(\boldsymbol{x}_p) + \widetilde{W}_{kp}(\boldsymbol{x}_p)\right)$$

for class k, where $\boldsymbol{x}_p = (x_j)_{j=\ell_p+1,\ldots,\ell_{p+1}}$.

For any $1 \leq k, k' \leq K$, define the density difference sets

$$\widetilde{A}_{kk'} = \left\{p : \widetilde{\eta}_{kp}(\boldsymbol{\theta}_{kp})\widetilde{U}_{kp}(\boldsymbol{x}_p) \neq \eta_{k'p}(\boldsymbol{\theta}_{k'p})\widetilde{U}_{k'p}(\boldsymbol{x}_p), \forall \boldsymbol{x}_p \in \mathbb{R}^{|\mathcal{E}_p|}\right\}$$

and $\widetilde{B}_{kk'} = \left\{p : \widetilde{W}_{kp} \neq \widetilde{W}_{k'p}\right\}$, and the pairwise log likelihood is thus given by

$$\begin{aligned}&\log\left(h_k/h_{k'}\right)\\ =&\sum_{p \in \widetilde{A}_{kk'} \bigcup \widetilde{B}_{kk'}} \left\{\left[\widetilde{\eta}_{kp}(\boldsymbol{\theta}_{kp})\widetilde{U}_{kp}(\boldsymbol{x}) - \widetilde{\eta}_{k'p}(\boldsymbol{\theta}_{k'p})\widetilde{U}_{k'p}(\boldsymbol{x})\right] + \left[\widetilde{W}_{kp}(\boldsymbol{x}) - \widetilde{W}_{k'p}(\boldsymbol{x})\right]\right\}.\end{aligned}$$

Given some finite positive number $N_{kk'}$, such that $|A_{kk'} \bigcup B_{kk'}| \leq N_{kk'}$, the verification can be similarly derived from Section 4.1.

5. Simulation studies

In this section, we provide numerical evidences to demonstrate the superior performance of mfDNN. In all simulations, we generated $n_k = 200, 350, 700$ training samples for each class, and testing sample sizes $100, 150$, and 300, respectively. We denote m as the sampling frequency, meaning that we choose m observations evenly on the domain of the functional data. For mfDNN, we use tensor of Fourier basis to extract projection scores by integration. The structure parameters $(L, J, \boldsymbol{p}, s)$ are selected by Algorithm 1, where the candidates are given based on Theorem 3.2. To simplify the structure, each element of vector $\boldsymbol{p}$ takes the same value. Particularly, the candidates of $L = 2$ or 3, J and $\boldsymbol{p}$ are increasing with n and m, with three candidates. For example, when we consider the scenario with $n_k = 200$ and $m = 8$, the candidates for J are $3, 4$ and 5 and the candidates for $\boldsymbol{p}$ are $30, 50$ and 100. When the sample size grows to $n_k = 700$ with more observations $m = 125$, we increase the values of candidates for J as $20, 30$ and 40 and the values of candidates for $\boldsymbol{p}$ as $300, 500$ and 800. The sparsity parameter is $s = 0.2, 0.5$ or 0.8. We summarize R codes and examples for the proposed mfDNN algorithms on `GitHub` (https://github.com/FDASTATAUBURN/mfdnn). To achieve the sparsity of the DNN, we use dropout techniques. Alternatively, one can also apply the ℓ_1 regularizer to achieve sparse neural networks.

The computation job of this paper is conducted at Yale University high performance computing center, where each compute node containing multiple CPU cores and substantial memory (RAM). The specific hardware configurations may vary, but it always includes 1 CPU and 8 GB RAM as default for each replicates of the parallel computing.

5.1. Alternative methods

Based on the fact that there is no existing multi-classification method specifically designed for multidimensional functional data, for comparison, we first consider two sparse discriminate analysis methods: ℓ_1 penalized Fisher's discriminant analysis (MSDA) approach introduced in [23] and penalized linear discriminant analysis (PLDA) classifier in [38]. Both of them are efficient classifiers designed for high-dimensional i.i.d. observations. To make these two methods directly applicable to functional data, we first pre-processed 2D or 3D functional data by vectorization. The realization is via the `R` packages `msda` and `PenalizedLDA`, where the tuning parameter candidates for the penalty term are generated by default. We use the default five-fold and six-fold cross-validation to tune MSDA and PLDA, respectively.

We also consider two machine learning alternatives: random forest and convolutional neural network (CNN). Specifically, we apply random forest to the extracted projection scores, referred to as mfRF, to test the efficiency of DNN in analyzing multi-dimensional functional data. We set the default number of trees as 500, and the number of features considered at each split in the decision

trees as 3. The implementation of mfRF is via the `R` package `randomForest`. We employ a promising approach in imaging processing, CNN method, to conduct the 2D and 3D functional data classification. It is implemented by `R` packages `keras` and `tensorflow`. Specifically, we first construct the CNN model with a convolutional layer consisting of 32 filters of size 2×2. The ReLU activation function is applied to introduce the non-linearity. Next, a max pooling layer with a pool size of 2×2 is added to reduce the spatial dimensions of the data while retaining crucial features. The model then proceeds with another convolutional layer, this time with 64 filters of size 2×2 and ReLU activation. Subsequently, another max pooling layer with pool size 2×2 is included. To prevent overfitting, a dropout layer with a dropout rate 0.25 is introduced. The data is then flattened to convert it from a 2D or 3D tensor into a 1D vector. The flattened data is passed through a dense layer with 128 units and ReLU activation, followed by another dropout layer with a dropout rate of 0.5. In settings with fewer sampling frequency, we adjust the size of filters and pool to accommodate the data.

5.2. 2D functional data

For $k = 1, 2, 3$, we generated functional data $X_i^{(k)}(s, s') = \sum_{j=1}^{5} \xi_{ij}^{(k)} \psi_j(s, s')$, $s, s' \in [0, 1]$. Let $\psi_1(s, s') = s$, $\psi_2(s, s') = s'$, $\psi_3(s, s') = ss'$, $\psi_4(s, s') = s^2 s'$, $\psi_5(s, s') = s{s'}^2$. Define $\mathbf{1}_k$ be a $k \times 1$ vector with all the elements one. We specify the distribution of $\xi_{ij}^{(k)}$'s as following.

Model 1 (2D Gaussian): Let $\left(\xi_{i1}^{(k)}, \ldots, \xi_{i5}^{(k)}\right)^\intercal \sim N(\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)$, where $\boldsymbol{\mu}_1 = (4, 4, 3, 3, 3)^\intercal$, $\boldsymbol{\Sigma}_1^{1/2} = \operatorname{diag}(8, 7, 6, 5, 4)$, $\boldsymbol{\mu}_2 = -\mathbf{1}_5$, $\boldsymbol{\Sigma}_2^{1/2} = \operatorname{diag}(5, 4, 3, 2, 1)$, $\boldsymbol{\mu}_3 = \mathbf{0}_5$, $\boldsymbol{\Sigma}_3^{1/2} = \operatorname{diag}(2.5, 2, 1.5, 1, 0.5)$.

Model 2 (2D Mixed 1): Let $\left(\xi_{i1}^{(k)}, \ldots, \xi_{i5}^{(k)}\right)^\intercal \sim N(\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)$ for $k = 1, 2$, and $\xi_{ij}^{(3)} \sim t_{2j+1}(\nu_j)$, where $\boldsymbol{\mu}_1 = -\mathbf{1}_5$, $\boldsymbol{\Sigma}_1^{1/2} = \operatorname{diag}(5, 4, 3, 2, 1)$, $\boldsymbol{\mu}_2 = \mathbf{0}_5$, $\boldsymbol{\Sigma}_2^{1/2} = \operatorname{diag}\left(\frac{5}{2}, 2, \frac{3}{2}, 1, \frac{1}{2}\right)$, $(\nu_1, \ldots, \nu_5)^\intercal = 3 \cdot \mathbf{1}_5$.

Model 3 (2D Mixed 2): Let $\left(\xi_{i1}^{(1)}, \ldots, \xi_{i5}^{(1)}\right)^\intercal \sim N(\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)$, $\xi_{ij}^{(2)} \sim t_{j+1}(\nu_{2j})$, $\xi_{ij}^{(2)} \sim t_{2j+1}(\nu_{3j})$, where $\boldsymbol{\mu}_1 = \mathbf{0}_5$, $\boldsymbol{\Sigma}_1^{1/2} = \operatorname{diag}\left(\frac{5}{2}, 2, \frac{3}{2}, 1, \frac{1}{2}\right)$, $(\nu_{21}, \ldots, \nu_{25})^\intercal = \mathbf{1}_5$, $(\nu_{31}, \ldots, \nu_{35})^\intercal = 3 \cdot \mathbf{1}_5$.

Model 4 (2D Mixed 3): Let $\xi_{ij}^{(1)} \sim Exp(r_{1j})$, $\xi_{ij}^{(2)} \sim t_{2j+1}(\nu_{2j})$, and $\left(\xi_{i1}^{(3)}, \ldots, \xi_{i5}^{(3)}\right)^\intercal \sim N(\boldsymbol{\mu}_3, \boldsymbol{\Sigma}_3)$, where $(r_{11}, \ldots, r_{15})^\intercal = (0.1, 0.3, 0.5, 0.7, 0.9)^\intercal$, $(\nu_{21}, \ldots, \nu_{25})^\intercal = 3 \times \mathbf{1}_5$, $\boldsymbol{\mu}_3 = \mathbf{0}_5$, $\boldsymbol{\Sigma}_3^{1/2} = \operatorname{diag}(2.5, 2, 1.5, 1, 0.5)$.

For each model, we observe the functional data on 3×3, 5×5, 10×10, and 20×20 grid points over $[0, 1]^2$, respectively. As a result, the sampling frequency $m = 9, 25, 100, 400$, which indicates that the functional observations are from sparse to dense. Tables 1 to 4 demonstrate the results of 100 simulations. For mfDNN, it can be seen that the misclassification risks decrease as the sample size n increasing, as well as the increase of the sampling frequency m. This founding

TABLE 1
Averaged misclassification rates with standard errors in brackets for 2D simulations when $m = 9$ over 100 replicates.

Model	n_k	$m = 9$				
		mfDNN	MSDA	PLDA	mfRF	CNN
2D Gaussian	200	0.259	**0.246**	0.323	0.253	0.458
		(0.031)	(0.027)	(0.028)	(0.028)	(0.022)
	350	**0.232**	0.243	0.325	0.238	0.453
		(0.020)	(0.022)	(0.022)	(0.023)	(0.015)
	700	**0.227**	0.241	0.323	0.224	0.442
		(0.014)	(0.014)	(0.016)	(0.013)	(0.010)
2D Mixed 1	200	0.158	**0.150**	0.227	0.174	0.399
		(0.020)	(0.021)	(0.026)	(0.023)	(0.014)
	350	0.153	**0.144**	0.227	0.166	0.386
		(0.015)	(0.016)	(0.020)	(0.019)	(0.011)
	700	**0.152**	0.156	0.229	0.156	0.368
		(0.011)	(0.014)	(0.014)	(0.013)	(0.007)
2D Mixed 2	200	0.166	**0.152**	0.198	0.181	0.499
		(0.023)	(0.022)	(0.025)	(0.023)	(0.054)
	350	0.165	**0.152**	0.200	0.176	0.451
		(0.016)	(0.016)	(0.022)	(0.020)	(0.035)
	700	0.163	**0.148**	0.197	0.170	0.331
		(0.011)	(0.012)	(0.013)	(0.014)	(0.025)
2D Mixed 3	200	0.168	**0.115**	0.264	0.246	0.527
		(0.030)	(0.018)	(0.023)	(0.025)	(0.053)
	350	0.164	**0.115**	0.265	0.241	0.439
		(0.024)	(0.014)	(0.021)	(0.020)	(0.039)
	700	0.156	**0.114**	0.264	0.237	0.321
		(0.014)	(0.011)	(0.016)	(0.014)	(0.024)

TABLE 2
Averaged misclassification rates with standard errors in brackets for 2D simulations when $m = 25$ over 100 replicates.

Model	n_k	$m = 25$				
		mfDNN	MSDA	PLDA	mfRF	CNN
2D Gaussian	200	**0.205**	0.245	0.325	0.250	0.457
		(0.026)	(0.028)	(0.028)	(0.028)	(0.020)
	350	**0.200**	0.243	0.328	0.237	0.451
		(0.020)	(0.021)	(0.022)	(0.022)	(0.014)
	700	**0.196**	0.241	0.325	0.224	0.435
		(0.014)	(0.015)	(0.016)	(0.014)	(0.012)
2D Mixed 1	200	**0.100**	0.150	0.229	0.171	0.391
		(0.018)	(0.022)	(0.025)	(0.022)	(0.012)
	350	**0.085**	0.144	0.229	0.164	0.374
		(0.012)	(0.016)	(0.019)	(0.018)	(0.011)
	700	**0.081**	0.123	0.231	0.155	0.354
		(0.010)	(0.014)	(0.014)	(0.013)	(0.006)
2D Mixed 2	200	**0.140**	0.153	0.200	0.183	0.453
		(0.020)	(0.022)	(0.025)	(0.022)	(0.024)
	350	**0.135**	0.152	0.202	0.175	0.389
		(0.016)	(0.016)	(0.022)	(0.020)	(0.030)
	700	**0.123**	0.148	0.199	0.169	0.237
		(0.010)	(0.012)	(0.013)	(0.014)	(0.022)
2D Mixed 3	200	0.136	**0.116**	0.264	0.245	0.421
		(0.022)	(0.018)	(0.024)	(0.026)	(0.038)
	350	0.132	**0.115**	0.265	0.242	0.344
		(0.019)	(0.014)	(0.021)	(0.022)	(0.030)
	700	0.123	**0.114**	0.264	0.236	0.269
		(0.014)	(0.011)	(0.016)	(0.015)	(0.017)

TABLE 3
Averaged misclassification rates with standard errors in brackets for 2D simulations when $m = 100$ over 100 replicates.

Model	n_k	$m = 100$				
		mfDNN	MSDA	PLDA	mfRF	CNN
2D Gaussian	200	**0.147**	0.244	0.326	0.177	0.463
		(0.024)	(0.027)	(0.029)	(0.025)	(0.021)
	350	**0.139**	0.242	0.328	0.163	0.460
		(0.017)	(0.021)	(0.023)	(0.019)	(0.014)
	700	**0.132**	0.241	0.327	0.151	0.446
		(0.011)	(0.014)	(0.016)	(0.010)	(0.011)
2D Mixed 1	200	**0.090**	0.150	0.229	0.122	0.405
		(0.018)	(0.021)	(0.025)	(0.020)	(0.013)
	350	**0.076**	0.144	0.229	0.111	0.391
		(0.010)	(0.016)	(0.020)	(0.017)	(0.011)
	700	**0.070**	0.142	0.232	0.101	0.372
		(0.009)	(0.011)	(0.014)	(0.011)	(0.008)
2D Mixed 2	200	**0.121**	0.152	0.201	0.139	0.467
		(0.019)	(0.022)	(0.025)	(0.024)	(0.020)
	350	**0.116**	0.152	0.202	0.132	0.417
		(0.014)	(0.016)	(0.021)	(0.015)	(0.023)
	700	**0.108**	0.148	0.199	0.123	0.299
		(0.009)	(0.012)	(0.012)	(0.009)	(0.027)
2D Mixed 3	200	**0.107**	0.116	0.264	0.208	0.344
		(0.024)	(0.018)	(0.023)	(0.022)	(0.036)
	350	**0.098**	0.116	0.265	0.200	0.280
		(0.012)	(0.014)	(0.021)	(0.018)	(0.025)
	700	**0.097**	0.114	0.265	0.192	0.250
		(0.012)	(0.011)	(0.016)	(0.013)	(0.015)

TABLE 4
Averaged misclassification rates with standard errors in brackets for 2D simulations when $m = 400$ over 100 replicates.

