Constructive Neural Network Learning

Shaobo Lin, Jinshan Zeng, and Xiaoqin Zhang

***Abstract*—In this paper, we aim at developing scalable neural network-type learning systems. Motivated by the idea of *constructive neural networks* in approximation theory, we focus on *constructing* rather than *training* feed-forward neural networks (FNNs) for learning, and propose a novel FNNs learning system called the constructive FNN (CFN). Theoretically, we prove that the proposed method not only overcomes the classical saturation problem for constructive FNN approximation, but also reaches the optimal learning rate when the regression function is smooth, while the state-of-the-art learning rates established for traditional FNNs are only near optimal (up to a logarithmic factor). A series of numerical simulations are provided to show the efficiency and feasibility of CFN.**

***Index Terms*—Constructive neural network learning, generalization error, neural networks, saturation.**

I. Introduction

TECHNOLOGICAL innovations bring a profound impact on the process of knowledge discovery. Collecting data of huge size becomes increasingly frequent in diverse areas of modern scientific research [44]. When the amount of data is huge, many traditional learning strategies, such as kernel methods [38] and neural networks [15] become infeasible due to their heavy computational burden. Designing effective and efficient approaches to extract useful information from massive data has been a recent focus in machine learning.

Scalable learning systems based on kernel methods have been designed for this purpose, such as the low-rank approximations of kernel matrices [3], incomplete Cholesky decomposition [13], early stopping of iterative regularization [24], and distributed learning equipped with a divide-and-conquer scheme [43]. However, most of the existing methods including the gradient-based method, such as the back propagation [37], second order optimization [40], greedy search [4], and the randomization methods like random vector functional-link networks [33], echo-state networks [20], extreme learning machines (ELMs) [17], fail in generating scalable neural network-type learning systems of high quality, since the gradient-based method usually suffers from the local minima and time-consuming problems, while the randomization methods sometimes brings an additional *uncertainty* and generalization capability degeneration phenomenon in some special learning tasks [23]. In this paper, we aim at introducing a novel scalable feed-forward neural network (FNN) learning system to tackle massive data.

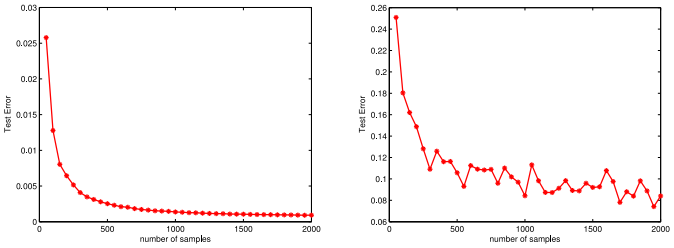


Fig. 1. Learning capability of [7]'s networks. (a) Noise-free data. (b) Noise data.

A. Motivations

FNN can be mathematically represented by

$$\sum_{i=1}^{n} c_i \sigma(a_i \cdot x + b_i), \quad x \in \mathbf{R}^d \tag{1}$$

where $\sigma : \mathbf{R} \to \mathbf{R}$ is an activation function, $a_i \in \mathbf{R}^d$, $b_i \in \mathbf{R}$, and $c_i \in \mathbf{R}$ are the inner weight, threshold and outer weight of FNN, respectively. From the approximation theory viewpoint, a_i, b_i, and c_i can be either determined via training [28] or constructed based on data directly [26]. Although various FNNs possessing optimal approximation property have been constructed in [1], [7]–[9], [22], [26], and [29], the idea of *constructing* FNN has not attracted lots of attention in the machine learning community. The main reason is that the constructed FNN possesses good learning performance for noise-free data only, which is usually impossible for real world applications.

Fig. 1 shows the learning performance of FNN constructed in [7]. The experimental setting of Fig. 1 can be found in Section IV. From approximation to learning, the tug of war between bias and variance [10] dictates that besides the approximation capability, a learning system of high quality should take the cost to reach the approximation accuracy into

Manuscript received August 9, 2017; accepted November 2, 2017. Date of publication March 20, 2018; date of current version December 14, 2018. The work of S. Lin was supported by the National Natural Science Foundation of China under Grant 61502342 and Grant 11771012. The work of J. Zeng was supported in part by the National Natural Science Foundation of China under Grant 61603162, Grant 61603163, Grant 61772246, and Grant 11501440, and in part by the Doctoral Start-Up Foundation of Jiangxi Normal University. This work of X. Zhang was supported in part by NSFC under Grant 61472285, in part by the Zhejiang Provincial Natural Science Foundation under Grant LR17F030001, in part by the Project of Science and Technology Plans of Zhejiang Province under Grant 2015C31168, and in part by the Project of Science and Technology Plans of Wenzhou City under Grant G20150017 and Grant ZG2017016. This paper was recommended by Associate Editor G.-B. Huang. *(Corresponding author: Jinshan Zeng.)*

S. Lin and X. Zhang are with the College of Mathematics and Information Science, Wenzhou University, Wenzhou 325035, China (e-mail: sblin1983@gmail.com; zhangxiaoqinnan@gmail.com).

J. Zeng is with the School of Computer and Information Engineering, Jiangxi Normal University, Nanchang 330022, China (e-mail: jsh.zeng@gmail.com).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TCYB.2017.2771463

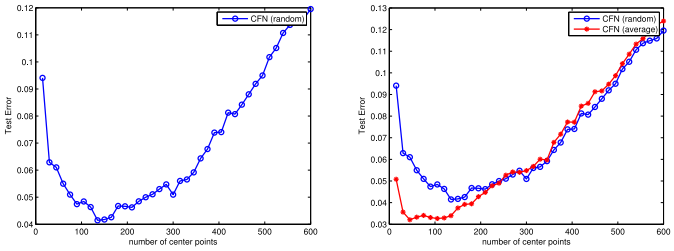


Fig. 2. Performance of refined construction of [7]'s networks. (a) Learning by part of samples. (b) Learning through average.

account. Using the constructed FNN for learning is doomed to be over-fitting, since the variance is neglected. A preferable way to reduce the variance is to cut down the number of neurons. Noting that the number of neurons of [7]'s FNN is the same as that of inputs of samples, we select part of samples whose inputs possess good geometrical distribution to construct FNN instead. It can be found in Figs. 1 and 2(a) that the accuracy of constructed FNN based on part of samples is 0.04, which is much better than 0.08 for the constructed FNN with the whole data. This implies that the over-fitting does exist for the constructed FNN and reducing the number of neurons is feasible to conquer it.

It is generally unreasonable to directly throw away a part of samples. Our idea stems from the local average argument in statistics [14], showing that noise-resistent estimators can be derived by averaging samples in small regions. Our construction starts with selecting a set of centers (not the samples) with good geometrical distribution and generating the Voronoi partitions [31] based on them. Then, an FNN is constructed according to the well-developed constructive technique in [7] by averaging outputs whose corresponding inputs are in the same small region. As the constructed FNN suffers from the saturation in the sense that the learning rate cannot be improved once the smoothness of the regression function goes beyond a specific value, we present a Landweber-type iterative method to overcome it at the last step. As Fig. 2(b) shows, the accuracy of the constructed FNN in this way is 0.03, exhibiting the necessity of the average and Landweber-type iteration.

B. Contributions

In this paper, we adopt the ideas from *constructive neural networks* in approximation theory [7], [22], [26], *local average* in statistics [6], [14], *Voronoi partition* in numerical analysis [31], [39], and *Landweber-type iteration for saturation problem* in inverse problems [12] to propose a new scalable learning scheme for FNN. In short, we aim at constructing an FNN, called the constructive FNN (CFN) for learning. Our main contributions are threefold.

First, different from the previous optimization-based training schemes [21], [42], our approach shows that parameters of FNN can be constructed directly based on samples, which essentially reduces the computational burden and provides a scalable FNN learning system to tackle massive data. The method is novel in terms of providing a step stone from FNN approximation to FNN learning by using tools in statistics, numerical analysis, and inverse problems.

The saturation problem, which was proposed in [7] as an open question, is a classical and long-standing problem of constructive FNN approximation. In fact, all of FNNs constructed in [1], [7]–[9], [22], and [26] suffer from the saturation problem. We highlight our second contribution to provide an efficient iterative method to overcome the saturation without affecting the variance very much.

Our last contribution is the feasibility verification of the proposed CFN. We verify both theoretical optimality and numerical efficiency of it. Theoretically, we prove that if the regression function is smooth, then CFN achieves the optimal learning rate in the sense that the upper and lower bounds are identical and equal to the best learning rates [14]. Experimentally, we run a series of numerical simulations to verify the theoretical assertions of CFN via comparing it with a benchmark algorithm for regression: regularized least squares (RLSs) with Gaussian kernel [11] and a popular neural networks learning scheme: ELMs [17].

C. Outline

The rest of this paper is organized as follows. In the next section, we present the details of CFN. In Section III, we study the theoretical behavior of CFN and compare it with some related work. In Section IV, some simulation results are reported to verify the theoretical assertions. In Section V, we prove the main results. In Section VI, we conclude this paper and present some discussions.

II. Construction of CFN

The construction of CFN is based on three factors: a Voronoi partition of the input space, a partition-based distance, and Landweber-type iterations.

A. Voronoi Partition

Let $\mathcal{X} \subset \mathbf{R}^d$ be a compact set and $\Xi_n := \{\xi_j\}_{j=1}^n$ be a set of points in $\mathcal{X}$. Define the mesh norm h_Ξ and separate radius q_Ξ [39] of Ξ_n by

$$h_\Xi := \max_{x\in\mathcal{X}} \min_j d(x, \xi_j), \quad q_\Xi := \frac{1}{2} \min_{1\le j\ne k\le n} d(\xi_j, \xi_k)$$

where $d(x, x')$ denotes the Euclidean distance between x and x'. If there exists a constant $\tau \ge 1$ satisfying $h_\Xi/q_\Xi \le \tau$, then Ξ_n is said to be quasi-uniform [39] with parameter τ. Throughout this paper, we assume Ξ_n is a quasi-uniform set with parameter 2.[1] That is

$$h_\Xi \le 2q_\Xi \le \frac{1}{n^d} \le 2h_\Xi \le 4q_\Xi. \tag{2}$$

Due to the definition of the mesh norm, we get

$$\mathcal{X} \subseteq \bigcup_{j=1}^n B_j(h_\Xi)$$

[1]The existence of a quasi-uniform set with parameter 2 was verified in [32]. Setting $\tau = 2$ is only for the sake of brevity. Our theory is feasible for arbitrary finite τ independent of n, however, the good numerical performance requires a small τ.

