Statistics Innovation

ARTICLE

Open Access

https://doi.org/10.48130/stati-0026-0008

Statistics Innovation **2026**, 3: e008

Feature selection with annealing for shallow neural networks using the multistage stochastic algorithm

Xinglei Li[1#], Chenlu Zhang[1#], Lizhe Sun[1*], Zhufeng Meng[1] and He Zhang[2]

[1] *School of Statistics, Shanxi University of Finance and Economics, Taiyuan, Shanxi 030006, China*

[2] *Beijing International Center for Mathematical Research, Peking University, Beijing 100871, China*

[#] Authors contributed equally: Xinglei Li, Chenlu Zhang

[*] Correspondence: lsun@sxufe.edu.cn (Sun L)

Abstract

Feature selection is a fundamental challenge in statistics and machine learning, playing a critical role in improving prediction accuracy and enhancing model interpretability. It is widely applied across various scientific and practical domains. However, most existing variable selection methods are developed within the framework of linear and generalized linear models, which limit their applicability to more complex data structures. In this article, we introduce two novel nonlinear variable selection approaches that integrate an iterative hard-thresholding operator into stochastic training of shallow neural networks. The proposed methods help to simultaneously identify important features and build a predictive model based on the selected subset using shallow neural networks. Furthermore, we propose a new Bayesian information criterion that can facilitate effective model selection in the high-dimensional setting. Extensive numerical experiments on both simulated and real-world data sets demonstrated that the proposed methods show remarkable performance compared with the classical approaches in the literature.

Citation: Li X, Zhang C, Sun L, Meng Z, Zhang H. 2026. Feature selection with annealing for shallow neural networks using the multistage stochastic algorithm. *Statistics Innovation* 3: e008 https://doi.org/10.48130/stati-0026-0008

Introduction

Feature selection is a fundamental topic of research in statistics and machine learning, as it improves predictive accuracy, enhances interpretability, and reduces computational costs by eliminating irrelevant or redundant variables. Consequently, it has been widely used in real-world data analysis across various application domains.

Several methods for variable selection have been proposed for both parametric and nonparametric models. The classical techniques include the ℓ_0-penalized approach[1,2], the ℓ_1-penalized method[3–5], as well as the SCAD[6] and MCP[7] methods. In addition, group-penalized methods, such as group Lasso[8] and adaptive group Lasso[9], have been proposed for variable selection. Although these methods are widely applied in practice, they have primarily been developed for high-dimensional linear and generalized linear models. Extensions to high-dimensional varying-coefficient models[10] and sparse additive models[11] have also been explored, which are, however, less applicable to neural networks.

Being a fundamental component of deep learning, neural networks have undergone significant theoretical and empirical advances, and can now be used to approximate complex data structures and tackle intricate modeling tasks[12–16]. However, neural networks are often over-parameterized, posing challenges for training, prediction, and interpretation. Several methods have been developed to reduce the number of model parameters and compress network architectures in order to enhance the predictive performance. A widely adopted solution is dropout[17], which randomly drops nodes during training to prevent overfitting. Subsequently, sparse neural networks have been proposed to reduce the number of parameters and compress model architectures. The existing research on sparse neural networks falls into two categories: Bayesian and frequentist. Within the Bayesian framework, various sparsity-inducing priors, such as the hierarchical prior[18,19], the mixture Gaussian prior[20], and the spike-and-slab prior[21], have been proposed for learning sparse Bayesian neural networks. In the frequentist framework, the group ℓ_1-regularized methods[22–25] have been proposed to learn sparse neural networks. In addition, node pruning techniques[26,27] have been introduced to compress neural network architectures.

Although the existing methods encourage sparsity in the weight matrix and compress the network architectures, few methods directly address the sparse-input problem. However, these methods are not specifically designed for variable selection. To identify the important covariates in neural network models, several nonlinear variable selection techniques have been proposed. First, the group ℓ_1 penalty is applied to induce sparsity into the input weight matrix[28]. Next, an additional selection layer is introduced, and both ℓ_0[29] and ℓ_1[30] penalties are used for variable selection. More recently, a group ℓ_0-regularized method has been proposed for variable selection[31], using the iterative hard-thresholding (IHT) algorithm to estimate the parameters[32]. Furthermore, the concept of group ℓ_0 penalty has been extended to nonparametric expectile regression within the framework of deep neural networks[33].

However, the current neural network-based variable selection methods face challenges in both variable selection accuracy and predictive performance. First, these ℓ_1-penalized neural network approaches rely heavily on tuning parameters, leading to instability in both variable selection and prediction. Moreover, using the ℓ_1 penalty may incur unwanted bias and lead to inconsistency in variable selection[5,34]. Therefore, the ℓ_0 penalty may be an ideal choice, as it does not incur any bias[2].

Just like the ℓ_1 penalty is associated with the soft-thresholding operator, the ℓ_0 penalty corresponds to the hard-thresholding operator[35]. Accordingly, the IHT algorithm[32], which combines gradient descent with a hard-thresholding operator, has been proposed to minimize ℓ_0-penalized loss functions. Although this gradient-based approach is computationally efficient and easy to implement, it may converge to the local optima when applied to neural networks.

© The Author(s)

Moreover, ℓ_0- and ℓ_1-regularized methods require users to pre-specify the number of selected variables k and the regularization parameter λ.

In comparison with the IHT algorithm, which is based on deterministic learning[31], we propose the multistage stochastic iterative hard-thresholding (StoIHT) algorithm. The proposed methods, which are based on stochastic learning algorithms, are used to select important variables by estimating the weights connecting the input and the first hidden layer. Then, we develop a novel variable selection method with an annealing strategy within the framework of a multistage stochastic algorithm[36]. This approach can help in substantially enhancing both variable selection and prediction accuracy. To simplify the discussion and provide a clear description of our methodology, we will focus on shallow neural networks in this article. Furthermore, we investigate the criteria for variable selection and develop a novel Bayesian information criterion (BIC) tailored to the high-dimensional setting (the large-p–small-n scenario), along with the conventional BIC for the low-dimensional setting ($n > p$ scenario)[37]. The main contributions of this paper are as follows:

(1) This article proposes two novel variable selection methods that use neural networks as surrogate models with a group ℓ_0 penalty. To enhance the selection accuracy, we introduce an annealing strategy and develop a multistage stochastic optimization framework to identify key variables. The performance and effectiveness of the proposed methods are demonstrated through numerical experiments and real-data applications.

(2) This paper develops a novel BIC for model selection in the high-dimensional setting. Numerical experiments demonstrate that the proposed criterion helps to substantially improve variable selection accuracy relative to the conventional BIC.

The remainder of this paper is organized as follows. The next section introduces the notation and outlines the model framework. The section "*Methodology*" presents the proposed variable selection approaches based on a stochastic optimization strategy. The section "*Simulation*" assesses the performance of the proposed methods through numerical experiments and real data applications. The last section provides the conclusions and discusses the potential directions for future research.

Setting and notation

Given a collection of independent training instances $\{(\mathbf{x}_i, y_i)\}_{i=1}^n$, where $\mathbf{x}_i \in \mathbb{R}^{p_0}$ represents a sample of input features and $y_i \in \mathbb{R}$ denotes the corresponding output, we consider the following nonparametric regression model:

$$y_i = f^*(\mathbf{x}_i) + \epsilon_i, \quad \epsilon_i \sim \mathcal{N}(0, \sigma^2), \tag{1}$$

where, f^* represents the unknown true model relating the input $\mathbf{x}_i$ to the output y_i, and ϵ_i is the random noise term assumed to have a Gaussian distribution with mean zero and variance σ^2. Suppose that p_0 is large and f^* in Eq. (1) only depends on a subset of the input features, which are referred to as important features. In other words, there exists an unknown index set $S^* \subset \{1, 2, \ldots, p_0\}$ with $|S^*| = k^* \ll p_0$, where k^* denotes the number of important features. Moreover, we can extend the previous setting to the nonparametric logistic regression model:

$$\log\left(\frac{\mathbb{P}(y_i = 1 \mid \mathbf{x}_i)}{\mathbb{P}(y_i = 0 \mid \mathbf{x}_i)}\right) = f^*(\mathbf{x}_i), \tag{2}$$

where, f^* represents the unknown relationship between the input $\mathbf{x}_i$ and the log-odds of the binary outcome $y_i \in \{0, 1\}$. In the literature of neural networks[14,15,31], f^* is often assumed to belong to the (d, C)-smoothness function class, which is presented in Definition 2.1.

Definition 2.1. Let $d = q + \zeta$, where $\zeta \in (0, 1]$ and $q = \lfloor d \rfloor \in \mathbb{N}_0$ denotes the largest integer strictly smaller than d ($\mathbb{N}_0$ denotes the set of nonnegative integers). A function $f : \mathbb{R}^{p_0} \to \mathbb{R}$ is called (d, C)-smooth function if for every $\boldsymbol{\alpha} = (\alpha_1, \alpha_2, \cdots, \alpha_{p_0})^\top \in \mathbb{N}_0^{p_0}$ with $\sum_{j=1}^{p_0} \alpha_j = q$, the partial derivative $\partial^q f / \partial x_1^{\alpha_1} \partial x_2^{\alpha_2} \cdots \partial x_{p_0}^{\alpha_{p_0}}$ exists and satisfies

$$\left| \frac{\partial^q f}{\partial x_1^{\alpha_1} \partial x_2^{\alpha_2} \cdots \partial x_{p_0}^{\alpha_{p_0}}}(\mathbf{x}_1) - \frac{\partial^q f}{\partial x_1^{\alpha_1} \partial x_2^{\alpha_2} \cdots \partial x_{p_0}^{\alpha_{p_0}}}(\mathbf{x}_2) \right| \le C \|\mathbf{x}_1 - \mathbf{x}_2\|_2^{\zeta},$$

for all $\mathbf{x}_1, \mathbf{x}_2 \in \mathbb{R}^{p_0}$, where $\|\cdot\|_2$ denotes the ℓ_2-norm, otherwise known as the Euclidean norm.