Model		$m = 400$				
		mfDNN	MSDA	PLDA	mfRF	CNN
2D Gaussian	200	**0.145**	0.245	0.329	0.155	0.465
		(0.025)	(0.027)	(0.029)	(0.024)	(0.022)
	350	**0.139**	0.242	0.331	0.141	0.459
		(0.017)	(0.021)	(0.022)	(0.017)	(0.015)
	700	**0.131**	0.241	0.329	0.132	0.438
		(0.011)	(0.014)	(0.016)	(0.011)	(0.011)
2D Mixed 1	200	**0.089**	0.150	0.232	0.105	0.399
		(0.017)	(0.022)	(0.026)	(0.018)	(0.014)
	350	**0.075**	0.144	0.232	0.098	0.382
		(0.010)	(0.016)	(0.019)	(0.014)	(0.010)
	700	**0.069**	0.142	0.235	0.089	0.359
		(0.008)	(0.011)	(0.015)	(0.011)	(0.007)
2D Mixed 2	200	**0.119**	0.152	0.204	0.132	0.443
		(0.018)	(0.022)	(0.025)	(0.019)	(0.023)
	350	**0.114**	0.151	0.206	0.127	0.365
		(0.013)	(0.016)	(0.021)	(0.015)	(0.030)
	700	**0.099**	0.148	0.203	0.118	0.222
		(0.009)	(0.012)	(0.013)	(0.010)	(0.019)
2D Mixed 3	200	**0.099**	0.116	0.264	0.199	0.281
		(0.020)	(0.019)	(0.024)	(0.022)	(0.036)
	350	**0.090**	0.115	0.267	0.191	0.250
		(0.012)	(0.014)	(0.021)	(0.017)	(0.022)
	700	**0.085**	0.114	0.265	0.186	0.235
		(0.010)	(0.011)	(0.016)	(0.013)	(0.017)

further confirms Theorem 3.2. Given the relatively sparse sampling frequency, i.e., $m = 9$, MSDA has slightly better performance than mfDNN does. However, despite the increase of m in Tables 3 and 4, there is no improvement of both MSDA and PLDA methods in terms of misclassification risks. This finding indicates that MSDA and PLDA classifiers can not be improved with more gathered information. The misclassification errror rates of the mfRF and CNN decrease slightly as the sample size and increasing. The method mfRF outperforms CNN and PLDA in most cases, showcasing that the proposed dimension reduction strategy by mfDNN effectively collects the information of the functional data observations. In addition, compared to MSDA and PLDA, mfRF and CNN have better performance with larger sampling frequency. This is because the inherent nature of both mfRF and CNN involves processing imaging data as a whole. In contrast to traditional multivariate analysis methods, which treat the sampling frequency (pixel or voxel number) as the data dimension, mfRF and CNN do not face the curse of dimensionality. Instead, they benefit from higher sampling frequencies, yielding better results as they capture more information from the data. In summary, the simulation results illustrate that the proposed mfDNN method dominates the existing penalized discriminant analysis and two modern deep learning methods when classifying 2D dense functional data. Additionally, in Appendix B, Figure 4 demonstrates the phase transition phenomenon of mfDNN method for Models 1 to 4.

5.3. 3D functional data

For $k = 1, 2, 3$ and $s_1, s_2, s_3 \in [0, 1]$, we generate 3D functional data $X_i^{(k)}(s_1, s_2, s_3) = \sum_{j=1}^{9} \xi_{ij}^{(k)} \psi_j(s_1, s_2, s_3)$ where $\psi_1(s_1, s_2, s_3) = s_1$, $\psi_2(s_1, s_2, s_3) = s_2$, $\psi_3(s_1, s_2, s_3) = s_3$, $\psi_4(s_1, s_2, s_3) = s_1 s_2$, $\psi_5(s_1, s_2, s_3) = s_1 s_3$, $\psi_6(s_1, s_2, s_3) = s_2 s_3$, $\psi_7(s_1, s_2, s_3) = s_1^2$, $\psi_8(s_1, s_2, s_3) = s_2^2$, $\psi_9(s_1, s_2, s_3) = s_3^2$, and the distribution of $\xi_{ij}^{(k)}$'s are specified as below.

Model 5 (3D Gaussian): Let $\left(\xi_{i1}^{(k)}, \ldots, \xi_{i9}^{(k)}\right)^\intercal \sim N(\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)$, where $\boldsymbol{\mu}_1 = 2 \times \mathbf{1}_9$, $\boldsymbol{\Sigma}_1^{1/2} = \boldsymbol{\Sigma}_2^{1/2} = \text{diag}\,(9, 8, 7, 6, 5, 4, 3, 2, 1)$, $\boldsymbol{\mu}_2 = \boldsymbol{\mu}_3 = \mathbf{0}_9$, $\boldsymbol{\Sigma}_3^{1/2} = 1/3 \times \boldsymbol{\Sigma}_1^{1/2}$.

Model 6 (3D Mixed 1): Let $\left(\xi_{i1}^{(k)}, \ldots, \xi_{i9}^{(k)}\right)^\intercal \sim N(\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)$ for $k = 1, 2$, and $\xi_{ij}^{(3)} \sim t_{j+1}(\nu_j)$, where $(\nu_1, \ldots, \nu_5)^\intercal = 3 \times \mathbf{1}_5$, $\boldsymbol{\mu}_1 = -\mathbf{1}_9$, $\boldsymbol{\Sigma}_1^{1/2} = \text{diag}(5.5, 5, 4.5, 4, 3.5, 3, 2.5, 2, 1.5)$, $\boldsymbol{\mu}_2 = \mathbf{0}_9$, $\boldsymbol{\Sigma}_2^{1/2} = \text{diag}\,(4.5, 4, 3.5, 3, 2.5, 2, 1.5, 1, 0.5)$.

Model 7 (3D Mixed 2): Let $\left(\xi_{i1}^{(1)}, \ldots, \xi_{i9}^{(1)}\right)^\intercal \sim N(\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)$, $\xi_{ij}^{(2)} \sim t_{j+1}(\nu_{2j})$, where $\boldsymbol{\mu}_1 = \mathbf{0}_9$, $\boldsymbol{\Sigma}_1^{1/2} = \text{diag}\,(4.5, 4, 3.5, 3, 2.5, 2, 1.5, 1, 0.5)$, $(\nu_{21}, \ldots, \nu_{29})^\intercal = -\mathbf{1}_9$, $(\nu_{31}, \ldots, \nu_{39})^\intercal = 0.5 \times \mathbf{1}_9$.

Model 8 (3D Mixed 3): Let $\xi_{ij}^{(1)} \sim Exp(r_{1j})$, $\xi_{ij}^{(2)} \sim t_{j+1}(\nu_{2j})$, and $\left(\xi_{i1}^{(3)}, \ldots, \xi_{i9}^{(3)}\right)^\intercal \sim N(\boldsymbol{\mu}_3, \boldsymbol{\Sigma}_3)$, where $(r_{11}, \ldots, r_{19})^\intercal = 0.1 \times (1, 3, 5, 7, 9, 11, 13, 15, 17)^\intercal$, $(\nu_{21}, \ldots, \nu_{29})^\intercal = 0.6 \times \mathbf{1}_9$, $\boldsymbol{\mu}_3 = \mathbf{0}_9$, $\boldsymbol{\Sigma}_3^{1/2} = \text{diag}(4.5, 4, 3.5, 3, 2.5, 2, 1.5, 1, 0.5)$.

TABLE 5
Averaged misclassification rates with standard errors in brackets for 3D simulations when $m = 8$ over 100 replicates.

Model	n_k	$m = 8$				
		mfDNN	MSDA	PLDA	mfRF	CNN
3D Gaussian	200	**0.374**	0.465	0.485	0.380	–
		(0.024)	(0.037)	(0.045)	(0.028)	–
	350	**0.373**	0.474	0.495	0.383	–
		(0.020)	(0.029)	(0.042)	(0.022)	–
	700	**0.363**	0.472	0.488	0.374	–
		(0.014)	(0.025)	(0.041)	(0.017)	–
3D Mixed 1	200	**0.215**	0.224	0.238	0.259	–
		(0.022)	(0.024)	(0.022)	(0.027)	–
	350	**0.200**	0.221	0.236	0.253	–
		(0.018)	(0.019)	(0.020)	(0.021)	–
	700	**0.188**	0.220	0.237	0.252	–
		(0.011)	(0.014)	(0.012)	(0.014)	–
3D Mixed 2	200	**0.311**	0.325	0.346	0.308	–
		(0.018)	(0.025)	(0.026)	(0.025)	–
	350	**0.307**	0.328	0.345	0.300	–
		(0.018)	(0.021)	(0.022)	(0.021)	–
	700	**0.253**	0.326	0.347	0.295	–
		(0.014)	(0.015)	(0.015)	(0.015)	–
3D Mixed 3	200	**0.272**	0.308	0.315	0.295	–
		(0.026)	(0.031)	(0.030)	(0.025)	–
	350	**0.268**	0.303	0.309	0.288	–
		(0.019)	(0.023)	(0.024)	(0.020)	–
	700	**0.253**	0.302	0.311	0.279	–
		(0.014)	(0.017)	(0.017)	(0.015)	–

For the 3D functional data, we apply similar setups as 2D cases. We observe the functional data on $2 \times 2 \times 2$, $3 \times 3 \times 3$, $4 \times 4 \times 4$, and $5 \times 5 \times 5$ grid points over $[0, 1]^3$, respectively, and the sampling frequency $m = 8, 27, 64, 125$. Tables 5 to 8 demonstrate the results of 100 simulations. The proposed mfDNN classifier is superior to its counterparts for all 3D functional data cases. Meanwhile, there also exists the phase transition patterns for mfDNN method. However, the performance of MSDA and PLDA methods lacks of improvement with the increase of m. It can be seen that when $m = 125$, the misclassification error rates of mfDNN are almost one third of MSDA's and one forth of PLDA's in Gaussian case, and almost a half of either MSDA's or PLDA's error rates in Models 7 and 8. A plausible reason is that given the functional data framework, our proposed mfDNN can properly accommodate the repeatedly observed data over pixels or voxels, while other competitors only treat those information as common high-dimensional covariates and ignore the underlining smoothing structures. By efficiently extracting the projection scores of the continuum, the proposed mfDNN has full potential to discover the underlying distributions of the functional data clusters. Similar as 2D functional data cases, the method mfRF has the second best performance in most cases. Note that in Table 5, the CNN architecture is not suitable for the given design of $2 \times 2 \times 2$ due to its extreme sparsity. In order to effectively capture spatial information, a CNN typically necessitates a filter dimension of at least $2 \times 2 \times 2$, resulting in an

TABLE 6
Averaged misclassification rates with standard errors in brackets for 3D simulations when $m = 27$ over 100 replicates.

Model		$m = 27$				
		mfDNN	MSDA	PLDA	mfRF	CNN
3D Gaussian	200	**0.234**	0.367	0.491	0.335	0.540
		(0.020)	(0.030)	(0.046)	(0.029)	(0.024)
	350	**0.233**	0.369	0.500	0.334	0.537
		(0.020)	(0.023)	(0.041)	(0.020)	(0.017)
	700	**0.232**	0.374	0.493	0.319	0.513
		(0.013)	(0.018)	(0.041)	(0.015)	(0.014)
3D Mixed 1	200	**0.168**	0.164	0.242	0.245	0.367
		(0.021)	(0.022)	(0.022)	(0.024)	(0.011)
	350	**0.151**	0.156	0.240	0.234	0.355
		(0.017)	(0.016)	(0.020)	(0.020)	(0.008)
	700	**0.149**	0.152	0.241	0.227	0.340
		(0.010)	(0.011)	(0.013)	(0.014)	(0.006)
3D Mixed 2	200	0.277	0.294	0.348	**0.276**	0.349
		(0.020)	(0.022)	(0.025)	(0.026)	(0.012)
	350	0.268	0.289	0.348	**0.267**	0.337
		(0.018)	(0.018)	(0.022)	(0.020)	(0.012)
	700	**0.234**	0.286	0.350	0.258	0.312
		(0.013)	(0.014)	(0.015)	(0.014)	(0.010)
3D Mixed 3	200	**0.268**	0.292	0.313	0.264	0.354
		(0.026)	(0.029)	(0.030)	(0.028)	(0.036)
	350	**0.239**	0.286	0.310	0.253	0.298
		(0.018)	(0.024)	(0.023)	(0.022)	(0.024)
	700	**0.234**	0.286	0.312	0.240	0.254
		(0.013)	(0.015)	(0.017)	(0.015)	(0.015)

TABLE 7
Averaged misclassification rates with standard errors in brackets for 3D simulations when $m = 64$ over 100 replicates.

Model	n_k	$m = 64$				
		mfDNN	MSDA	PLDA	mfRF	CNN
3D Gaussian	200	**0.135**	0.369	0.493	0.245	0.533
		(0.018)	(0.029)	(0.048)	(0.024)	(0.021)
	350	**0.123**	0.369	0.502	0.231	0.528
		(0.015)	(0.022)	(0.042)	(0.019)	(0.018)
	700	**0.114**	0.374	0.495	0.207	0.486
		(0.010)	(0.019)	(0.04)	(0.014)	(0.015)
3D Mixed 1	200	**0.130**	0.163	0.243	0.190	0.356
		(0.021)	(0.022)	(0.022)	(0.026)	(0.011)
	350	**0.109**	0.156	0.241	0.171	0.343
		(0.014)	(0.016)	(0.020)	(0.018)	(0.009)
	700	**0.098**	0.153	0.243	0.155	0.325
		(0.010)	(0.011)	(0.013)	(0.014)	(0.009)
3D Mixed 2	200	**0.166**	0.294	0.350	0.202	0.324
		(0.020)	(0.022)	(0.025)	(0.022)	(0.015)
	350	**0.147**	0.289	0.349	0.187	0.308
		(0.015)	(0.018)	(0.023)	(0.018)	(0.016)
	700	**0.141**	0.286	0.351	0.175	0.271
		(0.012)	(0.015)	(0.014)	(0.013)	(0.014)
3D Mixed 3	200	**0.184**	0.292	0.312	0.205	0.308
		(0.022)	(0.029)	(0.030)	(0.024)	(0.031)
	350	**0.171**	0.287	0.310	0.190	0.269
		(0.016)	(0.023)	(0.024)	(0.017)	(0.023)
	700	**0.160**	0.286	0.312	0.172	0.239
		(0.011)	(0.015)	(0.017)	(0.013)	(0.015)

TABLE 8
Averaged misclassification rates with standard errors in brackets for 3D simulations when $m = 125$ over 100 replicates.