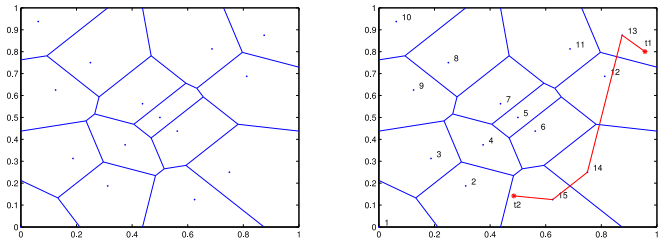


Fig. 3. (a) Voronoi partition and (b) partition-based distance.

where $B_j(r)$ denotes the Euclidean ball with radius r and center ξ_j.

A Voronoi partition $(A_j)_{j=1}^n$ of $\mathcal{X}$ with respect to Ξ_n is defined (e.g., [31]) by

$$A_j = \left\{x \in \mathcal{X} : j = \min\left\{\arg\min_{1\le k\le n} d(x,\xi_k)\right\}\right\}.$$

It is obvious that A_j contains all $x \in \mathcal{X}$ such that the center ξ_j is the nearest center to x. Moreover, if there exist j_1 and j_2 with $j_1 < j_2$, and

$$d\big(x,\xi_{j_1}\big) = d\big(x,\xi_{j_2}\big) < d(x,\xi_k)$$

for all $k \in \{1,2,\ldots,n\}/\{j_1,j_2\}$, then $x \in A_{j_1}$, since $j_1 < j_2$. It is obvious that $A_j \neq \varnothing$, $A_j \subset B_j(h_\Xi)$ for all $j \in \{1,\ldots,n\}$, $A_{j_1} \cap A_{j_2} = \varnothing$ $(j_1 \neq j_2)$, and $\mathcal{X} = \bigcup_{j=1}^n A_j$. Fig. 3(a) presents a specific example of the Voronoi partition in $[0,1]^2$.

It should be mentioned that our construction scheme is also feasible for other partition approaches satisfying $d(x,\xi_j) \le h_\Xi$ for each x in the small region with center ξ_j. Since the Voronoi partition is a classical partition technique for scattered data fitting and has been widely used in numerical analysis [39] and statistics [31]. We adopt the Voronoi partition in this paper.

B. Partition-Based Distance and the First Order CFN

To introduce the partition-based distance, we rearrange the points in Ξ_n in the following way: let ξ_1' be an arbitrary point in Ξ_n, and then recursively set ξ_{j+1}' satisfying $\|\xi_j' - \xi_{j+1}'\|_{l^2} \le 2h_\Xi$, $1 \le j \le n-1$. For the sake of brevity, we denote by Ξ_n its rearrangement $\{\xi_j'\}_{j=1}^n$. Let $(A_j)_{j=1}^n$ be a Voronoi partition with respect to Ξ_n. Then, for any $x \in \mathcal{X}$, there exists a unique $k_0 \le n$ such that $x \in A_{k_0}$. Given any two points $x, x' \in X$, we define the partition-based distance [22] between $x \in A_{k_0}$ and $x' \in A_{j_0}$ by

$$\overline{d}(x,x') = d(x,x'), \quad \text{if } k_0 = j_0$$

and otherwise

$$\overline{d}(x,x') = \sum_{j=\min\{j_0,k_0\}}^{\max\{j_0,k_0\}-1} d\big(\xi_j,\xi_{j+1}\big) + d\big(\xi_{k_0},x\big) + d\big(\xi_{j_0},x'\big).$$

In the above definition, $\{\xi_j\}_{j=1}^n$ are used as freight stations in computing the distance. The purpose of introducing the partition-based distance $\overline{d}(\cdot,\cdot)$ is to guarantee a strict order of points in different A_j. In this way, we can extend the construction technique in [7] from $d = 1$ to $d > 1$, and also preserve the good approximation performance. Fig. 3(b) presents an example for the partition-based distance between t_1 and t_2 in $[0,1]^2$.

Let $S_m = (x_i,y_i)_{i=1}^m$ be the set of samples and $I_m = (x_i)_{i=1}^m$ be the set of inputs. Let $T_j = A_j \cap I_m$ be the set of inputs locating in A_j (T_j may be an empty set). Denote by $T_j = \{x_1^j,\ldots,x_{|T_j|}^j\}$ and its corresponding outputs as $\{y_1^j,\ldots,y_{|T_j|}^j\}$. Throughout this paper, $|A|$ denotes the cardinality of the set A. The first order CFN is then given by

$$N_{n,w}^1(x) := \frac{\sum_{i=1}^{|T_1|} y_i^1}{|T_1|} + \sum_{j=1}^{n-1}\left(\frac{\sum_{i=1}^{|T_{j+1}|} y_i^{j+1}}{|T_{j+1}|} - \frac{\sum_{i=1}^{|T_j|} y_i^j}{|T_j|}\right) \times \sigma\big(w\big(\overline{d}(\xi_1,x) - \overline{d}\big(\xi_1,\xi_j\big)\big)\big) \tag{3}$$

where $(0/0) = 0$, σ is a sigmoidal function and $w \in \mathbf{R}_+$ is a localized parameter to embody the sigmoidal property of σ. Its value depends on σ, m, and d. We refer the readers to Section III-B for detailed information of w.

C. Iterative Scheme for the rth Order CFN

The constructed neural network in (3) suffers from the saturation. Indeed, it can be found in [7] and [22] that the approximation rate of (3) cannot exceed $n^{-2/d}$, no matter how smooth the regression function is. Such a saturation phenomenon also exists for the Tikhonov regularization algorithms [12] in inverse problems and Nadaraya–Watson kernel estimate in statistics [14]. It was shown in [12] and [34] that the saturation can be avoided by iteratively learning the residual. Borrowing the ideas from [12] and [34], we introduce an iterative scheme to avoid the saturation of CFN. For $k = 1,2,\ldots,r-1$.

1) Compute residuals $e_{i,k} = y_i - N_{n,w}^k(x_i)$, $i = 1,2,\ldots,m$.
2) Fit $U_{n,w}^k$ to the data $\{(x_i,e_{i,k})\}_{i=1}^m$, that is

$$U_{n,w}^k(x) := \frac{\sum_{i=1}^{|T_1|} e_{i,k}^1}{|T_1|} + \sum_{j=1}^{n-1}\left(\frac{\sum_{i=1}^{|T_{j+1}|} e_{i,k}^{j+1}}{|T_{j+1}|} - \frac{\sum_{i=1}^{|T_j|} e_{i,k}^j}{|T_j|}\right) \times \sigma\big(w\big(\overline{d}(\xi_1,x) - \overline{d}\big(\xi_1,\xi_j\big)\big)\big). \tag{4}$$

3) Update $N_{n,w}^{k+1}(x) = N_{n,w}^k(x) + U_{n,w}^k(x)$. We obtain the rth order CFN $N_{n,w}^r$ after repeating the above procedures $r-1$ times.

D. Summary of CFN

The proposed CFN can be formulated in Algorithm 1. In step 2, we focus on selecting n quasi-uniform points in $\mathcal{X}$. Theoretically, the distribution of Ξ_n significantly affects the approximation capability of CFN, as well as the learning performance. Therefore, Ξ_n is arranged the more uniformly, the better. In practice, we use some well developed low-discrepancy sequences, such as the Sobol sequence [5] and the Halton sequence [2] to generate Ξ_n. As there are some existing MATLAB codes (such as the *Sobolset* comment for the Sobol sequence), we use the Sobol sequence in this paper, which requires $\mathcal{O}(n\log n)$ floating computations to generate n points. In step 3, to implement the rearrangement, we use the

Algorithm 1 CFN

Step 1 (Initialization): Given data $S_m = \{(X_i, Y_i): i = 1, \ldots, m\}$, the iteration number r, the number of neurons n, the activation function σ and the parameter w.

Step 2 (Sampling): Select n quasi-uniform points $\Xi_n := \{\xi_j\}_{j=1}^n$ in $\mathcal{X}$.

Step 3 (Rearranging): Rearrange n quasi-uniform points in Ξ_n such that $d(\xi_j, \xi_{j+1}) \le \frac{2}{n^{1/d}}$.

Step 4 (Voronoi partition): Present a Voronoi partition with respect to Ξ_n and to get a set of subsets $(A_j)_{j=1}^n$ satisfying $A_j \cap A_k = \varnothing$, and $\cup_{j=1}^n A_j = \mathcal{X}$.

Step 5 (Constructing the first order CFN): Define the first order CFN by (3).

Step 6 (Initializing for iteration): For $k \in \mathbf{N}$, define $e_{i,k} = y_i - N_n^k(x_i)$ and generate a new set of sample $(x_i, e_{i,k})_{i=1}^m$.

Step 7 (Iterative updating function): Define the updating function $U_{n,w}^k$ by (4).

Step 8 (Updating): Define the $(k+1)$th order CFN by $N_{n,w}^{k+1}(x) = N_{n,w}^k(x) + U_{n,w}^k(x)$.

Step 9 (Repeating and stopping): Increase k by one and repeat Step 6-Step 8 if $k < r$, otherwise, the algorithm stops.

greedy scheme via searching the nearest point one by one.[2] It requires $\mathcal{O}(n^2)$ floating computations. In step 4, it requires $\mathcal{O}(n^2)$ floating computations to generate a Voronoi partition. In step 5, the partition-based distance requires $\mathcal{O}(n)$ floating computations. Then it can be deduced from (3) that there are $\mathcal{O}(mn)$ floating computations in this step. From steps 6 to 9, there is an iterative process and it requires $\mathcal{O}(rmn + rn^2)$ floating computations. Summarily, the total floating computations of Algorithm 1 are of the order $\mathcal{O}(rmn + rn^2)$.

III. Theoretical Behavior

In this section, we analyze the theoretical behavior of CFN in the framework of statistical learning theory [10] and compare it with some related work.