In this paper, we aim to estimate the index set S^* from the given training data and to simultaneously approximate the underlying function f^*. To address the feature selection and model learning problems, we employ a one-hidden-layer neural network as a surrogate to approximate the unknown true function f^*. We now introduce the class of one-hidden-layer neural networks used throughout this paper. Let $\mathcal{H}_{p_1,\varphi}$ denote the class of one-hidden-layer neural networks with width p_1 and activation function φ:

$$\mathcal{H}_{p_1,\varphi} = \left\{ \boldsymbol{\omega}_2^\top \varphi(\mathbf{W}_1 \mathbf{x} + \mathbf{b}_1) + b_2 : b_2 \in \mathbb{R}, \boldsymbol{\omega}_2, \mathbf{b}_1 \in \mathbb{R}^{p_1}, \mathbf{W}_1 \in \mathbb{R}^{p_1 \times p_0} \right\}.$$

Therefore, models (1) and (2) can be approximated as

$$y_i \approx \boldsymbol{\omega}_2{}^\top \varphi(\mathbf{W}_1 \mathbf{x}_i + \mathbf{b}_1) + b_2 + \epsilon_i,$$

and

$$\log\left(\frac{\mathbb{P}(y_i = 1 \mid \mathbf{x}_i)}{\mathbb{P}(y_i = 0 \mid \mathbf{x}_i)}\right) \approx \boldsymbol{\omega}_2{}^\top \varphi(\mathbf{W}_1 \mathbf{x}_i + \mathbf{b}_1) + b_2,$$

respectively, where $\mathbf{W}_1 \in \mathbb{R}^{p_1 \times p_0}$ is the weight matrix connecting the input layer to the first hidden layer, and $\mathbf{b}_1 \in \mathbb{R}^{p_1}$ is the bias vector. The vector $\boldsymbol{\omega}_2 \in \mathbb{R}^{p_1}$ represents the weights from the hidden layer to the output layer, with a bias term $b_2 \in \mathbb{R}$. The activation function $\varphi : \mathbb{R}^{p_1} \to \mathbb{R}^{p_1}$ is coordinate-wise and piecewise differentiable function. We use the $ReLU$ function $\varphi(x) = \max\{x, 0\}$ in this article. Let $\ell(y_i, f(\mathbf{x}_i))$ be the loss function for each training instance $(\mathbf{x}_i, y_i)$, where $f(\mathbf{x}_i) = \boldsymbol{\omega}_2^\top \varphi(\mathbf{W}_1 \mathbf{x}_i + \mathbf{b}_1) + b_2$ represents the one-hidden-layer neural network model. Hence, the empirical loss function based on n samples is

$$L(\mathbf{W}_1, \mathbf{b}_1, \boldsymbol{\omega}_2, b_2) = \frac{1}{n} \sum_{i=1}^{n} \ell\left(y_i, f(\mathbf{x}_i; \mathbf{W}_1, \mathbf{b}_1, \boldsymbol{\omega}_2, b_2)\right).$$

We propose the following optimization problem with a sparsity constraint on the weight matrix to address the feature selection problem:

$$\min_{\mathbf{W}_1, \mathbf{b}_1, \boldsymbol{\omega}_2, b_2} \frac{1}{n} \sum_{i=1}^{n} \ell(y_i, f(\mathbf{x}_i; \mathbf{W}_1, \mathbf{b}_1, \boldsymbol{\omega}_2, b_2)), \quad \|\mathbf{W}_1\|_{2,0} \le k, \tag{3}$$

where, $\|\mathbf{W}_1\|_{2,0} = \#\{j : \|\mathbf{w}_{\cdot j}\|_2 \neq 0, j = 1, 2, \cdots, p_0\}$ and $\mathbf{w}_{\cdot j}$ denotes the j-th column of the weight matrix $\mathbf{W}_1$. Therefore, $\|\mathbf{W}_1\|_{2,0}$ corresponds to the number of nonzero column vectors of $\mathbf{W}_1$. The index set of the nonzero columns of $\mathbf{W}_1$ is denoted by S.

It is worth noting that the group ℓ_0 penalty is similar to conventional group penalty methods[8]: if predictor j is unimportant, all entries in the vector $\mathbf{w}_{\cdot j}$ are zero; otherwise, all entries in the vector $\mathbf{w}_{\cdot j}$ are nonzero. Figure 1 shows the architecture of the one-hidden-layer neural network model derived using the group ℓ_0 penalty. This architecture can be naturally extended to deep neural networks, as the sparsity constraint is independent of the hidden layers. Since neural networks are interpreted as surrogate models approximating the unknown model f^*, assuming that f^* belongs to the (d, C)-smooth function class, we can pre-select the neural

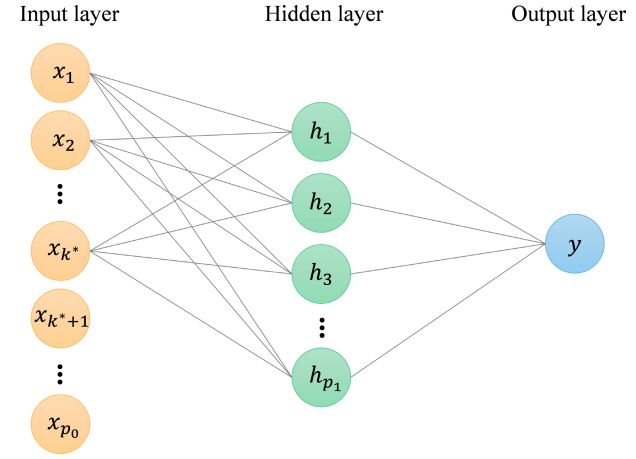

Fig. 1 The architecture of a one-hidden-layer neural network derived using the group ℓ_0 penalty is illustrated. For simplicity, the bias terms are omitted. The figure shows that there are no edges between the input nodes $x_{k^*+1}, \cdots, x_{p_0}$ and the neurons in the hidden layer, indicating that the (k^*+1)-th to p_0-th column vectors of $\mathbf{W}_1$ are zeros. This sparsity pattern is induced by the group ℓ_0 penalty.

network architecture to achieve a desired approximation error. We then formulate the feature selection problem as shown by Eq. (3). In other words, p_1, which denotes the number of neurons in the first hidden layer, is fixed by solving the feature selection problem in Eq. (3).

We now delineate the notations used throughout the paper. Vectors are denoted by lowercase bold letters, such as $\mathbf{x} \in \mathbb{R}^d$, and scalars by lowercase letters, for example, $x \in \mathbb{R}$. Subscripts index a sequence of vectors, for example $\mathbf{w}_1, \mathbf{w}_2, \ldots$. The components of a vector are indicated by nonbold subscripts, such as w_j. Matrices are represented by uppercase bold letters, for example, $\mathbf{M} \in \mathbb{R}^{d \times d}$, while random variables are denoted by uppercase letters, such as Z. For a vector $\boldsymbol{\gamma} = (\gamma_1, \gamma_2, \cdots, \gamma_n)^\top \in \mathbb{R}^n$, we use the following norms: $\|\boldsymbol{\gamma}\|_1 = \sum_{i=1}^n |\gamma_i|$, $\|\boldsymbol{\gamma}\|_2 = \sqrt{\sum_{i=1}^n \gamma_i^2}$, and $\|\boldsymbol{\gamma}\|_0 = \#\{i : \gamma_i \neq 0\}$.

Methodology

In the section "Setting and notation," we introduce the feature selection problem and derive the corresponding constrained optimization problem (Eq. [3]) using a one-hidden-layer neural network as a surrogate model. In this section, we propose two variable selection methods within a multistage stochastic learning framework to solve the constrained optimization problem (Eq. [3]). To mitigate the risk of gradient descent converging to local optima, we first extend the IHT algorithm[31,32] to a multistage StoIHT algorithm[38] to train a one-hidden-layer neural network. Next, we develop a variant of StoIHT that uses an annealing strategy for variable selection, known as feature selection with annealing (FSA). Compared with the existing IHT method, the proposed multistage StoIHT and multistage FSA methods show a superior overall performance on regression and classification tasks, as validated by numerical experiments.

Multistage stochastic IHT

We begin by introducing the multistage stochastic learning framework. In the stochastic algorithm, each iteration operates on only a mini-batch of the data set, dividing the training process into distinct stages, each comprising several iterations. This staged training structure enables the use of different learning rates and other hyperparameters across stages. The output of the current training stage is used as the initial value for the subsequent stage. In neural network training, a stage can naturally correspond to a single epoch.