Model		$m = 125$				
		mfDNN	MSDA	PLDA	mfRF	CNN
3D Gaussian	200	**0.134**	0.369	0.494	0.228	0.526
		(0.017)	(0.029)	(0.048)	(0.026)	(0.022)
	350	**0.118**	0.370	0.501	0.215	0.517
		(0.012)	(0.022)	(0.042)	(0.019)	(0.018)
	700	**0.108**	0.374	0.495	0.193	0.457
		(0.009)	(0.019)	(0.042)	(0.015)	(0.019)
3D Mixed 1	200	**0.127**	0.162	0.244	0.185	0.345
		(0.020)	(0.022)	(0.023)	(0.025)	(0.013)
	350	**0.106**	0.155	0.241	0.169	0.331
		(0.014)	(0.016)	(0.020)	(0.018)	(0.010)
	700	**0.098**	0.152	0.242	0.151	0.307
		(0.009)	(0.011)	(0.013)	(0.013)	(0.012)
3D Mixed 2	200	**0.154**	0.295	0.348	0.174	0.301
		(0.019)	(0.024)	(0.026)	(0.024)	(0.019)
	350	**0.135**	0.289	0.349	0.165	0.283
		(0.013)	(0.018)	(0.023)	(0.019)	(0.017)
	700	**0.126**	0.286	0.351	0.154	0.239
		(0.011)	(0.014)	(0.014)	(0.013)	(0.015)
3D Mixed 3	200	**0.176**	0.292	0.312	0.187	0.282
		(0.021)	(0.028)	(0.030)	(0.022)	(0.029)
	350	**0.162**	0.286	0.310	0.170	0.251
		(0.016)	(0.024)	(0.023)	(0.017)	(0.021)
	700	**0.151**	0.285	0.312	0.151	0.224
		(0.012)	(0.015)	(0.017)	(0.013)	(0.015)

output of $1 \times 1 \times 1$. Consequently, the dimension of max pooling needs to be $1 \times 1 \times 1$, rendering the pooling step unnecessary. As a result, the CNN does not adequately extract meaningful features, and its applicability in this context is questionable. This finding indicates that the popular CNN algorithms can not be adaptive the sparsely observed functional data automatically. Meanwhile, the proposal mfDNN is more flexible for both sparse and densely observed data. This again demonstrates our proposed classifier has a distinct advantage over these competitors in complex imaging data classification problems.

We also compare the computational time for each method in Tables 9 and 10 in both 2D and 3D cases under the Gaussian setting (Models 1 and 5). Both mfDNN and CNN are relatively time-consuming compared with other algorithms. Notably, CNN exhibits increasing computation time as the sampling frequency m grows. In contrast, our proposed method mfDNN remains constant across different data sizes m. This consistency is achieved by employing the extracted projection scores as inputs, which are relatively stable and do not significantly vary with the sampling frequency. Additionally, in Appendix B, Figure 5 demonstrates the phase transition phenomenon of mfDNN method for Models 5 to 8.

TABLE 9
Averaged computational time (in minitues) for 2D simulations over 100 *replicates.*

n_k	$m = 9$					$m = 25$				
	mfDNN	MSDA	PLDA	mfRF	CNN	mfDNN	MSDA	PLDA	mfRF	CNN
200	2.075	0.061	0.001	0.035	1.893	1.904	0.063	0.019	0.046	2.463
350	6.812	0.067	0.017	0.076	2.871	6.250	0.068	0.025	0.098	3.557
700	13.061	0.081	0.022	0.237	4.786	12.998	0.083	0.029	0.314	6.015
n_k	$m = 100$					$m = 400$				
	mfDNN	MSDA	PLDA	mfRF	CNN	mfDNN	MSDA	PLDA	mfRF	CNN
200	2.042	0.083	0.035	0.088	5.470	1.956	0.181	0.084	0.114	21.246
350	6.874	0.087	0.052	0.187	6.708	6.537	0.205	0.101	0.234	26.703
700	13.245	0.115	0.068	0.529	13.153	13.331	0.262	0.143	0.664	46.376

TABLE 10
Averaged computational time (in minitues) for 3D simulations over 100 *replicates.*

n_k	$m = 8$					$m = 27$				
	mfDNN	MSDA	PLDA	mfRF	CNN	mfDNN	MSDA	PLDA	mfRF	CNN
200	2.145	0.053	0.006	0.042	–	2.174	0.065	0.018	0.052	3.507
350	5.867	0.059	0.006	0.085	–	6.473	0.072	0.028	0.109	5.016
700	12.904	0.084	0.024	0.250	–	13.514	0.100	0.039	0.323	9.074
n_k	$m = 64$					$m = 125$				
	mfDNN	MSDA	PLDA	mfRF	CNN	mfDNN	MSDA	PLDA	mfRF	CNN
200	2.224	0.069	0.031	0.108	3.665	2.251	0.141	0.045	0.135	5.070
350	7.083	0.080	0.040	0.223	6.196	6.938	0.152	0.055	0.276	8.389
700	13.742	0.109	0.056	0.597	10.947	14.246	0.182	0.080	0.740	15.320

6. Real data analysis

6.1. Handwritten digits

The first benchmark data example was extracted from the MNIST database (`http://yann.lecun.com/exdb/mnist/`). This classical MNIST database contains 60,000 training images and 10,000 testing images of handwritten digits $(0, 1, \ldots, 9)$, and the black and white images were normalized to fit into a 28×28 pixel bounding box and anti-aliased. We used tensor of Fourier basis for data processing. According to our numerical experience, we choose candidates for $(L, J, \boldsymbol{p}, s)$, such that $L = (2, 3, 4)^\intercal$, $J = (300, 500, 800)^\intercal$, $\|\boldsymbol{p}\|_\infty = (500, 1000, 2000)^\intercal$, and $s = (0.01, 0.1, 0.5)$ for dropout rate. Here we abuse the notation of s, as the dropout is the technique we choose to sparsify the neural network. With the optimal parameters $L_{opt} = 3$, $J_{opt} = 500$, $\|\boldsymbol{p}_{opt}\|_\infty = 1000$, $s_{opt} = 0.01$ through validation, we demonstrate the misclassification risk in Table 11. We estimate the rules given by MSDA, PLDA, mfRF, CNN and our proposal on the training set. As most observations for each subject are zeros, `PenalizedLDA` reports errors and does not work any more. It can be seen that our proposal achieves the second best training accuracy compared with mfRF, and also second best performance on testing accuracy. However, when considering both training and testing accuracy and computational cost as a whole aspect, our method is relatively competitive and efficient.

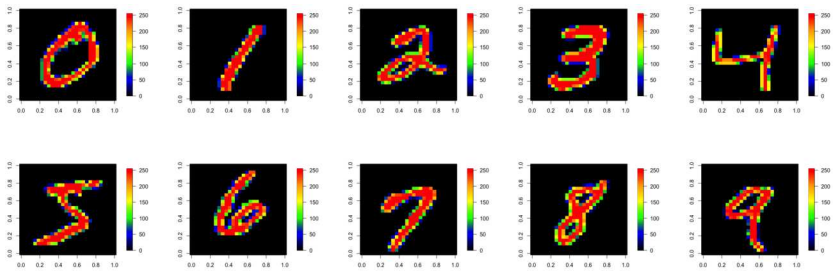

FIG 1. *Samples from MNIST data.*

TABLE 11
Classification accuracy for MNIST data.

methods	training accuracy	testing accuracy
mfDNN	0.998	0.983
MSDA	0.869	0.875
PLDA	–	–
mfRF	**1.000**	0.838
CNN	0.997	**0.994**

6.2. ADNI database

The dataset used in the preparation of this article were obtained from the ADNI database (http://adni.loni.usc.edu). The ADNI is a longitudinal multicenter study designed to develop clinical, imaging, genetic, and biochemical biomarkers for the early detection and tracking of Alzheimer's Disease (AD). From this database, we collect PET data from 79 patients in AD group, 45 patients in Early Mild Cognitive Impairment (EMCI) group, and 101 people in Control (CN) group. This PET dataset has been spatially normalized and post-processed. These AD patients have three to six times doctor visits and we select the PET scans obtained in the third visits. People in EMCI group only have the second visit, and we select the PET scans obtained in the second visits. For AD group, patients' age ranges from 59 to 88 and average age is 76.49, and there are 33 females and 46 males among these 79 subjects. For EMCI group, patients' age ranges from 57 to 89 and average age is 72.33, and there are 26 females and 19 males among these 45 subjects. For CN group, patients' age ranges from 62 to 87 and average age is 75.98, and there are 40 females and 61 males among these 101 subjects. All scans were reoriented into $79 \times 95 \times 68$ voxels, which means each patient has 68 sliced 2D images with 79×95 pixels. For 2D case, it means each subject has $N = 79 \times 95 = 7,505$ observed pixels for each selected image slice. For 3D case, the observed number of voxels for each patient's brain image observation is $N = 79 \times 95 \times 68 = 510,340$. We randomly split the datasets with a $7:3$ ratio in a balanced manner to form the training set and the testing set, with 100 repetitions.

We choose candidates for $(L, J, \boldsymbol{p}, s)$, such that $L = (2,3)^\intercal$, $J = (50,100,200)^\intercal$, $\|\boldsymbol{p}\|_\infty = (200,500,800)^\intercal$, and $s = (0.01, 0.1, 0.5)$ for dropout rate. We still

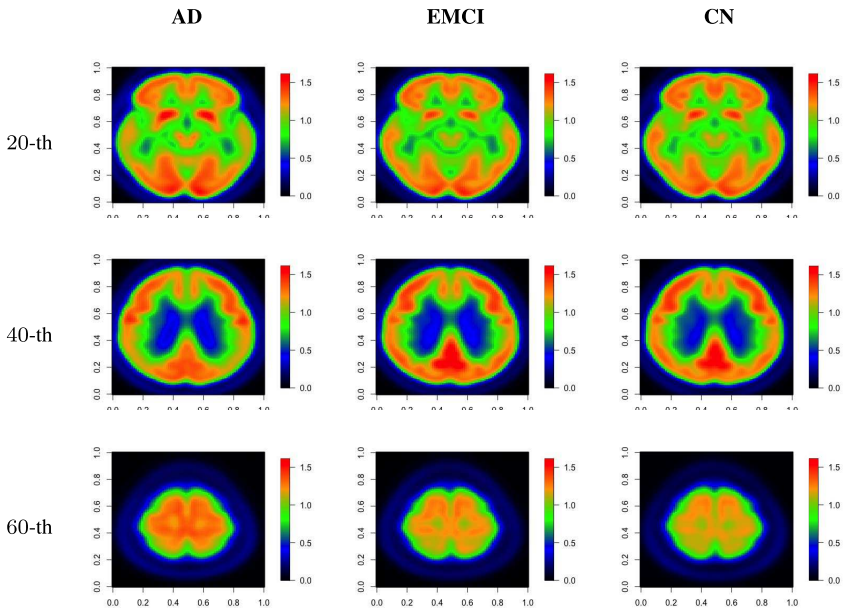

FIG 2. *Averaged images of the the* 20*-th, the* 40*-th and the* 60*-th slices of AD group (left column), EMCI group (middle column), and CN group (right column).*

compare our method with MSDA. For 2D case, it means each subject has $N = 79 \times 95 = 7,505$ observed pixels for each selected image slice. Table 12 displays the miclassification rates for 2D brain imaging data of AD, EMCI and CN. The proposed mfDNN has the smallest risk for most cases, while mfRF and CNN have fairly similar performance. For 3D case, the observed number of voxels for each patient's brain sample is $N = 79 \times 95 \times 68 = 510,340$. Regrettably, the MSDA method encountered a critical failure when confronted with over half a million covariates. The collapse ensued due to the colossal size of the covariance matrix, reaching nearly a million by a million dimensions, demanding an impractical 2TB RAM for storage. This magnitude surpasses the memory constraints of the typical supercomputer, leading to operational limitations. Hence, MSDA for 3D data results are unavailable. Meanwhile, as PLDA recasts Fisher's discriminant problem as a biconvex problem that can be optimized using a simple iterative algorithm, PLDA avoids the heavy computation burden of the covariance matrix and it still works for this 3D case. We also note that the computational cost and time for CNN in 3D data setting is extremely heavy due to the voxel size of the image. It takes more than hundreds of hours for a single replicate even with simple network structure, where the CNN structure has two convolutional layers, each comprising 32 and 64 filters, respectively. Table 13 presents the empirical misclassification risk for mfDNN and PLDA.

There are several interesting findings in Tables 12 and 13. First, our proposed classifier has better performance than other competitors in any 2D slice data or

TABLE 12
Averaged misclassification rates with standard errors in brackets for ADNI 2D brain images.

methods	10-th	20-th	30-th	40-th	50-th	60-th
mfDNN	0.341	**0.325**	0.409	**0.359**	**0.365**	**0.452**
	(0.057)	(0.049)	(0.049)	(0.050)	(0.041)	(0.055)
MSDA	0.347	0.374	0.444	0.366	0.385	0.453
	(0.038)	(0.034)	(0.035)	(0.034)	(0.039)	(0.040)
PLDA	0.354	0.339	0.440	0.376	0.424	0.474
	(0.057)	(0.059)	(0.067)	(0.060)	(0.068)	(0.064)
mfRF	0.353	0.366	0.428	0.328	0.396	0.435
	(0.045)	(0.048)	(0.045)	(0.034)	(0.048)	(0.049)
CNN	**0.304**	0.341	**0.376**	0.371	0.395	0.503
	(0.050)	(0.052)	(0.059)	(0.068)	(0.062)	(0.058)

TABLE 13
Averaged misclassification rates with standard errors in brackets for ADNI 3D brain images.

methods	Misclassification rates
mfDNN	**0.274 (0.044)**
MSDA	– (–)
PLDA	0.295 (0.056)
mfRF	0.358 (0.041)
CNN	– (–)

3D data cases. Second, from Table 13, we can conclude that given a single slice of 2D imaging data, the misclassification rates are consistently larger than using the 3D data. It indicates that 3D data contains more helpful information to label the brain images among three stages of the disease. Third, the 10-th and the 20-th slices provide the lowest misclassification error rates among all 2D data. As a matter of fact, it is well known that Alzheimer's disease destroys neurons and their connections in hippocampus, the entorhinal cortex, and the cerebral cortex. The related regions are corresponding to the first 25 slices. This is a promising finding for neurologists, as this smallest risk indicates this particular slice presents useful information to distinguish the CN, EMCI and AD groups. Further medical checkups are meaningful for this special location in the brain.

7. Summary

In this paper, we propose a new formulation to derive multiclass classifiers for multidimensional functional data. We show that our proposal has a solid theoretical foundation and can be solved by a very efficient computational algorithm. Our proposal actually gives a unified treatment of both one-dimensional and multidimensional classification problems. In light of this evidence, our proposal is regarded as an efficient multiclass and generalization of the multiclassification methods from i.i.d. data to multidimensional functional data cases. To the best of our knowledge, the present work is the first work on multiclassification for multidimensional functional data with theoretical justification.

Appendix A: Proofs of Theorems 3.1 and 3.2

Throughout the section for any two vectors $\boldsymbol{a}=(a_1,\ldots,a_K)^\intercal$ and $\boldsymbol{b}=(b_1,\ldots,b_K)^\intercal$ with length K, denote

$$\boldsymbol{a}^\intercal \log\left(\frac{\boldsymbol{a}}{\boldsymbol{b}}\right)=\sum_{k=1}^{K} a_k \log\left(\frac{a_k}{b_k}\right).$$

Let $\mathcal{F}_{id}$ be the network class with identity activation function as the last-layer activation function. We have the following lemma for approximation of the exponential output.

