Journal of the American Statistical Association

ISSN: 0162-1459 (Print) 1537-274X (Online) Journal homepage: www.tandfonline.com/journals/uasa20

Deep Regression Learning with Optimal Loss Function

Xuancheng Wang, Ling Zhou & Huazhen Lin

JOURNAL OF THE AMERICAN STATISTICAL ASSOCIATION
2025, VOL. 120, NO. 550, 1305–1317: Theory and Methods
https://doi.org/10.1080/01621459.2024.2412364

Deep Regression Learning with Optimal Loss Function

Xuancheng Wang*, Ling Zhou*, and Huazhen Lin

New Cornerstone Science Laboratory, Center of Statistical Research and School of Statistics, Southwestern University of Finance and Economics, Chengdu, China

ABSTRACT
In this article, we develop a novel efficient and robust nonparametric regression estimator under a framework of a feedforward neural network (FNN). There are several interesting characteristics for the proposed estimator. First, the loss function is built upon an estimated maximum likelihood function, which integrates the information from observed data as well as the information from the data distribution. Consequently, the resulting estimator has desirable optimal properties, such as efficiency. Second, different from the traditional maximum likelihood estimation (MLE), the proposed method avoids the specification of the distribution, making it adaptable to various distributions such as heavy tails and multimodal or heterogeneous distributions. Third, the proposed loss function relies on probabilities rather than direct observations as in least square loss, contributing to the robustness of the proposed estimator. Finally, the proposed loss function involves a nonparametric regression function only. This enables the direct application of the existing packages, simplifying the computational and programming requirements. We establish the large sample property of the proposed estimator in terms of its excess risk and minimax near-optimal rate. The theoretical results demonstrate that the proposed estimator is equivalent to the true MLE where the density function is known in terms of excess risk. Our simulation studies show that the proposed estimator outperforms the existing methods based on prediction accuracy, efficiency and robustness. Particularly, it is comparable to the MLE with the known density and even gets slightly better as the sample size increases. This implies that the adaptive and data-driven loss function from the estimated density may offer an additional avenue for capturing valuable information. We further apply the proposed method to four real data examples, resulting in significantly reduced out-of-sample prediction errors compared to existing methods. Supplementary materials for this article are available online, including a standardized description of the materials available for reproducing the work.

ARTICLE HISTORY
Received October 2023
Accepted September 2024

KEYWORDS
Estimated maximum likelihood estimation; Excess risk; Feedforward neural network; Kernel density estimation

1. Introduction

Let us consider a fully nonparametric regression model,

$$Y = g(\boldsymbol{X}) + \epsilon, \tag{1}$$

where $Y \in \mathbb{R}$ is a response variable, $\boldsymbol{X} \in \mathcal{X} \subseteq \mathbb{R}^d$ is an d-dimensional vector of predictors, ϵ is an error independent of $\boldsymbol{X}$ with mean zero, that is, $\mathbb{E}(\epsilon) = 0$, and $g : \mathcal{X} \to \mathbb{R}$ is an unknown regression function. Nonparametric regression (1) is a basic and core problem in statistics and machine learning, where the purpose is estimating the unknown regression function g given iid samples $S \equiv (\boldsymbol{X}_i, Y_i)_{i=1}^n$. Since the distribution of ϵ is unknown, $g(\cdot)$ is usually estimated based on the least square (LS) criterion, that is,

$$\hat{g} = \underset{g:\mathbb{R}^d \to \mathbb{R}}{\arg\min} \frac{1}{n} \sum_{i=1}^n \left\{Y_i - g(\boldsymbol{X}_i)\right\}^2. \tag{2}$$

Driven by various nonparametric approximation techniques, there is a vast literature on nonparametric regression. For example, local polynomial regression (Fan and Gijbels 2018), regression with spline approximation (Schumaker 2007) and reproducing kernel regression (Berlinet and Thomas-Agnan 2011; Lv et al. 2018). However, in practice, these methods frequently encounter the "curse of dimensionality" in scenarios where the dimension d of $\boldsymbol{X}$ is moderate or high, for example, when $d > 3$.

Recently, due to powerful function fitting ability, well-designed neural network architectures, effective training algorithms and high-performance computing technologies, deep neural networks (DNNs) with the empirical least square (LS) loss function have enjoyed tremendous success in a variety of applications, such as the fields of computer vision, natural language processing, speech recognition. In theory, when employing a feedforward neural network (FNN) to approximate $g(\cdot)$ from the β-Hölder class, promising findings have emerged regarding the minimax near-optimal learning rate of $n^{-\frac{2\beta}{2\beta+d}}(\log n)^s$ for the LS estimator of g. These results hold under various error assumptions, such as the assumption that the error term is bounded (Györfi et al. 2002; Farrell, Liang, and Misra 2021), possesses a finite pth moment with $p > 1$

CONTACT Huazhen Lin linhz@swufe.edu.cn New Cornerstone Science Laboratory, Center of Statistical Research and School of Statistics, Southwestern University of Finance and Economics, 555 Liutai Road, Wenjiang, Chengdu, Sichuan Province, China.
* These authors are co-first authors.
Supplementary materials for this article are available online. Please go to www.tandfonline.com/r/JASA.

(Kohler and Langer 2021; Kohler, Krzyzak, and Langer 2022), is sub-Gaussian (Schmidt-Hieber 2019, 2020; Bauer and Kohler 2019; Chen et al. 2022; Fan and Gu 2023; Bhattacharya, Fan, and Mukherjee 2023), is subexponential (Jiao et al. 2023; Yan and Yao 2023), or has finite variance (Liu, Boukai, and Shang 2022).

The LS criterion-based estimators are mathematically convenient, easily implemented, and efficient when the error ϵ is normally distributed. However, as expressed in (2), the LS loss is quite sensitive to the observation with large error, making the LS estimator vulnerable to outliers, leading to unstable and unreliable estimations. On the other hand, in the era of "big data," data generation mechanism and collection are unmanageable, and thus non-Gaussian noises or outliers are almost inevitable. Recently, several robust FNN methods have been introduced to address non-Gaussian noise problems, and corresponding convergence rates for learning the function g have also been established. For instance, Lederer (2020), Shen et al. (2021) and Fan, Gu, and Zhou (2022) have explored nonasymptotic error bounds of the estimators that minimize robust loss functions, such as the least-absolute deviation loss (Bassett Jr and Koenker 1978), the Huber loss (Huber 1973), the Cauchy loss and Tukey's biweight loss (Beaton and Tukey 1974). Particularly, based on a general loss function satisfying a Lipschitz continuity, Farrell, Liang, and Misra (2021) demonstrated the convergence rate $n^{-\frac{2\beta}{2\beta+d}}(\log n)^4$ with the assumption that the response is bounded, which means that heavy-tail error is not applicable. To allow a heavy-tailed response Y, Shen et al. (2021) established the convergence rate $n^{-\frac{2\beta}{2\beta+d}+1/p}(\log n)^c$ under the assumption that the pth moment of response is bounded for some $p > 1$. These methods are proposed for improving the robustness; however, they are suboptimal in terms of efficiency.

In this work, we provide a loss function that is both efficient and robust for nonparametric regression functions g within the framework of FNNs. As mentioned earlier, in the least squares (LS) criterion, observations where the response variable Y_i significantly deviates from the conditional mean $g(\boldsymbol{X}_i)$ play a prominent role due to higher noise levels, which may seem counterintuitive. In fact, when aiming to estimate the conditional mean $g(\cdot)$, observations that Y_i is closer to $g(\boldsymbol{X}_i)$ are expected to be more important compared to those where the response deviates significantly from the conditional mean. The importance of an observation can be quantified in terms of observation probabilities. Therefore, we propose a loss function based on the estimated likelihood function, which takes the following form:

$$\hat{g} = \arg\max_{g:\mathbb{R}^d\to\mathbb{R}} \frac{1}{n}\sum_{i=1}^{n} \log \hat{f}_g(Y_i - g(\boldsymbol{X}_i)), \tag{3}$$

where $\hat{f}_g$ is an estimator of the density function $f(\cdot)$ for $\epsilon_i = Y_i - g(\boldsymbol{X}_i)$. Throughout the article, we call the FNN estimators of $g(\cdot)$ based on maximizing estimated log-likelihood functions (3) the EML-FNN. Since we use the likelihood-based criterion, the EML-FNN should have desirable optimal properties, such as efficiency. In addition, different from the traditional maximum likelihood estimator (MLE), where $f(\cdot)$ is known, the proposed EML-FNN does not require specifying the error distribution and hence is flexible to various distributions, such as heavy tails, multimodal or heterogeneous distribution. Moreover, the quasi-likelihood loss (3), which relies on probabilities rather than direct observations as in LSE, contributes to the robustness of the proposed EML-FNN. Interestingly, in comparison to the true MLE where $f(\cdot)$ is known, our simulation studies reveal that the proposed EML-FNN is comparable to the true MLE and even gets slightly but consistently better as the sample size increases. This implies the adaptive loss of (3) by estimating the density $f(\cdot)$ is effective in learning the data structure and may offer an additional avenue to improve accuracy due to its efficiency as well as robustness.

By using the explicit form of the Nadaraya-Watson kernel estimator for the density function, we developed the EML-FNN estimation based on an objective function that exclusively relies on g. This enables the direct application of existing packages PyTorch (Paszke et al. 2019) and Scikit-learn (Pedregosa et al. 2011) in Python, simplifying the computation and programming. Furthermore, we establish the large sample property of the resulting EML-FNN in terms of its excess risk and the minimax rate, which demonstrate that the proposed EML-FNN is equivalent to the true MLE under mild and manageable conditions. As a result, the proposed deep learning approach for g exhibits the desired optimal properties, such as efficiency (Zhou, Lin, and Liang 2018; Zhou et al. 2019). Finally, we apply the proposed method to analyze four real datasets. As shown in Table 4, the EML-FNN method consistently outperforms existing methods in terms of prediction accuracy for each dataset. Examining the Q-Q plot of the estimated error distribution in Figure 6 demonstrates the substantial impact of distribution characteristics on the resulting estimator. This underscores the importance of integrating distribution information into the analysis.

The article is structured as follows. In Section 2, we introduce the proposed EML-FNN. In Section 3, we establish the large sample property of $\hat{g}$ in terms of its excess risk and the minimax rate. Section 4 contains simulation studies to investigate the performance of the proposed method via a comparison with competing estimation methods. In Section 5, we apply the proposed method to analyze four real data. We conclude the article with a discussion in Section 6. Technical proofs are included in the supplementary material.

2. Method

Set $\mathcal{G}$ to be a function class consisting of ReLU feedforward neural networks, that is, $\mathcal{G} := \mathcal{G}_{\mathcal{D},\mathcal{U},\mathcal{W},\mathcal{S},\mathcal{B}}$, where the input data are the predictor $\boldsymbol{X}$, forming the first layer, and the output is the last layer of the network. Such a network $\mathcal{G}$ has $\mathcal{D}$ hidden layers and a total of $(\mathcal{D}+2)$ layers. Denote the width of layer j by d_j, $j = 0, \ldots, \mathcal{D}, \mathcal{D}+1$ with $d_0 = d$ representing the dimension of the input $\boldsymbol{X}$, and $d_{\mathcal{D}+1} = 1$ representing the dimension of the response Y. The width $\mathcal{W}$ is defined as the maximum width among the hidden layers, that is, $\mathcal{W} = \max(d_1, \ldots, d_{\mathcal{D}})$. The size $\mathcal{S}$ is defined as the total number of parameters in the network $\mathcal{G}$, given by $\mathcal{S} = \sum_{i=0}^{\mathcal{D}} d_{i+1} \times (d_i + 1)$. The number of neurons $\mathcal{U}$ is defined as the total number of computational units in the hidden layers, given by $\mathcal{U} = \sum_{i=1}^{\mathcal{D}} d_i$. Furthermore, we assume that every function $g \in \mathcal{G}$ satisfies $|g|_\infty \le \mathcal{B}$ with $\mathcal{B}$ being a positive constant.