Then, we introduce the StoIHT algorithm within a multistage framework. The algorithm consists of N_{iter} stages. In each stage, the stochastic gradient descent (SGD) algorithm updates model parameters using only a mini-batch of data per iteration, thereby introducing randomness into the parameter updates. Consequently, applying a hard-thresholding operator immediately after each iteration may yield poor variable selection performance[39]. To address this issue, we apply the hard-thresholding operator after all the SGD iterations within a stage are complete, thereby stabilizing the variable selection procedure. Recall that the hard-thresholding operator selects the top-k column vectors of the weight matrix $\mathbf{W}_1$ based on the ℓ_2-norms and sets the rest of the column vectors to zeros. At the end of each stage, the resulting weight matrix, after implementing a hard-thresholding operator, is consistent with the constraint in the optimization problem (3). All parameter estimates from the shallow neural network model at each stage are treated as the initial values for the next stage. In other words, the multistage StoIHT algorithm stabilizes the variable selection procedure by using the hard-thresholding output of the current stage as a "warm start" for stochastic learning in the next stage. The multistage StoIHT algorithm is summarized in Algorithm 1. When $N_{iter} = 1$, the multistage StoIHT algorithm is equivalent to the SGD with truncation algorithm[39].

Multistage FSA

In this section, we present the proposed multistage FSA method. As the true variables may not reveal their significance early in an iterative algorithm, the ℓ_2-norm of their updated parameters $\|\mathbf{w}_{\cdot j}\|_2$ may not rank among the top-k largest values. As a result, these true variables could be excluded by setting their weights to zero, preventing their selection.

To address the abovementioned challenge and enhance the selection accuracy, we propose a variable selection method that

Algorithm 1. Multistage StoIHT algorithm.

Input: Standardized training data $\{(\mathbf{x}_i, y_i)\}_{i=1}^n$, initial learning rate η_0, sparse level k, and total iteration times N_{iter}.

Output: Trained weights $\mathbf{W}_1$, $\mathbf{b}_1$, $\boldsymbol{\omega}_2$ and b_2.

Initialize $\mathbf{W}_1^{(1,1)}$, $\mathbf{b}_1^{(1,1)}$, $\boldsymbol{\omega}_2^{(1,1)}$, $b_2^{(1,1)}$.

for $t = 1$ to N_{iter} **do**

Shuffle the training data $\{(\mathbf{x}_i, y_i)\}_{i=1}^n$, and relabel $i = 1, 2, \cdots, n$.

for $i = 1$ to n **do**

$$\boldsymbol{\omega}_2^{(t,i+1)} \leftarrow \boldsymbol{\omega}_2^{(t,i)} - \eta_t \frac{\partial \ell(y_i, f(\mathbf{x}_i; \cdots))}{\partial \boldsymbol{\omega}_2}, \; b_2^{(t,i+1)} \leftarrow b_2^{(t,i)} - \eta_t \frac{\partial \ell(y_i, f(\mathbf{x}_i; \cdots))}{\partial b_2},$$

$$\mathbf{W}_1^{(t,i+1)} \leftarrow \mathbf{W}_1^{(t,i)} - \eta_t \frac{\partial \ell(y_i, f(\mathbf{x}_i; \cdots))}{\partial \mathbf{W}_1},$$

$$\mathbf{b}_1^{(t,i+1)} \leftarrow \mathbf{b}_1^{(t,i)} - \eta_t \frac{\partial \ell(y_i, f(\mathbf{x}_i; \cdots))}{\partial \mathbf{b}_1},$$

end for

Let $\mathbf{W}_1^{(t)} = \mathbf{W}_1^{(t,n+1)}$. Denote $\mathbf{w}_{\cdot j}^{(t)}$ as the j-th column for $\mathbf{W}_1^{(t)}$.

Ranking $\{\|\mathbf{w}_{\cdot j}^{(t)}\|_2\}$ in descending order, and retaining the index set $\hat{S}^{(t)}$ corresponding to top-k largest values of $\{\|\mathbf{w}_{\cdot j}^{(t)}\|_2\}$.

Keep the $\mathbf{w}_{\cdot j}^{(t)}$ for $j \in \hat{S}^{(t)}$ and set $\mathbf{w}_{\cdot j}^{(t)} = \mathbf{0}$ for all others $j \notin \hat{S}^{(t)}$.

Set $\mathbf{W}_1^{(t+1,1)} = \mathbf{W}_1^{(t)}$, $\mathbf{b}_1^{(t+1,1)} = \mathbf{b}_1^{(t,n+1)}$, $\boldsymbol{\omega}_2^{(t+1,1)} = \boldsymbol{\omega}_2^{(t,n+1)}$, and $b_2^{(t+1,1)} = b_2^{(t,n+1)}$.

end for

incorporates an annealing strategy. After all iterations in a given stage are completed, we retain only those variables whose ℓ_2-norm of updated parameter vectors $\|\mathbf{w}_{\cdot j}^{(t)}\|_2$ ranks among the top-M_t largest values. Variables with smaller $\|\mathbf{w}_{\cdot j}^{(t)}\|_2$ values are considered unimportant and will be removed. Their parameters will not be updated in subsequent iterations. An annealing schedule for M_t has been proposed[1,39], representing the number of features kept at the t-th iteration such that

$$M_t = k + (p_0 - k)\max\left\{0, \frac{N_{iter} - t}{N_{iter} + t\mu}\right\}, \qquad t = 1,\ 2,\ ...,\ N_{iter},$$

where, p_0 is the total number of variables, k is the desired sparsity level, μ is the annealing parameter, and N_{iter} is the total number of iterations for this schedule when exactly k features are selected.

Figure 2 illustrates the annealing schedule for various annealing parameters μ. A larger μ leads to a sharp decline in the number of retained variables M_t in the initial stage, whereas a smaller μ results in a far more gradual decrease. Compared with linear decay, the annealing schedule is computationally more efficient, as it truncates more input variables and reduces the dimensionality of the neural network weight parameters in the first few iterations. The multistage FSA algorithm is summarized in Algorithm 2.

Given that the neural network models are trained with SGD per epoch, backpropagation is performed accordingly. As a result, ω_2 and b_2 are updated first, followed by the updates of $\mathbf{W}_1$ and $\mathbf{b}_1$. While sharing the same SGD iterations in each stage, Algorithm 1 employs a hard-thresholding operator and sets the unimportant weights to zero, keeping only k variables. Algorithm 2 removes the unimportant variables according to the current schedule M_t, which reaches k in the last stage. It is worth noting that during the annealing procedure, the learning rate may need to be adapted based on the number of remaining variables M_t, due to changes in the problem dimensions[2]. Consequently, the learning rate is kept constant within each stage but decays across stages.

The BIC for neural network models

Since the true number of features k^* is generally unknown in practice, it is necessary to determine the number of relevant features using a model selection criterion. In the neural network model, we can select the number of variables k using the BIC[37]. According to the conventional definition of the BIC for a sparse model[40], the BIC value for a one-hidden-layer neural network model is given by

$$\text{BIC} = -2\log(\text{likelihood})/n + k \cdot p_1 \cdot \log(n)/n, \tag{4}$$

where, p_1 denotes the width of the hidden layer, k is the selected number of variables, and n is the sample size. Eq. (4) is based on the fact that the number of free parameters in a one-hidden-layer neural network satisfying the sparsity constraint in Eq. (3) is of order $O(k \cdot p_1)$.

However, in the high-dimensional scenario with sample size $n < p_0$, an increase in the number of parameters k causes the penalty term $k \cdot p_1 \cdot \log(n)/n$ to intensify sharply. Consequently, to avoid excessive penalization, the variable selection criterion inevitably favors models with smaller k, which increases the risk of omitting important variables. To address this issue, a novel high-dimensional BIC (HD-BIC) is proposed:

$$\text{HD-BIC} = -2\log(\text{likelihood})/n + k \cdot \log(p_1) \cdot \log(n)/n\ . \tag{5}$$

Here, we modify the penalty term in Eq. (4) to $k \cdot \log(p_1) \cdot \log(n)/n$ for enabling the model to select a more inclusive set of important variables. Furthermore, the high-dimensional setting corresponds to the few-shot supervised learning paradigm within the proposed feature selection framework, particularly when neural networks are employed as the surrogate model. The modification introduced to the penalty term enables us to incorporate neural networks with wider hidden layers, thereby introducing a moderate degree of over-parameterization. Such a design effectively alleviates the difficulties posed by insufficient training samples, a critical challenge commonly encountered in high-dimensional few-shot scenarios. The numerical results for these two BIC values are reported in the experiments of the section "*Simulation*".