Lemma 7.1. *(Theorem 4.1 in [4]) For all $M \geq 2$ and $\beta > 0$, there exists a $G \in \mathcal{F}_{id}(L,1,\boldsymbol{p},s)$, with $L=\lfloor 40(\beta+2)^2\log_2(M)\rfloor$, $\boldsymbol{p}=(1,\lfloor 48\lceil\beta\rceil^3 2^\beta M^{1/\beta}\rfloor,\ldots,\lfloor 48\lceil\beta\rceil^3 2^\beta M^{1/\beta}\rfloor,1)$, and $s \leq 4284(\beta+5)^2 2^\beta M^{1/\beta}\log_2(M)$, such that $\forall x \in [0,1]$, $|e^{G(x)}-x| \leq 4M^{-1}$ and $G(x) \geq \log_2(4M^{-1})$.*

Lemma 7.2. *(Lemma 3 in [28]) Let h_{uv} be the function defined as follows:*

$$h_0=\frac{g_0}{2R_0}+\frac{1}{2},\quad h_i=\frac{g_i(2R_{i-1}\boldsymbol{x}-R_{i-1})}{2R_i}+\frac{1}{2},\quad h_q=g_q(2R_{q-1}\boldsymbol{x}-R_{q-1}),$$

where for any $\boldsymbol{x} \in [0,1]^{d_u}$, $u \in 0,\ldots,q$, $2R_{u-1}\boldsymbol{x}-R_{u-1}$ is equivalent to the transform of $2R_{u-1}x_u-R_{u-1}$ for every $u=1,\ldots,d_u$. Then for any functions $\widetilde{h}_u=\left(\widetilde{h}_{uv}\right)^\intercal$ with $\widetilde{h}_{uv}:[0,1]^{t_u}\to[0,1]$,

$$\|h_q\circ\ldots\circ h_0-\widetilde{h}_q\circ\ldots\circ\widetilde{h}_0\|_\infty \leq R_q\prod_{\ell_0}^{q-1}(2R_\ell)^{\beta_{\ell+1}}\sum_{u=0}^{q}\|\|h_u-\widetilde{h}_u|_\infty\|_\infty^{\prod_{\ell=u+1}^{q}\min\{\beta_\ell,1\}}.$$

Lemma 7.3. *(Theorem 5 in [28]) For ant function $f \in \mathcal{C}^\beta([0,1]^r,R)$ and any integers $m_0 \geq 1$ and $N \geq \max\{(\beta+1)^r,(R+1)e^r\}$, there exists a network*

$$\widetilde{f} \in \mathcal{F}(L,r,(r,6(r+\lceil\beta\rceil)N,\ldots,6(r+\lceil\beta\rceil)N,1),s,\infty)$$

with depth

$$L=8+(m_0+5)(1+\lceil\log_2(\max\{r,\beta\})\rceil)$$

and number of parameters

$$s \leq 141(r+\beta+1)^{3+r}N(m_0+6),$$

such that

$$\|\widetilde{f}-f\|_\infty \leq (2R+1)(1+r^2+\beta^2)6^r N 2^{-m_0}+K3^\beta N^{-\beta/r}.$$

Lemma 7.4. *(Network approximation of $p_J^{(k)}$) For any sufficiently large $M>0$, there exists a constant $c_k>0$ and a $H^{(k)} \in \mathcal{F}(L^{(k)},J,\boldsymbol{p}^{(k)},s^{(k)})$, with*

$$L^{(k)}=3(q^{(k)}-1)+\sum_{u=0}^{q^{(k)}}\left\lceil 8+\left(\max_u\left(\frac{1}{\beta_u^{(k)**}}\log_2\left(\frac{A_u^{(k)}C(t_u^{(k)})}{B_u^{(k)}}\right)\right.\right.\right.$$

$$+\left(1+\frac{\beta_u^{(k)}}{t_u^{(k)}}\right)\left(\max_u \frac{t_u^{(k)}}{\tilde{\beta}_u^{(k)}}\right)\log_2 M\Bigg)+5\Bigg)\left(1+\lceil\log_2(t_u^{(k)}\vee\beta_u^{(k)})\rceil\right)\Bigg],$$

$$\boldsymbol{p}^{(k)} =(J, 6\max_u d_{u+1}^{(k)}(t_u^{(k)}+\lceil\beta_u^{(k)}\rceil)M^{\max_u \frac{t_u^{(k)}}{\tilde{\beta}_u^{(k)}}},\ldots,$$

$$6\max_u d_{u+1}^{(k)}(t_u^{(k)}+\lceil\beta_u^{(k)}\rceil)M^{\max_u \frac{t_u^{(k)}}{\tilde{\beta}_u^{(k)}}},1),$$

$$s^{(k)} = \sum_{u=0}^{q^{(k)}} d_{u+1}^{(k)}\Bigg[141(t_u^{(k)}+\beta_u^{(k)}+1)^{3+t_u^{(k)}}M^{\max_u \frac{t_u^{(k)}}{\tilde{\beta}_u^{(k)}}}$$

$$\times\Bigg\{\max_u\left(\frac{1}{\beta_u^{(k)**}}\log_2\left(\frac{A_u^{(k)}C(t_u^{(k)})}{B_u^{(k)}}\right)+\left(1+\frac{\beta_u^{(k)}}{t_u^{(k)}}\right)\left(\max_u\frac{t_u^{(k)}}{\tilde{\beta}_u^{(k)}}\right)\log_2 M\right)$$

$$+6\Bigg\}+4\Bigg],$$

where $\beta_u^{(k)**}=\prod_{\ell=u+1}^{q^{(k)}}\beta_\ell^{(k)}\wedge 1$, $A_u^{(k)}=\left\{R_{q^{(k)}}\prod_{\ell=0}^{q^{(k)}-1}(2R_\ell)^{\beta_{\ell+1}^{(k)}}\right\}(2Q_u^{(k)}+1)$, $B_u^{(k)}=\left\{R_{q^{(k)}}\prod_{\ell=0}^{q^{(k)}-1}(2R_\ell)^{\beta_{\ell+1}^{(k)}}\right\}(Q_u^{(k)}3^{\beta_u^{(k)}})^{\beta_u^{(k)**}}$, $C(t_u^{(k)})=2(t_u^{(k)})^2 6^{t_u^{(k)}\beta_u^{(k)**}}$, $Q_u^{(k)}$ *are some absolute constants only depending on* $\beta_u^{(k)}$, $R_u^{(k)}$, *such that*

$$\|H^{(k)}-\pi_k^{(J)}\|_\infty\le c_k M^{-1}.$$

Proof. Let m_k be the depth parameter, and N_k be the width parameter. Following Lemmas 7.2 and 7.3, we have

$$\begin{aligned}\inf_{\widetilde{H}^{(k)}\in\mathcal{F}(L,J,\boldsymbol{p},s)}\|\widetilde{H}^{(k)}-\pi_k^{(J)}\|_\infty &\le \|h_{q^{(k)}}\circ\ldots\circ h_0-\widetilde{h}_{q^{(k)}}\circ\ldots\circ\widetilde{h}_0\|_\infty\\ &\le \sum_{u=0}^{q^{(k)}}A_u^{(k)}C(t_u^{(k)})(N_k 2^{-m_k})^{\beta_u^{(k)**}}\\ &\quad+\sum_{u=0}^{q^{(k)}}B_u^{(k)}N_k^{-\tilde{\beta}_u^{(k)}/t_u^{(k)}}.\end{aligned}$$

The corresponding network structure under the specific approximation error is given by

$$\begin{aligned}L^{(k)} &= 3(q^{(k)}-1)+\sum_{u=0}^{q^{(k)}}\left[8+(m_k+5)(1+\lceil\log_2(t_u^{(k)}\vee\beta_u^{(k)})\rceil)\right],\\ \boldsymbol{p}^{(k)} &= (J,6r_kN_k,\ldots,6r_kN_k,1),\\ s^{(k)} &= \sum_{u=0}^{q^{(k)}}d_{u+1}^{(k)}\left[141(t_u^{(k)}+\beta_u^{(k)}+1)^{3+t_u^{(k)}}N_k(m_k+6)+4\right],\end{aligned}$$

where $r_k=\max_u d_{u+1}^{(k)}(t_u^{(k)}+\lceil\beta_u^{(k)}\rceil)$.

Let $m_k = \max_u \left\{ \frac{1}{\beta_u^{(k)**}} \log_2 \left(\frac{A_u^{(k)} C(t_u^{(k)})}{B_u^{(k)}} \right) + \left(1 + \frac{\beta_u^{(k)}}{t_u^{(k)}} \right) \log_2 N_k \right\}$ and $N_k^{\min_u \frac{\tilde{\beta}_u^{(k)}}{t_u^{(k)}}} = M$, the proof is complete with $c_k = 2(q^{(k)} + 1) \max_u B_u^{(k)}$. □

Define absolute constants

$$
\begin{aligned}
D_1^{(k)} &= 3(q^{(k)} - 1) + \sum_{u=0}^{q^{(k)}} \left\{ 8 + \max_u \left[\left(\frac{1}{\beta_u^{(k)**}} \log_2 \left(\frac{A_u^{(k)} C(t_u^{(k)})}{B_u^{(k)}} \right) \right. \right. \right. \\
&\quad \left. \left. \left. + 5 \right) \left(1 + \lceil \log_2 (t_u^{(k)} \vee \beta_u^{(k)}) \rceil \right) \right] \right\}, \\
D_2^{(k)} &= \sum_{u=0}^{q^{(k)}} (1 + \lceil \log_2 (t_u^{(k)} \vee \beta_u^{(k)}) \rceil) \max_u \left(1 + \frac{\beta_u^{(k)}}{t_u^{(k)}} \right) \left(\max_u \frac{t_u^{(k)}}{\tilde{\beta}_u^{(k)}} \right), \\
D_3^{(k)} &= 6 \max_u d_{u+1}^{(k)} (t_u^{(k)} + \lceil \beta_u^{(k)} \rceil), \quad D_4^{(k)} = 4 \sum_{u=0}^{q^{(k)}} d_{u+1}^{(k)}, \\
D_5^{(k)} &= \sum_{u=0}^{q^{(k)}} d_{u+1}^{(k)} \left[141 (t_u^{(k)} + \beta_u^{(k)} + 1)^{3 + t_u^{(k)}} \right], \\
D_6^{(k)} &= \max_u \left\{ \frac{1}{\beta_u^{(k)**}} \log_2 \left(\frac{A_u^{(k)} C(t_u^{(k)})}{B_u^{(k)}} \right) \right\} + 6, \\
D_7^{(k)} &= \max_u \left(1 + \frac{\beta_u^{(k)}}{t_u^{(k)}} \right) \left(\max_u \frac{t_u^{(k)}}{\tilde{\beta}_u^{(k)}} \right), \\
D_8 &= 2 \max_k D_2^{(k)}, \quad D_9 = \max_k D_3^{(k)}, \quad D_{10} = 4 \max_k D_5^{(k)} D_7^{(k)}.
\end{aligned}
$$

We can simplify the aforementioned network structure as

$$
\begin{aligned}
L^{(k)} &= D_1^{(k)} + D_2^{(k)} \log_2 M, \\
\boldsymbol{p}^{(k)} &= (J, D_3^{(k)} M^{\max_u \frac{t_u^{(k)}}{\tilde{\beta}_u^{(k)}}}, \ldots, D_3^{(k)} M^{\max_u \frac{t_u^{(k)}}{\tilde{\beta}_u^{(k)}}}, 1), \\
s^{(k)} &= D_4^{(k)} + D_5^{(k)} M^{\max_u \frac{t_u^{(k)}}{\tilde{\beta}_u^{(k)}}} \left(D_6^{(k)} + D_7^{(k)} \log_2 M \right).
\end{aligned}
$$

In the following, we consider relatively large $M \geq M_0$, such that

$$
M_0 = \max \left\{ 2^{\max_k D_1^{(k)}/D_2^{(k)}}, 2^{\max_k D_6^{(k)}/D_7^{(k)}}, \max_k \left(\frac{D_4^{(k)}}{D_5^{(k)}} \right)^{\min_u \tilde{\beta}_u^{(k)}/t_u^{(k)}} \right\}.
$$

Define

$$
\begin{aligned}
D_{11} &= D_8 + 40 \left(\tilde{\beta}_{\hat{u}}^{(\hat{k})} + 2 \right)^2 + 1, \\
D_{12} &= K \left(D_9 + 48 \lceil \tilde{\beta}_{\hat{u}}^{(\hat{k})} \rceil^3 2^{\tilde{\beta}_{\hat{u}}^{(\hat{k})}} \right),
\end{aligned}
$$

$$D_{13} = K\left\{D_{10} + 4284\left(\tilde{\beta}_{\hat{u}}^{(\hat{k})} + 5\right)^2 2^{\tilde{\beta}_{\hat{u}}^{(\hat{k})}}\right\}.$$

We have the following lemma for approximating $\boldsymbol{\pi}_J$.

Lemma 7.5. *For any $M \geq M_0$, $a, b \in \mathbb{R}$, there exists $\widetilde{\boldsymbol{\pi}}_J \in \mathcal{F}(L, J, \boldsymbol{p}, s)$ and an absolute constant C_4, with*

$$L = D_{11}\log_2 M, \tag{7.1}$$

$$\boldsymbol{p} = \left(J, D_{12}M^{t_{\hat{u}}^{(\hat{k})}/\tilde{\beta}_{\hat{u}}^{(\hat{k})}}, \ldots, D_{12}M^{t_{\hat{u}}^{(\hat{k})}/\tilde{\beta}_{\hat{u}}^{(\hat{k})}}, K\right), \tag{7.2}$$

and

$$s \leq D_{13}M^{t_{\hat{u}}^{(\hat{k})}/\tilde{\beta}_{\hat{u}}^{(\hat{k})}}\log_2(M), \tag{7.3}$$

such that

$$\|\widetilde{\pi}_J^{(k)} - \pi_k^{(J)}\|_\infty \leq \frac{2K(4+C_4)}{M},$$

and $\widetilde{\pi}_J^{(k)}(\boldsymbol{x}_J) \geq M^{-1}$ for all $k \in \{1, \ldots, K\}$ and $\boldsymbol{x}_J \in [a,b]^J$.

Proof. According to Lemma 7.4, for all sufficiently large M, there exists a constant $C_4 = \max_k c_k$ and a $H^{(k)} \in \mathcal{F}(L', J, \boldsymbol{p}', s')$, with $L' = \max_k L^{(k)}$, $\boldsymbol{p}' = \max_k \boldsymbol{p}^{(k)}$, $s' = \max_k s^{(k)}$, such that

$$\|H^{(k)} - \pi_k^{(J)}\|_\infty \leq C_4 M^{-1}, \quad \forall k \in 1, \ldots, K.$$

Let G be the function defined in Lemma 7.1, with the corresponding network class, denoted as $\mathcal{F}(L'', J, \boldsymbol{p}'', s'')$, and $\widetilde{\pi}_J^{(k)}(\boldsymbol{x}_J) = \frac{\exp\left(G(H^{(k)})\right)}{\sum_{\ell=1}^K \exp(G(H^{(\ell)}))}$. Directly following the technical constructions in Lemma 4.3 in [4] and Lemma 7.1, we have $\|\widetilde{\boldsymbol{\pi}}_J^{(k)} - \boldsymbol{\pi}_k^{(J)}\|_\infty \leq 2K(4+C_4)M^{-1}$ and $\widetilde{\pi}_J^{(k)}(\boldsymbol{x}_J) \geq M^{-1}$ for all $k \in 1, \ldots, K$ and $\boldsymbol{x}_J \in [a,b]^J$.