We estimate g under the framework of FNN, that is, g is estimated by

$$\arg\min_{g\in\mathcal{G}} \left\{ \frac{1}{n}\sum_{i=1}^{n} \rho(Y_i - g(\boldsymbol{X}_i)) \right\}, \tag{4}$$

where $\rho(\cdot)$ is a given loss function, for example, the least squares $\rho(t) = t^2$, the least absolute criteria $\rho(t) = |t|$ and other robust loss functions. The LS-based estimator is efficient only when the error ϵ is normally distributed. The estimators based on robust loss functions such as least absolute, Huber, Cauchy and Tukey's biweight are robust, but they are suboptimal in terms of efficiency.

When the error distribution $f(\cdot)$ is known, an ideal estimator of g can be obtained by

$$\begin{aligned}\hat{g} &= \arg\min_{g\in\mathcal{G}} \mathcal{R}_n(g) \\ &:= \arg\min_{g\in\mathcal{G}} \left\{ \frac{1}{n}\sum_{i=1}^{n} \left(-\log f(Y_i - g(\boldsymbol{X}_i))\right) \right\}.\end{aligned} \tag{5}$$

However, in reality, f is usually unknown. To prevent potential distribution misspecification and, at the same time, to derive an estimator based on an optimal loss, we employ kernel techniques to estimate the density function f. That is,

$$\hat{f}_g(z) = \frac{1}{n}\sum_{i=1}^{n} \mathcal{K}_h(\epsilon_i, z), \tag{6}$$

where $\epsilon_i = Y_i - g(\boldsymbol{X}_i)$, $\mathcal{K}_h(y_1, y_2) = K(\frac{y_1 - y_2}{h})/h$, h is a bandwidth and $K(\cdot)$ is a kernel function.

Replacing $f(\cdot)$ in (5) with $\hat{f}_g$, we obtain the empirical loss, that is, $\hat{\mathcal{R}}_n(g) := n^{-1}\sum_{i=1}^{n}\left(-\log \hat{f}_g(Y_i - g(\boldsymbol{X}_i))\right) = n^{-1}\sum_{i=1}^{n}\left(-\log \frac{1}{n}\sum_{j=1}^{n}\mathcal{K}_h(Y_j - g(\boldsymbol{X}_j), Y_i - g(\boldsymbol{X}_i))\right)$. Note that $\hat{\mathcal{R}}_n(g)$ shares the same value for any constant shift of g, that is, $\hat{\mathcal{R}}_n(g_1) \equiv \hat{\mathcal{R}}_n(g_1 + c)$. Thus, in practical calculations, after obtaining one solution $\hat{g}$, we rescale $\hat{g}(\boldsymbol{x})$ by $\hat{g}(\boldsymbol{x}) + n^{-1}\sum_{i=1}^{n}\left(Y_i - \hat{g}(\boldsymbol{X}_i)\right)$ to implement the identification condition $\mathbb{E}(\epsilon) = 0$. Particularly, we estimate g by

$$\begin{aligned}\hat{g} &\in \arg\min_{g\in\mathcal{G}} \hat{\mathcal{R}}_n(g) = \arg\min_{g\in\mathcal{G}} n^{-1}\sum_{i=1}^{n} \\ &\quad \left(-\log\frac{1}{n}\sum_{j=1}^{n}\mathcal{K}_h(Y_j - g(\boldsymbol{X}_j), Y_i - g(\boldsymbol{X}_i))\right), \\ \tilde{g} &= \hat{g}(\boldsymbol{x}) + n^{-1}\sum_{i=1}^{n}\left(Y_i - \hat{g}(\boldsymbol{X}_i)\right).\end{aligned} \tag{7}$$

The loss function involves g only. This enables the direct application of the packages PyTorch (Paszke et al. 2019) and Scikit-learn (Pedregosa et al. 2011) in Python, and hence the computation and programming are simple. In addition, the kernel-based estimation approach benefits from the smooth continuity of kernel functions, facilitating gradient calculations and overcoming nondifferentiability issues when dealing with densities such as the uniform distribution, mixture distribution, and heteroscedasticity.

The proposed $\hat{\mathcal{R}}_n(g)$ involves a tuning parameter h. According to the property of kernel approximation (Fan and Gijbels 2018), a smaller h yields a more accurate density approximation but with a larger variance. Fortunately, the summation over individuals mitigates the increased variance caused by a small h. Therefore, when computational feasibility allows, a smaller value of h is preferred. This conclusion is supported by both our theoretical and numerical results. In practice, we use the Gaussian kernel function and set $\hat{f}_g = 1e-5$ when $\hat{f}_g < 1e-5$ because logarithmic transformation is required in the objective function.

Remark 1. Recall that the conventional FNN minimizes a least square objective and is sensitive to the distribution type and outliers. The robust FNN is developed to address the instability of the LS loss, however, the enhanced robustness comes at the cost of efficiency. In contrast to existing methods, our approach uses an MLE criterion as the objective function, achieving both efficiency and robustness. Particularly, efficiency is attained by fully leveraging the data distribution, and robustness is added because our proposed loss function relies on probabilities rather than direct observations.

3. Large Sample Properties

For simplicity, we continue using $\hat{g}(\boldsymbol{x})$ for its rescaled version here and thereafter, unless a distinction is necessary. In this section, we establish the large sample property of $\hat{g}$ in terms of its excess risk, which is defined as the difference between the risks of g and g^*:

$$\begin{aligned}\mathcal{R}(g) - \mathcal{R}(g^*) = \mathbb{E}\left(-\log f(Y_i - g(\boldsymbol{X}_i))\right) \\ - \mathbb{E}\left(-\log f(Y_i - g^*(\boldsymbol{X}_i))\right),\end{aligned}$$

where g^* is defined as

$$g^* := \arg\min_{g} \mathcal{R}(g) = \arg\min_{g} \mathbb{E}\left(-\log f\left(Y_i - g(\boldsymbol{X}_i)\right)\right),$$

where the minimizer is taken over the entire space and thus implies that g^* does not necessarily belong to the FNN set $\mathcal{G}$.

We denote $f^{(r)}(\cdot)$ as the rth derivative of f and $f_{\boldsymbol{x}}(\cdot)$ as the density function of covariates $\boldsymbol{X}$, which is supported on a bounded set, and for simplicity, we assume this bounded set to be $[0,1]^d$. In the rest of the article, the symbol c denotes a positive constant that may vary across different contexts. The following conditions are required for establishing the rate of excess risk:

(C1) Kernel: Let $U_r = \int K(t)t^r dt$ and $v_r = \int K^2(t)t^r dt$. Assume the kernel function $K(\cdot)$ has a bounded second derivative, $U_0 = 1$ and $U_1 = 0$.

(C2) Bandwidth: $h \to 0$ and $nh \to \infty$ as $n \to \infty$.

(C3) Density function f: (C3a) For any $\zeta > 0$, there exists a $\beta_\zeta > 0$ such that $\mathbb{E}(|\log f(\epsilon) I(f(\epsilon) < \beta_\zeta)|) < \zeta$. For $\zeta = O(n^{-1}\log n)$, there exists a $\beta_\zeta = O(n^{-r})$ for some constant $r \geq 1$ satisfying the above inequality. (C3b) Assume the density function $f(\cdot)$ has a continuous first-order derivative and satisfies $\mathbb{E}(|f^{(1)}/f|) < \mathcal{B}$. (C3c) Assume the density function $f(\cdot)$ has a continuous third-order derivative and satisfies $\mathbb{E}|f^{(r_1)}/f|^{r_2} < \mathcal{B}$ for $r_1 = 1,2,3$, $r_2 = 1,2$.

(C4) Function class for g and g^*: For any function $g \in \mathcal{G}$ and the true function g^*, we assume $\|g\|_\infty < \mathcal{B}$ and $\|g^*\|_\infty < \mathcal{B}$.

Condition (C1) is a mild condition for the kernel function, which is easily satisfied when the kernel function is a symmetric density function. Condition (C2) is the most commonly used assumption for the bandwidth. Condition (C2) ensures that the proposed estimator is asymptotically equivalent to the oracle estimator, where the density function is known. Condition (C3a) requires a tail condition on the log transformation of the density. Conditions (C3b) and (C3c) require bounded moment for the density and its derivatives to avoid tail-related problems. Clearly, when the errors have bounded supports, conditions (C3b) and (C3c) are automatically satisfied. For errors with unbounded supports, Gaussian errors and several sub-Gaussian errors also meet these conditions. An example is a variable whose characteristic function takes the form: $\varphi(t) \propto \exp(-\gamma t^2)\psi_b(t)$, where γ is a positive constant and $\psi_b(t)$ is a polynomial function of t with order $b > 0$. In particular, a variable with the characteristic function $\varphi(t) = \exp(-t^2/2)(1 - \alpha t^2 + \beta t^4)$ for $\alpha \geq \sqrt{2\beta}$ is classified as a strictly sub-Gaussian variable, as defined in Proposition 5.1 in Bobkov, Chistyakov, and Götze (2024). Condition (C4) is a bounded condition for the function class $\mathcal{G}$ and the true function g^*, which has been used in Shen (2020), Lu et al. (2021), Chen et al. (2022), and Yarotsky (2017). Notably, in cases where the explicit depiction of the approximation error for the function class $\mathcal{G}$ to g^* becomes necessary, an additional condition is introduced concerning the category to which g^* belongs. This is demonstrated in Corollary 1.

We define $\mathcal{G}|_{\boldsymbol{x}} := \{g(\boldsymbol{x}_1), g(\boldsymbol{x}_2), \ldots, g(\boldsymbol{x}_n) : g \in \mathcal{G}\}$ for a given sequence $\boldsymbol{x} = (\boldsymbol{x}_1, \ldots, \boldsymbol{x}_n)$ and denote $\mathcal{N}_{2n}(\delta, \|\cdot\|_\infty, \mathcal{G}|_{\boldsymbol{x}})$ as the covering number of $\mathcal{G}|_{\boldsymbol{x}}$ under the norm $\|\cdot\|_\infty$ with radius δ. Let $A \preceq B$ represent $A \leq cB$ for a positive constant c. In the following Theorems 1 and 2, we show the excess risk of the proposed estimator under the true and estimated density function coupled with rescaling techniques, respectively, to see the difference between the proposed estimator and the true MLE estimator, which is defined as

$$\hat{g}_{oracle} = \arg\min_{g \in \mathcal{G}} \left\{ \frac{1}{n} \sum_{i=1}^{n} \left(-\log f(Y_i - g(\boldsymbol{X}_i)) \right) \right\}. \quad (8)$$

Theorem 1. Under Conditions (C3a), (C3b), and (C4), we have that, as $n \to \infty$,

$$\mathbb{E}\left(\mathcal{R}(\hat{g}_{oracle}) - \mathcal{R}(g^*)\right) \preceq n^{-1} \log n \log \mathcal{N}_{2n}(n^{-1}, \|\cdot\|_\infty, \mathcal{G}|_{\boldsymbol{x}}) + \left(\mathcal{R}(g^*_{\mathcal{G}}) - \mathcal{R}(g^*)\right),$$

where the two terms on the right side represent estimation error and approximation error, respectively.