A comparison of existing feature selection methods in neural networks

The methods introduced in this paper, regardless of the specific algorithmic details, can be broadly classified as nonparametric

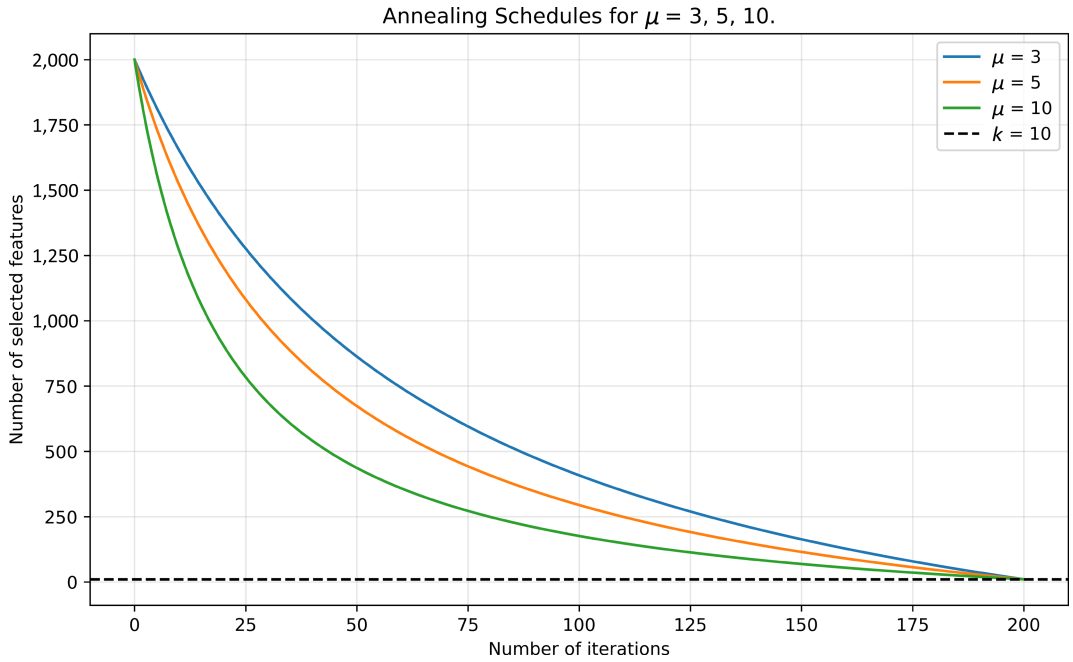

Fig. 2 Annealing schedule for $p_0 = 2{,}000$, $k = 10$, $N_{\text{iter}} = 200$, and multiple annealing parameters $\mu = 3$, 5, and 10.

Algorithm 2. Multistage FSA algorithm.

Input: Standardized training data $\{(\mathbf{x}_i, y_i)\}_{i=1}^n$, initial learning rate η_0, sparse level k, annealing parameter μ, and total iteration times N_{iter}.

Output: Trained weights $\mathbf{W}_1$, $\mathbf{b}_1$, ω_2 and b_2.

Initialize $\mathbf{W}_1^{(1,1)}$, $\mathbf{b}_1^{(1,1)}$, $\omega_2^{(1,1)}$, $b_2^{(1,1)}$.

for $t = 1$ to N_{iter} **do**

Shuffle the training data $\{(\mathbf{x}_i, y_i)\}_{i=1}^n$, and relabel $i = 1, 2, \cdots, n$.

for $i = 1$ to n **do**

$$\omega_2^{(t,i+1)} \leftarrow \omega_2^{(t,i)} - \eta_t \frac{\partial \ell(y_i, f(\mathbf{x}_i; \cdots))}{\partial \omega_2}, \; b_2^{(t,i+1)} \leftarrow b_2^{(t,i)} - \eta_t \frac{\partial \ell(y_i, f(\mathbf{x}_i; \cdots))}{\partial b_2},$$

$$\mathbf{W}_1^{(t,i+1)} \leftarrow \mathbf{W}_1^{(t,i)} - \eta_t \frac{\partial \ell(y_i, f(\mathbf{x}_i; \cdots))}{\partial \mathbf{W}_1},$$

$$\mathbf{b}_1^{(t,i+1)} \leftarrow \mathbf{b}_1^{(t,i)} - \eta_t \frac{\partial \ell(y_i, f(\mathbf{x}_i; \cdots))}{\partial \mathbf{b}_1},$$

end for

Let $\mathbf{W}_1^{(t)} = \mathbf{W}_1^{(t,n+1)}$. Denote $\mathbf{w}_{\cdot j}^{(t)}$ as the j-th column for $\mathbf{W}_1^{(t)}$.

Ranking $\left\{\|\mathbf{w}_{\cdot j}^{(t)}\|_2\right\}$ in descending order, and retaining the index set $\hat{S}^{(t)}$ corresponding to top-M_t largest values of $\left\{\|\mathbf{w}_{\cdot j}^{(t)}\|_2\right\}$.

Keep the $\mathbf{W}_1^{(t)} = [\mathbf{w}_{\cdot j}^{(t)}]$ for $j \in \hat{S}^{(t)}$ and remove $\mathbf{w}_{\cdot j}^{(t)}$ for all others $j \notin \hat{S}^{(t)}$.

Set $\mathbf{W}_1^{(t+1,1)} = \mathbf{W}_1^{(t)}$, $\mathbf{b}_1^{(t+1,1)} = \mathbf{b}_1^{(t,n+1)}$, $\omega_2^{(t+1,1)} = \omega_2^{(t,n+1)}$, and $b_2^{(t+1,1)} = b_2^{(t,n+1)}$.

end for

feature selection techniques. In these approaches, surrogate models are employed to approximate the underlying model. The index set S^* is estimated by determining the parameters of the surrogate models from the available data. Given the powerful approximation capabilities of neural networks, many studies have leveraged them as surrogate models. In this section, we review the relevant studies to the best of the our knowledge and compare their methodologies with those in our own work.

In the literature of the latest method[41], the authors estimate the index set of the nonzero columns of $\mathbf{W}_1$ using a second-order Stein's formula and establish consistency results under the sub-Gaussian assumption on the input covariates and Hölder continuity of the underlying model. Although the resulting estimator is independent of the neural network architecture, the proposed algorithms rely on accurate estimation of the inverse covariance matrix, which is challenging in high dimensions. As variants of the SGD algorithm, our methods incur low storage costs in the high-dimensional setting and are independent of the distribution of the input variables.

In works featuring Deep Feature Selection[29] and LassoNet[30], the sparsity constraint is imposed on $\mathbf{W}_1$, leading to a sparse-constrained nonconvex optimization problem that is typically solved using Alternating Direction Method of Multipliers (ADMM) algorithms. Although our methods address the same class of constrained optimization problems, we instead adopt the "Slow Kill" strategy[2], which gradually enforces sparsity through a multistage feature selection procedure. Moreover, the optimization at each stage is not restricted to SGD; the corresponding iterations can be implemented using any mainstream machine-learning optimization algorithm, thereby providing greater flexibility and additional modeling capabilities.

Simulation

In this section, we describe the evaluation of the proposed variable selection methods using a one-hidden-layer neural network. First, we present the simulation results for a high-dimensional linear regression model as an initial example. Next, we evaluate the performance of the proposed variable selection methods on the nonlinear regression model using large-scale data sets. We then extend the nonlinear regression model to the nonlinear logistic regression model. In addition, we tackle a particularly challenging Exclusive OR (XOR) problem[42]. Finally, we present the results of the real-data analysis. Each simulation setting is repeated 100 times using independently generated data sets, and the cosine-annealing method is used to schedule the learning-rate decay. Since initializing neural network parameters to zero leads to symmetry issues during training, we adopt randomized initialization schemes proposed in the literature[43]. The number of hidden-layer nodes is set to 128 for all experiments, except for the XOR problem, where it is increased to 256. All simulations are performed on a desktop equipped with an AMD Ryzen 9 9950X3D CPU running at 4.30 GHz and 64 GB of memory.

High-dimensional linear regression case

Here, we present the results of a numerical experiment for a high-dimensional linear regression model. The samples $\mathbf{x}_i \in \mathbb{R}^{p_0}$, for $i = 1, 2, \cdots, n$, are generated from a multivariate Gaussian distribution with mean zero and an AR(2) covariance matrix $\mathbf{\Sigma}$, which has the structure:

$$\mathbf{\Sigma}_{i,j} = \begin{cases} \rho^{|i-j|}, & \text{if } |i-j| \le 2, \quad i,\ j = 1,\ 2,\ ...,\ p_0 \\ 0, & \text{others} \end{cases}$$

The response y_i is generated using the high-dimensional linear regression model

$$y_i = \mathbf{x}_i^\top \boldsymbol{\beta}^* + \epsilon_i,\ \epsilon_i \sim \mathcal{N}(0, 1),$$

where, the parameter vector β^* = (2, −2, 2, 3, 3, −2, 3, 3, −3, 2, 0, 0, ..., 0)$^T \in \mathbb{R}^{p_0}$, and ϵ_i is the random noise term. In this simulation, we set p_0 = 2,000 and sample sizes n = 1,000, 1,500, and 3,000. We also generate test data with sample size n = 1,000.

In this simulation, we compare the performance of the proposed multistage StoIHT method with the multistage FSA method. We also include the IHT method as a baseline[31]. First, we consider the case in which the number of variables k^* is known. Next, we assume that the number of variables k^* is unknown and use both the conventional BIC and the proposed HD-BIC, as defined in Eqs. (4) and (5), for variable selection. The evaluation criteria for prediction performance and variable selection include model size (i.e., the number of selected covariates), false selection rate (FSR), negative selection rate (NSR)[44,45], and root-mean-square error (RMSE) for the test data. The FSR and NSR are defined as

$$\text{FSR} = \frac{|\hat{S} \setminus S^*|}{|\hat{S}|} \text{ and } \text{NSR} = \frac{|S^* \setminus \hat{S}|}{|S^*|},$$

where, S^* denotes the index set of true important covariates and $\hat{S}$ is the estimated index set. The results for the numerical experiments are presented in Table 1.