Applying the composition rule, we have

$$G \circ \sigma_{\mathbf{0}}(H^{(k)}) \in \mathcal{F}\left(L' + L'' + 1, J, (p_0', \ldots, p_{L'+1}', p_0'', \ldots, p_{L''+1}''), s' + s''\right).$$

Together with the parallelization rule, the network $\boldsymbol{G} = \left(G(H^{(1)}), \ldots, G(H^{(K)})\right)$ with softmax output belongs to

$$\mathcal{F}\left(L' + L'' + 2, J, (J, Kp, \ldots, Kp), K(s' + s'')\right),$$

where $p = (\max_{\ell=1,\ldots,L'+1} p_\ell') \vee \|\boldsymbol{p}''\|_\infty$. Since we have

$$L' \leq D_8\log_2 M, \quad \max_{\ell=1,\ldots,L'+1} p_\ell' \leq D_9 M^{t_{\hat{u}}^{(\hat{k})}/\tilde{\beta}_{\hat{u}}^{(\hat{k})}}, \quad s' \leq D_{10}M^{t_{\hat{u}}^{(\hat{k})}/\tilde{\beta}_{\hat{u}}^{(\hat{k})}}\log_2 M,$$

let β in Lemma 7.1 be $\beta = \tilde{\beta}_{\hat{u}}^{(\hat{k})}$, and

$$L' + L'' \quad \leq \quad \left(D_8 + 40\left(\tilde{\beta}_{\hat{u}}^{(\hat{k})} + 2\right)^2\right)\log_2 M,$$

$$\begin{aligned} p &\leq \left(D_9 + 48\lceil\tilde{\beta}_{\hat{u}}^{(\hat{k})}\rceil^3 2^{\tilde{\beta}_{\hat{u}}^{(\hat{k})}}\right) M^{t_{\hat{u}}^{(\hat{k})}/\tilde{\beta}_{\hat{u}}^{(\hat{k})}}, \\ s'+s'' &\leq \left\{D_{10} + 4284\left(\tilde{\beta}_{\hat{u}}^{(\hat{k})}+5\right)^2 2^{\tilde{\beta}_{\hat{u}}^{(\hat{k})}}\right\} M^{t_{\hat{u}}^{(\hat{k})}/\tilde{\beta}_{\hat{u}}^{(\hat{k})}} \log_2(M), \end{aligned}$$

thus proof is complete. □

Lemma 7.6. *There exists some constants C_1, C_2, C_3 only depending on $\mathcal{H}$ and C_0, C_3^* depending on $\mathcal{H}$, C_0, a, and b, and a network $\widehat{\pi} \in \mathcal{F}(L, J, \boldsymbol{p}, s)$, where*

- $L \leq C_1 \log n$,
- $\|\boldsymbol{p}\|_\infty \leq \max\{J, K, C_2 n^\nu\}$,
- $s \leq C_3 n^\nu \log n$,

such that

$$\mathrm{E}_{\boldsymbol{h}}\left\{\boldsymbol{\pi}_J^\intercal(\boldsymbol{\xi}_J)\left[(C_0\epsilon^{-1}) \wedge \log\left(\frac{\boldsymbol{\pi}_J(\boldsymbol{\xi}_J)}{\widehat{\boldsymbol{\pi}}(\boldsymbol{\xi}_J)}\right)\right]\right\} \leq C_3^* n^{-\theta}\log^3 n$$

for $\widehat{\boldsymbol{\pi}}$ taking values on $[a,b]$, $a<b\in\mathbb{R}$.

Proof. According to Equations (7.1), (7.2), (7.3), and proof of Theorem 3.3 in [4], let $M = \lfloor c_K n^{\bar{\mu}} \rfloor$ for some small constant c_K depending on K, where $\bar{\mu} = \frac{1+\alpha_{\hat{l}}}{1+\bar{\alpha}} \cdot \frac{\beta_{\hat{v}}^{(\hat{l})^*}}{(1+\alpha_{\hat{l}})\beta_{\hat{v}}^{(\hat{l})^*} + t_{\hat{v}}^{(\hat{l})}}$, the conclusion follows for $L \leq C_1 \log n$, $\|\boldsymbol{p}\|_\infty \leq \max\{J, K, C_2 n^\nu\}$, $s \leq C_3 n^\nu \log n$, where C_1, C_2, C_3 depend on D_{11}, D_{12} and D_{13} respectively along with c_K. □

Lemma 7.7. *Assumption 1 implies that there exists a constant C_5, such that*

$$\left|E\left(\pi_k(\boldsymbol{\xi}) - \pi_k^{(J)}(\boldsymbol{\xi}_J)\right)\right| \leq C_5\zeta(J)$$

for all $k = 1, \ldots, K$.

Proof. Without loss of generality, suppose $\int_0^\infty \pi(x)dx = 1$, otherwise we scale it by $C_5 = \int_0^\infty \pi(x)dx$.

$$\begin{aligned} \left|E\left(\pi_k - \pi_k^{(J)}\right)\right| &\leq E|\pi_k - \pi_k^{(J)}| = \int_0^\infty \mathrm{P}(|\pi_k - \pi_k^{(J)}| > x)dx \\ &\leq \int_0^\infty \zeta(J)\Gamma(x)dx = C_5\zeta(J). \end{aligned}$$

□

Lemma 7.8. *Assumptions 1 and 2 imply that there exists a constant C_6, such that*

$$\mathrm{P}\left(\pi_k^{(J)}(\boldsymbol{\xi}_J) \leq x\right) \leq C_6\left(x^{\alpha_k} + \zeta(J)\right), \quad \forall x \in (0,1], \quad k = 1, \ldots, K, \quad \forall J \geq J_0.$$

Proof. According to Assumption 1 and Assumption 2, for any $k = 1, \ldots, K$, we have

$$\begin{aligned}
\mathrm{P}\left(\pi_k^{(J)}(\boldsymbol{\xi}_J) \leq x\right) &= \mathrm{P}\left(\pi_k \leq x + \pi_k - \pi_k^{(J)}\right) \\
&\leq \mathrm{P}\left(\pi_k \leq x + |\pi_k - \pi_k^{(J)}|\right) \\
&= \mathrm{P}\left(\pi_k \leq x + |\pi_k - \pi_k^{(J)}|, |\pi_k - \pi_k^{(J)}| \leq x\right) \\
&\quad + \mathrm{P}\left(\pi_k \leq x + |\pi_k - \pi_k^{(J)}|, |\pi_k - \pi_k^{(J)}| > x\right) \\
&\leq \mathrm{P}\left(\pi_k \leq 2x\right) + \mathrm{P}\left(|\pi_k - \pi_k^{(J)}| > x\right) \\
&\leq C(2x)^{\alpha_k} + \zeta(J)\Gamma(x) \leq \left((C2^{\alpha_k}) \vee \Gamma(x)\right)\left(x^{\alpha_k} + \zeta(J)\right).
\end{aligned}$$

Let $C_6 = \max\{C2^{\alpha_1}, \ldots, C2^{\alpha_K}, \sup_{x\in(0,1)} \Gamma(x)\}$, and the proof is complete. □

Lemma 7.9. *Assumption 1 and Assumption 2 imply that given the ϵ, for any $x \in (0, \epsilon)$,*

$$\mathrm{P}\left(\pi_k^{(J)}(\boldsymbol{\xi}_J) > x\right) \geq 1 - \Gamma(\epsilon - x)\zeta(J)$$

for all $k = 1, \ldots, K$ and $J \geq J_0$.

Proof. According to Assumption 1 and Assumption 2, for any $k = 1, \ldots, K$, we have

$$\begin{aligned}
&\mathrm{P}\left(\pi_k^{(J)} \geq x\right) \\
=\ &\mathrm{P}\left(\pi_k \geq x + \pi_k - \pi_k^{(J)}\right) \\
\geq\ &\mathrm{P}\left(\pi_k \geq x + |\pi_k - \pi_k^{(J)}|\right) \\
\geq\ &\mathrm{P}\left(\pi_k \geq x + |\pi_k - \pi_k^{(J)}|, |\pi_k - \pi_k^{(J)}| \leq \epsilon - x\right) \\
=\ &\mathrm{P}\left(\pi_k \geq x + |\pi_k - \pi_k^{(J)}| \middle| |\pi_k - \pi_k^{(J)}| \leq \epsilon - x\right) \mathrm{P}\left(|\pi_k - \pi_k^{(J)}| \leq \epsilon - x\right) \\
\geq\ &\mathrm{P}\left(\pi_k \geq \epsilon\right) \mathrm{P}\left(|\pi_k - \pi_k^{(J)}| \leq \epsilon - x\right) \\
\geq\ &1 - \Gamma(\epsilon - x)\zeta(J).
\end{aligned}$$

The proof is complete. □

Lemma 7.10. *Assumption 1 and Assumption 2 implies that there exists constants C_7 and C_8, such that $\mathrm{E}\left(\frac{1}{\pi_k}\right) \leq C_7$, and $\mathrm{E}\left(\frac{1}{\pi_k^{(J)}}\right) \leq C_8$ for all $k = 1, \ldots, K$ and $J \geq J_0$.*

Proof. Since $\mathrm{P}\left(\pi_k > \epsilon\right) = 1$, we have $\mathrm{P}\left(\frac{1}{\pi_k} \geq x\right) = 0$ for all $x > 1/\epsilon$. Therefore

$$\mathrm{E}\left(\frac{1}{\pi_k}\right) = \int_0^\infty \mathrm{P}\left(\frac{1}{\pi_k} \geq x\right) dx = \int_0^{\epsilon^{-1}} \mathrm{P}\left(\frac{1}{\pi_k} \geq x\right) dx$$

$$\begin{aligned}
&= \int_0^1 \mathrm{P}\left(\frac{1}{\pi_k} \geq x\right) dx + \int_1^{\epsilon^{-1}} \mathrm{P}\left(\frac{1}{\pi_k} \geq x\right) dx \\
&= \int_0^1 \mathrm{P}\left(\pi_k \leq \frac{1}{x}\right) dx + \int_1^{\epsilon^{-1}} \mathrm{P}\left(\frac{1}{\pi_k} \geq x\right) dx \\
&\leq 1 + \int_1^{\epsilon^{-1}} C x^{-\alpha_k} dx = C\epsilon^{\alpha_k - 1} + 1 - C,
\end{aligned}$$

where the inequality is owing to the derivation of Assumption 2, such that

$$\mathrm{P}\left(\frac{1}{\pi_k} \geq x\right) \leq C x^{-\alpha_k}, \quad \forall x \in [1, \infty)\,.$$

According to Lemma 7.9, since $\mathrm{P}\left(\pi_k^{(J)} > x\right) \geq 1 - \Gamma(\epsilon - x)\zeta(J)$ for all $x \in (0, \epsilon)$, we have $\mathrm{P}\left(\frac{1}{\pi_k^{(J)}} \geq x\right) \leq \Gamma(\epsilon - 1/x)\zeta(J)$ for all $x > 1/\epsilon$. Together with Lemma 7.8, we have

$$\begin{aligned}
&\mathrm{E}\left(\frac{1}{\pi_k^{(J)}}\right) \\
&= \int_0^1 \mathrm{P}\left(\frac{1}{\pi_k^{(J)}} \geq x\right) dx + \int_1^{\epsilon^{-1}} \mathrm{P}\left(\frac{1}{\pi_k^{(J)}} \geq x\right) dx + \int_{\epsilon^{-1}}^{\infty} \mathrm{P}\left(\frac{1}{\pi_k^{(J)}} \geq x\right) dx \\
&\leq \int_0^1 \mathrm{P}\left(\pi_k^{(J)} \leq \frac{1}{x}\right) dx + \int_1^{\epsilon^{-1}} \mathrm{P}\left(\frac{1}{\pi_k^{(J)}} \geq x\right) dx + \int_{\epsilon^{-1}}^{\infty} \Gamma(\epsilon - 1/x)\zeta(J) dx \\
&\leq 1 + \int_1^{\epsilon^{-1}} C_6 \left(x^{-\alpha_k} + \zeta(J)\right) dx + \Delta.
\end{aligned}$$

Let $C_7 = \max\{C\epsilon^{\alpha_1 - 1} + 1 - C, \ldots, C\epsilon^{\alpha_K - 1} + 1 - C\}$ and $C_8 = \max_k\{C_6\epsilon^{\alpha_1 - 1} + 1 - C_6 + (\epsilon^{-1} - 1)e(J_0) + \Delta, \ldots, C_6\epsilon^{\alpha_K - 1} + 1 - C_6 + (\epsilon^{-1} - 1)e(J_0) + \Delta\}$, the proof is complete. □

Lemma 7.11. *When $E|\xi_j - \widehat{\xi}_j| \leq m^{-\tau}$ for all $j = 1, \ldots, J$, where J is some relatively large constant, Assumption 1 and Assumption 2 imply that there exists a constant C_9, such that*

$$\mathrm{P}\left(\pi_k^{(J)}(\widehat{\boldsymbol{\xi}}_J) \leq x\right) \leq C_9 \left(x^{\alpha_k} + \zeta(J) + x m^{-\tau\widetilde{\beta}}\right), \qquad \forall x \in (0, 1]\,, \forall k \in \{1, \ldots, K\}.$$

Proof. We first give the expected distance between $\pi_k(\boldsymbol{\xi}_J)$ and $\pi_k^{(J)}(\widehat{\boldsymbol{\xi}}_J)$. By the definition of $\pi_k^{(J)}$, we have $\left|\pi_k^{(J)}(\boldsymbol{\xi}_J) - \pi_k^{(J)}(\widehat{\boldsymbol{\xi}}_J)\right| \leq R^{q^{(k)}+1} \|\boldsymbol{\xi}_J - \widehat{\boldsymbol{\xi}}_J\|_\infty^{\widetilde{\beta}_0^{(k)} \wedge 1}$, and thus

$$\mathrm{E}\left|\pi_k^{(J)}(\boldsymbol{\xi}_J) - \pi_k^{(J)}(\widehat{\boldsymbol{\xi}}_J)\right| \leq C_R m^{-\tau\widetilde{\beta}}, \forall k = 1, \ldots, K,$$

where $C_R = \max_{k=1,\ldots,K} R^{q^{(k)}+1}$. Together with Lemma 7.7, we have

$$\left|E\left(\pi_k(\boldsymbol{\xi}) - \pi_k^{(J)}(\widehat{\boldsymbol{\xi}}_J)\right)\right| \leq C_5\zeta(J) + C_R m^{-\tau\widetilde{\beta}}, \forall k = 1, \ldots, K.$$