Note that the excess risk of the LS estimator takes the form $n^{-1}\mathcal{B}^2 \log \mathcal{N}_{2n}(n^{-1}, \|\cdot\|_\infty, \mathcal{G}|_{\boldsymbol{x}})(\log n)^c + \left(\mathcal{R}(g^*_{\mathcal{G}}) - \mathcal{R}(g^*)\right)$ for some positive constant c with the condition of bounded response (Györfi et al. 2002; Farrell, Liang, and Misra 2021) or bounded pth moment (Kohler and Langer 2021; Kohler, Krzyzak, and Langer 2022). For the robust loss considered in Shen et al. (2021), the excess risk has the form of $n^{-(1-1/p)}\lambda_L \mathcal{B} \log \mathcal{N}_{2n}(n^{-1}, \|\cdot\|_\infty, \mathcal{G}|_{\boldsymbol{x}})(\log n)^c + \left(\mathcal{R}(g^*_{\mathcal{G}}) - \mathcal{R}(g^*)\right)$ with the condition of the bounded pth moment, and λ_L represents the Lipschitz coefficient of the robust loss function. Additionally, our estimator converges faster than the robust estimators with $(\log n)^c/n^{1-1/p}$ for robust estimators versus a reduced rate of $\log n/n$ for our estimator in estimation error. It is important to highlight that, unlike the requirement of a Lipschitz condition for the robust loss, we instead invoke a lower bound condition (C3) for the density function. The introduction of a lower bound to the density function seems to be helpful to the stability of our estimator. Moreover, by leveraging the inherent benefits of the density function, our proposed estimator exemplifies a harmonious blend of robustness and efficiency that is crucial for practical applications.

Theorem 2. For the proposed estimator $\hat{g}$, under conditions (C1), (C2), (C3a), (C3c), and (C4), we have

$$\begin{aligned}\mathbb{E}\left(\mathcal{R}(\hat{g}) - \mathcal{R}(g^*)\right) &\preceq n^{-1} \log n \log \mathcal{N}_{2n}(n^{-1}, \|\cdot\|_\infty, \mathcal{G}|_{\boldsymbol{x}}) \\ &\quad + \left(\mathcal{R}(g^*_{\mathcal{G}}) - \mathcal{R}(g^*)\right) \\ &\quad + \left(\|g^*_{\mathcal{G}} - g^*\|^2_\infty + h^2\right).\end{aligned}$$

The proofs of Theorems 1 and 2 are given in supplementary materials S.2. Theorems 1 and 2 show that the upper bounds of the excess risk for both $\hat{g}_{oracle}$ and $\hat{g}$ encompass two terms: $n^{-1} \log n \log \mathcal{N}_{2n}(n^{-1}, \|\cdot\|\infty, \mathcal{G}|_{\boldsymbol{x}})$ and $\mathcal{R}(g^*_{\mathcal{G}}) - \mathcal{R}(g^*)$, which represent the estimation error of $\hat{g}$ evaluated at the true density function f with mean zero and the approximate bias of the FNN space toward the true function g^*, respectively. The disparity in excess risks between $\hat{g}_{oracle}$ and $\hat{g}$ is encapsulated in $\|g^*_{\mathcal{G}} - g^*\|^2_\infty + h^2$, which describes the error introduced by substituting f with its kernel estimator $\hat{f}$ and the error due to the rescaling for identification. The error implies that using the kernel estimator $\hat{f}$ in lieu of f and the rescaling technique do not introduce additional variance. However, when using a larger value of h, it introduces significant approximation bias. Therefore, it is advisable to opt for a smaller h to reduce this bias. Particularly the error from $\hat{f}$ in lieu of f is ignorable if $h^2 \preceq n^{-1} \log n \log \mathcal{N}_{2n}(n^{-1}, \|\cdot\|_\infty, \mathcal{G}|_{\boldsymbol{x}})$ and the FNN function closely approximates the true function g^*. The former can be satisfied by taking a small h, and the latter is assured by the powerful function fitting ability of the FNN. The simulation studies in Section 4 further validate this conclusion. In practical applications, the conditions on h including condition (C2) and the above requirement $h^2 \preceq n^{-1} \log n \log \mathcal{N}_{2n}(n^{-1}, \|\cdot\|_\infty, \mathcal{G}|_{\boldsymbol{x}})$ can only provide guidance for selecting a small h such that $h^2 \leq Cn^{-1} \log(n) \log \mathcal{N}_{2n}$ for some constant C.

Now, we further explore how the excess risk relies on the FNN structure and the function class to which g^* belongs. Denote $\lceil a \rceil$ and $\lfloor a \rfloor$ to be the smallest integer no less than a and the largest integer strictly smaller than a, respectively. Let $\mathbb{N}^+$ be the set of positive integers and $\mathbb{N}_0$ be the set of nonnegative integers. Let $\beta = s + r$, $r \in (0, 1]$ and $s = \lfloor \beta \rfloor \in \mathbb{N}_0$. For a finite constant $B_0 > 0$, the *Hölder* class $\mathcal{H}_\beta([0,1]^d, B_0)$ is defined as

$$\begin{aligned}\mathcal{H}_\beta([0,1]^d, B_0) = \{g : [0,1]^d \mapsto \mathbb{R}, &\max_{\|\alpha\|_1 < s} \|\partial^\alpha g\|_\infty \\ &\leq B_0, \max_{\|\alpha\|_1 = s} \sup_{x \neq y} \frac{|\partial^\alpha g(x) - \partial^\alpha g(y)|}{\|x - y\|_2^r} \leq B_0\}\end{aligned}$$

where $\partial^\alpha = \partial^{\alpha_1} \cdots \partial^{\alpha_d}$ with $\alpha = (\alpha_1, \ldots, \alpha_d)^\top \in \mathbb{N}_0^d$ and $\|\alpha\|_1 = \sum_{i=1}^d \alpha_i$.

Based on Theorem 3.3 of Jiao et al. (2023) for the approximation error in terms of FNN structures and Theorem 3 and 7 of Bartlett et al. (2019) for the bounding covering number, we can conclude the following Corollary 1 from Theorem 2:

Corollary 1. Given Hölder smooth functions $g^* \in \mathcal{H}_\beta([0,1]^d, B_0)$, for any $D \in \mathbb{N}^+$, $W \in \mathbb{N}^+$, under conditions of Theorem 2, Theorem 3.3 in Jiao et al. (2023) and Theorem 3 and 7 in Bartlett et al. (2019), if the FNN with a ReLU activation function has width $\mathcal{W} = c(\lfloor\beta\rfloor + 1)^2 d^{\lfloor\beta\rfloor+1} W \lceil \log_2(8W) \rceil$ and depth $\mathcal{D} = c(\lfloor\beta\rfloor + 1)^2 D \lceil \log_2(8D) \rceil$, then

$$\mathbb{E}\left(\mathcal{R}(\hat{g}) - \mathcal{R}(g^*)\right) \preceq n^{-1}\mathcal{S}\mathcal{D}\log(\mathcal{S}) + (WD)^{-4\beta/d} + h^2.$$

To facilitate reading, Theorem 3.3 of Jiao et al. (2023) and Theorems 3 and 7 of Bartlett et al. (2019) are also shown in Lemmas S.1 and S.2 in the supplementary materials S.3. In Corollary 1, the first term comes from the covering number of $\mathcal{G}$, which is bounded by its VC dimension $\mathcal{N}_{2n}(n^{-1}, \|\cdot\|_\infty, \mathcal{G}|_{\boldsymbol{x}}) = O(\mathcal{S}\mathcal{D}\log(\mathcal{S}))$ (Bartlett et al. 2019), where $\mathcal{S}$ and $\mathcal{D}$ are the total number of parameters and hidden layers, respectively. The second term follows from the approximation results from Jiao et al. (2023) that $\|g^* - g^*_{\mathcal{G}}\|_\infty \le 18B_0(\lfloor\beta\rfloor + 1)^2 d^{\lfloor\beta\rfloor+\max\{\beta,1\}/2}(WD)^{-2\beta/d}$ and $\mathbb{E}(\mathcal{R}(g^*_{\mathcal{G}}) - \mathcal{R}(g^*)) \simeq \|g^*_{\mathcal{G}} - g^*\|^2_\infty$, where $A \simeq B$ represents $A \preceq B$ and $B \preceq A$. The third term comes from the approximation error of the kernel estimator $\hat{f}$ to f.

If given $\mathcal{S} = \mathcal{O}(n^{\frac{d}{2\beta+d}}\log n)$ and $\mathcal{D} = \log n$, following Corollary 1 and $\mathcal{S} = O(\mathcal{W}^2\mathcal{D})$, it holds that $\mathbb{E}\left(\mathcal{R}(\hat{g}) - \mathcal{R}(g^*)\right) \preceq n^{-\frac{2\beta}{2\beta+d}}(\log n)^3 + n^{-\frac{2\beta}{2\beta+d}} + h^2$. Hence, we have the following Corollary 2.

Corollary 2. Under the conditions in Corollary 1, if $h^2 = O(n^{-\frac{2\beta}{2\beta+d}})$, then

$$\mathbb{E}\left(\mathcal{R}(\hat{g}) - \mathcal{R}(g^*)\right) \preceq n^{-\frac{2\beta}{2\beta+d}}(\log n)^3,$$

which is comparable to the lower bound $n^{-\frac{2\beta}{2\beta+d}}$ of the minimax learning rate of g^*, that is, $\min_{\check{g}} \max_{g^* \in \mathcal{H}_\beta([0,1]^d, B_0)} \mathbb{E}\left[\int_{[0,1]^d} (\check{g}(\boldsymbol{x}) - g^*(\boldsymbol{x}))^2 f_{\boldsymbol{x}}(\boldsymbol{x}) d\boldsymbol{x}\right] \succeq n^{-\frac{2\beta}{2\beta+d}}$ (Stone 1982), where $\check{g}$ is an estimator of g^* based on the dataset S and the expectation is taken with respect to the randomness of S.

Using the equality that $\mathbb{E}(\mathcal{R}(g_1) - \mathcal{R}(g_2)) \simeq \|g_1 - g_2\|^2_\infty$, we have $\|\hat{g} - g^*\|_\infty \to 0$ from Corollary 2, which shows the consistency of the proposed estimator. It is interesting to compare several established convergence rates under the framework of FNN. Particularly, by using the LS loss for $g \in \mathcal{H}_\beta([0,1]^d, B_0)$, Chen et al. (2022), Nakada and Imaizumi (2020), Schmidt-Hieber (2020), Jiao et al. (2023), Liu, Boukai, and Shang (2022), Bhattacharya, Fan, and Mukherjee (2023), and Yan and Yao (2023) obtained the upper bound of the minimax learning rate of g at $n^{-\frac{2\beta}{2\beta+d}}(\log n)^s$, which is nearly minimax optimal (Donoho et al. 1995; Stone 1982). By using a Lipschitz continuous loss function, Farrell, Liang, and Misra (2021) obtained the convergence rate at $n^{-\frac{2\beta}{2\beta+d}}(\log n)^4$ under a bounded response condition; Shen et al. (2021) obtained the convergence rate $n^{-\frac{2\beta}{2\beta+d}+1/p}(\log n)^c$ under the assumption of the bounded pth moment for some $p > 1$ to allow a heavy-tailed response Y. The severity of the heavy tail decreases as p increases. In particular, if the response Y is subexponentially distributed, $p = \infty$, the convergence rate achieves the minimax near-optimal rate. Obviously, the proposed EML-FNN also enjoys a nearly minimax optimal rate under conditions (C3a) and (C3c) with bounded moments for the density and its derivatives to avoid tail-related problems.