From Table 1, we observe that the performances of the multistage StoIHT and multistage FSA methods are similar in this high-dimensional linear regression setting. Specifically, in the high-dimensional scenario with n = 1,000 and p_0 = 2,000, the multistage FSA outperforms the multistage StoIHT and the IHT when using the proposed HD-BIC. As the sample size n increases, the multistage StoIHT performs slightly better than the multistage FSA. When the sample size n is sufficiently large, for example, n = 3,000, the results obtained with the conventional BIC are better than those obtained with the HD-BIC. We also conclude that multistage FSA and multi-

Table 1. Comparison of the proposed multistage FSA, multistage StoIHT, and IHT methods for the high-dimensional linear regression model with the sample size n = 1,000, 1,500, and 3,000 and p_0 = 2,000.

Sample size	Parameter	Multistage FSA			Multistage StoIHT			IHT		
		–	HD-BIC	BIC	–	HD-BIC	BIC	–	HD-BIC	BIC
n = 1,000	Model size†	10 (0)	11.67 (1.556)	6 (0)	10 (0)	11.99 (1.972)	6 (0)	10 (0)	10.91 (1.167)	6 (0)
	FSR†	0.091 (0.071)	0.207 (0.113)	0 (0)	0.185 (0.074)	0.284 (0.135)	0 (0)	0.075 (0.099)	0.141 (0.120)	0 (0)
	NSR†	0.091 (0.071)	0.089 (0.073)	0.4 (0)	0.185 (0.074)	0.165 (0.077)	0.4 (0)	0.075 (0.099)	0.072 (0.094)	0.4 (0)
	RMSE†	1.745 (0.510)	1.743 (0.533)	3.333 (0.142)	2.353 (0.461)	2.234 (0.499)	3.222 (0.101)	1.520 (0.617)	1.492 (0.593)	3.197 (0.095)
	Time‡	2.972	34.27	32.63	11.58	128.1	130.0	2.017	22.01	23.17
n = 1,500	Model size†	10 (0)	10.43 (0.711)	7.160 (1.815)	10 (0)	10.41 (0.861)	6.040 (0.398)	10 (0)	10.43 (0.778)	6 (0)
	FSR†	0.022 (0.044)	0.061 (0.075)	0.006 (0.024)	0.013 (0.034)	0.041 (0.068)	0 (0)	0.073 (0.094)	0.090 (0.106)	0 (0)
	NSR†	0.022 (0.044)	0.025 (0.048)	0.290 (0.173)	0.013 (0.034)	0.007 (0.026)	0.396 (0.040)	0.073 (0.094)	0.056 (0.089)	0.4 (0)
	RMSE†	1.192 (0.343)	1.228 (0.400)	2.802 (1.054)	1.119 (0.270)	1.075 (0.205)	3.206 (0.236)	1.467 (0.563)	1.368 (0.538)	3.179 (0.097)
	Time‡	4.551	51.80	52.55	17.22	192.0	192.8	3.183	34.48	36.30
n = 3,000	Model size†	10 (0)	10.02 (0.140)	10 (0)	10 (0)	10.35 (0.921)	10 (0)	10 (0)	10.51 (0.877)	10.34 (0.681)
	FSR†	0 (0)	0.002 (0.013)	0 (0)	0 (0)	0.028 (0.067)	0 (0)	0.070 (0.094)	0.089 (0.100)	0.081 (0.098)
	NSR†	0 (0)	0 (0)	0 (0)	0 (0)	0 (0)	0 (0)	0.070 (0.094)	0.048 (0.083)	0.053 (0.085)
	RMSE†	1.192 (0.343)	1.228 (0.400)	2.802 (1.054)	1.119 (0.270)	1.075 (0.205)	3.206 (0.236)	1.467 (0.563)	1.313 (0.506)	1.347 (0.525)
	Time‡	4.551	51.80	52.55	17.22	192.0	192.8	3.183	56.24	59.04

All experiments are conducted on 100 independent data sets. † Average FSR, NSR, and RMSE of test data are presented; ‡ average computational time (s).

stage StoIHT outperform the IHT method in the large-n case.

Nonlinear regression case

The numerical results for the nonlinear regression model are presented in this section. The predictors $\mathbf{x}_i \in \mathbb{R}^{p_0}$, for $i = 1, 2, \cdots, n$, are generated from the previous multivariate Gaussian distribution with mean zero and an AR(2) covariance matrix $\mathbf{\Sigma}$. The response y_i is generated using the high-dimensional nonlinear regression model:

$$y_i = \frac{6x_{i2}}{1+x_{i1}^2} + 5\sin(x_{i3}x_{i4}) + 4x_{i5} - 5x_{i6} + 0 \cdot x_{i7} + \cdots + 0 \cdot x_{ip_0} + \epsilon_i,\ \epsilon_i \sim \mathcal{N}(0,1).$$

In this case, we set p_0 = 1,000 and $n = 10^4$, 2×10^4, 3×10^4 as a large-scale data set, and compare the multistage FSA, multistage StoIHT, and IHT under the conditions of known and unknown k^*. In addition, we generate test data with n = 1,000. The results of numerical experiments are presented in Table 2.

From Table 2, we can conclude that the performance of the multistage StoIHT slightly outperforms that of the multistage FSA when the sample size n is large, regardless of whether k^* is known. However, the multistage StoIHT is more computationally expensive than the multistage FSA. Both the multistage FSA and multistage StoIHT methods outperform the IHT method. When the sample size n is large, the variable selection performance of HD-BIC is inferior of the conventional BIC. The former may select more unimportant variables compared to the latter.

Nonlinear logistic regression case

We extend the nonlinear regression model to the nonlinear logistic regression model. The instances $\mathbf{x}_i \in \mathbb{R}^{p_0}$, for $i = 1, 2, \cdots, n$, are generated using the multivariate Gaussian distribution described above. The response y_i is generated using the model

$$y_i = \mathbb{I}\left(2\exp(x_{i1}) + 3x_{i2}^2 + 5\sin(x_{i3}x_{i4}) - 4x_{i5} - 3x_{i6} + 0 \cdot x_{i7} + \cdots + 0 \cdot x_{ip_0} \geq \epsilon_i\right),$$

where $\mathbb{I}(\cdot)$ is an indicator function and the random noise term ϵ_i has a

Table 2. Comparison of the multistage FSA, multistage StoIHT, and IHT methods for the nonlinear regression model with the sample size $n = 10^4$, 2×10^4, 3×10^4 , and p_0 = 1,000.

Sample size	Parameter	Multistage FSA			Multistage StoIHT			IHT		
		–	HD-BIC	BIC	–	HD-BIC	BIC	–	HD-BIC	BIC
$n = 10^4$	Model size†	6 (0)	8.270 (1.399)	5.850 (0.766)	6 (0)	7.810 (1.270)	6.120 (0.325)	6 (0)	7.640 (1.797)	5.120 (1.283)
	FSR†	0.100 (0.082)	0.309 (0.142)	0.070 (0.090)	0 (0)	0.212 (0.126)	0.017 (0.046)	0.275 (0,128)	0.401 (0.164)	0.172 (0.175)
	NSR†	0.100 (0.082)	0.077 (0.083)	0.102 (0.081)	0 (0)	0 (0)	0 (0)	0.275 (0,128)	0.278 (0.106)	0.323 (0.099)
	RMSE†	2.518 (0.176)	2.453 (0.168)	2.543 (0.222)	1.506 (0.084)	1.531 (0.082)	1.519 (0.082)	2.888 (0.207)	2.943 (0.183)	3.020 (0.198)
	Time‡	23.78	195.6	192.8	27.75	223.5	219.7	10.80	83.04	89.33
$n = 2 \times 10^4$	Model size†	6 (0)	8.150 (1.352)	6.350 (0.684)	6 (0)	8.180 (1.403)	6.090 (0.286)	6 (0)	7.740 (1.659)	6.510 (1.895)
	FSR†	0.015 (0.048)	0.257 (0.125)	0.059 (0.094)	0 (0)	0.243 (0.139)	0.013 (0.041)	0.282 (0.122)	0.406 (0.164)	0.291 (0.205)
	NSR†	0.015 (0.048)	0.017 (0.050)	0.013 (0.045)	0 (0)	0 (0)	0 (0)	0.282 (0.122)	0.267 (0.139)	0.283 (0.130)
	RMSE†	2.110 (0.150)	2.087 (0.159)	2.076 (0.159)	1.408 (0.075)	1.393 (0.083)	1.390 (0.083)	2.906 (0.210)	2.885 (0.280)	2.943 (0.308)
	Time‡	47.67	379.9	386.5	55.51	457.8	446.5	18.11	156.6	162.2
$n = 3 \times 10^4$	Model size†	6 (0)	7.790 (1.409)	6.610 (0.937)	6 (0)	7.760 (1.556)	6.020 (0.140)	6 (0)	7.660 (1.589)	6.590 (1.650)
	FSR†	0.007 (0.033)	0.209 (0.148)	0.080 (0.113)	0 (0)	0.195 (0.159)	0.003 (0.020)	0.252 (0.124)	0.388 (0.159)	0.296 (0.180)
	NSR†	0.007 (0.033)	0.007 (0.033)	0.003 (0.023)	0 (0)	0 (0)	0 (0)	0.252 (0.124)	0.252 (0.119)	0.267 (0.120)
	RMSE†	1.823 (0.149)	1.816 (0.143)	1.800 (0.130)	1.357 (0.080)	1.358 (0.069)	1.354 (0.067)	2.859 (0.242)	2.852 (0.246)	2.876 (0.241)
	Time‡	72.70	587.9	586.0	82.18	682.6	670.2	33.98	264.8	268.5

All experiments are conducted on 100 independent data sets. †Average FSR, NSR, and RMSE of the test data are presented; ‡average computational time (s).

logistic distribution. We set p_0 = 1,000 and $n = 10^4$, 2×10^4, 3×10^4 in this nonlinear logistic regression case, and compare the multistage FSA, multistage StoIHT, and IHT under the conditions of known and unknown k^*. The test data is generated with n = 1,000. We use the area under the receiver operating characteristic curve (AUC) to evaluate the predictive performance of the proposed methods. Recall that, for binary classification problems, the ROC curve shows the trade-off between false-positive rate and true positive rate across all possible classification thresholds. AUC indicates the probability that the model ranks a random positive sample higher than a random negative sample; it equals 1 for perfect classifiers. AUC was computed using sklearn.metrics.roc_auc_score[46], which implements a pairwise ranking approach: positive–negative samples are ranked by predicted scores, valid pairs (positive score > negative score) are counted, with ties contributing 0.5 per pair, and the total is normalized by the number of positive–negative pairs. The results of numerical experiments are presented in Table 3.