A. Assumptions and Main Result

Let $S_m = (x_i, y_i)_{i=1}^m \subset \mathcal{Z} := \mathcal{X} \times \mathcal{Y}$ be a set of samples drawn independently according to an unknown joint distribution ρ, which satisfies $\rho(x, y) = \rho_X(x)\rho(y|x)$. Here, ρ_X is the marginal distribution and $\rho(y|x)$ is the conditional distribution. Without loss of generality, we assume $|y_i| \le M$ almost surely for some positive number M. The performance of an estimate, f, is measured by the generalization error $\mathcal{E}(f) := \int_{\mathcal{Z}} (f(x) - y)^2 d\rho$, which is minimized by the regression function defined by $f_\rho(x) := \int_{\mathcal{Y}} y d\rho(y|x)$. Let $L^2_{\rho_X}$ be the Hilbert space of ρ_X square integrable functions on $\mathcal{X}$, with norm $\|\cdot\|_\rho$. According to [10], there holds

$$\mathcal{E}(f) - \mathcal{E}(f_\rho) = \|f - f_\rho\|_\rho^2. \tag{5}$$

To state the main result, some assumptions on the regression function and activation function should be imposed. Let $s = u + \beta$ for some $u \in \mathbf{N}$ and $0 < \beta \le 1$. A function $f: \mathcal{X} \to \mathbf{R}$ is said to be s-smooth if for every $\alpha = (\alpha_1, \ldots, \alpha_d)$, $\alpha_l \in \mathbf{N}$, $\sum_{l=1}^d \alpha_l = u$ and for all $x, x' \in \mathcal{X}$, the partial derivative $(\partial^u f / [\partial x_1^{\alpha_1} \cdots \partial x_d^{\alpha_d}])$ exists and satisfies

$$\left| \frac{\partial^u f}{\partial x_1^{\alpha_1} \cdots \partial x_d^{\alpha_d}}(x) - \frac{\partial^u f}{\partial x_1^{\alpha_1} \cdots \partial x_d^{\alpha_d}}(x') \right| \le C\big[d(x, x')\big]^\beta$$

for some universal positive constant C. Let $\mathcal{F}^s$ be the set of all s-smooth functions.

Assumption 1: $f_\rho \in \mathcal{F}^s$ for some $s > 0$.

Assumption 1 describes the smoothness of f_ρ, and is a regular assumption in learning theory. It has been adopted in [23], [25], [30], [35], [36], and [41] to quantify the learning performance of various neural network-type learning systems. Let $\mathcal{M}(\Theta)$ be the class of all Borel measures ρ on $\mathcal{Z}$ such that $f_\rho \in \Theta$. Let $\mathbf{G}_m$ be the set of all estimators derived from the sample S_m. Define

$$e_m(\Theta) := \inf_{f_S \in \mathbf{G}_m} \sup_{\rho \in \mathcal{M}(\Theta)} \mathbf{E}\Big\{ \|f_\rho - f_S\|_\rho^2 \Big\}.$$

Obviously, $e_m(\Theta)$ quantitatively measures the quality of f_S. It can be found in [14, Th. 3.2] that

$$e_m\big(\mathcal{F}^s\big) \ge C_0 m^{-\frac{2s}{2s+d}}, \ \ m = 1, 2, \ldots \tag{6}$$

where C_0 is a constant depending only on C, M, s, and d. If an S_m-based estimator f_S reaches the bound

$$\sup_{\rho \in \mathcal{M}(\mathcal{F}^s)} \mathbf{E}\Big\{ \|f_\rho - f_S\|_\rho^2 \Big\} \le C_1 m^{-\frac{2s}{2s+d}}$$

where C_1 is a constant independent of m, then f_m is rate-optimal for $\mathcal{F}^s$.

Let $\sigma: \mathbf{R} \to \mathbf{R}$ be a sigmoidal function, that is

$$\lim_{t \to +\infty} \sigma(t) = 1, \quad \lim_{t \to -\infty} \sigma(t) = 0.$$

Then, there exists a $K > 0$ such that

$$|\sigma(t) - 1| < n^{-(s+d)/d} \quad \text{if} \quad t \ge K \tag{7}$$

and

$$|\sigma(t)| < n^{-(s+d)/d} \quad \text{if} \quad t \le -K. \tag{8}$$

For a positive number a, we denote by $[a]$, $\lceil a \rceil$, and $\lfloor a \rfloor$ the integer part of a, the smallest integer not smaller than a and the largest integer smaller than a.

Assumption 2: The following assumption presents some restrictions on the activation functions.

1) σ is a bounded sigmoidal function.
2) For $s > 0$, σ is at least $\lfloor s \rfloor$ differentiable.

Conditions 1) and 2) are mild. Indeed, there are numerous examples satisfying 1) and 2) for arbitrary s, such as the logistic function

$$\sigma(t) = \frac{1}{1 + e^{-t}}$$

hyperbolic tangent sigmoidal function

$$\sigma(t) = \frac{1}{2}(\tanh(t) + 1)$$

[2] This greedy scheme sometimes generates a rearrangement that $d(\xi_j, \xi_{j+1}) \le c_0 n^{-1/d}$ with $c_0 > 2$. We highlight that the constant 2 in the algorithm is only for the sake of theoretical convenience.

with $\tanh(t) = (e^{2t}-1)/(e^{2t}+1)$, arctan sigmoidal function

$$\sigma(t) = \frac{1}{\pi}\arctan(t) + \frac{1}{2}$$

and Gompertz function

$$\sigma(t) = e^{-ae^{-bt}}$$

with $a, b > 0$.

The following Theorem 1 is the main result of this paper, which illustrates the rate-optimality of CFN.

Theorem 1: Let $s > 0$, $r \in \mathbf{N}$ with $r \geq s$, and $N^r_{n,w}$ be the estimator defined by Algorithm 1. Under Assumptions 1 and 2, if $n \sim m^{(d/[2s+d])}$ and $w \geq 4Kn^{1/d}$ with K satisfying (7) and (8), then there exists a constant C_2 independent of m or n such that

$$C_0 m^{-\frac{2s}{2s+d}} \leq \sup_{f_\rho \in \mathcal{F}^s} \mathbf{E}\left\{\left\|N^r_{n,w} - f_\rho\right\|^2_\rho\right\} \leq C_2 m^{-\frac{2s}{2s+d}}. \quad (9)$$

B. Remarks and Comparisons

There are three parameters in CFN: 1) the number of centers n; 2) the parameter w; and 3) the number of iterations r. Theorem 1 shows that if some *a priori* information of the regression function is known, i.e., $f_\rho \in \mathcal{F}^s$, then all these parameters can be determined. In particular, we can set $w \geq 4Kn^{1/d}$, $r = \lceil s \rceil$, and $n = [m^{d/(2s+d)}]$. K in (7) and (8) depends on s, σ, and n. So K can be specified when σ and n are given. For example, if the logistic function is utilized, then $w \geq 4([(s+d)n^{1/d}\log n]/d)$ is preferable.

However, in real world applications, it is difficult to check the smoothness of the regression function. We provide some suggestions about the parameter selection. Since Theorem 1 holds for all $w \geq 4Kn^{1/d}$, w can be selected to be sufficiently large. The iterative scheme is imposed only for overcoming the saturation problem and regression functions in real world applications are usually not very smooth. We find that a few iterations (say, $r \leq 5$) are commonly sufficient. Thus, we set $r \in [1, 5]$ in real world applications and use the cross-validation [14] to determine it. The key parameter is n, which is also crucial for almost all neural network-type learning systems [21], [25], [30], since n reflects the tradeoff between bias and variance. We also use the cross-validation to determine it. Compared with the optimization-based neural network-type learning systems, the total computational complexity of CFN (with cross-validation) is much lower, since the computational complexity of CFN is of order $\mathcal{O}(mn)$ for fixed r and that of optimization-based methods is at least $\mathcal{O}(mn^2)$.

The saturation problem of FNN approximation was proposed as an open question by Chen [7]. It can be found in [1], [2], [8], [9], and [22] that the saturation problem for constructive neural network approximation has not been settled, since all these results were established on the assumption that the regression function belongs to $\mathcal{F}^s$ with $0 < s \leq 1$. However, for optimization-based FNN, the saturation problem did not exist as shown in the prominent work [28], [29]. In this paper, we succeed in settling the saturation problem by using the proposed iterative scheme. Theorem 1 states that if $f_\rho \in \mathcal{F}^s$, then CFN is rate-optimal for all $s > 0$, since its learning rate can reach the lower bound in (6).

Finally, we compare Theorem 1 with two related theoretical results about learning performances of optimization-based FNN learning [30] and ELM [25]. Denote

$$\mathcal{N}^M_n := \left\{f = \sum_{k=1}^n c_k \phi(a_k \cdot x + b_k) : \|f\|_\infty \leq M\right\}$$

where $a_k \in \mathbf{R}^d$, b_k, and $c_k \in \mathbf{R}$. Define

$$f_{1,S} = \arg\min_{f \in \mathcal{N}^M_n} \frac{1}{m}\sum_{i=1}^m (f(x_i) - y_i)^2. \quad (10)$$

Maiorov [30] proved that for some activation function ϕ, if $f_\rho \in \mathcal{F}^s$ and $n \sim m^{d/(s+d)}$, then there holds

$$C_0 m^{-\frac{2s}{2s+d}} \leq \sup_{f_\rho \in \mathcal{F}^s} \mathbf{E}\left\{\left\|f_{1,S} - f_\rho\right\|^2_\rho\right\} \leq C_3 m^{-\frac{2s}{2s+d}} \log m \quad (11)$$

where C_3 is a constant independent of m or n. It should be noted from (10) that $f_{1,S}$ is obtained by solving a nonlinear least squares problem, which generally requires higher computational complexity than CFN. Furthermore, comparing (9) with (11), we find that CFN is rate-optimal while (10) is near rate-optimal (up to a logarithmic factor).

Denote

$$\mathcal{N}^R_n := \left\{f = \sum_{k=1}^n c_k \phi\left(a^*_k \cdot x + b^*_k\right) : c_k \in \mathbf{R}\right\}$$

where $a^*_k \in \mathbf{R}^d$ and $b^*_k \in \mathbf{R}$ are randomly selected according to a distribution μ. The ELM estimator is defined by

$$f_{2,S} = \arg\min_{f \in \mathcal{N}^R_n} \frac{1}{m}\sum_{i=1}^m (f(x_i) - y_i)^2. \quad (12)$$

It is easy to see that the optimization problem (12) is a linear problem and can be solved by using the pseudo-inverse technique [17]. The theoretical performance of ELM was justified in [25], asserting that for some activation function, if $f_\rho \in \mathcal{F}^s$ and $n \sim m^{d/(s+d)}$, then there holds

$$\mathbf{E}_\mu \mathbf{E}\left\{\left\|f_{2,S} - f_\rho\right\|^2_\rho\right\} \leq C_4 m^{-\frac{2s}{2s+d}} \log m \quad (13)$$

where C_4 is a constant independent of m or n. It follows from (6) and (13) that (12) is near rate-optimal in the sense of expectation, since there is an additional $\mathbf{E}_\mu$ in (13) due to the randomness of $f_{2,S}$. Compared with (13), our result in (9) shows that CFN can remove the logarithmic factor and avoid the randomness of ELM.