It seems that to achieve the optimal convergence rate, a bounded condition on the tail probability may be necessary. In fact, a similar condition also appears for a quantile regression model. Under the assumption that the conditional density of Y given $\boldsymbol{X}$ around the τth quantile is bounded by below, Padilla, Tansey, and Chen (2022) obtained the convergence rate $n^{-\frac{2\beta}{2\beta+d}}(\log n)^2$ for a quantile regression model under the framework of FNN.

With the discussion above, we next investigate the efficiency of the proposed estimator. It is worthy of noting that the nearly minimax optimal rate indicates that the learning rate of the proposed estimator is nearly minimax optimal in terms of the order. Nevertheless, despite two estimators achieving the nearly minimax optimal rate, there still exist variance disparities between them. The estimator with a smaller variance is more efficient.

For classical models with finite parameters, we refer to an estimator as efficient if it is unbiased and has the lowest variance among all unbiased estimators (Rao 1962). However, when the number of parameters is infinite, the Fisher information matrix (FIM), that is,

$$J := \mathbb{E}\left[\left(\frac{\partial \log f(Y - g(\boldsymbol{X};\boldsymbol{\theta}))}{\partial \boldsymbol{\theta}} - \mathbb{E}\left(\frac{\partial \log f(Y - g(\boldsymbol{X};\boldsymbol{\theta}))}{\partial \boldsymbol{\theta}}\right)\right)^{\otimes 2}\right]|_{\boldsymbol{\theta}=\boldsymbol{\theta}^*},$$

is singular. According to Stoica and Marzetta (2001), when J is singular, all unbiased estimator for $g(x)$ through $g(x;\boldsymbol{\theta})$ must have infinite variance, except under unusual circumstances where certain linear combinations of $\boldsymbol{\theta}$ are to be estimated with the requirement that the rows of the linear combiner matrix belong to the range space of J. Therefore, the classical definition of efficiency is inadequate for $g(x)$ under the framework of FNN $g(x;\boldsymbol{\theta})$, which has the form of

$$g(\boldsymbol{x};\boldsymbol{\theta}) = \boldsymbol{W}_{\mathcal{D}}^\top \sigma\left(\boldsymbol{W}_{\mathcal{D}-1}^\top \sigma(\boldsymbol{W}_{\mathcal{D}-2}^\top \sigma(\boldsymbol{W}_{\mathcal{D}-3} \cdots \sigma(\boldsymbol{W}_0^\top \boldsymbol{x} + \boldsymbol{a}_0)) + \boldsymbol{a}_{\mathcal{D}-2}) + \boldsymbol{a}_{\mathcal{D}-1}\right) + a_{\mathcal{D}},$$

where $\sigma(\cdot)$ is a given activation function and $\boldsymbol{\theta} = \{\boldsymbol{W}_r, \boldsymbol{a}_r, r = 0, \ldots, \mathcal{D}\}$ are parameters.

Focusing on the estimators that achieve nearly minimax optimal rate, we define an FNN-based estimator $\hat{g}(\boldsymbol{x}) = g(\boldsymbol{x};\hat{\boldsymbol{\theta}})$ as FNN-efficient if it is asymptotically unbiased, achieves nearly minimax optimal rate, and exhibits the lowest variance among all asymptotically unbiased FNN estimators $g(\boldsymbol{x};\check{\boldsymbol{\theta}})$ that attain nearly minimax optimal rate, where the asymptotic unbiasedness is characterized by $E\{g(\boldsymbol{x};\check{\boldsymbol{\theta}})\} - g(\boldsymbol{x}) = o(1)$. Clearly, the current FNN-based estimators are typically asymptotically unbiased and approach nearly minimax optimal rates, except for certain robust estimators (Shen et al. 2021) which prioritize robustness over efficiency, but can still attain nearly minimax optimal rates if the response variable Y is subexponentially distributed. With the definition above, we can derive that

Proposition 1. For any asymptotically unbiased FNN based estimator $\breve{g} := g(\boldsymbol{x}; \breve{\boldsymbol{\theta}})$ with nearly minimax optimal rate, it holds that

$$\operatorname{var}(\breve{g}) \geq \operatorname{var}(\hat{g}) > 0.$$

The proof of Proposition 1 is given in supplementary materials S.2.4. Combining with Corollary 2, the Proposition 1 indicates the proposed EML-FNN is FNN-efficient.

4. Simulation Study

We investigate the performance of the proposed EML-FNN (simplified as EML) by comparing it with the least square estimator (LSE) and several robust methods with g approximated by the FNN. We consider four commonly used robust methods: The Least absolute deviation method (LAD) with the loss function $\rho(x) = |x|$; the Huber method with the loss function $\rho(x) = 0.5x^2 I(|x| \leq \zeta) + (\zeta\,|x| - \zeta^2/2) I(|x| > \zeta)$ at $\zeta = 1.345$; the Cauchy method with the loss function $\rho(x) = \log\{1 + \kappa^2 x^2\}$ at $\kappa = 1$; and Tukey's biweight method with the loss function $\rho(x) = t^2[1 - \{1 - (x/t)^2\}^3] I(|x| \leq t)/6 + t^2 I(|x| > t)/6$ at $t = 4.685$. We also investigate the effect of bandwidth on our method in Section 4.3.

Feedforward network architectures, initial values, and data for training are the same for all methods involved. The computations were implemented via the packages PyTorch (Paszke et al. 2019) and Scikit-learn (Pedregosa et al. 2011) in Python. Specifically, we use the network *Net-d5-w256* with ReLU-activated functions, which comprises 3 hidden layers, resulting in a network depth of 5 with the corresponding network widths $(d, 256, 256, 256, 1)$. We use the Adam optimization algorithm (Kingma and Ba 2014) with a learning rate of 0.0003 for network parameters initialized by uniform distribution (He et al. 2015). The coefficients used for calculating the running average of the gradient and the squared gradient are $\beta = (0.9, 0.99)$. When the training dataset is small, we set the training batch size to match the dataset size (n). However, with larger training datasets, such as $n \geq 10{,}000$, we decrease the batch size to a level sufficient for reasonably estimating a density function, such as setting it to 500. We train the network for at least 1000 epochs using a dropout rate of 0.01 until the training loss converges or reaches a satisfactory level.

To enhance flexibility and simplicity, we adopt a varying bandwidth $h(\epsilon_i) = |\max(\epsilon_i(\nu)) - \min(\epsilon_i(\nu))|$, where the set $\epsilon_i(\nu)$ is the neighborhood of ϵ_i encompassing a proportion ν of the total sample (Loftsgaarden and Quesenberry 1965). Then, the selection of the bandwidth is translated into selecting a value for ν from the interval $(0, 1]$. The constrained interval simplifies the process of bandwidth selection. In practical computations, we first estimate g by using a robust method such as Cauchy method. Then we calculate $h(\epsilon_i)$ by $h(\tilde{\epsilon}_i)$ based on the estimated g, $\tilde{g}$, through $\tilde{\epsilon}_i = Y_i - \tilde{g}(\boldsymbol{X}_i)$. To reduce computational load, we fixed and no longer update $h(\tilde{\epsilon}_i)$ throughout the subsequent computation. Here, we choose the Cauchy method for $\tilde{g}$ to prevent $\tilde{\epsilon}_i$ from concentrating around 0, which may happen when we use the least squares method due to the neural network's ability to overfit to training data. In addition, we also check whether the selected bandwith satisfy Condition (C2) and $h^2 \lesssim n^{-1}\log(n)\log\mathcal{N}_{2n}$. For example, in the following numerical studies for $n = 1024$, the maximum bandwidth used is $h = 1.75$, while $n^{-1}\log(n)\log\mathcal{N}_{2N} \leq n^{-1}\mathcal{SD}\log(\mathcal{S}) \leq n^{-1}\mathcal{WD}^2\log(\mathcal{WD}) = 19.42$. Comparing with $n = 1024$, which is regarded as large enough, the conditions $h \to 0$, $nh \to \infty$ and $h^2 \leq Cn^{-1}\log(n)\log\mathcal{N}_{2N}$ are approximately met when $h = 1.75$.

We evaluate the performance of $\hat{g}$ by the bias, standard deviation (SD) and root mean square error (RMSE) and prediction error (PE), defined as bias $= \left[\frac{1}{n_{grid}}\sum_{i=1}^{n_{grid}}(E\widehat{g}(z_i) - g(z_i))^2\right]^{\frac{1}{2}}$, $SD = \left[\frac{1}{n_{grid}}\sum_{i=1}^{n_{grid}} E(\widehat{g}(z_i) - E\widehat{g}(z_i))^2\right]^{\frac{1}{2}}$, $RMSE = \sqrt{\text{bias}^2 + SD^2}$, and $PE = \frac{1}{t}\sum_{i=1}^{t}\left\|g(X_i^{test}) - Y_i^{test}\right\|^2$, where $z_i (i = 1, \ldots, n_{grid})$ are grid points on which $g(\cdot)$ is evaluated, which are first randomly generated from the distribution of $\boldsymbol{X}$ and then fixed, $n_{grid} = 2048$, and $E\widehat{g}(z_i)$ is approximated by its sample mean based on 100 replications, $\left\{X_i^{test}, Y_i^{test}\right\}_{i=1}^{t}$ represents the test data, which shares the same distribution as the training data.

4.1. Data Generation

Let us write $\boldsymbol{X}_i = (X_{i1}, \ldots, X_{id})^\top$ with each component of $\boldsymbol{X}_i$ being iid generated from a uniform distribution $U(0, 1)$. We consider three target functions: (a) $g_5(\boldsymbol{X}_i) = x_{i1}^3 + x_{i2}^2 + x_{i3} + |x_{i4}| + \cos(x_{i5})$; (b) $g_{10}(\boldsymbol{X}_i) = x_{i1}^3 + x_{i2}^2 + x_{i3} + |x_{i4}| + \cos(x_{i5}) + \sin(x_{i6}) + e^{x_{i7}} + \log(1 + x_{i8}) + x_{i9}^{\frac{1}{2}} + x_{i10}^{\frac{1}{3}}$; and (c) $g_{20}(\boldsymbol{X}_i) = x_{i1}^5 + x_{i2}^4 + x_{i3}^3 + x_{i4}^2 + x_{i5} + |x_{i6}| + x_{i7}^{\frac{1}{2}} + x_{i8}^{\frac{1}{3}} + x_{i9}^{\frac{1}{4}} + x_{i10}^{\frac{1}{5}} + \left|x_{i11}^3\right| + \cos(x_{i12}) + \sin(x_{i13}) + \cos(x_{i14}^2) + \sin(x_{i15}^2) + e^{x_{i16}} + \log(1 + x_{i17}) + e^{x_{i18}^2} + \log(1 + x_{i19}^2) + \log(1 + x_{i20}^{\frac{1}{2}})$, which are $p = 5, 10,$ and 20-dimensional functions, respectively, where $x_{ij} = \boldsymbol{X}_i^\top \boldsymbol{\beta}_j, j = 1, \ldots, 20$, $\boldsymbol{\beta}_j$ is a d-dimensional vector with $\boldsymbol{\beta}_j\left[\left((j-1) \times \left\lfloor \frac{d}{p} \right\rfloor + 1\right) : \left(j \times \left\lfloor \frac{d}{p} \right\rfloor\right)\right] = \frac{(\boldsymbol{\gamma}^\top, \ldots, \boldsymbol{\gamma}^\top)}{\left\lfloor \frac{d}{20p} \right\rfloor \times \|\boldsymbol{\gamma}\|_1}$ and the remaining components of $\boldsymbol{\beta}_j$ are 0, $\boldsymbol{\gamma} = (1, 2, \ldots, 20)^\top$. That is, the nonzero elements of $\boldsymbol{\beta}_j$ are integer values ranging from 1 to 20 but scaled according to L_1 norms.