From Table 3, we observe that the multistage FSA significantly outperforms the multistage StoIHT and IHT methods in both variable selection and prediction, regardless of whether k^* is known. For large sample sizes n, HD-BIC may select more unimportant variables.

XOR problem

In this section, we present numerical results on the challenging XOR problem for the proposed multistage FSA and multistage StoIHT methods. The IHT method is also included as a baseline for comparison. The XOR problem is a binary classification task. The p_0-dimensional predictor vectors $\mathbf{x}_i \in \mathbb{R}^{p_0}$ are independently generated from the uniform distribution $\mathcal{U}(-1, 1)$. The response variable y_i is defined as

$$y_i = \mathbb{I}\left(\prod_{j=1}^{k^*} x_{ij} > 0\right),$$

where, $\mathbb{I}(\cdot)$ denotes the indicator function. This model corresponds to a k^*-dimensional XOR problem with nonlinear separability. We set p_0 = 20 and n = 3,000 in this XOR problem and compare the multi-stage FSA, multi-stage StoIHT, and IHT methods. The test data is

Table 3. Comparison of the multistage FSA, multistage StoIHT, and IHT methods for the nonlinear logistic regression model with the sample size $n = 10^4$, 2×10^4, 3×10^4, and p_0 = 1,000.

Sample size	Parameter	Multistage FSA			Multistage StoIHT			IHT		
		–	HD-BIC	BIC	–	HD-BIC	BIC	–	HD-BIC	BIC
$n = 10^4$	Model size†	6 (0)	6.530 (0.830)	4.680 (0.760)	6 (0)	5.450 (1.982)	4.020 (0.140)	6 (0)	7.500 (1.825)	4.040 (0.196)
	FSR†	0.072 (0.118)	0.135 (0.161)	0.021 (0.069)	0.650 (0.065)	0.580 (0.131)	0.497 (0.088)	0.330 (0.023)	0.411 (0.146)	0.248 (0.010)
	NSR†	0.072 (0.118)	0.073 (0.134)	0.238 (0.125)	0.650 (0.065)	0.652 (0.075)	0.663 (0.058)	0.330 (0.023)	0.305 (0.071)	0.493 (0.033)
	AUC†	0.960 (0.008)	0.959 (0.011)	0.942 (0.021)	0.903 (0.015)	0.903 (0.016)	0.902 (0.015)	0.933 (0.009)	0.936 (0.013)	0.904 (0.013)
	Time‡	24.22	201.6	196.3	44.29	365.7	375.2	12.23	105.1	101.6
$n = 2 \times 10^4$	Model size†	6 (0)	7.460 (1.315)	5.890 (0.467)	6 (0)	6.880 (1.856)	4.310 (0.523)	6 (0)	7.670 (1.750)	5.190 (0.674)
	FSR†	0.013 (0.056)	0.175 (0.138)	0.005 (0.027)	0.512 (0.114)	0.505 (0.159)	0.317 (0.157)	0.330 (0.023)	0.426 (0.133)	0.219 (0.065)
	NSR†	0.013 (0.056)	0.003 (0.023)	0.023 (0.078)	0.512 (0.114)	0.468 (0.112)	0.515 (0.100)	0.330 (0.023)	0.302 (0.073)	0.332 (0.029)
	AUC†	0.967 (0.007)	0.967 (0.006)	0.956 (0.049)	0.915 (0.016)	0.918 (0.018)	0.915 (0.017)	0.934 (0.009)	0.938 (0.013)	0.934 (0.009)
	Time‡	50.43	398.4	397.6	88.29	711.3	750.4	22.64	198.6	189.5
$n = 3 \times 10^4$	Model size†	6 (0)	8.810 (1.102)	5.970 (0.386)	6 (0)	4.710 (1.458)	4.050 (0.218)	6 (0)	7.640 (1.616)	5.450 (1.117)
	FSR†	0.008 (0.036)	0.309 (0.100)	0.008 (0.033)	0.350 (0.055)	0.150 (0.173)	0.075 (0.112)	0.332 (0.017)	0.431 (0.119)	0.244 (0.097)
	NSR†	0.008 (0.036)	0.003 (0.023)	0.013 (0.061)	0.350 (0.055)	0.367 (0.067)	0.377 (0.073)	0.332 (0.017)	0.303 (0.080)	0.330 (0.053)
	AUC†	0.967 (0.007)	0.968 (0.006)	0.962 (0.034)	0.936 (0.008)	0.935 (0.008)	0.934 (0.009)	0.933 (0.008)	0.938 (0.014)	0.935 (0.012)
	Time‡	73.86	592.2	592.6	124.3	1069	1138	33.84	256.8	250.2

All experiments are conducted on 100 independent data sets. † Average FSR, NSR, and AUC for test data are presented; ‡ average computational time (s).

Table 4. Comparison of the multistage FSA, multistage StoIHT, and IHT methods for the XOR model with sample size n = 3,000 and p_0 = 20.

	Parameter	Multistage FSA			Multistage StoIHT			IHT		
		–	HD-BIC	BIC	–	HD-BIC	BIC	–	HD-BIC	BIC
$k^* = 3$	Model size†	3 (0)	3.580 (0.982)	1 (0)	3 (0)	6.940 (0.276)	1 (0)	3 (0)	1.160 (0.913)	1 (0)
	FSR†	0.277 (0.430)	0.333 (0.405)	0.250 (0.433)	0.870 (0.199)	0.834 (0.117)	0.880 (0.325)	0.997 (0.033)	0.887 (0.313)	0.890 (0.313)
	NSR†	0.277 (0.430)	0.253 (0.419)	0.750 (0.144)	0.870 (0.199)	0.613 (0.274)	0.960 (0.108)	0.997 (0.033)	0.957 (0.112)	0.963 (0.104)
	AUC†	0.848 (0.227)	0.856 (0.225)	0.501 (0.020)	0.505 (0.053)	0.522 (0.110)	0.500 (0.019)	0.500 (0.009)	0.500 (0.000)	0.500 (0.000)
	Time‡	2.378	18.58	19.81	2.833	22.64	23.22	4.458	17.71	17.59
$k^* = 4$	Model size†	4 (0)	6.130 (1.101)	1 (0)	4 (0)	7 (0)	1 (0)	4 (0)	1 (0)	1 (0)
	FSR†	0.290 (0.396)	0.473 (0.260)	0.320 (0.466)	0.840 (0.175)	0.786 (0.132)	0.880 (0.325)	1 (0)	0.850 (0.357)	0.850 (0.357)
	NSR†	0.290 (0.396)	0.223 (0.337)	0.830 (0.117)	0.840 (0.175)	0.625 (0.230)	0.970 (0.081)	1 (0)	0.963 (0.089)	0.963 (0.089)
	AUC†	0.812 (0.231)	0.823 (0.221)	0.497 (0.018)	0.498 (0.019)	0.506 (0.050)	0.497 (0.020)	0.498 (0.010)	0.500 (0.001)	0.500 (0.001)
	Time‡	2.615	20.51	20.47	2.788	24.52	24.50	4.611	17.46	17.43
$k^* = 5$	Model size†	5 (0)	6.590 (0.750)	1 (0)	5 (0)	7 (0)	1 (0)	5 (0)	1 (0)	1 (0)
	FSR†	0.636 (0.313)	0.624 (0.264)	0.610 (0.488)	0.792 (0.160)	0.734 (0.146)	0.830 (0.376)	0.998 (0.020)	0.820 (0.384)	0.820 (0.384)
	NSR†	0.636 (0.313)	0.512 (0.325)	0.922 (0.098)	0.792 (0.160)	0.628 (0.204)	0.966 (0.075)	0.998 (0.020)	0.964 (0.077)	0.964 (0.077)
	AUC†	0.573 (0.166)	0.598 (0.185)	0.500 (0.019)	0.503 (0.020)	0.508 (0.048)	0.500 (0.017)	0.498 (0.011)	0.500 (0.000)	0.500 (0.000)
	Time‡	2.616	20.88	20.97	2.828	25.30	23.83	4.618	17.37	17.57

All experiments are conducted on 100 independent data sets. † Average FSR, NSR, and AUC for test data are presented; ‡ average computational time (s).

generated with n = 1,000. We use the AUC to evaluate the predictive performance of the proposed methods. The results of numerical experiments are presented in Table 4.

Table 4 shows that the performance of all methods is relatively poor, attributed to the XOR problem being particularly challenging. Nevertheless, compared with the multistage StoIHT and IHT methods, the multistage FSA achieves better variable selection and prediction performance when using the HD-BIC.