IV. Simulations

In this section, we verify our theoretical assertions via a series of numerical simulations. When $d = 1$, the rearranging step (step 3 in Algorithm 1) is trivial. Thus, we divide the simulations into two cases: $d = 1$ and $d > 1$. All numerical studies are implemented by using MATLAB R2014a on a Windows personal computer with Core i7-3770 3.40 GHz CPUs and RAM 4.00 GB. Throughout the simulations, the logistic activation function is used and the statistics are averaged based on 50 independent trails.

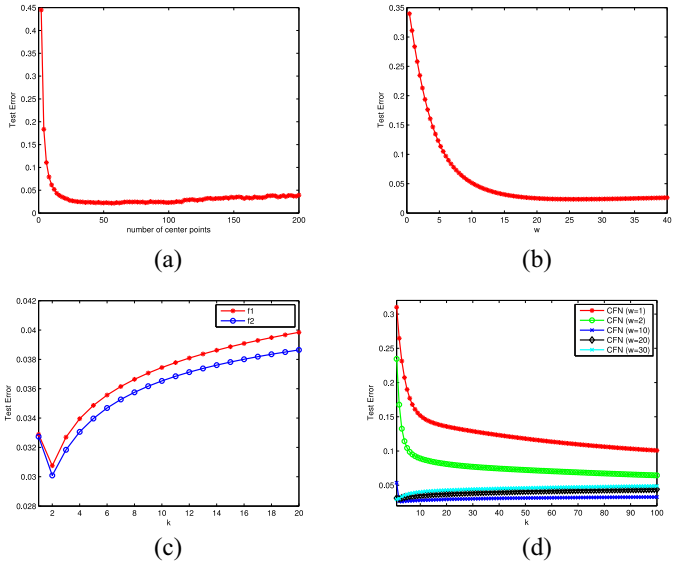


Fig. 4. Roles of parameters. (a) Role of n for f_1. (b) Role of w for f_1. (c) Role of r for f_1 and f_2. (d) Small w and large r for f_1.

A. Simulations for $d = 1$

In the simulations, we generate training data from the following model:

$$Y = f(X) + \varepsilon \tag{14}$$

where ε is the Gaussian noise with variance 0.1 and independent of X. The centers are n equally spaced points in $[-1, 1]$. The test data are sampled from $Y = f(X)$. Let

$$f_1(x) = 1 + \frac{80}{3}x^2 - 40x^3 + 15x^4 + \frac{8}{3}x^5 + 20x^2 \log(|x|)$$

and

$$f_2(x) = (1-x)_+^5\left(8x^2 + 5x + 1\right)$$

where $a_+ = a$ when $a \geq 0$ and $a_+ = 0$ when $a < 0$. It is easy to check that $f_1 \in \mathcal{F}^1$ but $f_1 \notin \mathcal{F}^2$ and $f_2 \in \mathcal{F}^4$ but $f_2 \notin \mathcal{F}^5$.

In the first simulation, we numerically study the roles of three parameters n, w, and r. Since we are interested in the role of a specified parameter, the other two parameters are selected to be optimal according to the test data directly. In this simulation, the number of training and test samples are 1024 and 1000. The results of the simulation are shown in Fig. 4. Specifically, Fig. 4(a) describes the relation between the test error and the number of centers. From Fig. 4(a), it follows that there exists an optimal n ranged in $[10, 20]$ minimizing the test error, which coincides with the theoretical assertions $n \sim m^{d/(2s+d)} = 1024^{1/3}$. Fig. 4(b) demonstrates the relation between the test error and w. It can be found in Fig. 4(b) that after a crucial value of w, the test error does not vary with respect to w, which coincides with the assertion in Theorem 1. Fig. 4(c) presents the relation between the test error and the number of iterations. It is shown in Fig. 4(c) that such a scheme is feasible and can improve the learning performance. Furthermore, there is an optimal r in $[1, 5]$ minimizing the test error (the average value of r is 2.2 via average for 50 trails). This also verifies our assertions in Section III.

TABLE I
COMPARISONS OF LEARNING PERFORMANCE FOR $d = 1$

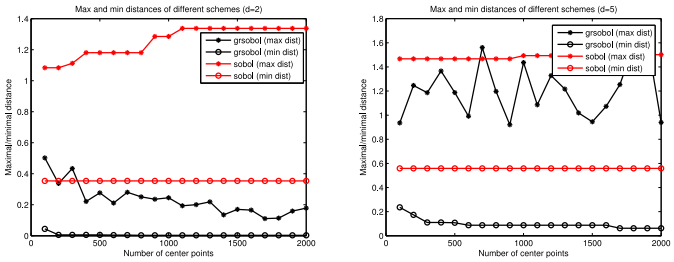
	TrainRMSE	TestRMSE	TrainingTime	TestTime
		f_1		
CFN	0.3305(0.0070)	**0.0182(0.0035)**	0.51[0.01]	0.91
RLS	0.3303(0.0070)	0.0203(0.0064)	9.95[0.19]	8.22
ELM	**0.3302(0.0070)**	0.0223(0.0072)	**0.42[0.01]**	**0.31**
		f_2		
CFN	0.3304(0.0070)	**0.0199(0.0037)**	0.48[0.01]	0.88
RLS	0.3303(0.0070)	0.0204(0.0067)	10.6[0.21]	9.14
ELM	**0.3302(0.0070)**	0.0225(0.0072)	**0.42[0.01]**	**0.33**

Fig. 4(d) presents another application of the iterative scheme in CFN. It is shown in Fig. 4(d) that once the parameter w is selected to be small, then we can use large r to reduce the test error.

In the second simulation, we compare CFN with two widely used learning schemes. One is the RLS with Gaussian kernel [11], which is recognized as the benchmark for regression. The other is the ELM [17], which is one of the most popular neural network-type learning systems. The corresponding parameters including the width of the Gaussian kernel and the regularization parameters in RLS, the number of hidden neurons in ELM, and the number of centers, the number of iterations in CFN are selected via fivefold cross validation. We record the mean test rooted square mean error (RMSE) and the mean training RMSE as TestRMSE and TrainRMSE, respectively. We also record their standard deviations in the parentheses. Furthermore, we record the average total training time as TrainTime. We also record the time of training for fixed parameters in the bracket. Since the training time of ELM is different for different number of neurons, we record in the bracket the training time for ELM with optimal parameter. We finally record the mean test time as TestTime. The results are reported in Table I. It is shown in Table I that the learning performance of CFN is at least not worse than RLS and ELM. In fact, the test error of CFN is comparable with RLS but better than ELM and the training price (training time and memory requirement) of CFN is comparable with ELM but lower than RLS. This verifies the feasibility of CFN and coincides with our theoretical assertions.

B. Simulations for $d > 1$

When $d > 1$, the strategies of selecting centers and rearrangement operator are required. In this part, we draw the Sobol sequence to build up the set of centers and then use a greedy strategy to rearrange them. Specifically, we start with an arbitrary point in the Sobol sequence, then select the next point to be the nearest point of the current point, and then repeat this procedure until all the points are rearranged. We show in the following Fig. 5 that such a greedy scheme essentially cuts down the maximum distance between arbitrary two adjacent points (max dist). For comparison, we also exhibit the change of the minimum distance (min dist).



(a) (b)

Fig. 5. Distribution of centers. (a) $d = 2$. (b) $d = 5$.

We demonstrate the feasibility of CFN by comparing it with RLS and ELM for various regression functions. Let

$$f_3(x) = (1 - \|x\|_2)_+^6 \left(35/3\|x\|_2^2 + 6\|x\|_2 + 1\right)$$

$$f_4(x) = (1 - \|x\|_2)_+^7 \left(35\|x\|_2^6 + 245\|x\|_2^5 + 720\|x\|_2^4 + 1120\|x\|_2^3 + 928\|x\|_2^2 + 336\|x\|_2 + 48\right)$$

and

$$f_5(x) = 1 + \frac{80}{3}\|x\|_2^5 - 40\|x\|_2^3 + 15\|x\|_2^4 + \frac{8}{3}\|x\|_2^5 + 20\|x\|_2^2 \log(\|x\|_2)$$

where $\|x\|_2$ denotes the Euclidean norm of the vector x. It is easy to check that for $d = 2, 3$, $f_3 \in \mathcal{F}^4$ but $f_3 \notin \mathcal{F}^5$, and for $d = 5$, $f_4 \in \mathcal{F}^4$ but $f_4 \notin \mathcal{F}^5$, and for $d = 2, 3, 5$, $f_5 \in \mathcal{F}^1$ but $f_5 \notin \mathcal{F}^2$. The simulation results are reported in Table II. It can be found in Table II that, similar to the 1-D simulations, the learning performance of CFN is a bit better than RLS (in terms of training time), and slightly better than ELM (in terms of test error). This also verifies our theoretical assertions.

C. Challenge for Massive Data

Since our main motivation for introducing CFN is to tackle massive data, we pursue the advantage of CFN for massive data sets. For this purpose, we set the number of training samples to be 50 000, 100 000, 150 000 and the number of test samples to be 100 000, 200 000, 300 000. As the memory requirements and training time of RLS are huge, we only compare CFN with ELM in this simulation. The simulation results are reported in Table III. It is shown that when applied to the massive data, the performance of CFN is at least not worse than that of ELM. In particular, CFN possesses a slight smaller test errors and dominates in the training time.