We consider the following four distributions for the error ϵ_i: (I) Standard normal distribution: $\epsilon_i \sim \mathcal{N}(0, 1)$; (II) mixture Gaussian distribution: $\epsilon_i \sim 0.7\mathcal{N}(0, 1) + 0.3\mathcal{N}(0, 5)$; (III) Student-$t$ distribution: $\epsilon_i \sim t(2)$; (IV) Heteroscedasticity: $\epsilon_i \sim \mathcal{N}(0, 3X_{i1} + 4X_{i2})$.

Finally, we generate Y_i by $Y_i = g_p(\boldsymbol{X}_i) + \varepsilon_i$ with $p = 5, 10, 20$. We set $n = 256, 1024, 10{,}000$ and consider $d = 100, 200, 400, 500, 600, 800$ for the above three target functions and four error distributions, respectively. All simulation results, based on 100 replications, are depicted in Figures 1–3, Tables 1, 2 and Tables S1 and S2 in the supplementary material S.4.

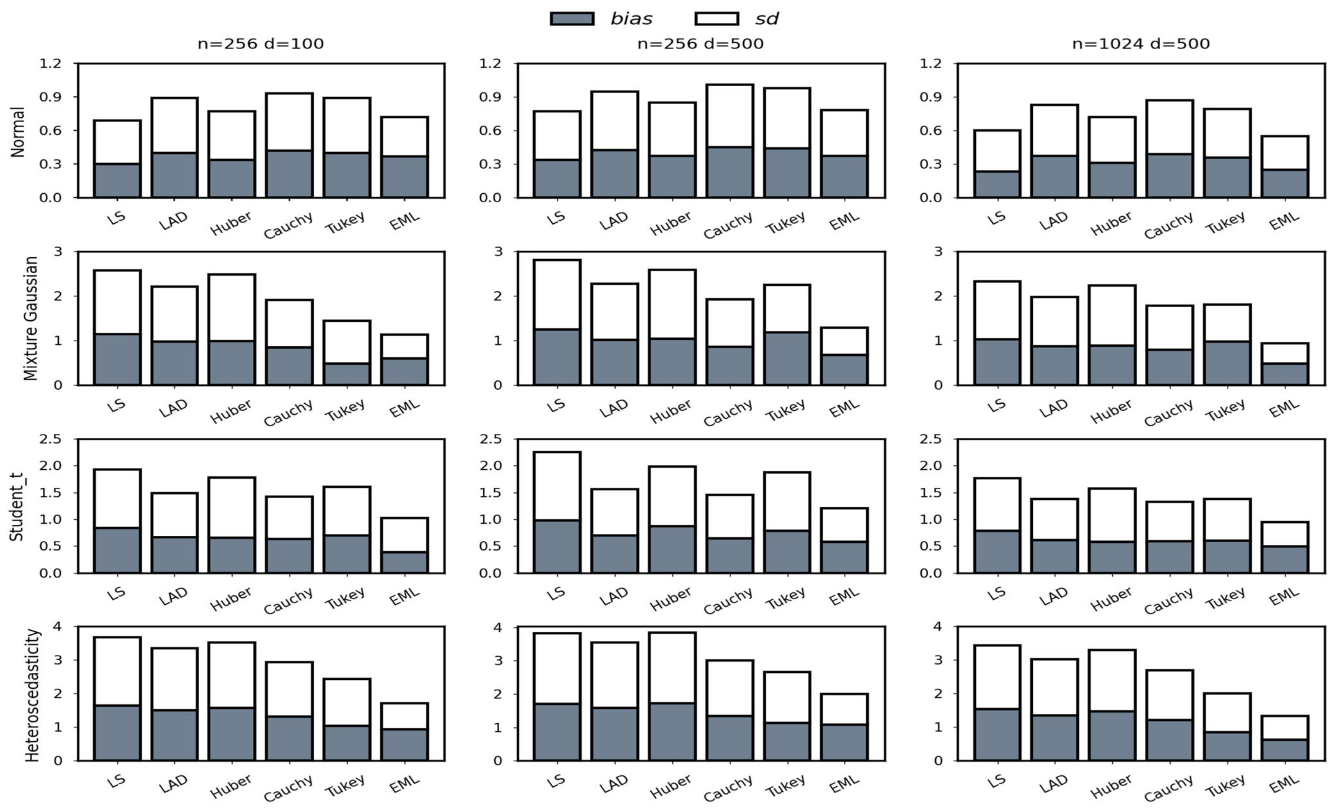

Figure 1. The bar chart of the bias and standard deviation of g_5 when using six methods for four error distributions with sample sizes $n = 256, 1024$ and input dimensions $d = 100, 500$.

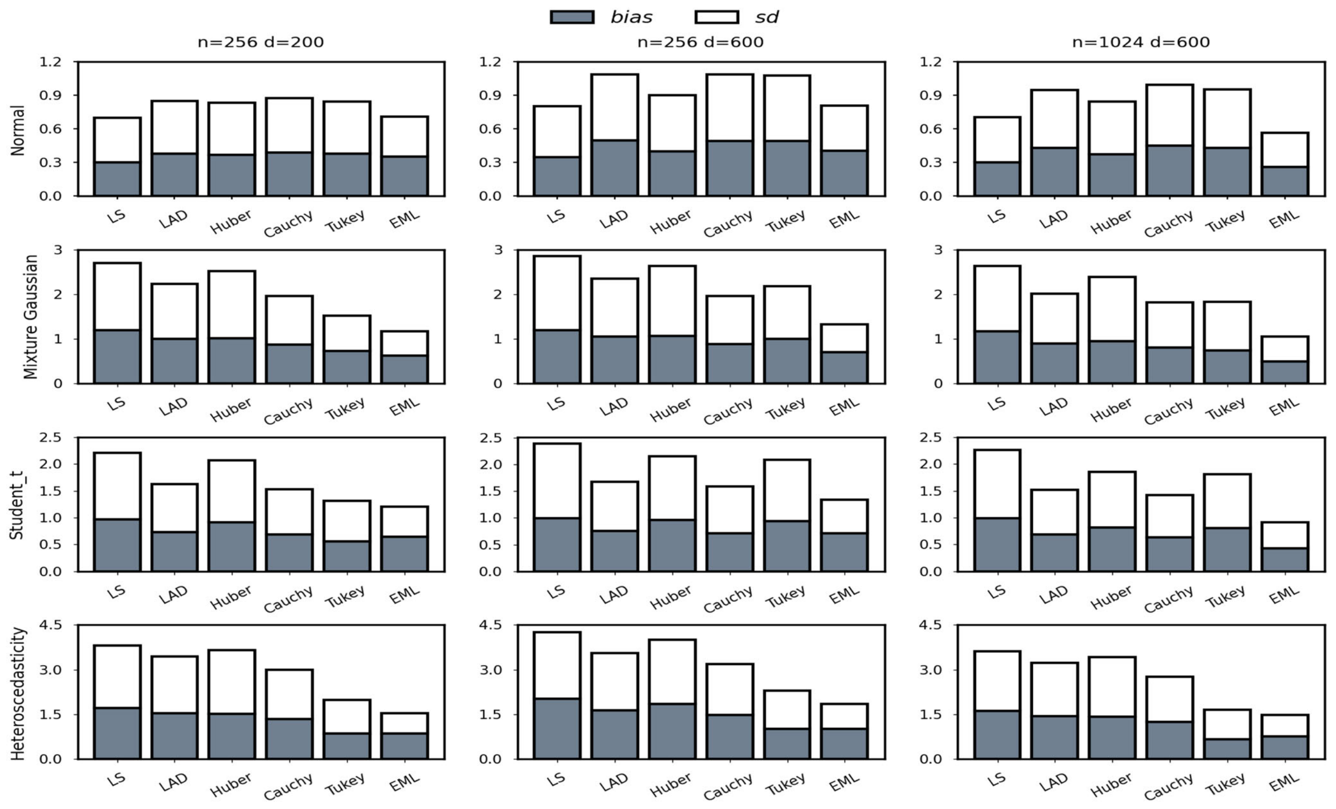

Figure 2. The bar chart of the bias and standard deviation of g_{10} when using six methods for four error distributions with sample sizes $n = 256, 1024$ and input dimensions $d = 200, 600$.

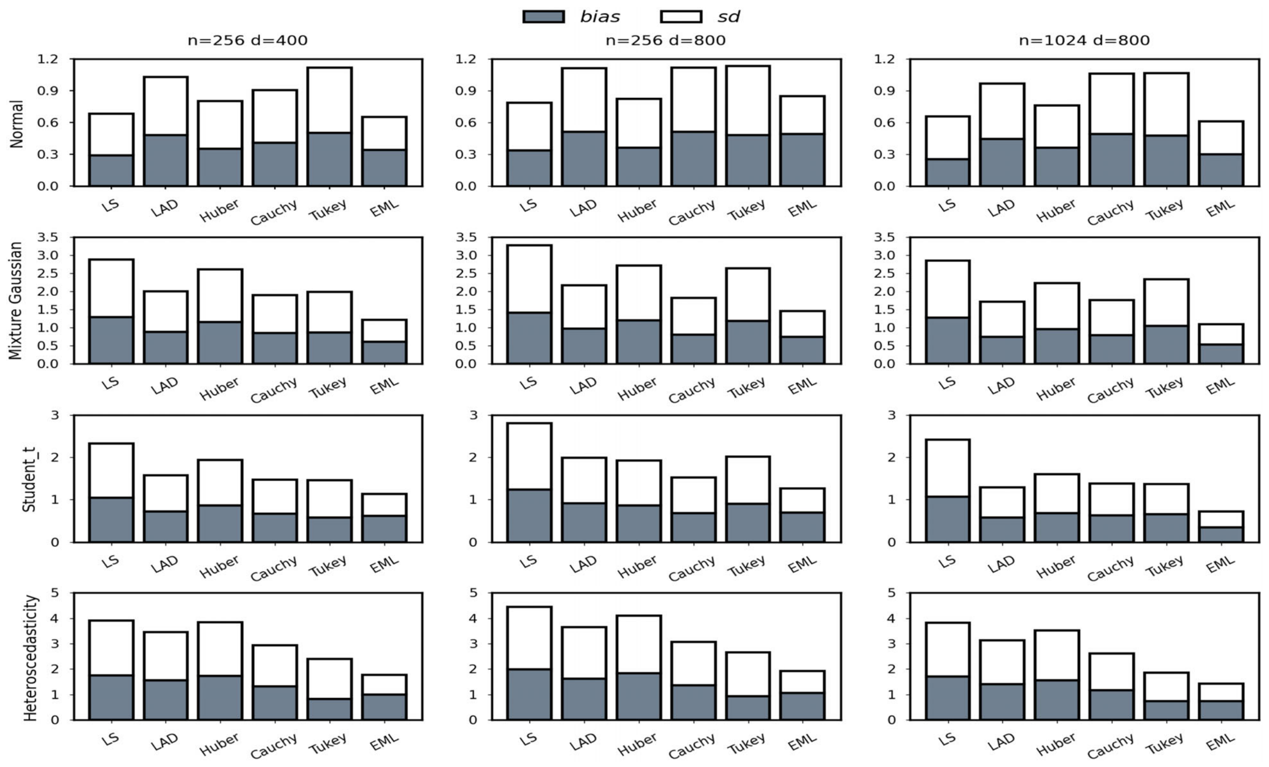

Figure 3. The bar chart of the bias and standard deviation of g_{20} when using six methods for four error distributions with sample sizes $n = 256, 1024$ and input dimensions $d = 400, 800$.