According to Assumption 1 and Assumption 2, for any $k = 1, \ldots, K$, we have

$$\begin{aligned}
& \mathrm{P}\left(\pi_k^{(J)}(\widehat{\boldsymbol{\xi}}_J) \le x\right) \\
= \; & \mathrm{P}\left(\pi_k^{(J)}(\boldsymbol{\xi}_J) \le x + \pi_k^{(J)}(\boldsymbol{\xi}_J) - \pi_k^{(J)}(\widehat{\boldsymbol{\xi}}_J)\right) \\
\le \; & \mathrm{P}\left(\pi_k^{(J)}(\boldsymbol{\xi}_J) \le x + \left|\pi_k^{(J)}(\boldsymbol{\xi}_J) - \pi_k^{(J)}(\widehat{\boldsymbol{\xi}}_J)\right|\right) \\
\le \; & \mathrm{P}\left(\pi_k^{(J)}(\boldsymbol{\xi}_J) \le x + \left|\pi_k^{(J)}(\boldsymbol{\xi}_J) - \pi_k^{(J)}(\widehat{\boldsymbol{\xi}}_J)\right|, \left|\pi_k^{(J)}(\boldsymbol{\xi}_J) - \pi_k^{(J)}(\widehat{\boldsymbol{\xi}}_J)\right| \le x\right) \\
& + \mathrm{P}\left(\pi_k^{(J)}(\boldsymbol{\xi}_J) \le x + \left|\pi_k^{(J)}(\boldsymbol{\xi}_J) - \pi_k^{(J)}(\widehat{\boldsymbol{\xi}}_J)\right|, \left|\pi_k^{(J)}(\boldsymbol{\xi}_J) - \pi_k^{(J)}(\widehat{\boldsymbol{\xi}}_J)\right| > x\right) \\
\le \; & \mathrm{P}\left(\pi_k^{(J)}(\boldsymbol{\xi}_J) \le 2x\right) + \mathrm{P}\left(\left|\pi_k^{(J)}(\boldsymbol{\xi}_J) - \pi_k^{(J)}(\widehat{\boldsymbol{\xi}}_J)\right| > x\right) \\
\le \; & C_6(x^{\alpha_k} + \zeta(J)) + x\,\mathrm{E}\left|\pi_k^{(J)}(\boldsymbol{\xi}_J) - \pi_k^{(J)}(\widehat{\boldsymbol{\xi}}_J)\right| \\
\le \; & C_6(x^{\alpha_k} + \zeta(J)) + xC_5\zeta(J) + xC_R m^{-\tau\widetilde{\beta}} \\
\le \; & C_9\left(x^{\alpha_k} + \zeta(J) + xm^{-\tau\widetilde{\beta}}\right),
\end{aligned}$$

where $C_9 = (C_5 + C_6) \vee C_R$, the proof is complete. □

Define the set of the effective inputs of the k-th group by

$$\mathcal{A}_k = \left\{j : \xi_j \text{ is effective for } g_{0v}^{(k)} \text{ for all } v = 1, \ldots, d_1^{(k)}\right\},$$

such that $|\mathcal{A}_k| \le t_0^{(k)} d_1^{(k)} < \infty$. Let $\mathcal{A} = \bigcup_{k=1}^K \mathcal{A}_k$ be the effective inputs among all groups, where $|\mathcal{A}| \le \sum_{k=1}^K t_0^{(k)} d_1^{(k)} < \infty$. Given an integer M_0, let $\mathcal{B}_{M_0} = \{|\xi_j| \le M_0 \text{ for all } j \in \mathcal{A}\}$ be the concentration set for all effective inputs.

Lemma 7.12. *Under Assumptions 1 and 2, for any $\boldsymbol{h} \in \mathcal{H}$ and $C_0 > 0$, there exists some constants M_0, M_0^*, such that the functional deep neural network classifier $\widehat{\boldsymbol{\pi}}$ in network space $\mathcal{F}(L, J, \boldsymbol{p}, s)$ which is derived in Lemma 7.6 satisfies*

$$\begin{aligned}
R_{\boldsymbol{h},C_0}(\widehat{\boldsymbol{\pi}}) \le \; & 4C_0K^2\left\{(C_5+1)^{2+\max_k \alpha_k \wedge 1} C_6\epsilon^{-1} + (4+C_4)C_5M_0^*\right\}\zeta(J) \\
& + \mathrm{E}_{\boldsymbol{h}}\left\{\boldsymbol{\pi}_J^{\intercal}(\boldsymbol{\xi}_J)\left[(C_0\epsilon^{-1}) \wedge \log\left(\frac{\boldsymbol{\pi}_J(\boldsymbol{\xi}_J)}{\widehat{\boldsymbol{\pi}}(\boldsymbol{\xi}_J)}\right)\right]\right\},
\end{aligned}$$

where $\widehat{\boldsymbol{\pi}}$ takes value on $(-M_0, M_0)$.

Proof. For any $a, b, c \ge 0$, we have a simple fact that

$$a \wedge (b + c) \le (a + c) \wedge (b + c) = (a \wedge b) + c.$$

Let $\mathbf{1}_K = (1, \ldots, 1)^{\intercal}$ with length K, we have

$$\begin{aligned}
& \mathrm{E}_{\boldsymbol{h}}\left[\boldsymbol{\pi}(\boldsymbol{\xi})^{\intercal}\left\{C_0\mathbf{1}_K \wedge \log\left(\frac{\boldsymbol{\pi}(\boldsymbol{\xi})}{\widehat{\boldsymbol{\pi}}(\boldsymbol{\xi}_J)}\right)\right\}\right] \\
= \; & \mathrm{E}_{\boldsymbol{h}}\left[(\boldsymbol{\pi}(\boldsymbol{\xi})^{\intercal}C_0) \wedge \left\{\boldsymbol{\pi}(\boldsymbol{\xi})^{\intercal}\log\left(\frac{\boldsymbol{\pi}(\boldsymbol{\xi})}{\widehat{\boldsymbol{\pi}}(\boldsymbol{\xi}_J)}\right)\right\}\right]
\end{aligned}$$

$$
\begin{aligned}
&\leq \mathrm{E}_{\boldsymbol{h}}\left\{C_0 \wedge \left[(\boldsymbol{\pi}(\boldsymbol{\xi}) - \boldsymbol{\pi}_J(\boldsymbol{\xi}_J) + \boldsymbol{\pi}_J(\boldsymbol{\xi}_J))^{\intercal} \log\left(\frac{\boldsymbol{\pi}(\boldsymbol{\xi})}{\boldsymbol{\pi}_J(\boldsymbol{\xi}_J)} \cdot \frac{\boldsymbol{\pi}_J(\boldsymbol{\xi}_J)}{\widehat{\boldsymbol{\pi}}(\boldsymbol{\xi}_J)}\right)\right]\right\} \\
&\leq \mathrm{E}_{\boldsymbol{h}}\left[\boldsymbol{\pi}(\boldsymbol{\xi})^{\intercal} \log\left(\frac{\boldsymbol{\pi}(\boldsymbol{\xi})}{\boldsymbol{\pi}_J(\boldsymbol{\xi}_J)}\right)\right] + \mathrm{E}_{\boldsymbol{h}}\left[C_0 \wedge \left|(\boldsymbol{\pi}(\boldsymbol{\xi}) - \boldsymbol{\pi}_J(\boldsymbol{\xi}_J))^{\intercal} \log\left(\frac{\boldsymbol{\pi}_J(\boldsymbol{\xi}_J)}{\widehat{\boldsymbol{\pi}}(\boldsymbol{\xi}_J)}\right)\right|\right] \\
&\quad + \mathrm{E}_{\boldsymbol{h}}\left\{C_0 \wedge \left[\boldsymbol{\pi}_J^{\intercal}(\boldsymbol{\xi}_J) \log\left(\frac{\boldsymbol{\pi}_J(\boldsymbol{\xi}_J)}{\widehat{\boldsymbol{\pi}}(\boldsymbol{\xi}_J)}\right)\right]\right\} \\
&\leq \sum_{k=1}^{K} \mathrm{E}_{\boldsymbol{h}}\left[\frac{\left(\pi_k(\boldsymbol{\xi}) - \pi_k^{(J)}(\boldsymbol{\xi}_J)\right)^2}{\pi_k^{(J)}(\boldsymbol{\xi}_J)}\right] \\
&\quad + \sum_{k=1}^{K} \mathrm{E}_{\boldsymbol{h}}\left[C_0 \wedge \left|\left(\pi_k(\boldsymbol{\xi}) - \pi_k^{(J)}(\boldsymbol{\xi}_J)\right) \log\left(1 + \frac{\pi_k^{(J)}(\boldsymbol{\xi}_J) - \widehat{\pi}^{(k)}(\boldsymbol{\xi}_J)}{\widehat{\pi}^{(k)}(\boldsymbol{\xi}_J)}\right)\right| \mathbb{I}\left(\mathcal{B}_{M_0}\right)\right] \\
&\quad + \mathrm{E}_{\boldsymbol{h}}\left\{C_0 \wedge \left[\boldsymbol{\pi}_J^{\intercal}(\boldsymbol{\xi}_J) \log\left(\frac{\boldsymbol{\pi}_J(\boldsymbol{\xi}_J)}{\widehat{\boldsymbol{\pi}}(\boldsymbol{\xi}_J)}\right) \mathbb{I}\left(\mathcal{B}_{M_0}\right)\right]\right\} + 2C_0 K \mathrm{P}\left(\mathcal{B}_{M_0}^c\right),
\end{aligned}
$$

where the first inequality is because of $\pi_k \leq 1$, the second inequality is owing to the aforementioned simple fact, and the last inequality holds for the first term since KL-divergence is upper bounded by χ^2-divergence.

For the first term, when J is relatively large, we have

$$
\begin{aligned}
&\sum_{k=1}^{K} \mathrm{E}_{\boldsymbol{h}}\left[\frac{\left(\pi_k(\boldsymbol{\xi}) - \pi_k^{(J)}(\boldsymbol{\xi}_J)\right)^2}{\pi_k^{(J)}(\boldsymbol{\xi}_J)}\right] \\
= &\sum_{k=1}^{K} \mathrm{E}_{\boldsymbol{h}}\left[\frac{\left(\pi_k(\boldsymbol{\xi}) - \pi_k^{(J)}(\boldsymbol{\xi}_J)\right)^2}{\pi_k^{(J)}(\boldsymbol{\xi}_J)} \mathbb{I}\left(\pi_k^{(J)}(\boldsymbol{\xi}_J) > \epsilon\right)\right] \\
&+ \sum_{k=1}^{K} \mathrm{E}_{\boldsymbol{h}}\left[\frac{\left(\pi_k(\boldsymbol{\xi}) - \pi_k^{(J)}(\boldsymbol{\xi}_J)\right)^2}{\pi_k^{(J)}(\boldsymbol{\xi}_J)} \mathbb{I}\left(\pi_k^{(J)}(\boldsymbol{\xi}_J) \leq \epsilon\right)\right] \\
< &\sum_{k=1}^{K} \mathrm{E}_{\boldsymbol{h}}\left[\frac{\left(\pi_k(\boldsymbol{\xi}) - \pi_k^{(J)}(\boldsymbol{\xi}_J)\right)^2}{\epsilon}\right] \\
&+ \sum_{k=1}^{K} \mathrm{E}_{\boldsymbol{h}}\left[\frac{\left(\pi_k(\boldsymbol{\xi}) - \pi_k^{(J)}(\boldsymbol{\xi}_J)\right)^2}{\pi_k^{(J)}(\boldsymbol{\xi}_J)}\right] \mathrm{P}\left(\pi_k^{(J)}(\boldsymbol{\xi}_J) \leq \epsilon\right) \\
\leq &\; K\epsilon^{-1} C_5 \zeta(J) + 2C_0 K (C_5 + 1)^{2 + \max_k \alpha_k \wedge 1} C_6 \epsilon^{\max_k \alpha_k} \zeta(J) \\
\leq &\; 4C_0 K (C_5 + 1)^{2 + \max_k \alpha_k \wedge 1} C_6 \epsilon^{-1} \zeta(J),
\end{aligned}
$$

where the second last inequality is derived by Lemmas 7.7 and 7.8, and Theorem 3.2 in [4].

For any $x, y \in \mathbb{R}$, there exists C_0', such that

$$|x \log(1+y)| \wedge C_0 \leq |x \log(1+y)| \mathbb{I}(|y| \leq C_0').$$

Therefore, combining the fact that given M_0, within the concentrated set $\mathcal{B}_{M_0}$, there exists an M_0^*, such that $\left[\widehat{\pi}^{(k)}(\boldsymbol{\xi}_J)\right]^{-1} \mathbb{I}(\mathcal{B}_{M_0}) \leq M_0^*$ for $\forall \boldsymbol{\xi}_J$, the second term can be upper bounded as

$$\begin{aligned}
& \sum_{k=1}^{K} \mathrm{E}_{\boldsymbol{h}} \left[C_0 \wedge \left| \left(\pi_k(\boldsymbol{\xi}) - \pi_k^{(J)}(\boldsymbol{\xi}_J) \right) \log \left(1 + \frac{\pi_k^{(J)}(\boldsymbol{\xi}_J) - \widehat{\pi}^{(k)}(\boldsymbol{\xi}_J)}{\widehat{\pi}^{(k)}(\boldsymbol{\xi}_J)} \right) \right| \mathbb{I}(\mathcal{B}_{M_0}) \right] \\
\leq & \sum_{k=1}^{K} \mathrm{E}_{\boldsymbol{h}} \left[\left| \pi_k(\boldsymbol{\xi}) - \pi_k^{(J)}(\boldsymbol{\xi}_J) \right| \log \left(1 + \frac{\left| \pi_k^{(J)}(\boldsymbol{\xi}_J) - \widehat{\pi}^{(k)}(\boldsymbol{\xi}_J) \right| \wedge C_0'}{\widehat{\pi}^{(k)}(\boldsymbol{\xi}_J)} \right) \mathbb{I}(\mathcal{B}_{M_0}) \right] \\
\leq & \sum_{k=1}^{K} \mathrm{E}_{\boldsymbol{h}} \left[\left| \pi_k(\boldsymbol{\xi}) - \pi_k^{(J)}(\boldsymbol{\xi}_J) \right| \frac{\left| \pi_k^{(J)}(\boldsymbol{\xi}_J) - \widehat{\pi}^{(k)}(\boldsymbol{\xi}_J) \right| \wedge C_0'}{\widehat{\pi}^{(k)}(\boldsymbol{\xi}_J)} \mathbb{I}(\mathcal{B}_{M_0}) \right] \\
\leq & 2K(4+C_4) \sum_{k=1}^{K} \mathrm{E}_{\boldsymbol{h}} \left(\left| \pi_k(\boldsymbol{\xi}) - \pi_k^{(J)}(\boldsymbol{\xi}_J) \right| M_0^* \right) \\
\leq & 2K^2(4+C_4) C_5 M_0^* \zeta(J),
\end{aligned}$$

where the second last inequality is based on Lemma 7.5 and the choice of M in Lemma 7.6 is greater than 1.