Based on the numerical results, we find that CFN performs slightly better than ELM in terms of generalization error but requires more training time for data set with middle size. When $r \leq 5$ and $n \leq m$, the computational complexities of CFN and ELM are $\mathcal{O}(rmn)$ and $\mathcal{O}(mn^2)$, respectively. The reason for the time-consuming of CFN over ELM is that for data set of middle size, the selected n in ELM is a bit smaller than CFN and both of them are very small (reflected by the testing time). Under this circumstance, the computational complexity of ELM $\mathcal{O}(mn^2)$ is smaller than that of CFN (rmn). However, for massive data set, CFN performs at least not worse than ELM in generalization but requires less time. The reason is that for massive data, a large number of neurons in ELM and CFN are required. Large n inevitably leads to much more training time for ELM. Thus, besides the perfect theoretical behaviors, CFN is also better than ELM for this massive data set. It should be mentioned that there are several variants of ELM [16], [18], [19]. We compare the proposed CFN here only with the original learning algorithm proposed in [17] to facilitate verifying our theoretical assertions. We believe that

TABLE II
COMPARISONS OF LEARNING PERFORMANCE FOR $d > 1$

	TrainRMSE	TestRMSE	TrainingTime	TestTime
f_3, $d = 2$				
CFN	0.3303(0.010)	**0.0341 (0.0094)**	4.67 [0.067]	1.05
RLS	**0.3259(0.0102)**	0.0412(0.0047)	10.6[0.217]	9.43
ELM	0.3291(0.010)	0.0391 (0.0055)	**0.42 [0.003]**	**0.34**
f_3, $d = 3$				
CFN	0.3312 (0.008)	**0.0495 (0.0057)**	4.77[0.09]	1.05
RLS	**0.3204(0.007)**	0.0593(0.0065)	11.0 [0.27]	9.65
ELM	0.3253(0.007)	0.0542 (0.0066)	**0.61 [0.01]**	**0.31**
f_4, $d = 5$				
CFN	0.3302(0.009)	0.0278(0.004)	4.96 [0.09]	1.12
RLS	**0.3294(0.009)**	**0.0272 (0.006)**	10.8 [0.20]	9.79
ELM	0.3303(0.009)	0.0311(0.004)	**0.52[0.01]**	**0.35**
f_5, $d = 2$				
CFN	0.3318(0.010)	**0.0286(0.007)**	4.73 [0.08]	0.81
RLS	**0.3282(0.011)**	0.0404 (0.006)	10.8 [0.22]	9.70
ELM	0.3302 (0.011)	0.0359 (0.007)	**0.49 [0.01]**	**0.45**
f_5, $d = 3$				
CFN	0.3361(0.009)	**0.0465 (0.009)**	4.86[0.08]	0.82
RLS	0.3260(0.007)	0.0573 (0.006)	10.6 [0.21]	9.63
ELM	**0.3315(0.008)**	0.0494 (0.004)	**0.56 [0.01]**	**0.38**
f_5, $d = 5$				
CFN	0.3310(0.007)	**0.0299 (0.006)**	4.91 [0.08]	0.80
RLS	**0.3270(0.007)**	0.0400 (0.003)	10.7 [0.20]	9.77
ELM	0.3329(0.007)	0.0438(0.004)	**0.51 [0.01]**	**0.47**

TABLE III
COMPARISONS OF LEARNING PERFORMANCE FOR MASSIVE DATA

	TrainRMSE	TestRMSE	TrainTime	TestTime
f_1, $d = 1, m = 5 \times 10^4$				
CFN	**0.3298(0.001)**	**0.0033(5.0e-004)**	**13.6[0.34]**	21.8
ELM	**0.3298(0.001)**	0.0035(7.0e-004)	78.9[0.16]	**19.2**
f_1, $d = 1, m = 10^5$				
CFN	**0.3299(0.001)**	0.0029(7.3e-004)	**28.3[0.71]**	45.3
ELM	**0.3299(0.001)**	**0.0027(5.3e-004)**	167[3.90]	**38.3**
f_1, $d = 1, m = 1.5 \times 10^5$				
CFN	**0.3303(0.001)**	**0.0021(4.8e-004)**	**42.5[1.27]**	67.8
ELM	**0.3303(0.001)**	0.0022(6.6e-004)	252[7.65]	**55.8**
f_5, $d = 5, m = 5 \times 10^4$				
CFN	0.3309(0.001)	**0.0161(0.005)**	**95.8[4.51]**	44.6
ELM	**0.3304(0.001)**	0.019(0.001)	134[2.70]	**36.5**
f_5, $d = 5, m = 10^5$				
CFN	0.3302(0.001)	**0.0131(5.0e-04)**	**191[8.93]**	89.1
ELM	**0.3299(0.001)**	0.0159(7.0e-04)	260[7.05]	**71.6**
f_5, $d = 5, m = 1.5 \times 10^5$				
CFN	0.3307(0.001)	**0.0122(3.9e-04)**	**291[13.2]**	134
ELM	**0.3305(0.001)**	0.0138(6.6e-04)	399[9.81]	**108**

for some delicate variants the learning performance of ELM can be essentially improved, but it is out of our scope.

V. Proof of Theorem 1

The lower bound of (9) can be derived directly by (6). Therefore, it suffices to prove the upper bound of (9). For $1 \le k \le r$, define

$$\widetilde{N^k_{n,w}}(x) := \mathbf{E}\Big[N^k_{n,w}(x)|X_1, \ldots, X_m|\Big].$$

Then

$$\mathbf{E}\Big[\|N^r_{n,w} - f_\rho\|^2_\rho\Big] = \mathbf{E}\Big[\big\|N^r_{n,w} - \widetilde{N^r_{n,w}}\big\|^2_\rho\Big] + \mathbf{E}\Big[\|\widetilde{N^r_{n,w}} - f_\rho\|^2_\rho\Big] \tag{15}$$

since $\mathbf{E}\langle N^r_{n,w} - \widetilde{N^r_{n,w}}, \widetilde{N^r_{n,w}} - f_\rho\rangle_\rho = 0$. We call the first term in the right-hand side of (15) as the sample error (or estimate error) and the second term as the approximation error.

A. Bounding Sample Error

We bound the sample error in two cases: $r = 1$ and $r > 1$. We first consider the case $r = 1$. Since $f_\rho(x) = \mathbf{E}[Y|X = x]$, we have

$$\begin{aligned}\widetilde{N^1_{n,w}}(x) = &\frac{\sum_{i=1}^{|T_1|} f_\rho\big(X_i^1\big)}{|T_1|} \\ &+ \sum_{j=1}^{n-1}\left(\frac{\sum_{i=1}^{|T_{j+1}|} f_\rho\Big(X_i^{j+1}\Big)}{|T_{j+1}|} - \frac{\sum_{i=1}^{|T_j|} f_\rho(X_i^j)}{|T_j|}\right) \\ &\times \sigma\big(w\big(\overline{d}(\xi_1, x) - \overline{d}\big(\xi_1, \xi_j\big)\big)\big).\end{aligned} \tag{16}$$

Denote

$$g_j := \frac{\sum_{i=1}^{|T_j|} Y_i^j}{\big|T_j\big|} = \sum_{i=1}^m Y_i \frac{I_{\{X_i \in A_j\}}}{\sum_{l=1}^m I_{\{X_l \in A_l\}}}$$

and

$$g_j^* := \frac{\sum_{i=1}^{|T_j|} f_\rho\Big(X_i^j\Big)}{\big|T_j\big|} = \sum_{i=1}^m f_\rho(X_i) \frac{I_{\{X_i \in A_j\}}}{\sum_{l=1}^m I_{\{X_l \in A_l\}}}$$

where I_A denotes the indicator function of set A. Define further

$$\begin{aligned}c_1(x) &:= 1 - \sigma\big(w\overline{d}(\xi_1, x)\big) \\ c_n(x) &:= \sigma\big(w\big(\overline{d}(\xi_1, x) - \overline{d}(\xi_1, \xi_{n-1})\big)\big)\end{aligned}$$

and

$$\begin{aligned}c_j(x) := &\sigma\big(w\big(\overline{d}(\xi_1, x) - \overline{d}\big(\xi_1, \xi_{j-1}\big)\big)\big) \\ &- \sigma\big(w\big(\overline{d}(\xi_1, x) - \overline{d}\big(\xi_1, \xi_j\big)\big)\big),\ 2 \le j \le n-1.\end{aligned}$$

We have

$$N^1_{n,w}(x) = \sum_{j=1}^n g_j c_j(x) \tag{17}$$

and

$$\widetilde{N^1_{n,w}}(x) = \sum_{j=1}^n g_j^* c_j(x). \tag{18}$$

Since $A_{j_1} \cap A_{j_2} = \varnothing$ $(j_1 \ne j_2)$ and $\mathcal{X} = \bigcup_{j=1}^n A_j$, for arbitrary x, there is a unique k_0 such that $x \in A_{k_0}$. Without loss of generality, we assume $k_0 \ge 2$. Then it follows from (2) and the definition of $\overline{d}(\xi_1, x)$ that:

$$\begin{aligned}&\overline{d}(\xi_1, x) - \overline{d}(\xi_1, \xi_k) \ge \frac{1}{4}n^{-1/d},\ 1 \le k \le k_0 - 1 \\ &\big|\overline{d}(\xi_1, x) - \overline{d}\big(\xi_1, \xi_{k_0}\big)\big| \le 2n^{-1/d}\end{aligned}$$

and

$$\overline{d}(\xi_1, x) - \overline{d}(\xi_1, \xi_k) \le -\frac{1}{4}n^{-1/d},\ k \ge k_0 + 1.$$

Due to (7), (8), and $w \ge 4Kn^{1/d}$, we have when $k \le k_0 - 1$

$$\big|\sigma\big(w\big(\overline{d}(\xi_1, x) - \overline{d}(\xi_1, \xi_k)\big)\big) - 1\big| < n^{-(s+d)/d} \tag{19}$$

and when $k \ge k_0 + 1$

$$\big|\sigma\big(w\big(\overline{d}(\xi_1, x) - \overline{d}(\xi_1, \xi_k)\big)\big)\big| < n^{-(s+d)/d}. \tag{20}$$

Then for arbitrary $j \le k_0 - 1$ and $j \ge k_0 + 2$, (19), (20) and the definition of $c_j(x)$ yield

$$\big|c_j(x)\big| \le 2n^{-(s+d)/d}. \tag{21}$$

Since $|Y_j| \le M$ almost surely, we have $|g_j| \le M$ and $|g_j^*| \le M$ almost surely for any $j = 1, \ldots, n$. Since

$$\mathbf{E}\big[g_j|X_1, \ldots, X_m\big] = g_j^*$$

we get

$$\begin{aligned}&\mathbf{E}\left[\Big(N^1_{n,w}(x) - \widetilde{N^1_{n,w}}(x)\Big)^2|X_1, \ldots, X_m\right] \\ &\quad\le 2M^2n^{-2s/d} \\ &\quad+ \mathbf{E}\left[\Big(\Big(g_{k_0} - g^*_{k_0}\Big)c_{k_0}(x)\Big)^2|X_1, \ldots, X_m\right] \\ &\quad+ \mathbf{E}\left[\Big(\Big(g_{k_0+1} - g^*_{k_0+1}\Big)c_{k_0+1}(x)\Big)^2|X_1, \ldots, X_m\right].\end{aligned}$$