Table 1. The mean and standard deviation of prediction error of g_{20} when using six methods for four error distributions with training sample sizes $n = 256, 1024$, testing sample size $t = 2048$ and input dimensions $d = 400, 800$.

		LS	LAD	Huber	Cauchy	Turkey	EML
				$n = 256, d = 400$			
Normal	PE	1.3398	1.4207	1.3675	1.4079	1.5115	**1.1867**
	SD	0.082	0.1243	0.0859	0.1117	0.1521	**0.0598**
Mixture Gaussian	PE	10.9579	9.7114	10.3718	9.482	10.1371	**8.3613**
	SD	1.2149	0.8584	0.9356	0.7084	0.93	**0.6696**
Student-t	PE	14.9209	14.0095	14.3943	13.9566	13.8578	**12.3858**
	SD	10.5185	9.9556	10.3562	9.56	9.5931	**8.5225**
Heteroscedasticit	PE	19.1514	18.1451	19.0234	17.0328	17.6632	**14.474**
	SD	1.4022	1.1605	1.3683	1.1008	1.1053	**0.8049**
				$n = 256, d = 800$			
Normal	PE	**1.3772**	1.5959	1.376	1.6002	1.6214	1.3887
	SD	0.0758	0.125	0.1063	0.1373	0.1765	**0.0624**
Mixture Gaussian	PE	11.7864	9.8367	10.5045	9.5207	10.3528	**8.4616**
	SD	1.2957	0.8698	0.9859	0.7829	0.9573	**0.6457**
Student-t	PE	17.4688	16.0819	15.6975	15.127	16.1709	**14.0685**
	SD	11.8648	11.9311	11.1637	10.8954	11.8253	**9.8341**
Heteroscedasticit	PE	20.4291	18.4613	19.4358	18.0389	17.855	**14.4865**
	SD	1.5972	1.5221	1.5666	1.3392	1.0813	**0.8017**
				$n = 1024, d = 800$			
Normal	PE	1.2612	1.3264	1.2831	1.3933	1.4477	**1.1363**
	SD	0.0688	0.0787	0.0705	0.1101	0.151	**0.0586**
Mixture Gaussian	PE	10.7343	9.1828	9.5585	9.2162	9.8933	**8.3509**
	SD	1.172	0.7593	0.9106	0.811	0.8216	**0.5373**
Student-t	PE	13.4447	10.538	10.9382	10.782	10.7184	**9.636**
	SD	8.4321	7.6277	8.1895	7.9653	7.9205	**7.0795**
Heteroscedasticit	PE	18.937	17.4363	18.2855	16.4426	16.1865	**14.4336**
	SD	1.31	1.035	1.2081	1.0103	1.0032	**0.6618**

Bold values indicate the best performance.

4.2. Numerical Results

From Figures 1–3, it is evident that the proposed EML consistently and significantly outperforms the robust-based methods in terms of bias, SD, and RMSE across all the simulation scenarios. For nonnormal and heteroscedasticity situations, the robust-based methods perform better than LSE, and the proposed EML significantly outperforms LSE in terms of both bias and

Table 2. The bias, standard deviation, the mean and standard deviation of prediction error of g_{10} and running time when using six methods for heteroscedasticity distributions with sample sizes $n = 10{,}000$, testing sample size $t = 20{,}000$, input dimensions $d = 200$ and 1000 training epochs.

	LS	LAD	Huber	Cauchy	Turkey	EML
Bias	1.2146	0.4941	0.513	0.4384	0.4952	**0.3136**
SD	1.5578	0.5161	0.7237	0.6312	0.6309	**0.335**
PE	22.1131	15.6201	16.7824	15.7653	15.8863	**14.4557**
SD	1.3652	0.3522	0.8916	0.4792	0.4824	**0.1666**
Runtime	129.3189	131.6143	**123.3842**	125.4246	127.6436	139.3014

Bold values indicate the best performance.

SD. When the errors follow the normal distribution, the LSE is optimal. In this case, the proposed EML performs comparably to LSE, and both outperform the robust-based methods. The comparability of the EML and the LSE indicates that the loss caused by estimating the density function can be ignored, which aligns with the theoretical findings in Theorem 2. Upon closer examination across Figures 1–3, we can see that EML even slightly but consistently outperforms LSE for normal data when the sample size is larger, such as when $n = 1024$. This suggests that the ability of the proposed EML to learn the data structure may become more pronounced as the sample size grows. Figures 1–3 also show that the performance of all the methods improves with increasing sample sizes or decreasing dimensions.

In scenarios with large training datasets, our method encounters computational challenges due to the summation in (7) involving n^2 terms. To tackle this issue, we used the Adam optimization algorithm in conjunction with batch methods. An advantage of the Adam algorithm is its adaptive learning rate, which facilitates faster convergence. In addition, we note that the inner summation in $\hat{g} = \arg\min_{g\in\mathcal{G}} n^{-1}\sum_{i=1}^{n}\Big(-\log\frac{1}{n}\sum_{j=1}^{n}\mathcal{K}_h(Y_j - g(\boldsymbol{X}_j), Y_i - g(\boldsymbol{X}_i))\Big)$ is used to estimate the density function at $Y_i - g(\boldsymbol{X}_i)$. Recognizing that the density function is one-dimensional and requiring limited data for estimation, for large training datasets, we can use batch methods with a specified batch size to ensure reasonable density estimation, for instance, setting it to 500. Consequently, for $n = 10{,}000$, the terms in the summation of (8) are reduced to $\frac{n^2}{20} = 20 \times \frac{n}{20} \times \frac{n}{20}$. The simulation results are presented in Table 2. All methods undergo 1000 training epochs on CPUs with equal capacity. Compared to the other method, our approach exhibits a mere 16-sec delay compared to Huber, which has the shortest average training time of 123 sec among all baseline models. Despite this, our approach still yields conclusions similar to those in Figures 1–3, where the batch size equals the training data size.

To clarify, we also run a simulation to evaluate the impact of batch size on stability, accuracy, and computational time. Particularly, using the simulation setting with target function g_5 for Mixture Gaussian noise, sample sizes $n = 1024$, testing sample size $t = 2048$ and input dimensions $d = 100$, we assess the proposed method's performance in terms of computation time, bias, SD, and PE across batch sizes: $16, 32, 64, 128, 256$, and 512. The experimental results in Table 3 demonstrate that our method using smaller batch sizes reduce computation time but often result in significantly higher bias, SD and PE. As the batch size increases, computation time grows, while prediction accuracy initially improves but then stabilizes. Table 3 suggests

Table 3. The bias, SD, PE and its SD, and running time for estimating g_5 using different batch sizes.

	Bacth size					
	16	32	64	128	256	512
Bias	1.3580	1.1974	1.1145	0.2794	0.2796	0.2761
SD	0.8329	0.8455	0.7468	0.3133	0.3156	0.3116
PE	14.6036	13.4594	12.6283	8.3448	8.3596	8.3426
SD	2.8155	2.0983	2.0554	0.6979	0.6953	0.6920
Runtime	4.0351	4.8535	4.8878	6.4537	8.7789	13.7178

Bold values indicate the best performance.

a batch size of 128 offers the best balance between computation time, accuracy, and stability for the simulation setting, yielding lower prediction errors in less time.

4.3. Effect of the Bandwidth

Now, we examine the effect of bandwidth on the proposed method. In Figures 4 and 5, we present the bias, SD, and prediction error (PE) of the EML-FNN estimator when the bandwidths vary from 0.2 to 0.8 for g_5 under four error distributions and $(n, d) = (1024, 500)$. From Figures 4 and 5, we can see that a smaller bandwidth provides a better estimator in terms of bias, SD, and PE, and the proposed EML estimator is robust to variations in bandwidth within a certain range that approaches zero. These findings are consistent with the theoretical result presented in Theorem 2, which indicates that a small bandwidth is favored, and the extra risk is independent of the bandwidth if the bandwidth is appropriately small. Additionally, the comparison of Figures 4 and 5 reveals that the PE is more stable than the bias and SD as the bandwidth changes.

5. Real Data Example

We applied our proposed EML-FNN and other competing methods to analyze four real datasets based on the model (1) when using observations $(\boldsymbol{X}_i, Y_i)_{i=1}^n$.

1. Boston House Price Dataset. It is available in the scikit-learn library (Pedregosa et al. 2011) and encompasses a total of $n = 506$ observations. The purpose of the analysis is to predict the house price based on 13 input variables $\boldsymbol{X}_i$, such as urban crime rates, nitric oxide levels, average number of rooms in a dwelling, weighted distance to central areas, and average owner-occupied house prices. Following Kong and Xia (2012) and Zhou et al. (2019), we employ the logarithm of the median price of owner-occupied residences in units of \$1000 as our response Y_i.
2. QSAR Aquatic Toxicity Dataset. The dataset was provided by Cassotti et al. (2014) and was used to develop quantitative

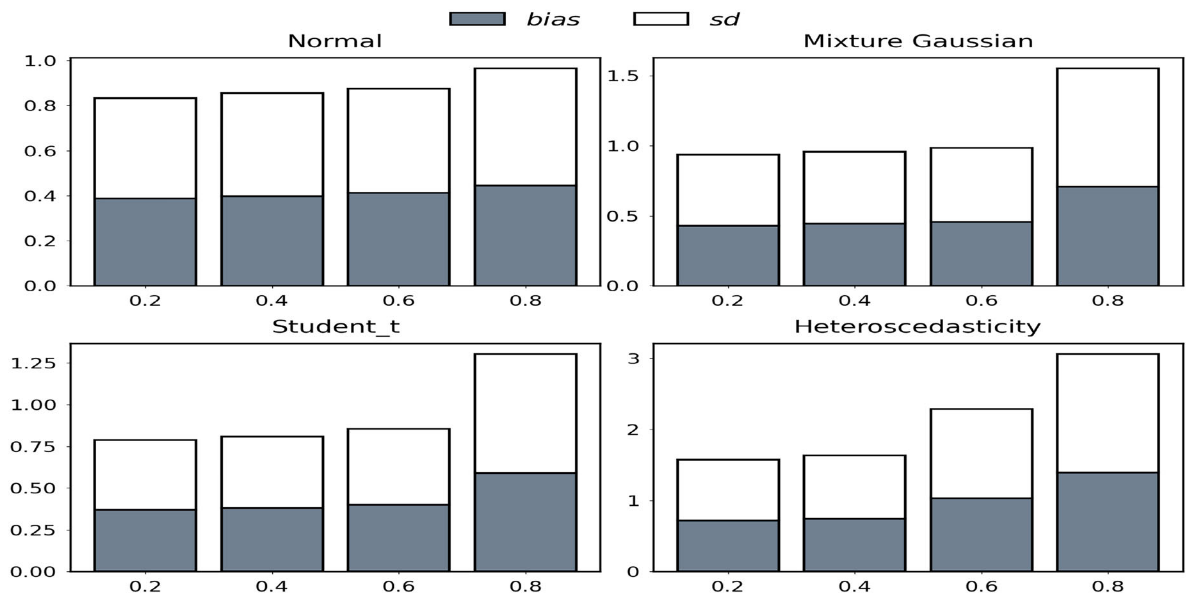

Figure 4. The bar chart of the bias and standard deviation of the proposed EML-FNN for g_5 under four error distributions with $(n, d) = (1024, 500)$ as the bandwidths vary from 0.2 to 0.8.