Real-data analysis

17-AAG data set

We apply the proposed multistage FSA, multistage StoIHT, and IHT methods to the 17-AAG data set from the Cancer Cell Line Encyclopedia (CCLE) to identify genes associated with sensitivity to 17-AAG. The CCLE data set comprises 8-point dose–response curves for 24 chemical compounds, measured across more than 400 cancer cell lines, and is publicly available at www.broadinstitute.org/ccle. For each cell line, gene expression levels of 18,926 genes are recorded. We use the area under the dose–response curve as the response variable to quantify drug sensitivity[47]. Specifically, for the 17-AAG data set, our objective is to apply the proposed variable selection methods to identify genes associated with sensitivity to 17-AAG.

First, we apply the proposed HD-BIC to the entire data set to select important variables. Figure 3 presents the BIC values for different numbers of selected variables k. We observe that when k = 2, the BIC value is minimal. The essential variable, NQO1, is selected using the multistage FSA method when $k = 3$ and $k = 4$.

We then randomly split the data set into training and validation sets, with 400 samples in the training set and 76 samples in the validation set. This procedure is repeated 20 times. The training data are used to fit the model, and the performance is evaluated on the validation data. We select k = 2, 3, 4 variables and compare the multistage FSA with the multistage StoIHT and IHT methods. The results are presented in Table 5.

The variable NQO1 selected by the multistage FSA method has been identified in the existing literature as the most important biomarker associated with sensitivity to 17-AAG[47,48]. From Table 5, we observe that, across multiple choices of the selected number k, the multistage FSA achieves a smaller predicted RMSE on the validation data compared with the multistage StoIHT and IHT methods.

Madelon data set

The Madelon data set is an artificial data set designed for variable selection and classification[49]. This data set consists of data points grouped into 32 clusters at the vertices of a five-dimensional hypercube, with labels assigned at random to +1 or −1 (relabeled as +1 or 0 in this paper). This data set contains k^* = 20 informative features, including 5 original features and their 15 linear combinations. In addition, this data set includes 480 irrelevant features with no predictive power, bringing the total to 500. The data set has a sample size of 2,600. The Madelon data set is publicly available at https://archive.ics.uci.edu/dataset/171/madelon. We train shallow neural networks on the training data and report the performance on the validation data in Table 6.

We randomly split the data set into training and validation sets 20 times, with 2,000 samples in the training set and 600 samples in the validation set each time. The training data are used to fit the model, while the performance is evaluated on the validation set. We compare the performance of the proposed multistage FSA, multistage stochastic IHT, and IHT methods. In addition, the classical ℓ_1-penalized method is included as a baseline for comparison. As the true number of variables is known (k^* = 20) in this data set, we use k^* as the known value, and the conventional BIC and proposed HD-

Table 5. Comparison of the multistage FSA, multistage StoIHT, and IHT methods for 17-AAG data set.

Number of selected variables	Multistage FSA	Multistage StoIHT	IHT
$k = 2$	0.991 (0.064)	0.991 (0.064)	0.991 (0.065)
$k = 3$	0.988 (0.065)	0.992 (0.062)	0.993 (0.056)
$k = 4$	0.987 (0.065)	0.994 (0.062)	0.989 (0.059)

The RMSE values for validation data are presented.

Table 6. Comparison of the multistage FSA, multistage StoIHT, and IHT methods for the Madelon data set.

	Multistage FSA	Multistage StoIHT	IHT	Lasso
k^* Known	0.642 (0.021)	0.535 (0.029)	0.534 (0.040)	0.608 (0.013)
BIC	0.656 (0.022)	0.533 (0.062)	0.522 (0.040)	0.610 (0.011)
HD-BIC	0.651 (0.029)	0.532 (0.049)	0.519 (0.039)	–

The average AUC values for validation data are presented.

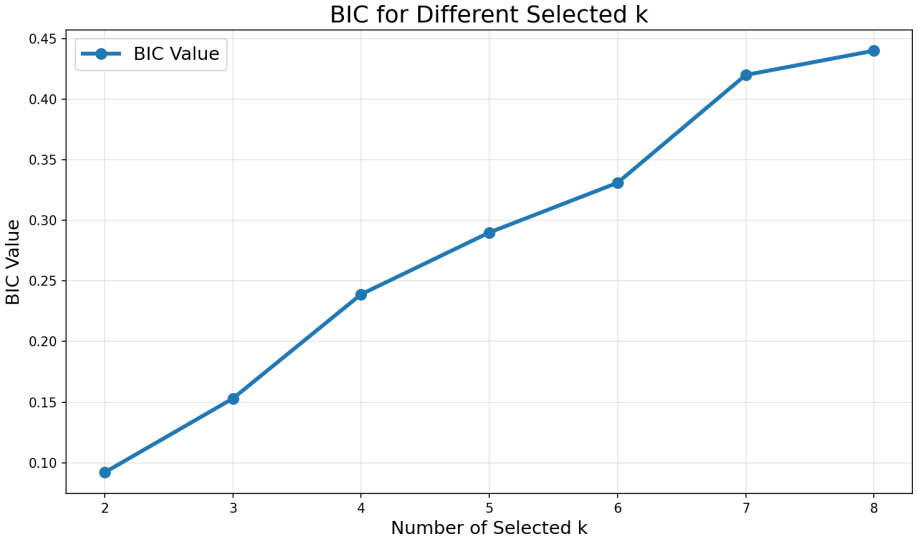

Fig. 3 BIC values for multiple selected k.

BIC as the criteria. For the ℓ_1-penalized method, which is proposed for linear models, we use k^* as the known value and the conventional BIC as the criteria. We select about 20 variables each time when k^* is the known value. From Table 6, we observe that the multistage FSA outperforms the other methods across all criteria on the validation data.

Discussion

This article proposes two novel stochastic algorithms for nonlinear variable selection via shallow neural networks. The first proposed algorithm is the multistage StoIHT algorithm, a stochastic version of the IHT algorithm[31,32]. The second proposed algorithm is the multistage FSA algorithm, a variable selection method that employs an annealing strategy with the multistage StoIHT. After a comprehensive evaluation in both regression and classification settings, we conclude that the proposed multistage FSA and multistage StoIHT methods slightly outperform the IHT method in the numerical experiments presented in this paper. Furthermore, in the high-dimensional setting (large-p–small-n setting), the proposed HD-BIC outperforms the conventional BIC for variable selection. In contrast, in the low-dimensional setting (large-n–small-p setting), the conventional BIC performs better for variable selection.

Although the proposed methods exhibit strong empirical performance, they have several limitations. First, this study focuses primarily on numerical investigations, with rigorous theoretical analysis left for future research. Unlike parameter learning in linear regression, which involves only the estimation error, training a neural network model requires accounting for both estimation and approximation errors[14]. As a result, the theoretical analysis of variable selection consistency for neural network models is substantially more complicated than that for linear regression models. Second, the present study is limited to nonlinear variable selection using shallow neural networks, and the performance of the proposed methods in the deep neural network setting remains unexplored. We will focus on these issues in our future research.

Author contributions

This work was collaboratively conducted by Li X, Zhang C, Sun L, Meng Z, and Zhang H. Li X and Zhang C contributed equally to this paper. The contributions of each author are as follows: Li X, Zhang C, and Sun L contributed to the study's conception and design. Li X and Zhang C conducted coding preparation, data collection, and analysis. Sun L wrote the first draft of the manuscript and revised it. Meng Z and Zhang C contributed to the final version of the manuscript through critical revisions. All authors reviewed the manuscript and approved the final version for submission.

Data availability

The real data sets used in this paper are publicly available as specified in the section "*Simulation*". The code will be made publicly available on https://github.com/lizhesun0507 upon acceptance of the paper.

Acknowledgments

Lizhe Sun's research was partially supported by the National Social Science Fund of China grant 24&ZD183.

Conflict of interest

The authors declare that they have no conflict of interest.

Dates

Received 30 December 2025; Revised 20 March 2026; Accepted 8 April 2026; Published online 31 May 2026

References

[1] Barbu A, She Y, Ding L, Gramajo G. 2017. Feature selection with annealing for computer vision and big data learning. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 39(2):272–286

[2] She Y, Shen J, Barbu A. 2023. Slow kill for big data learning. *IEEE Transactions on Information Theory* 69(9):5936–5955

[3] Tibshirani R. 1996. Regression shrinkage and selection *via* the lasso. *Journal of the Royal Statistical Society Series B: Methodological* 58(1):267–288

[4] Zou H, Hastie T. 2005. Regularization and variable selection *via* the elastic net. *Journal of the Royal Statistical Society Series B: Statistical Methodology* 67(2):301–320

[5] Zou H. 2006. The adaptive lasso and its oracle properties. *Journal of the American Statistical Association* 101(476):1418–1429

[6] Fan J, Li R. 2001. Variable selection *via* nonconcave penalized likelihood and its oracle properties. *Journal of the American Statistical Association* 96(456):1348–1360

[7] Zhang CH. 2010. Nearly unbiased variable selection under minimax concave penalty. *The Annals of Statistics* 38(2):894–942

[8] Yuan M, Lin Y. 2006. Model selection and estimation in regression with grouped variables. *Journal of the Royal Statistical Society Series B: Statistical Methodology* 68(1):49–67

[9] Wang M, Tian GL. 2019. Adaptive group lasso for high-dimensional generalized linear models. *Statistical Papers* 60(5):1469–1486