We hence only need to bound

$$\mathbf{E}\left[\Big(\Big(g_{k_0} - g^*_{k_0}\Big)c_{k_0}(x)\Big)^2|X_1, \ldots, X_m\right]$$

since

$$\mathbf{E}\left[\Big(\Big(g_{k_0+1} - g^*_{k_0+1}\Big)c_{k_0+1}(x)\Big)^2|X_1, \ldots, X_m\right]$$

can be bounded by using the same method. As σ is bounded, we have

$$\begin{aligned}&\mathbf{E}\left[\Big(\Big(g_{k_0} - g^*_{k_0}\Big)c_{k_0}(x)\Big)^2|X_1, \ldots, X_m\right] \\ &\quad\le 4\|\sigma\|^2_\infty \mathbf{E}\left[\Big(g_{k_0} - g^*_{k_0}\Big)^2|X_1, \ldots, X_m\right].\end{aligned}$$

Due to the definition of g_{k_0} and $g^*_{k_0}$, we get

$$\begin{aligned}&\mathbf{E}\left[\Big(\Big(g_{k_0} - g^*_{k_0}\Big)\Big)^2|X_1, \ldots, X_m\right] \\ &\quad= \frac{\sum_{i=1}^m \big(Y_i - f_\rho(X_i)\big)^2 I_{\left\{X_i \in A_{k_0}\right\}}}{\big(m\mu_m(A_{k_0})\big)^2} \\ &\quad\le \frac{4M^2}{m\mu_m\big(A_{k_0}\big)} I_{\left\{m\mu_m\left(A_{k_0}\right) > 0\right\}}\end{aligned}$$

where $\mu_m(A_{k_0})$ denotes the empirical measure of A_{k_0}. Then it can be found in [14, pp. 65–66] that

$$\mathbf{E}\left[\frac{I_{\left\{m\mu_m\left(A_{k_0}\right)>0\right\}}}{m\mu_m\left(A_{k_0}\right)}\right] \le \frac{4n}{m}.$$

This implies

$$\begin{aligned}&\mathbf{E}\left[\left\|N_{n,w}^1 - \widetilde{N_{n,w}^1}\right\|_\rho^2\right]\\ &= \mathbf{E}\left[\mathbf{E}\left[\left(N_{n,w}^1(X) - \widetilde{N_{n,w}^1}(X)\right)^2 | X, X_1, \ldots, X_m\right]\right]\\ &\le 2M^2 n^{-2s/d} + 32\|\sigma\|_\infty^2 M^2 n/m.\end{aligned}$$

Since $n \sim m^{d/(2s+d)}$, we obtain

$$\mathbf{E}\left[\left\|N_{n,w}^1 - \widetilde{N_{n,w}^1}\right\|_\rho^2\right] \le C' m^{-2s/(2s+d)} \tag{22}$$

where C' is a constant depending only on M.

We then bound the sample error for $r > 1$. It follows from the definition of $N_{n,w}^{k+1}(x)$ that:

$$\begin{aligned}N_{n,w}^{k+1}(x) = N_{n,w}^k(x) + N_{n,w}^1(x) - \frac{\sum_{i=1}^{|T_1|}\left(N_{n,w}^k\left(X_i^1\right)\right)}{|T_1|}\\ - \sum_{j=1}^{n-1}\left(\frac{\sum_{i=1}^{|T_{j+1}|}\left(N_{n,w}^k\left(X_i^{j+1}\right)\right)}{|T_{j+1}|}\right.\\ \left. - \frac{\sum_{i=1}^{|T_j|}\left(N_{n,w}^k\left(X_i^j\right)\right)}{|T_j|}\right)\\ \times \sigma\left(w\left(\overline{d}(\xi_1, x) - \overline{d}\left(\xi_1, \xi_j\right)\right)\right)\end{aligned}$$

where $T_j \coloneqq \{X_1^j, X_2^j, \ldots, X_{|T_j|}^j\}, j = 1, \ldots, n$. Since

$$\begin{aligned}&\frac{\sum_{i=1}^{|T_1|}\left(N_n^k\left(X_i^1\right)\right)}{|T_1|}\\ &+ \sum_{j=1}^{n-1}\left(\frac{\sum_{i=1}^{|T_{j+1}|}\left(N_n^k\left(X_i^{j+1}\right)\right)}{|T_{j+1}|} - \frac{\sum_{i=1}^{|T_j|}\left(N_n^k\left(X_i^j\right)\right)}{|T_j|}\right)\\ &\times \sigma\left(w\left(\overline{d}(\xi_1, x) - \overline{d}\left(\xi_1, \xi_j\right)\right)\right)\end{aligned}$$

is independent of $Y_1, \ldots, Y_m$, and

$$\widetilde{N_{n,w}^k}(x) = \mathbf{E}\left[N_{n,w}^k(x) | X_1, \ldots, X_m|\right]$$

we have

$$\begin{aligned}&\mathbf{E}\left[\left(N_{n,w}^{k+1}(x) - \widetilde{N_{n,w}^{k+1}}(x)\right)^2 | X_1, \ldots, X_m\right]\\ &\le 2\mathbf{E}\left[\left(N_{n,w}^1(x) - \widetilde{N_{n,w}^1}(x)\right)^2 | X_1, \ldots, X_m\right]\\ &+ 2\mathbf{E}\left[\left(N_{n,w}^k(x) - \widetilde{N_{n,w}^k}(x)\right)^2 | X_1, \ldots, X_m\right].\end{aligned}$$

Therefore, we obtain

$$\mathbf{E}\left[\left\|N_{n,w}^r - \widetilde{N_{n,w}^r}\right\|_\rho^2\right] \le 2r\mathbf{E}\left[\left\|N_{n,w}^1 - \widetilde{N_{n,w}^1}\right\|_\rho^2\right].$$

Then it follows from (22) that:

$$\mathbf{E}\left[\left\|N_{n,w}^r - \widetilde{N_{n,w}^r}\right\|_\rho^2\right] \le 2rC' m^{-2s/(2s+d)}. \tag{23}$$

B. Bounding Approximation Error

It is obvious that

$$\begin{aligned}\left(\widetilde{N_{n,w}^r}(x) - f_\rho(x)\right)^2 = \left(\widetilde{N_{n,w}^r}(x) - f_\rho(x)\right)^2 I_{\left\{\mu_m\left(A_{k_0}\right)>0\right\}}\\ + \left(\widetilde{N_{n,w}^r}(x) - f_\rho(x)\right)^2 I_{\left\{\mu_m\left(A_{k_0}\right)=0\right\}}.\end{aligned}$$

Due to the definition, we obtain

$$\left\|\widetilde{N_{n,w}^r}\right\|_\infty \le (2\|\sigma\|_\infty + 1)rM$$

almost surely. Then it follows from [14, pp. 66–67] that:

$$\begin{aligned}&\mathbf{E}\left[\left(\widetilde{N_{n,w}^r}(X) - f_\rho(X)\right)^2 I_{\left\{\mu_m\left(A_{k_0}\right)=0\right\}}\right]\\ &\le 3(2\|\sigma\|_\infty + 1)^2 r^2 M^2 \frac{n}{m}.\end{aligned}$$

Furthermore

$$\begin{aligned}&\left(\widetilde{N_{n,w}^r}(x) - f_\rho(x)\right)^2 I_{\left\{\mu_m\left(A_{k_0}\right)>0\right\}}\\ &= \left(\widetilde{N_{n,w}^r}(x) - f_\rho(x)\right)^2 I_{\left\{\mu_m\left(A_{k_0}\right)>0, \mu_m\left(A_{k_0+1}\right)>0\right\}}\\ &+ \left(\widetilde{N_{n,w}^r}(x) - f_\rho(x)\right)^2 I_{\left\{\mu_m\left(A_{k_0}\right)>0, \mu_m\left(A_{k_0+1}\right)=0\right\}}.\end{aligned}$$

Similar method in [14, pp. 66–67] yields

$$\begin{aligned}&\mathbf{E}\left[\left(\widetilde{N_{n,w}^r}(X) - f_\rho(X)\right)^2 I_{\left\{\mu_m\left(A_{k_0}\right)>0, \mu_m\left(A_{k_0+1}\right)=0\right\}}\right]\\ &\le 3(2\|\sigma\|_\infty + 1)^2 r^2 M^2 \frac{n}{m}.\end{aligned}$$

Hence, we have

$$\begin{aligned}\mathbf{E}\left[\left(\widetilde{N_{n,w}^r}(X) - f_\rho(X)\right)^2\right] \le 6(2\|\sigma\|_\infty + 1)^2 r^2 M^2 \frac{n}{m}\\ + \mathbf{E}\left[\left(\widetilde{N_{n,w}^r}(X) - f_\rho(X)\right)^2 I_{\left\{\mu_m\left(A_{k_0}\right)>0, \mu_m\left(A_{k_0+1}\right)>0\right\}}\right].\end{aligned} \tag{24}$$

To bound

$$\mathbf{E}\left[\left(\widetilde{N_{n,w}^r}(X) - f_\rho(X)\right)^2 I_{\left\{\mu_m\left(A_{k_0}\right)>0, \mu_m\left(A_{k_0+1}\right)>0\right\}}\right]$$

we need to introduce a series of auxiliary functions. Let X_j^* be arbitrary sample in A_j and Y_j^* be its corresponding output. If there is no point in A_j, then we denote by $Y_j^* = 0$, and $g(X_j^*) = 0$ for arbitrary function g. Define

$$\begin{aligned}L_{n,w}^1(x) = f_\rho\left(X_1^*\right) + \sum_{j=1}^{n-1}\left(f_\rho\left(X_{j+1}^*\right) - f_\rho\left(X_j^*\right)\right)\\ \times \sigma\left(w\left(\overline{d}(\xi_1, x) - \overline{d}\left(\xi_1, \xi_j\right)\right)\right)\end{aligned}$$

and

$$L_{n,w}^{k+1}(x) = L_{n,w}^k(x) + V_{n,w}^k(x)$$

with

$$V_{n,w}^{k}(x) = f_\rho\left(X_1^*\right) - L_n^k\left(X_1^*\right) + \sum_{j=1}^{n-1}\left(\left(f_\rho\left(X_{j+1}^*\right) - L_n^k\left(X_{j+1}^*\right)\right) - \left(f_\rho\left(X_j^*\right) - L_n^k\left(X_j^*\right)\right)\right) \times \sigma\left(w\left(\overline{d}(\xi_1, x) - \overline{d}\left(\xi_1, \xi_j\right)\right)\right).$$

Then for arbitrary $x \in \mathcal{X}$ and $1 \le k \le r$, there holds

$$\left|\widetilde{N_{n,w}^k}(x) - f_\rho(x)\right| \le \max_{X_1^*,\ldots,X_n^*}\left|L_{n,w}^k(x) - f_\rho(x)\right| \tag{25}$$

where the maximum runs over all the possible choices of $X_j^* \in A_j$, $j = 1, 2, \ldots, n$.