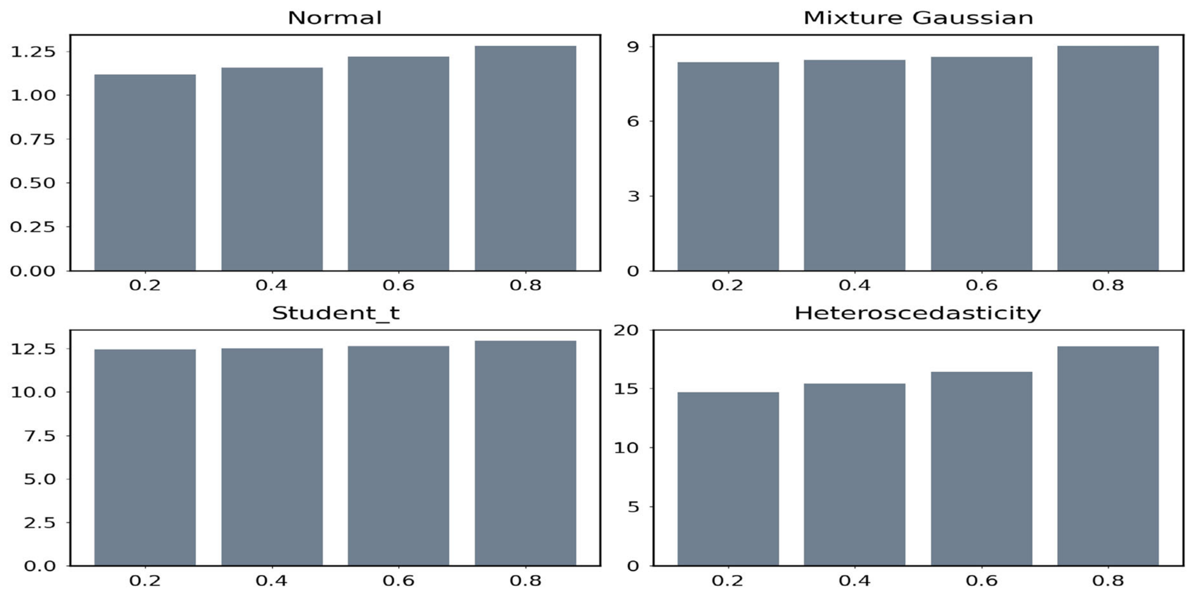

Figure 5. Bar chart of the mean prediction error (PE) of the proposed EML-FNN for g_5 under four error distributions with $(n, d) = (1024, 500)$ as the bandwidths vary from 0.2 to 0.8.

regression QSAR models for predicting acute aquatic toxicity toward Daphnia magna. It consists of a total of $n = 546$ observations, each with eight molecular descriptors serving as covariates $\boldsymbol{X}_i$, including PSA(Tot) (molecular properties), SAacc (molecular properties), H-050 (atom-centered fragments), MLOGP (molecular properties), RDCHI (connectivity indices), GATS1p (2D autocorrelations), nN (constitutional indices), and C-040 (atom-centered fragments). The response variable Y_i is the acute aquatic toxicity, specifically the LC50, which is defined as the concentration causing death in 50% of the test D. magna over a test duration of 48 hr.

3. QSAR Fish Toxicity Dataset. Another version of the dataset for quantitative regression QSAR models was provided by Cassotti et al. (2015). This dataset includes 908 observations, and each observation has six input variables ($\boldsymbol{X}_i$), including molecular descriptors: MLOGP (molecular properties), CIC0 (information indices), GATS1i (2D autocorrelations), NdssC (atom-type counts), NdsCH ((atom-type counts), and SM1_Dz(Z) (2D matrix-based descriptors). The response variable Y_i is the LC50, which is the concentration that causes death in 50% of test fish over a test duration of 96 hours.
4. Temperature Forecast Dataset. The dataset was provided by Cho et al. (2020) and aims to correct the bias of next-day maximum and minimum air temperature forecasts from the LDAPS model operated by the Korea Meteorological Administration over Seoul, South Korea. The data consist of summer data spanning from 2013 to 2017. The input data $\boldsymbol{X}_i$ are largely composed of predictions from the LDAPS model for

Table 4. Mean prediction error for four real datasets.

	Boston (Pedregosa et al. 2011)	Aquatic toxicity (Cassotti et al. 2014)	Fish toxicity (Cassotti et al. 2015)
LS	0.1045	1.2812	2.0153
LAD	0.1054	1.2184	2.142
Huber	0.1155	1.2003	2.1403
Cauchy	0.1192	1.2697	2.1179
Tukey's biweight	0.1153	1.3148	2.1436
EML	**0.0833**	**1.1497**	**1.8918**
	Temperature forecast (Cho et al. 2020)		
LS	3.3478	1.9861	
LAD	4.6311	2.5609	
Huber	4.2441	2.4549	
Cauchy	5.4385	2.7776	
Tukey's biweight	3.2366	1.9603	
EML	**2.1451**	**1.2302**	

Bold values indicate the best performance.

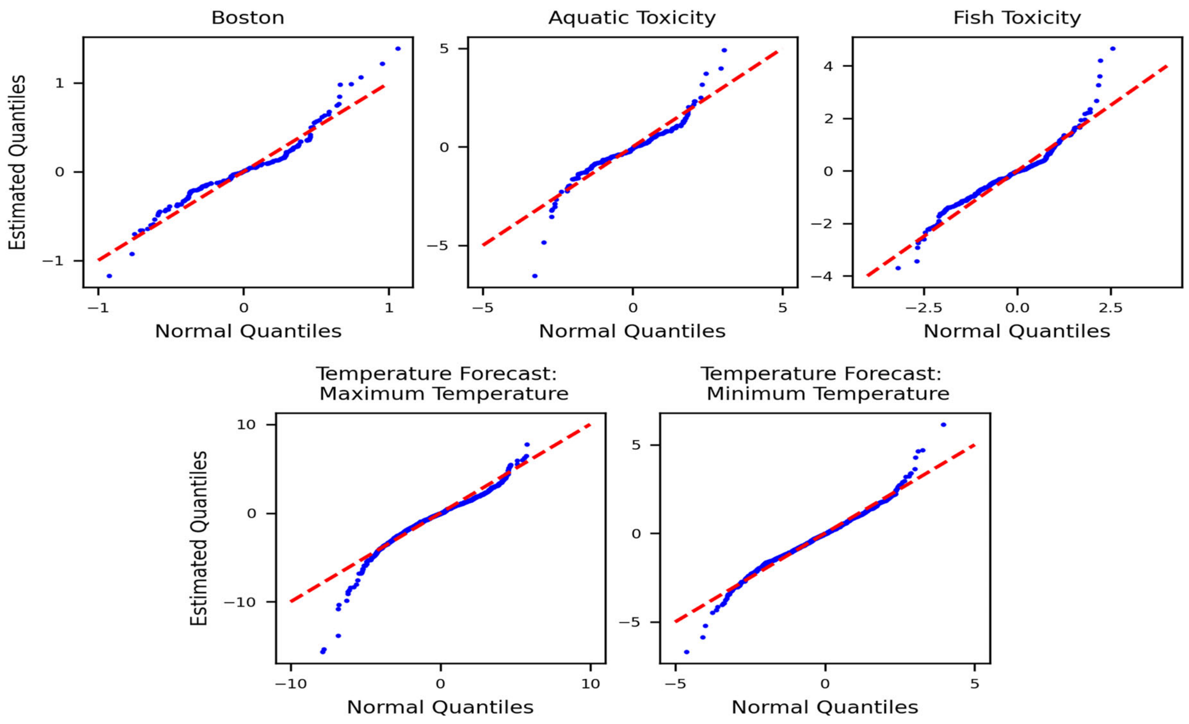

Figure 6. The Q-Q plot of the estimated density function (blue) for four real datasets. Red dotted lines indicate diagonal.

the subsequent day, in-situ records of present-day maximum and minimum temperatures, and five geographic auxiliary variables. In this dataset, two outputs (Y_i) are featured: next-day maximum and minimum air temperatures.

We preprocessed all the datasets by applying Z-score normalization to each predictor variable. Inspired by transfer learning, we employed the fine-tuning technique to simplify the computation. We initiated the process by training a single network model based on, for example, the Cauchy loss function by employing the methodology outlined in Section 4. Subsequently, we leveraged this trained model as a foundation to train all other models with a learning rate of 0.00003. All four datasets were randomly split into training and test sets at a ratio of 4:1 to calculate PE. The entire procedure was repeated 50 times, and the average PEs were calculated and are presented in Table 4. The results in Table 4 clearly demonstrate the superiority of our approach over other competing methods in terms of a remarkable improvement in prediction accuracy across all four datasets. Particularly noteworthy is the outstanding performance achieved when applying our EML technique to the Temperature Forecast dataset, where the improvement of the prediction accuracy achieves up to 50%. To understand the underlying reasons behind these improvements, we proceeded to plot the Q-Q plot in Figure 6 on the estimated error distribution for all four real datasets. The Q-Q plot indicates that the Boston House Prices closely approximate a normal distribution. In contrast, the Toxicity data deviate from normality, mainly due to a few outliers. Furthermore, the Temperature data exhibit a substantial departure from the normal distribution, marked by a notable prevalence of extreme values. Based on these findings, we can conclude that the prediction performances, as illustrated in Table 4, are linked to the degree to which the respective distributions adhere to normality; in particular, all the methods exhibit enhanced predictive accuracy when handling datasets that are more similar to a normal distribution. This observation highlights the influence of distribution characteristics on the resulting estimator and emphasizes the importance of incorporating distribution information into the analysis.

6. Concluding Remarks

In the article, we present an innovative approach to estimate nonparametric regression under the framework of FNN. This approach can be characterized by its efficiency in both estimation and in computation, its adaptability to diverse data distributions, and its robustness in the presence of noise and outliers. The main advantages are as follows. (a) Estimation efficiency: The method introduces a novel loss function that incorporates not only observed data but also potentially implicit information about the data structure. By organizing this hidden structural information, the loss function transforms into an estimated maximum likelihood function, resulting in desirable properties such as efficiency. (b) Distribution-free: The method is distribution-independent. Therefore, it adeptly handles data with diverse distribution patterns, including heavy tails, multimodal distributions, and heterogeneity. (c) Probabilistic Robustness: The loss function is formulated through a probabilistic framework, which effectively mitigates the influence of large noise and outliers, thereby improving its robustness. (d) Kernel-Based Smoothness: The method leverages the inherent smoothness of kernel functions, allowing for gradient calculations and addressing challenges related to nondifferentiability in scenarios involving densities such as uniform distributions, mixture distributions, and cases of heteroscedasticity. (e) Computational efficiency: The proposed loss function involves only the regression function g, making it easy to use with existing software packages and simplifying both computational and programming requirements. In summary, the ability of the method to handle diverse data distributions with efficient estimation and simple computation makes it versatile and applicable to a wide range of real-world scenarios. For example, reducing the requirement on the data size due to its efficiency. In fact, the need for large training samples is a point that DNN is often criticized. The proposed method alleviates this problem to some extent.