It is trivial to see that, for any $\delta > 0$, there exists an $M_0 > 0$, such that

$$\mathrm{P}(\mathcal{B}_{M_0}) = P(|\xi_j| \leq M_0 \text{ for all } j \in \mathcal{A}) \geq 1 - \delta,$$

for all $j \in \mathcal{A}$. Therefore, for any $J \geq J_0$, when δ is relatively small, there exists a corresponding M_0, such that

$$K(4+C_4) C_5 M_0^* \zeta(J) \geq C_0 \delta.$$

Consequently, under Assumption 2, we have

$$\begin{aligned}
& \mathrm{E}_{\boldsymbol{h}} \left[\boldsymbol{\pi}(\boldsymbol{\xi})^{\intercal} \left(C_0 \wedge \log \left(\frac{\boldsymbol{\pi}(\boldsymbol{\xi})}{\widehat{\boldsymbol{\pi}}(\boldsymbol{\xi}_J)} \right) \right) \right] \\
\leq & 4C_0 K (C_5+1)^{2+\max_k \alpha_k \wedge 1} C_6 \epsilon^{-1} \zeta(J) + 4K^2(4+C_4) C_5 M_0^* \zeta(J) \\
& + \mathrm{E}_{\boldsymbol{h}} \left\{ C_0 \wedge \left[\boldsymbol{\pi}_J^{\intercal}(\boldsymbol{\xi}_J) \log \left(\frac{\boldsymbol{\pi}_J(\boldsymbol{\xi}_J)}{\widehat{\boldsymbol{\pi}}(\boldsymbol{\xi}_J)} \right) \mathbb{I}(\mathcal{B}_{M_0}) \right] \right\} \\
\leq & 4C_0 K^2 \left\{ (C_5+1)^{2+\max_k \alpha_k \wedge 1} C_6 \epsilon^{-1} + (4+C_4) C_5 M_0^* \right\} \zeta(J) \\
& + \mathrm{E}_{\boldsymbol{h}} \left\{ \boldsymbol{\pi}_J^{\intercal}(\boldsymbol{\xi}_J) \left[(C_0 \epsilon^{-1}) \wedge \log \left(\frac{\boldsymbol{\pi}_J(\boldsymbol{\xi}_J)}{\widehat{\boldsymbol{\pi}}(\boldsymbol{\xi}_J)} \right) \right] \mathbb{I}(\mathcal{B}_{M_0}) \right\}.
\end{aligned}$$

□

7.1. Proof of Theorem 3.1

Proof. According to Lemma 7.6 and Lemma 7.12, there exists a constant C_4^* adjusted only by M_0 and C_3^*, such that for any $J_0 \geq J \leq C_2 n^\nu$, the optimal rate is achieved as

$$\begin{aligned} & \sup_{\boldsymbol{h}\in\mathcal{H}} \mathrm{E}_{\boldsymbol{h}} \left[\boldsymbol{\pi}(\boldsymbol{\xi})^\intercal \left\{ C_0 \wedge \log\left(\frac{\boldsymbol{\pi}(\boldsymbol{\xi})}{\widehat{\boldsymbol{\pi}}(\boldsymbol{\xi}_J)} \right) \right\} \right] \\ \leq \; & 4C_0K^2 \left((C_5+1)^{2+\max_k \alpha_k \wedge 1} C_6 \epsilon^{-1} + (4+C_4)C_5 M_0^* \right) \zeta(J) + C_4^* n^{-\theta} \log^3 n. \end{aligned}$$

By Assumption 1, there exists constants C_2' and ω_1, such that $\forall J \geq C_2' n^{\theta/\rho}$, we have $\zeta(J) \lesssim n^{-\theta}$, and $\sup_{\boldsymbol{h}\in\mathcal{H}} \mathrm{E}_{\boldsymbol{h}} \left[\boldsymbol{\pi}(\boldsymbol{\xi})^\intercal \left(C_0 \wedge \log\left(\frac{\boldsymbol{\pi}(\boldsymbol{\xi})}{\widehat{\boldsymbol{\pi}}(\boldsymbol{\xi}_J)} \right) \right) \right] \leq \omega_1 n^{-\theta} \log^3 n$. The range of optimal J is obtained as $J \in \left[C_2' n^{\theta/\rho}, C_2 n^\nu \right]$. □

7.2. Proof of Theorem 3.2

Proof. We follow the proof of Lemma 7.12 to decompose the truncated KL risk in terms of $\widehat{\boldsymbol{\pi}}(\widehat{\boldsymbol{\xi}}_J)$, such that

$$\begin{aligned} & \mathrm{E}_{\boldsymbol{h}} \left[\boldsymbol{\pi}(\boldsymbol{\xi})^\intercal \left(C_0 \mathbf{1}_K \wedge \log\left(\frac{\boldsymbol{\pi}(\boldsymbol{\xi})}{\widehat{\boldsymbol{\pi}}(\widehat{\boldsymbol{\xi}}_J)} \right) \right) \right] \\ = \; & \mathrm{E}_{\boldsymbol{h}} \left[(\boldsymbol{\pi}(\boldsymbol{\xi})^\intercal C_0) \wedge \left(\boldsymbol{\pi}(\boldsymbol{\xi})^\intercal \log\left(\frac{\boldsymbol{\pi}(\boldsymbol{\xi})}{\widehat{\boldsymbol{\pi}}(\widehat{\boldsymbol{\xi}}_J)} \right) \right) \right] \\ \leq \; & \mathrm{E}_{\boldsymbol{h}} \left\{ C_0 \wedge \left[\boldsymbol{\pi}(\boldsymbol{\xi})^\intercal \log\left(\frac{\boldsymbol{\pi}(\boldsymbol{\xi})}{\boldsymbol{\pi}_J(\widehat{\boldsymbol{\xi}}_J)} \right) + \left(\boldsymbol{\pi}(\boldsymbol{\xi}) - \boldsymbol{\pi}_J(\widehat{\boldsymbol{\xi}}_J) \right)^\intercal \log\left(\frac{\boldsymbol{\pi}_J(\widehat{\boldsymbol{\xi}}_J)}{\widehat{\boldsymbol{\pi}}(\widehat{\boldsymbol{\xi}}_J)} \right) \right. \right. \\ & \left. \left. + \boldsymbol{\pi}_J^\intercal(\widehat{\boldsymbol{\xi}}_J) \log\left(\frac{\boldsymbol{\pi}_J(\widehat{\boldsymbol{\xi}}_J)}{\widehat{\boldsymbol{\pi}}(\widehat{\boldsymbol{\xi}}_J)} \right) \right] \right\} \\ \leq \; & \mathrm{E}_{\boldsymbol{h}} \left[\boldsymbol{\pi}(\boldsymbol{\xi})^\intercal \log\left(\frac{\boldsymbol{\pi}(\boldsymbol{\xi})}{\boldsymbol{\pi}_J(\widehat{\boldsymbol{\xi}}_J)} \right) \right] \\ & + \mathrm{E}_{\boldsymbol{h}} \left[C_0 \wedge \left| \left(\boldsymbol{\pi}(\boldsymbol{\xi}) - \boldsymbol{\pi}_J(\widehat{\boldsymbol{\xi}}_J) \right)^\intercal \log\left(\frac{\boldsymbol{\pi}_J(\widehat{\boldsymbol{\xi}}_J)}{\widehat{\boldsymbol{\pi}}(\widehat{\boldsymbol{\xi}}_J)} \right) \right| \right] \\ & + \mathrm{E}_{\boldsymbol{h}} \left\{ C_0 \wedge \left[\boldsymbol{\pi}_J^\intercal(\widehat{\boldsymbol{\xi}}_J) \log\left(\frac{\boldsymbol{\pi}_J(\widehat{\boldsymbol{\xi}}_J)}{\widehat{\boldsymbol{\pi}}(\widehat{\boldsymbol{\xi}}_J)} \right) \right] \right\} \\ \leq \; & \sum_{k=1}^K \mathrm{E}_{\boldsymbol{h}} \left[\frac{\left(\pi_k(\boldsymbol{\xi}) - \pi_k^{(J)}(\boldsymbol{\xi}_J) \right)^2}{\pi_k^{(J)}(\widehat{\boldsymbol{\xi}}_J)} \right] \\ & + \sum_{k=1}^K \mathrm{E}_{\boldsymbol{h}} \left[C_0 \wedge \left| \left(\pi_k(\boldsymbol{\xi}) - \pi_k^{(J)}(\widehat{\boldsymbol{\xi}}_J) \right) \log\left(1 + \frac{\pi_k^{(J)}(\widehat{\boldsymbol{\xi}}_J) - \widehat{\pi}^{(k)}(\widehat{\boldsymbol{\xi}}_J)}{\widehat{\pi}^{(k)}(\widehat{\boldsymbol{\xi}}_J)} \right) \right| \mathbb{I}\left(\mathcal{B}_{M_0} \right) \right] \end{aligned}$$

$$+\mathrm{E}_{\boldsymbol{h}}\left\{C_0\wedge\left[\boldsymbol{\pi}_J^{\intercal}(\widehat{\boldsymbol{\xi}}_J)\log\left(\frac{\boldsymbol{\pi}_J(\widehat{\boldsymbol{\xi}}_J)}{\widehat{\boldsymbol{\pi}}(\widehat{\boldsymbol{\xi}}_J)}\right)\mathbb{I}\left(\mathcal{B}_{M_0}\right)\right]\right\}+2C_0K\mathrm{P}\left(\mathcal{B}_{M_0}^c\right)$$

For the first term, when J is relatively large, we have

$$\begin{aligned}
&\sum_{k=1}^{K}\mathrm{E}_{\boldsymbol{h}}\left[\frac{\left(\pi_k(\boldsymbol{\xi})-\pi_k^{(J)}(\widehat{\boldsymbol{\xi}}_J)\right)^2}{\pi_k^{(J)}(\widehat{\boldsymbol{\xi}}_J)}\right]\\
=&\sum_{k=1}^{K}\mathrm{E}_{\boldsymbol{h}}\left[\frac{\left(\pi_k(\boldsymbol{\xi})-\pi_k^{(J)}(\widehat{\boldsymbol{\xi}}_J)\right)^2}{\pi_k^{(J)}(\widehat{\boldsymbol{\xi}}_J)}\mathbb{I}\left(\pi_k^{(J)}(\widehat{\boldsymbol{\xi}}_J)>\epsilon\right)\right]\\
&+\sum_{k=1}^{K}\mathrm{E}_{\boldsymbol{h}}\left[\frac{\left(\pi_k(\boldsymbol{\xi})-\pi_k^{(J)}(\widehat{\boldsymbol{\xi}}_J)\right)^2}{\pi_k^{(J)}(\widehat{\boldsymbol{\xi}}_J)}\mathbb{I}\left(\pi_k^{(J)}(\widehat{\boldsymbol{\xi}}_J)\le\epsilon\right)\right]\\
<&\sum_{k=1}^{K}\mathrm{E}_{\boldsymbol{h}}\left[\frac{\left(\pi_k(\boldsymbol{\xi})-\pi_k^{(J)}(\widehat{\boldsymbol{\xi}}_J)\right)^2}{\epsilon}\right]\\
&+\sum_{k=1}^{K}\mathrm{E}_{\boldsymbol{h}}\left[\frac{\left(\pi_k(\boldsymbol{\xi})-\pi_k^{(J)}(\widehat{\boldsymbol{\xi}}_J)\right)^2}{\pi_k^{(J)}(\widehat{\boldsymbol{\xi}}_J)}\right]\mathrm{P}\left(\pi_k^{(J)}(\widehat{\boldsymbol{\xi}}_J)\le\epsilon\right)\\
\le&K\epsilon^{-1}\left(C_5\zeta(J)+C_Rm^{-\tau\tilde{\beta}}\right)\\
&+C_0K(C_5+1)^{2+\max_k\alpha_k\wedge1}C_9\left(2\zeta(J)+\epsilon m^{-\tau\tilde{\beta}}\right)\\
\le&\left(K\epsilon^{-1}C_5+2C_0K(C_5+1)^{2+\max_k\alpha_k\wedge1}C_9\right)\zeta(J)\\
&+\left(K\epsilon^{-1}C_R+C_0K(C_5+1)^{2+\max_k\alpha_k\wedge1}C_9\epsilon\right)m^{-\tau\tilde{\beta}},
\end{aligned}$$

where the second last inequality is derived by Lemma 7.11 and Theorem 3.2 in [4].

Given M_0, within the concentrated set $\mathcal{B}_{M_0}$, there exists an $\widetilde{M}_0^*$, such that $\left[\widehat{\pi}^{(k)}(\widehat{\boldsymbol{\xi}}_J)\right]^{-1}\mathbb{I}\left(\mathcal{B}_{M_0}\right)\le M_0^*$ for $\forall\boldsymbol{\xi}_J$, thus the second term can be upper bounded as

$$\begin{aligned}
&\sum_{k=1}^{K}\mathrm{E}_{\boldsymbol{h}}\left[C_0\wedge\left|\left(\pi_k(\boldsymbol{\xi})-\pi_k^{(J)}(\widehat{\boldsymbol{\xi}}_J)\right)\log\left(1+\frac{\pi_k^{(J)}(\widehat{\boldsymbol{\xi}}_J)-\widehat{\pi}^{(k)}(\widehat{\boldsymbol{\xi}}_J)}{\widehat{\pi}^{(k)}(\widehat{\boldsymbol{\xi}}_J)}\right)\right|\mathbb{I}\left(\mathcal{B}_{M_0}\right)\right]\\
\le&\sum_{k=1}^{K}\mathrm{E}_{\boldsymbol{h}}\left[\left|\pi_k(\boldsymbol{\xi})-\pi_k^{(J)}(\boldsymbol{\xi}_J)\right|\frac{\left|\pi_k^{(J)}(\widehat{\boldsymbol{\xi}}_J)-\widehat{\pi}^{(k)}(\widehat{\boldsymbol{\xi}}_J)\right|\wedge C_0'}{\widehat{\pi}^{(k)}(\widehat{\boldsymbol{\xi}}_J)}\mathbb{I}\left(\mathcal{B}_{M_0}\right)\right]\\
\le&2K(4+C_4)\sum_{k=1}^{K}\mathrm{E}_{\boldsymbol{h}}\left(\left|\pi_k(\boldsymbol{\xi})-\pi_k^{(J)}(\widehat{\boldsymbol{\xi}}_J)\right|\widetilde{M}_0^*\right)
\end{aligned}$$

$$\leq \quad 2K^2(4+C_4)\widetilde{M}_0^* \left(C_5\zeta(J)+C_R m^{-\tau\widetilde{\beta}}\right),$$

where the second last inequality is based on Lemma 7.5 and the choice of M in Lemma 7.6 is greater than 1. The last inequality is from the proof of Lemma 7.11.

Consequently, similar in proof of Theorem 3.1, we have

$$\begin{aligned}
&\mathrm{E}_{\boldsymbol{h}}\left[\boldsymbol{\pi}(\boldsymbol{\xi})^{\intercal}\left(C_0\mathbf{1}_K \wedge \log\left(\frac{\boldsymbol{\pi}(\boldsymbol{\xi})}{\widehat{\boldsymbol{\pi}}(\widehat{\boldsymbol{\xi}}_J)}\right)\right)\right]\\
\leq\ & 2\left(K\epsilon^{-1}C_5+2C_0K(C_5+1)^{2+\max_k \alpha_k\wedge 1}C_9+2K^2(4+C_4)\widetilde{M}_0^*C_5\right)\zeta(J)\\
&+2\left(K\epsilon^{-1}C_R+C_0K(C_5+1)^{2+\max_k \alpha_k\wedge 1}C_9\epsilon+2K^2(4+C_4)\widetilde{M}_0^*C_R\right)m^{-\tau\widetilde{\beta}}\\
&+\mathrm{E}_{\boldsymbol{h}}\left\{C_0\wedge\left[\boldsymbol{\pi}_J^{\intercal}(\widehat{\boldsymbol{\xi}}_J)\log\left(\frac{\boldsymbol{\pi}_J(\widehat{\boldsymbol{\xi}}_J)}{\widehat{\boldsymbol{\pi}}(\widehat{\boldsymbol{\xi}}_J)}\right)\mathbb{I}\left(\mathcal{B}_{M_0}\right)\right]\right\}\\
=\ & \widetilde{c}_1\zeta(J)+\widetilde{c}_2 m^{-\tau\widetilde{\beta}}+C_4^* n^{-\theta}\log^3 n.
\end{aligned}$$

When $\widetilde{c}_2 m^{-\tau\widetilde{\beta}} \leq C_4^* n^{-\theta}\log^3 n$, the result follows by Theorem 3.1, where we define the corresponding constants by C_1, C_2, C_2', and C_3.