Due to the definition of $c_j(x)$, we get

$$L_{n,w}^1(x) = \sum_{j=1}^{n} f_\rho\left(X_j^*\right)c_j(x), \quad \text{and} \quad \sum_{j=1}^{n} c_j(x) = 1.$$

We then prove that for arbitrary $k \ge 1$, there exists a set of functions $\{c_j^k(x)\}_{j=1}^n$ such that

$$L_{n,w}^k(x) = \sum_{j=1}^{n} f_\rho\left(X_j^*\right)c_j^k(x), \ \text{and} \ \sum_{j=1}^{n} c_j^k(x) = 1. \tag{26}$$

We prove (26) by induction. It is obvious that (26) holds for $k = 1$ with $c_j^1(x) = c_j(x)$. Assume (26) holds for $l \ge 1$, that is

$$L_{n,w}^l(x) = \sum_{j=1}^{n} f_\rho\left(X_j^*\right)c_j^l(x), \ \text{and} \ \sum_{j=1}^{n} c_j^l(x) = 1.$$

Obviously

$$V_{n,w}^l(x) = \sum_{j=1}^{n}\left(f_\rho\left(X_j^*\right) - L_{n,w}^l\left(X_j^*\right)\right)c_j^1(x).$$

Therefore

$$L_{n,w}^{l+1}(x) = L_{n,w}^l(x) + V_{n,w}^l(x) = \sum_{j=1}^{n} f_\rho\left(X_j^*\right)c_j^l(x) + \sum_{j=1}^{n}\left(f_\rho\left(X_j^*\right) - \sum_{i=1}^{n} f_\rho\left(X_i^*\right)c_i^l\left(X_j^*\right)\right)c_j^1(x) = \sum_{j=1}^{n} f_\rho\left(X_j^*\right)\left(c_j^l(x) + c_j^1(x) - \sum_{i=1}^{n} c_j^l\left(X_i^*\right)c_i^1(x)\right).$$

Define

$$c_j^{l+1}(x) \coloneqq c_j^l(x) + c_j^1(x) - \sum_{i=1}^{n} c_j^l\left(X_i^*\right)c_i^1(x). \tag{27}$$

Then it follows from $\sum_{j=1}^n c_j^1(x) = 1$ and $\sum_{j=1}^n c_j^l(x) = 1$ that:

$$\sum_{j=1}^{n} c_j^{l+1}(x) = 2 - \sum_{i=1}^{n}\sum_{j=1}^{n} c_j^l\left(X_i^*\right)c_i^1(x) = 1.$$

This proves (26). Equation (26) together with (27) implies

$$\begin{aligned} L_{n,w}^{k+1}(x) - f_\rho(x) &= \sum_{j=1}^{n} c_j^{k+1}(x)\left(f_\rho\left(X_j^*\right) - f_\rho(x)\right) \\ &= \sum_{j=1}^{n} c_j^k(x)\left(f_\rho\left(X_j^*\right) - f_\rho(x)\right) \\ &\quad - \sum_{i=1}^{n} c_i^1(x)\left[\sum_{j=1}^{n} c_j^k\left(X_i^*\right)\left(f_\rho\left(X_j^*\right) - f_\rho\left(X_i^*\right)\right)\right] \\ &= L_{n,w}^k(x) - f_\rho(x) - \sum_{j=1}^{n} c_j^1(x) \\ &\quad \times \left(L_{n,w}^k\left(X_j^*\right) - f_\rho\left(X_j^*\right)\right). \end{aligned}$$

If we denote by $\lambda^k(x) \coloneqq L_{n,w}^k(x) - f_\rho(x)$ with $\lambda^0(x) \coloneqq -f_\rho(x)$, then

$$\begin{aligned} \lambda^{k+1}(x) &= \lambda^k(x) - \sum_{j=1}^{n} c_j^1(x)\lambda^k\left(X_j^*\right) \\ &= \sum_{j=1}^{n} c_j^1(x)\left[\lambda^k(x) - \lambda^k\left(X_j^*\right)\right]. \end{aligned}$$

When $0 < s \le 1$, we get from (21) and $f_\rho \in \mathcal{F}^s$ that

$$\begin{aligned} \left\|\lambda^1\right\|_\infty &\le \left|\sum_{j\ne k_0, k_0+1}^{n} c_j^1(x)\left[f_\rho\left(X_j^*\right) - f_\rho(x)\right]\right| \\ &\quad + 4\left\|f_\rho\right\|_\infty n^{-s/d} + C\|\sigma\|_\infty\left(2^s n^{-s/d} + 4^s n^{-s/d}\right) \\ &= C_1' n^{-s/d} \end{aligned} \tag{28}$$

with

$$C_1' \coloneqq 4\left\|f_\rho\right\|_\infty + 6C\|\sigma\|_\infty.$$

When $s > 1$, it follows from the multivariate mean-valued theorem that:

$$\begin{aligned} \lambda^1(x) &= \sum_{j=1}^{n} c_j^1(x)\left[f_\rho\left(X_j^*\right) - f_\rho(x)\right] \\ &= n^{-1/d} n^{1/d} \sum_{j=1}^{n} c_j^1(x) \sum_{\ell=1}^{d}\left(x^{(\ell)} - X_j^{*(\ell)}\right)\beta_{1,\ell}\left(\xi_{x,j}\right) \end{aligned}$$

where $\xi_{x,j} \in [0, 1]^d$, $\beta_{1,\ell}$ is the first partial derivative of f_ρ with respect to $x^{(\ell)}$ and $x = (x^{(1)}, x^{(2)}, \ldots, x^{(d)})$ with $x^{(\ell)} \in [0, 1]$. Set

$$\alpha_{n,1}(x) = n^{1/d} \sum_{j=1}^{n} c_j^1(x) \sum_{\ell=1}^{d}\left(x^{(\ell)} - X_j^{*(\ell)}\right)\beta_{1,\ell}\left(\xi_{x,j}\right).$$

We have from (21) and $x \in A_{k_0}$ that

$$\begin{aligned} \left\|\alpha_{n,1}\right\|_\infty &\le n^{\frac{1}{d}}\|\sigma\|_\infty d \max_{1\le\ell\le d}\left|x^{(\ell)} - X_{k_0}^{*(\ell)}\right|\left\|\beta_{1,\ell}\right\|_\infty \\ &\quad + n^{\frac{1}{d}}\|\sigma\|_\infty d \max_{1\le\ell\le d}\left|x^{(\ell)} - X_{k_0+1}^{*(\ell)}\right|\left\|\beta_{1,\ell}\right\|_\infty \\ &\quad + n^{\frac{1}{d}} 2d \max_{1\le j\le n, 1\le\ell\le d}\left\|\beta_{1,\ell}\right\|_\infty n^{-\frac{s}{d}}. \end{aligned}$$

This means $\|\alpha_{n,1}\|_\infty \le \tilde{C}_{k+1}$ with

$$\tilde{C}_1 := 2d\|\sigma\|_\infty \max_{1\le \ell\le d} \left\|\beta_{1,\ell}\right\|_\infty (1+2\|\sigma\|_\infty).$$

Since $f_\rho \in \mathcal{F}^s$ with $s = u+\beta$, we can repeate the above procedure u times to obtain

$$\lambda^u(x) = n^{-u/d}\alpha_{n,u}(x) \tag{29}$$

where $\alpha_{n,u}$ satisfies

$$\left|\alpha_{n,u}(x) - \alpha_{n,u}(x')\right| \le Cd(x,x')^\beta$$

and

$$\left\|\alpha_{n,u}\right\|_\infty \le \tilde{C}_u$$

with some positive constant $\tilde{C}_u$ independent of m and n. For $r = u$, we then have

$$\begin{aligned}\left|\lambda^{u+1}(x)\right| &\le \sum_{j=1}^n c_j^1(x)\left|\lambda^u(x) - \lambda^u\left(X_j^*\right)\right| \\ &\le n^{-u/d}\sum_{j=1}^n c_j^1(x)\left|\alpha_{n,u}(x) - \alpha_{n,u}\left(X_j^*\right)\right| \\ &\le n^{-u/d}\sum_{j=1}^n c_j^1(x)\left|x - X_j^*\right|^\beta.\end{aligned}$$

It follows from (21) that:

$$\left\|\lambda^{u+1}\right\|_\infty \le 2\|\sigma\|_\infty\left(\tilde{C}_u + 2C\|\sigma\|_\infty\right)n^{-s/d}.$$

For $k > u$, the above estimate yields

$$\begin{aligned}\left|\lambda^{k+1}(x)\right| &\le \sum_{j=1}^n c_j^1(x)\left|\lambda^k(x) - \lambda^k\left(X_j^*\right)\right| \\ &\le \sum_{j=1}^n c_j^1(x)\left|\lambda^k(x)\right| + \left|\lambda^k\left(X_j^*\right)\right| \\ &\le \le 4\|\sigma\|_\infty\left(\tilde{C}_u + 2C\|\sigma\|_\infty\right)n^{-s/d}.\end{aligned}$$

That is

$$\left\|\lambda^k\right\|_\infty \le \tilde{C}_k, \quad k > u \tag{30}$$

with $\tilde{C}_k = 4\|\sigma\|_\infty(\tilde{C}_u + 2C\|\sigma\|_\infty)n^{-s/d}$. Thus, Theorem 1 follows from (15), (28), (30), (22), and (23).