By using a reasonably small bandwidth, the proposed estimator is proven to be equivalent to the maximum likelihood estimator (MLE), where the density function is known. Furthermore, it nearly attains the minimax optimal rate, with only an additional logarithmic factor. Its exceptional performance is further exemplified through comprehensive simulation studies and its successful application to four distinct real-world datasets.

There are several directions for future research. First, it might be possible to extend our method to a more complicated model, such as the generalized regression model for a discrete response. Second, practical scenarios often involve multiple responses that exhibit correlations, as seen in the maximum and minimum air temperature forecast dataset. By further modeling interresponse correlations, predictive capabilities may be enhanced. Third, simulation studies indicate that the proposed method seems to be superefficient. It is valuable to establish the theoretical foundations for investigating the existence of this high level of efficiency. Last, it remains our responsibility to consistently enhance the associated software packages, ensuring seamless application. Despite having introduced an efficient and user-friendly package named EML-FNN, continued optimization and refinement are necessary.

Supplementary Materials

In the supplementary materials, we provide detailed proofs of Theorems 1–2 and Proposition 1, along with some results from the simulation studies.

Disclosure Statement

The authors report there are no competing interests to declare.

Funding

The research was supported by National Key R&D Program of China (No.2022YFA1003702), National Natural Science Foundation of China (Nos. 11931014 and 12271441), and New Cornerstone Science Foundation.

References

Bartlett, P. L., Harvey, N., Liaw, C., and Mehrabian, A. (2019), "Nearly-Tight VC-dimension and Pseudodimension Bounds for Piecewise Linear Neural Networks," *The Journal of Machine Learning Research*, 20, 2285–2301. [1309]

Bassett Jr, G., and Koenker, R. (1978), "Asymptotic Theory of Least Absolute Error Regression," *Journal of the American Statistical Association*, 73, 618–622. [1306]

Bauer, B., and Kohler, M. (2019), "On Deep Learning as a Remedy for the Curse of Dimensionality in Nonparametric Regression," *The Annals of Statistics*, 47, 2261–2285. [1306]

Beaton, A. E., and Tukey, J. W. (1974), "The Fitting of Power Series, Meaning Polynomials, Illustrated on Band-Spectroscopic Data," *Technometrics*, 16, 147–185. [1306]

Berlinet, A., and Thomas-Agnan, C. (2011), *Reproducing Kernel Hilbert Spaces in Probability and Statistics*, Dordrecht: Springer. [1305]

Bhattacharya, S., Fan, J., and Mukherjee, D. (2023), "Deep Neural Networks for Nonparametric Interaction Models with Diverging Dimension," arXiv preprint arXiv:2302.05851. [1306,1309]

Bobkov, S., Chistyakov, G., and Götze, F. (2024), "Strictly Subgaussian Probability Distributions," *Electronic Journal of Probability*, 29, 1–28. [1308]

Cassotti, M., Ballabio, D., Consonni, V., Mauri, A., Tetko, I. V., and Todeschini, R. (2014), "Prediction of Acute Aquatic Toxicity Toward Daphnia Magna by Using the GA-kNN Method," *Alternatives to Laboratory Animals*, 42, 31–41. [1313,1315]

Cassotti, M., Ballabio, D., Todeschini, R., and Consonni, V. (2015), "A Similarity-based QSAR Model for Predicting Acute Toxicity Towards the Fathead Minnow (*Pimephales promelas*)," *SAR and QSAR in Environmental Research*, 26, 217–243. [1314,1315]

Chen, M., Jiang, H., Liao, W., and Zhao, T. (2022), "Nonparametric Regression on Low-Dimensional Manifolds Using Deep ReLU Networks: Function Approximation and Statistical Recovery," *Information and Inference: A Journal of the IMA*, 11, 1203–1253. [1306,1308,1309]

Cho, D., Yoo, C., Im, J., and Cha, D.-H. (2020), "Comparative Assessment of Various Machine Learning-based Bias Correction Methods for Numerical Weather Prediction Model Forecasts of Extreme Air Temperatures in Urban Areas," *Earth and Space Science*, 7, e2019EA000740. [1314,1315]

Donoho, D. L., Johnstone, I. M., Kerkyacharian, G., and Picard, D. (1995), "Wavelet Shrinkage: Asymptopia?" *Journal of the Royal Statistical Society*, Series B, 57, 301–337. [1309]

Fan, J., and Gijbels, I. (2018), *Local Polynomial Modelling and its Applications*, New York: Routledge. [1305,1307]

Fan, J., and Gu, Y. (2023), "Factor Augmented Sparse Throughput Deep ReLU Neural Networks for High Dimensional Regression," *Journal of the American Statistical Association*, 1–15. [1306]

Fan, J., Gu, Y., and Zhou, W.-X. (2022), "How Do Noise Tails Impact on Deep ReLU Networks?" arXiv preprint arXiv:2203.10418. [1306]

Farrell, M. H., Liang, T., and Misra, S. (2021), "Deep Neural Networks for Estimation and Inference," *Econometrica*, 89, 181–213. [1305,1306,1308,1309]

Györfi, L., Kohler, M., Krzyzak, A., and Walk, H. (2002), *A Distribution-Free Theory of Nonparametric Regression* (Vol. 1), New York: Springer. [1305,1308]

He, K., Zhang, X., Ren, S., and Sun, J. (2015), "Delving Deep into Rectifiers: Surpassing Human-Level Performance on Imagenet Classification," in *2015 IEEE International Conference on Computer Vision (ICCV)*, pp. 1026–1034. [1310]

Huber, P. J. (1973), "Robust Regression: Asymptotics, Conjectures and Monte Carlo," *The Annals of Statistics*, 1, 799–821. [1306]

Jiao, Y., Shen, G., Lin, Y., and Huang, J. (2023), "Deep Nonparametric Regression on Approximate Manifolds: Nonasymptotic Error Bounds with Polynomial Prefactors," *The Annals of Statistics*, 51, 691–716. [1306,1309]

Kingma, D. P., and Ba, J. (2014), "Adam: A Method for Stochastic Optimization," arXiv preprint arXiv:1412.6980. [1310]

Kohler, M., Krzyzak, A., and Langer, S. (2022), "Estimation of a Function of Low Local Dimensionality by Deep Neural Networks," *IEEE Transactions on Information Theory*, 68, 4032–4042. [1306,1308]

Kohler, M., and Langer, S. (2021), "On the Rate of Convergence of Fully Connected Deep Neural Network Regression Estimates," *The Annals of Statistics*, 49, 2231–2249. [1306,1308]

Kong, E., and Xia, Y. (2012), "A Single-Index Quantile Regression Model and its Estimation," *Econometric Theory*, 28, 730–768. [1313]

Lederer, J. (2020), "Risk Bounds for Robust Deep Learning," arXiv preprint arXiv:2009.06202. [1306]

Liu, R., Boukai, B., and Shang, Z. (2022), "Optimal Nonparametric Inference via Deep Neural Network," *Journal of Mathematical Analysis and Applications*, 505, 125561. [1306,1309]

Loftsgaarden, D. O., and Quesenberry, C. P. (1965), "A Nonparametric Estimate of a Multivariate Density Function," *The Annals of Mathematical Statistics*, 36, 1049–1051. [1310]

Lu, J., Shen, Z., Yang, H., and Zhang, S. (2021), "Deep Network Approximation for Smooth Functions," *SIAM Journal on Mathematical Analysis*, 53, 5465–5506. [1308]

Lv, S., Lin, H., Lian, H., and Huang, J. (2018), "Oracle Inequalities for Sparse Additive Quantile Regression in Reproducing Kernel Hilbert Space," *The Annals of Statistics*, 46, 781–813. [1305]

Nakada, R., and Imaizumi, M. (2020), "Adaptive Approximation and Generalization of Deep Neural Network with Intrinsic Dimensionality," *Journal of Machine Learning Research*, 21, 1–38. [1309]

Padilla, O. H. M., Tansey, W., and Chen, Y. (2022), "Quantile Regression with ReLU Networks: Estimators and Minimax Rates," *Journal of Machine Learning Research*, 23, 1–42. [1309]

Paszke, A., Gross, S., Massa, F., Lerer, A., Bradbury, J., Chanan, G., Killeen, T., Lin, Z., Gimelshein, N., Antiga, L., et al. (2019), "Pytorch: An Imperative Style, High-Performance Deep Learning Library," in *Advances in Neural Information Processing Systems* (Vol. 32). [1306,1307,1310]

Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M., and Duchesnay, E. (2011), "Scikit-learn: Machine Learning in Python," *Journal of Machine Learning Research*, 12, 2825–2830. [1306,1307,1310,1313,1315]

Rao, C. R. (1962), "Efficient Estimates and Optimum Inference Procedures in Large Samples," *Journal of the Royal Statistical Society*, Series B, 24, 46–63. [1309]

Schmidt-Hieber, J. (2019), "Deep ReLU Network Approximation of Functions on a Manifold," arXiv preprint arXiv:1908.00695. [1306]

——— (2020), "Nonparametric Regression Using Deep Neural Networks with ReLU Activation Function," *The Annals of Statistics*, 48, 1875–1897. [1306,1309]

Schumaker, L. (2007), *Spline Functions: Basic Theory*, Cambridge: Cambridge University Press. [1305]

Shen, G., Jiao, Y., Lin, Y., and Huang, J. (2021), "Robust Nonparametric Regression with Deep Neural Networks," arXiv preprint arXiv:2107.10343. [1306,1308,1309,1310]

Shen, Z. (2020), "Deep Network Approximation Characterized by Number of Neurons," *Communications in Computational Physics*, 28, 1768–1811. [1308]

Stoica, P., and Marzetta, T. L. (2001), "Parameter Estimation Problems with Singular Information Matrices," *IEEE Transactions on Signal Processing*, 49, 87–90. [1309]

Stone, C. J. (1982), "Optimal Global Rates of Convergence for Nonparametric Regression," *The Annals of Statistics*, 10, 1040–1053. [1309]

Yan, S., and Yao, F. (2023), "Nonparametric Regression for Repeated Measurements with Deep Neural Networks," arXiv preprint arXiv:2302.13908. [1306,1309]

Yarotsky, D. (2017), "Error Bounds for Approximations with Deep ReLU Networks," *Neural Networks*, 94, 103–114. [1308]

Zhou, L., Lin, H., Chen, K., and Liang, H. (2019), "Efficient Estimation and Computation of Parameters and Nonparametric Functions in Generalized Semi/non-parametric Regression Models," *Journal of Econometrics*, 213, 593–607. [1306,1313]

Zhou, L., Lin, H., and Liang, H. (2018), "Efficient Estimation of the Nonparametric Mean and Covariance Functions for Longitudinal and Sparse Functional Data," *Journal of the American Statistical Association*, 113, 1550–1564. [1306]