When $\widetilde{c}_2 m^{-\tau\widetilde{\beta}} < C_4^* n^{-\theta}\log^3 n$, according to Equations (7.1), (7.2), (7.3), and proof of Lemma 7.6, let $M = \lfloor \widetilde{c}_K n^{\widetilde{\mu}} \rfloor$ for some small constant $\widetilde{c}_K$ depending on K, where $\widetilde{\mu} = \frac{1}{1+\widetilde{\alpha}}\widetilde{\theta}$, the conclusion follows for $L \leq \widetilde{C}_1 \log m$, $\|\boldsymbol{p}\|_\infty \leq \max\{J, K, \widetilde{C}_2 m^{\nu'}\}$, $s \leq \widetilde{C}_3 m^{\nu'}\log n$, where $\widetilde{C}_1, \widetilde{C}_2, \widetilde{C}_3$ depend on D_{11}, D_{12}, D_{13} respectively along with $\widetilde{c}_K$. By Assumption 1, there exists constants $\widetilde{C}_2'$ and ω_2, such that $\forall J \geq \widetilde{C}_2' m^{\theta'/\rho}$, we have $\zeta(J) \lesssim m^{-\theta'}$, and

$$\sup_{\boldsymbol{h}\in\mathcal{H}} \mathrm{E}_{\boldsymbol{h}}\left[\boldsymbol{\pi}(\boldsymbol{\xi})^{\intercal}\left(C_0 \wedge \log\left(\frac{\boldsymbol{\pi}(\boldsymbol{\xi})}{\widehat{\boldsymbol{\pi}}(\boldsymbol{\xi}_J)}\right)\right)\right] \leq \omega_2 m^{-\theta'}.$$

The range of optimal J is obtained as $J \in \left[\widetilde{C}_2' m^{\theta'/\rho}, \widetilde{C}_2 m^{\nu'}\right]$. The phase transition happens at $m^* = \lfloor (C_4^*/\widetilde{c}_2)\left(n^\theta/\log^3 n\right)^{1/\theta'} \rfloor$. Let $\omega_3 = C_4^*/\widetilde{c}_2$, the proof is complete. □

Appendix B: Additional Figures

Figure 3 depicts empirical KL risk versus theoretical rate obtained in Theorem 3.1 in Model 2D Gaussian setting. The computation is based on sampling rate $m = 400$, which indicates the data are fully observed. Since the Gaussian density functions (quadratic function and exponential function) are infinitely smooth in any composition layer, the decay rate parameter in Theorem 3.1 is $\theta = 1$. We compare the empirical KL risk with theoretical KL risk, i.e., $R_{\boldsymbol{h},C_0}(\widehat{\boldsymbol{\pi}})$ (with constant $\omega_1 = 12$). Figure 3 shows that the theoretical convergence rate is tight enough to bound the empirical risk.

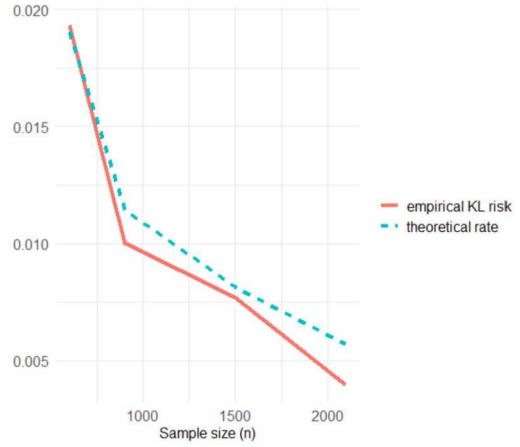

FIG 3. *Empirical KL risk versus theoretical rate obtained in Theorem 3.1 in Model 2D Gaussian*

Figures 4 and 5 illustrate the phase transition phenomenon of the proposed mfDNN method for all eight models in Sections 5.2 and 5.3. For three distinct sample sizes, it is observed that the misclassification rates remain constant when

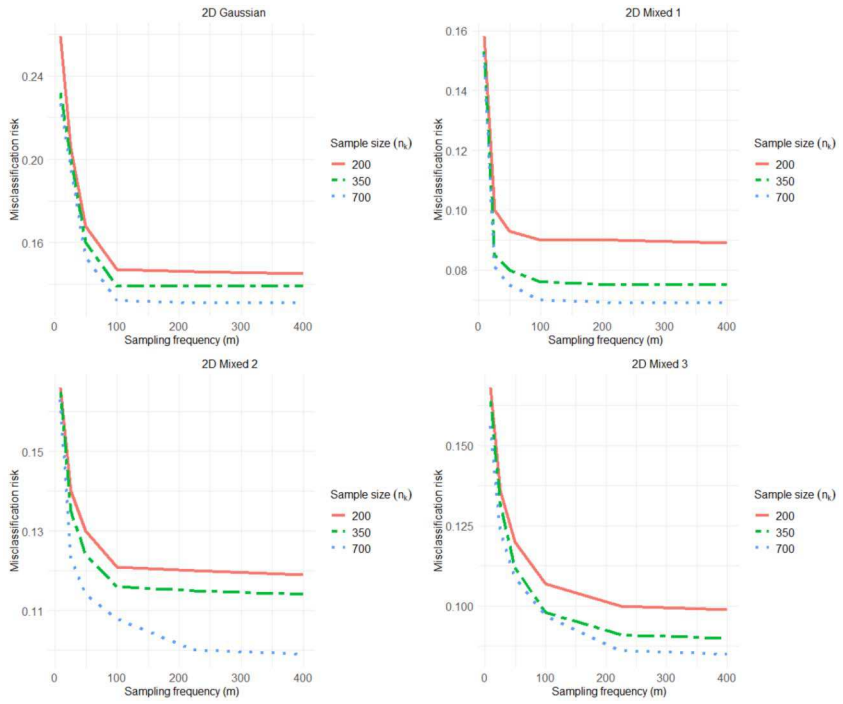

FIG 4. *Phase transition phenomenon for 2D simulation in three different sample sizes*

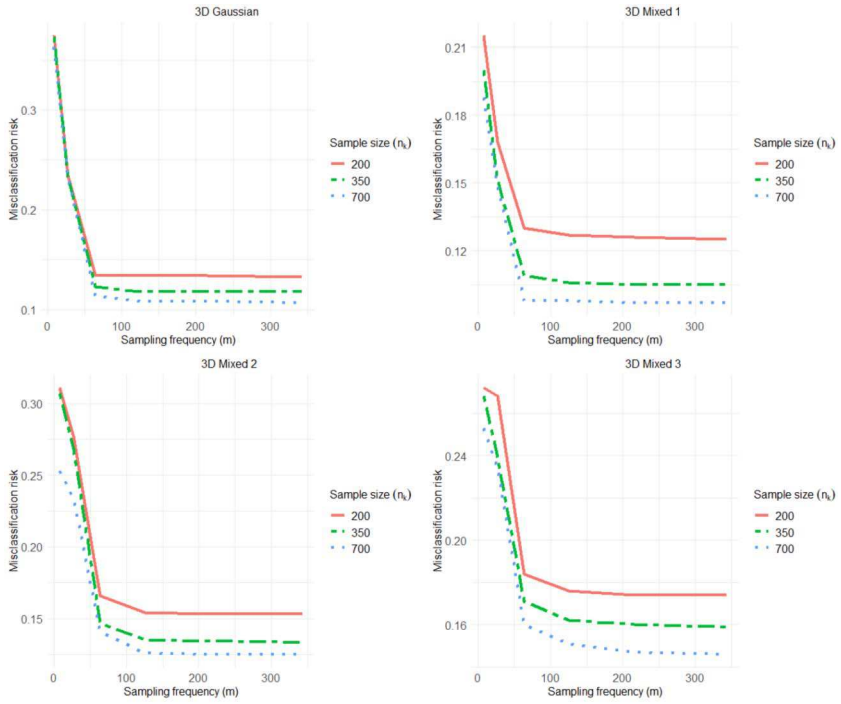

FIG 5. *Phase transition phenomenon for 3D simulation in three different sample sizes*

the sampling frequency exceeds 100 or 200. As the sample size increases, a larger sampling frequency is required for the risk to stabilize.

Acknowledgments

ADNI data used in preparation of this article were obtained from the Alzheimers Disease Neuroimaging Initiative (ADNI) database (adni.loni.usc.edu). As such, the investigators within the ADNI contributed to the design and implementation of ADNI and/or provided data but did not participate in analysis or writing of this report. A complete listing of ADNI investigators can be found at: http://adni.loni.usc.edu/wp-content/uploads/how_to_apply/ADNI_Acknowledgement_List.pdf.

Funding

Cao's research was also partially supported by the Simons Foundation under Grant #849413 and National Science Foundation under Grants CNS-2319342 and CNS-2319343.

References

[1] Jorge Adrover, Matias Salibian-Barrera, and Ruben Zamar. Globally robust inference for the location and simple linear regression models. *Journal of Statistical Planning and Inference*, 119(2):353–375, 2004. MR2019646

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

[3] J. R. Berrendero, A. Cuevas, and J. L. Torrecilla. On the use of reproducing kernel hilbert spaces in functional classification. *Journal of the American Statistical Association*, 113(523):1210–1218, 2018. MR3862351

[4] Thijs Bos and Johannes Schmidt-Hieber. Convergence rates of deep relu networks for multiclass classification. *Electronic Journal of Statistics*, 16:2724–2773, 2022. MR4406243

[5] T. Tony Cai and Linjun Zhang. A convex optimization approach to high-dimensional sparse quadratic discriminant analysis. *arXiv:1912.02872*, 2019. MR4298872

[6] T. Tony Cai and Linjun Zhang. High dimensional linear discriminant analysis: optimality, adaptive algorithm and missing data. *Journal of the Royal Statistical Society. Series B. Statistical Methodology*, 81(4):675–705, 2019. MR3997097

[7] F. Chamroukhi and H. Glotin. Mixture model-based functional discriminant analysis for curve classification. *Proceedings of the International Joint Conference on Neural Networks (IJCNN)*, pages 1–8, 2012.

[8] Xiongtao Dai, Hans-Georg Müller, and Fang Yao. Optimal Bayes classifiers for functional data and density ratios. *Biometrika*, 104(3):545–560, 2017. MR3694582

[9] A. Delaigle and P. Hall. Achieving near-perfect classification for functional data. *Journal of the Royal Statistical Society, Series B*, 74:267–286, 2012. MR2899863

[10] A. Delaigle, P. Hall, and N. Bathia. Componentwise classification and clustering of functional data. *Biometrika*, 99(2):299–313, 2012. MR2931255

[11] Aurore Delaigle and Peter Hall. Classification using censored functional data. *Journal of the American Statistical Association*, 108(504):1269–1283, 2013. MR3174707

[12] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby. An image is worth 16x16 words: Transformers for image recognition at scale. In *9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, May 3-7, 2021*. OpenReview.net, 2021.

[13] Pedro Galeano, Esdras Joseph, and Rosa E. Lillo. The Mahalanobis distance for functional data with applications to classification. *Technometrics*, 57(2):281–291, 2015. MR3369683

[14] Peter Hall and Hosseini-Nasab Mohammad. On properties of functional principal components analysis. *Journal of the Royal Statistical Society, Se-*

ries B, 68:109–126, 2006. MR2212577
[15] Travor J. Hastie and Robert J. Tibshirani. *Generalized Additive Models*. Chapman & Hall/CRC, 1990. MR1082147
[16] Tianyang Hu, Zuofeng Shang, and Guang Cheng. Sharp rate of convergence for deep neural network classifiers under the teacher-student setting. *arXiv: 2001.06892*, 2020.
[17] Yongdai Kim, Ilsang Ohn, and Dongha Kim. Fast convergence rates of deep neural networks for classification. *Neural Networks*, 138:179–197, 2021.
[18] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E. Hinton. Imagenet classification with deep convolutional neural networks. *Commun. ACM*, 60(6):84–90, may 2017.
[19] X. Leng and H.G. Müller. Classification using functional data analysis for temporal gene expression data. *Bioinformatics*, 22:68–76, 2006.
[20] Xiuqi Li and Subhashis Ghosal. Bayesian classification of multiclass functional data. *Electronic Journal of Statistics*, 12(2):4669–4696, 2018. MR3894067
[21] Yi Lin. Tensor product space anova models. *The Annals of Statistics*, 28:734 – 755, 2000. MR1792785
[22] Ruiqi Liu, Zuofeng Shang, and Guang Cheng. On deep instrumental variables estimate. *arXiv:2004.14954*, 2021.
[23] Qing Mai, Yi Yang, and Hui Zou. Multiclass sparse discriminant analysis. *Statistica Sinica*, 29:97–111, 2019. MR3889359
[24] Enno Mammen and Alexandre B. Tsybakov. Smooth discrimination analysis. *The Annals of Statistics*, 27:1808–1829, 1999. MR1765618
[25] Juhyun Park, Jeongyoun Ahn, and Yongho Jeon. Sparse functional linear discriminant analysis. *arXiv:2012.06488*, 2020. MR4374650
[26] J. O. Ramsay and B. W. Silverman. *Functional Data Analysis, Second Edition*. Springer Series in Statistics, New York, 2005. MR2168993
[27] Fabric Rossi, Delannay Nicolas, Brieuc Conan-Guez, and Michel Verleysen. Representation of functional data in neural networks. *Neurocomputing*, 64:183–210, 2005.
[28] J. Schmidt-Hieber. Nonparametric regression using deep neural networks with relu activation function. *The Annals of Statistics*, 48(4):1875–1897, 2020. MR4134774
[29] H. Shin. An extension of fisher's discriminant analysis for stochastic processes. *Journal of Multivariate Analysis*, 99:1191—-1216, 2008. MR2419344
[30] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. 2015.
[31] J. Song, W. Deng, H. Lee, and D. Kwon. Optimal classification for time-course gene expression data using functional data analysis. *Biometrika*, 103(1):147–159, 2016.
[32] C. Szegedy, Wei Liu, Yangqing Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich. Going deeper with convolutions. pages 1–9, 2015.
[33] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, and Z. Wojna. Rethinking the inception architecture for computer vision. In *2016 IEEE Conference*

on Computer Vision and Pattern Recognition (CVPR), pages 2818–2826, Los Alamitos, CA, USA, 2016. IEEE Computer Society.

[34] Alexandre B. Tsybakov. Optimal aggregation of classifiers in statistical learning. *The Annals of Statistics*, 32:135–166, 2004. MR2051002

[35] Shuoyang Wang, Guanqun Cao, and Zuofeng Shang. Estimation of the mean function of functional data via deep neural networks. *Stat*, e393, 2021. MR4319004

[36] Shuoyang Wang, Zuofeng Shang, and Guanqun Cao. Deep neural network classifier for multi-dimensional functional data. *arXiv:2205.08592*, 2022. MR4522373

[37] Shuoyang Wang, Zuofeng Shang, Guanqun Cao, and S. Jun Liu. Optimal classification for functional data. *Statistica Sinica*, 34, 2023.

[38] Daniela M. Witten and Robert Tibshirani. Penalized classification using Fisher's linear discriminant. *J. R. Stat. Soc. Ser. B Stat. Methodol.*, 73(5):753–772, 2011. MR2867457