VI. Conclusion

In this paper, we succeeded in constructing an FNN, called CFN, for learning purpose. Both theoretical and numerical results showed that CFN is efficient and effective. The idea of "constructive neural networks" for learning purpose provided a new springboard for developing scalable neural network-type learning systems.

We concluded in this paper by presenting some extensions of the constructive neural networks learning. this paper, the neural network was constructed by using the method in [22]. Besides [22], there are large portions of neural networks constructed to approximate smooth functions, such as [1], [8], and [9]. All these constructions are proved to possess prominent approximation capability and simultaneously, but suffer from the saturation problem. We guess that by using the approach in this paper, most of these neural networks can be used for learning. We will keep studying in this direction and report the progress in a future publication.

Acknowledgment

The authors would like to thank three anonymous reviewers and the associate editor for their constructive and helpful comments.

References

[1] G. A. Anastassiou, "Multivariate sigmoidal neural network approximation," *Neural Netw.*, vol. 24, no. 4, pp. 378–386, 2011.
[2] E. I. Atanassov, "On the discrepancy of the Halton sequences," *Math. Balkanica New Series*, vol. 18, nos. 1–2, pp. 15–32, 2004.
[3] F. Bach, "Sharp analysis of low-rank kernel matrix approximations," *ArXiv:1208.2015*, 2013.
[4] A. R. Barron, A. Cohen, W. Dahmen, and R. A. Devore, "Approximation and learning by greedy algorithms," *Ann. Stat.*, vol. 36, no. 1, pp. 64–94, 2008.
[5] P. Bratley and B. L. Fox, "Algorithm 659: Implementing Sobol's quasirandom sequence generator," *ACM Trans. Math. Softw.*, vol. 14, no. 1, pp. 88–100, 1988.
[6] X. Chang, S.-B. Lin, and Y. Wang, "Divide and conquer local average regression," *Electron. J. Stat.*, vol. 11, no. 1, pp. 1326–1350, 2017.
[7] D. B. Chen, "Degree of approximation by superpositions of a sigmoidal function," *Approx. Theory Appl.*, vol. 9, no. 3, pp. 17–28, 1993.
[8] D. Costarelli and R. Spigler, "Approximation results for neural network operators activated by sigmoidal functions," *Neural Netw.*, vol. 44, pp. 101–106, Aug. 2013.
[9] D. Costarelli and R. Spigler, "Multivariate neural network operators with sigmoidal activation functions," *Neural Netw.*, vol. 48, pp. 72–77, Dec. 2013.
[10] F. Cucker and D. X. Zhou, *Learning Theory: An Approximation Theory Viewpoint*. Cambridge, U.K.: Cambridge Univ. Press, 2007.
[11] M. Eberts and I. Steinwart, "Optimal learning rates for least squares SVMs using Gaussian kernels," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 24. 2011, pp. 1539–1547.
[12] H. Engl, M. Hanke, and A. Neubauer, *Regularization of Inverse Problem*. Amsterdam, The Netherlands: Kluwer Acad., 2000.
[13] S. Fine and K. Scheinberg, "Efficient SVM training using low-rank kernel representations," *J. Mach. Learn. Res.*, vol. 2, pp. 243–264, Mar. 2002.
[14] L. Györfy, M. Kohler, A. Krzyzak, and H. Walk, *A Distribution-Free Theory of Nonparametric Regression*. Berlin, Germany: Springer, 2002.
[15] M. T. Hagan, H. B. Demuth, and M. H. Beale, *Neural Network Design*. Boston, MA, USA: PWS, 1996.
[16] G. B. Huang, L. Chen, and C. K. Siew, "Universal approximation using incremental constructive feedforward networks with random hidden nodes," *IEEE Trans. Neural Netw.*, vol. 17, no. 4, pp. 879–892, Jul. 2006.
[17] G.-B. Huang, Q.-Y. Zhu, and C.-K. Siew, "Extreme learning machine: Theory and applications," *Neurocomputing*, vol. 70, nos. 1–3, pp. 489–501, 2006.
[18] G.-B. Huang and L. Chen, "Convex incremental extreme learning machine," *Neurocomputing*, vol. 70, nos. 16–18, pp. 3056–3062, 2007.
[19] G.-B. Huang, "What are extreme learning machines? Filling the gap between Frank Rosenblatt's dream and John von Neumann's puzzle," *Cogn. Comput.*, vol. 7, no. 3, pp. 263–278, 2015.
[20] H. Jaeger and H. Haas, "Harnessing nonlinearity: Predicting chaotic systems and saving energy in wireless communication," *Science*, vol. 304, no. 5667, pp. 78–80, 2004.
[21] M. Kohler and J. Mehnert, "Analysis of the rate of convergence of least squares neural network regression estimates in case of measurement errors," *Neural Netw.*, vol. 24, no. 3, pp. 273–279, 2011.
[22] S. Lin, Y. Rong, and Z. Xu, "Multivariate Jackson-type inequality for a new type neural network approximation," *Appl. Math. Model.*, vol. 38, no. 24, pp. 6031–6037, 2014.

[23] S. Lin, X. Liu, J. Fang, and Z. Xu, "Is extreme learning machine feasible? A theoretical assessment (part II)," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 26, no. 1, pp. 21–34, Jan. 2015.
[24] S. B. Lin and D. X. Zhou, "Distributed kernel-based gradient descent algorithms," *Construct. Approx*, vol. 47, no. 2, pp. 249–276, 2018.
[25] X. Liu, S. Lin, J. Fang, and Z. Xu, "Is extreme learning machine feasible? A theoretical assessment (part I)," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 26, no. 1, pp. 7–20, Jan. 2015.
[26] B. Llanas and F. J. Sainz, "Constructive approximate interpolation by neural networks," *J. Comput. Appl. Math.*, vol. 188, no. 2, pp. 283–308, 2006.
[27] V. Maiorov and R. S. Meir, "Approximation bounds for smooth functions in $C(\mathbf{R}^d)$ by neural and mixture networks," *IEEE Trans. Neural Netw.*, vol. 9, no. 5, pp. 969–978, Sep. 1998.
[28] V. E. Maiorov and R. Meir, "On the near optimality of the stochastic approximation of smooth functions by neural networks," *Adv. Comput. Math.*, vol. 13, no. 1, pp. 79–103, 2000.
[29] V. Maiorov, "Almost optimal estimates for best approximation by translates on a torus," *Construct. Approx.*, vol. 21, no. 3, pp. 337–349, 2005.
[30] V. Maiorov, "Approximation by neural networks and learning theory," *J. Complexity*, vol. 22, no. 1, pp. 102–117, 2006.
[31] M. Meister and I. Steinwart, "Optimal learning rates for localized SVMs," *J. Mach. Learn. Res.*, vol. 17, no. 1, pp. 6722–6765, 2016.
[32] F. J. Narcowich, X. Sun, J. D. Ward, and H. Wendland, "Direct and inverse Sobolev error estimates for scattered data interpolation via spherical basis functions," *Found. Comput. Math.*, vol. 7, no. 3, pp. 369–370, 2007.
[33] Y.-H. Pao, *Adaptive Pattern Recognition and Neural Networks*. Reading, MA, USA: Addison-Wesley, 1989.
[34] B. U. Park, Y. K. Lee, and S. Ha, "L_2 boosting in kernel regression," *Bernoulli*, vol. 15, no. 3, pp. 599–613, 2009.
[35] L. Shi, Y.-L. Feng, and D.-X. Zhou, "Concentration estimates for learning with l_1-regularizer and data dependent hypothesis spaces," *Appl. Comput. Harmonic Anal.*, vol. 31, no. 2, pp. 286–302, 2011.
[36] L. Shi, "Learning theory estimates for coefficient-based regularized regression," *Appl. Comput. Harmonic Anal.*, vol. 34, no. 2, pp. 252–265, 2013.
[37] D. E. Rumelhart, G. E. Hintont, and R. J. Williams, "Learning representations by back-propagating errors," *Nature*, vol. 323, no. 6088, pp. 533–536, 1986.
[38] J. Taylor and N. Cristianini, *Kernel Methods for Pattern Analysis*. Cambridge, U.K.: Springer, 2004.
[39] H. Wendland, *Scattered Data Approximation*. Cambridge, U.K.: Cambridge Univ. Press, 2005.
[40] B. M. Wilamowski and H. Yu, "Neural network learning without backpropagation," *IEEE Trans. Neural Netw.*, vol. 21, no. 11, pp. 1793–1803, Nov. 2010.
[41] Q. Wu and D.-X. Zhou, "Learning with sample dependent hypothesis space," *Comput. Math. Appl.*, vol. 56, no. 11, pp. 2896–2907, 2008.
[42] J. Zeng, S. Lin, and Z. Xu, "Sparse regularization: Convergence of iterative jumping thresholding algorithm," *IEEE Trans. Signal Process.*, vol. 64, no. 19, pp. 5106–5118, Oct. 2016.
[43] Y. C. Zhang, J. Duchi, and M. Wainwright, "Divide and conquer kernel ridge regression: A distributed algorithm with minimax optimal rates," *J. Mach. Learn. Res.*, vol. 16, no. 1, pp. 3299–3340, 2015.
[44] Z.-H. Zhou, N. V. Chawla, Y. Jin, and G. J. Williams, "Big data opportunities and challenges: Discussions from data analytics perspectives [discussion forum]," *IEEE Comput. Intell. Mag.*, vol. 9, no. 4, pp. 62–74, Nov. 2014.

Shaobo Lin received the Ph.D. degree in applied mathematics from Xi'an Jiaotong University, Xi'an, China, in 2014.

He is currently an Assistant Professor with the College of Mathematics and Information Science, Wenzhou University, Wenzhou, China.

Jinshan Zeng received the Ph.D. degree in applied mathematics from Xi'an Jiaotong University, Xi'an, China, in 2015.

He is currently an Assistant Professor with the School of Computer and Information Engineering, Jiangxi Normal University, Nanchang, China.

Xiaoqin Zhang received the B.Sc. degree in electronic information science and technology from Central South University, Changsha, China, in 2005 and the Ph.D. degree in pattern recognition and intelligent system from the National Laboratory of Pattern Recognition, Institute of Automation, Chinese Academy of Sciences, Beijing, China, in 2010.

He is currently a Professor with the College of Mathematics and Information Science, Wenzhou University, Wenzhou, China.