[10] Wei F, Huang J, Li H. 2011. Variable selection and estimation in high-dimensional varying-coefficient models. *Statistica Sinica* 21(4):1515–1540

[11] Ravikumar P, Lafferty J, Liu H, Wasserman L. 2009. Sparse additive models. *Journal of the Royal Statistical Society Series B: Statistical Methodology* 71(5):1009–1030

[12] Schmidt-Hieber J. 2020. Nonparametric regression using deep neural networks with ReLU activation function. *The Annals of Statistics* 48(4):1875–1897

[13] Nakada R, Imaizumi M. 2020. Adaptive approximation and generalization of deep neural network with intrinsic dimensionality. *Journal of Machine Learning Research* 21(174):1–38

[14] Kohler M, Langer S. 2021. On the rate of convergence of fully connected deep neural network regression estimates. *The Annals of Statistics* 49(4):2231–2249

[15] Jiao Y, Shen G, Lin Y, Huang J. 2023. Deep nonparametric regression on approximate manifolds: nonasymptotic error bounds with polynomial prefactors. *The Annals of Statistics* 51(2):691–716

[16] Siegel JW. 2023. Optimal approximation rates for deep relu neural networks on sobolev and besov spaces. *Journal of Machine Learning Research* 24(357):1–52

[17] Srivastava N, Hinton G, Krizhevsky A, Sutskever I, Salakhutdinov R. 2014. Dropout: a simple way to prevent neural networks from overfitting. *Journal of Machine Learning Research* 15(1):1929–1958

[18] Liang F, Li Q, Zhou L. 2018. Bayesian neural networks for selection of drug sensitive genes. *Journal of the American Statistical Association* 113(523):955–972

[19] Ghosh S, Yao J, Doshi-Velez F. 2019. Model selection in Bayesian neural networks *via* horseshoe priors. *Journal of Machine Learning Research* 20(182):1–46

[20] Sun Y, Song Q, Liang F. 2022. Consistent sparse deep learning: theory and computation. *Journal of the American Statistical Association* 117(540):1981–1995

[21] Sun Y, Song Q, Liang F. 2022. Learning sparse deep neural networks

with a spike-and-slab prior. *Statistics & Probability Letters* 180:109246

[22] Wen W, Wu C, Wang Y, Chen Y, Li H. 2016. Learning structured sparsity in deep neural networks. *NIPS'16: Proceedings of the 30th International Conference on Neural Information Processing Systems, Barcelona, Spain, 5–10 December 2016*. Vol. 29. Red Hook, NY, USA: Curran Associates, Inc. pp. 2082–2090 https://proceedings.neurips.cc/paper_files/paper/2016/file/41bfd20a38bb1b0bec75acf0845530a7-Paper.pdf (Accessed March 20, 2026)

[23] Scardapane S, Comminiello D, Hussain A, Uncini A. 2017. Group sparse regularization for deep neural networks. *Neurocomputing* 241:81–89

[24] Bungert L, Roith T, Tenbrinck D, Burger M. 2022. A Bregman learning framework for sparse neural networks. *Journal of Machine Learning Research* 23(192):1–43

[25] Li G, Wang G, Ding J. 2023. Provable identifiability of two-layer ReLU neural networks *via* LASSO regularization. *IEEE Transactions on Information Theory* 69(9):5921–5935

[26] Guo Y, She Y, Barbu A. 2021. Network pruning via annealing and direct sparsity control. *2021 International Joint Conference on Neural Networks (IJCNN). Shenzhen, China, 18–22 July 2021*. New Jersey: IEEE. pp. 1–8 doi: 10.1109/ijcnn52387.2021.9533741

[27] Jantre S, Bhattacharya S, Maiti T. 2025. Spike-and-slab shrinkage priors for structurally sparse Bayesian neural networks. *IEEE Transactions on Neural Networks and Learning Systems* 36(6):11176–11188

[28] Dinh VC, Ho LS. 2020. Consistent feature selection for analytic deep neural networks. *NIPS '20: Proceedings of the 34th International Conference on Neural Information Processing Systems, Vancouver, BC, Canada, 6–12 December 2020*, eds. Larochelle H, Ranzato M, Hadsell R, Balcan MF, Lin H. Vol. 33. Red Hook, NY, USA: Curran Associates, Inc. pp. 2420–2431 https://proceedings.neurips.cc/paper_files/paper/2020/file/1959eb9d5a0f7ebc58ebde81d5df400d-Paper.pdf (Accessed March 20, 2026)

[29] Chen Y, Gao Q, Liang F, Wang X. 2021. Nonlinear variable selection *via* deep neural networks. *Journal of Computational and Graphical Statistics* 30(2):484–492

[30] Lemhadri I, Ruan F, Abraham L, Tibshirani R. 2021. Lassonet: a neural network with feature sparsity. *Journal of Machine Learning Research* 22(127):1–29

[31] Yang Z, Zheng S, Tang N. 2026. Supervised predictive modeling of high-dimensional data with group ℓ^0-norm constrained neural networks. *Journal of Computational and Graphical Statistics* 00:1–14

[32] Yuan XT, Li P, Zhang T. 2018. Gradient hard thresholding pursuit. *Journal of Machine Learning Research* 18(166):1–43

[33] Yang R, Song Y. 2024. Nonparametric expectile regression meets deep neural networks: a robust nonlinear variable selection method. *Statistical Analysis and Data Mining: The ASA Data Science Journal* 17(6):e70002

[34] Zhao P, Yu B. 2006. On model selection consistency of lasso. *Journal of Machine Learning Research* 7(90):2541–2563

[35] She Y. 2009. Thresholding-based iterative selection procedures for model selection and shrinkage. *Electronic Journal of Statistics* 3:384–415

[36] Agarwal A, Negahban SN, Wainwright MJ. 2012. Stochastic optimization and sparse statistical recovery: optimal algorithms for high dimensions. *Advances in Neural Information Processing Systems 25 (NIPS 2012), Lake Tahoe, NV, USA*. Red Hook, NY, USA: Curran Associates, Inc. pp. 1547–1555 https://proceedings.neurips.cc/paper_files/paper/2012/file/5751ec3e9a4feab575962e78e006250d-Paper.pdf (Accessed March 20, 2026).

[37] McInerney A, Burke K. 2025. A statistical modelling approach to feed-forward neural network model selection. *Statistical Modelling* 25(4):323–342

[38] Nguyen N, Needell D, Woolf T. 2017. Linear convergence of stochastic iterative greedy algorithms with sparse constraints. *IEEE Transactions on Information Theory* 63(11):6869–6895

[39] Sun L, Barbu A. 2025. Stochastic feature selection with annealing and its applications to streaming data. *Journal of Nonparametric Statistics* 37(3):580–597

[40] Zou H, Hastie T, Tibshirani R. 2007. On the "degrees of freedom" of the lasso. *The Annals of Statistics* 35(5):2173–2192

[41] Du J, Li Z, Gu Z, Feng L. 2025. A nonparametric statistics approach to feature selection in deep neural networks with theoretical guarantees. *arXiv* 2512.13565

[42] Guo Y, Wu YN, Barbu A. 2021. A study of local optima for learning feature interactions using neural networks. *2021 International Joint Conference on Neural Networks (IJCNN), Shenzhen, China, 18–22 July 2021*. New Jersey: IEEE. pp. 1–8 doi: 10.1109/ijcnn52387.2021.9533833

[43] He K, Zhang X, Ren S, Sun J. 2015. Delving deep into rectifiers: surpassing human-level performance on ImageNet classification. *2015 IEEE International Conference on Computer Vision (ICCV), Santiago, Chile, 7–13 December 2015*. New Jersey: IEEE. pp. 1026–1034 doi: 10.1109/iccv.2015.123

[44] Liang F, Xue J, Jia B. 2022. Markov neighborhood regression for high-dimensional inference. *Journal of the American Statistical Association* 117(539):1200–1214

[45] Sun L, Liang F. 2022. Markov neighborhood regression for statistical inference of high-dimensional generalized linear models. *Statistics in Medicine* 41(20):4057–4078

[46] Pedregosa F, Varoquaux C, Gramfort A, Michel V, Thirion B, et al. 2011. Scikit-learn: machine learning in Python. *Journal of Machine Learning Research* 12(85):2825–2830

[47] Barretina J, Caponigro G, Stransky N, Venkatesan K, Margolin AA, et al. 2012. The cancer cell line encyclopedia enables predictive modelling of anticancer drug sensitivity. *Nature* 483:603–607

[48] Hadley KE, Hendricks DT. 2014. Use of NQO1 status as a selective biomarker for oesophageal squamous cell carcinomas with greater sensitivity to 17-AAG. *BMC Cancer* 14:334

[49] Guyon I, Gunn S, Ben-Hur A, Dror G. 2004. Result analysis of the NIPS 2003 feature selection challenge. *Advances in Neural Information Processing Systems 17 (NIPS 2004)*, eds. Saul L, Weiss Y, Bottou L. Cambridge, MA: MIT Press. pp. 545–552. https://proceedings.neurips.cc/paper_files/paper/2004/file/5e751896e527c862bf67251a474b3819-Paper.pdf (Accessed March 20, 2026)

Copyright: © 2026 by the author(s). Published by Maximum Academic Press, Fayetteville, GA. This article is an open access article distributed under Creative Commons Attribution License (CC BY 4.0), visit https://creativecommons.org/licenses/by/4.0/.