Sparse-Input Neural Network using Group Concave Regularization

Anonymous authors
Paper under double-blind review

Abstract

Simultaneous feature selection and non-linear function estimation are challenging, especially in high-dimensional settings where the number of variables exceeds the available sample size in modeling. In this article, we investigate the problem of feature selection in neural networks. Although the group LASSO has been utilized to select variables for learning with neural networks, it tends to select unimportant variables into the model to compensate for its over-shrinkage. To overcome this limitation, we propose a framework of sparse-input neural networks using group concave regularization for feature selection in both low-dimensional and high-dimensional settings. The main idea is to apply a proper concave penalty to the l_2 norm of weights from all outgoing connections of each input node, and thus obtain a neural net that only uses a small subset of the original variables. In addition, we develop an effective algorithm based on backward path-wise optimization to yield stable solution paths, in order to tackle the challenge of complex optimization landscapes. Our extensive simulation studies and real data examples demonstrate satisfactory finite-sample performances of the proposed estimator, in feature selection and prediction for modeling continuous, binary, and time-to-event outcomes.

1 Introduction

Over the past decades, advancements in molecular, imaging, and other laboratory tests have led to a growing interest in high-dimensional data analysis (HDDA) (Donoho et al., 2000). This type of data involves a large number of observed variables relative to the small sample size, which presents a considerable challenge in building accurate and interpretable models. For example, in bioinformatics, hundreds of thousands of RNA expressions, genome-wide association study (GWAS) data, and genomic data are used to understand disease biology and the correlation with clinical outcomes, with only hundreds of patients involved (Visscher et al., 2012; Hertz et al., 2016; Kim & Halabi, 2016; Beltran et al., 2017). To address the curse of dimensionality, feature selection is a critical step in HDDA modeling. By identifying the most relevant features that capture the relationship with clinical outcomes, feature selection enhances model interpretability and improves generalization.

There are various methods for feature selection, including filter methods (Koller & Sahami, 1996; Guyon & Elisseeff, 2003; Gu et al., 2012), wrapper methods (Kohavi & John, 1997; Inza et al., 2004; Tang et al., 2014), and embedded methods (Tibshirani, 1996; Zou, 2006; Fan & Li, 2001; Zhang et al., 2010). Among them, penalized regression methods have become very popular in HDDA since the introduction of the least absolute shrinkage and selection operator (LASSO) (Tibshirani, 1996). Penalized regression method can perform simultaneous parameter estimation and feature selection by shrinking some of the parameter coefficients to exact zeros. While LASSO has been widely used to obtain sparse estimates in machine learning and statistics, it tends to select unimportant variables to compensate for the over-shrinkage for relevant variables (Zou, 2006). To address the bias and inconsistent feature selection of LASSO, several methods have been proposed, including adaptive LASSO (Zou, 2006), the minimax concave penalty (MCP) (Zhang et al., 2010), and the smoothly clipped absolute deviation (SCAD) (Fan & Li, 2001).

However, most of these penalized methods assume linearity in the relationship between the variables and the outcomes, while the actual functional form of the relationship may not be available in many applications. Some additive non-parametric extensions have been proposed to resolve this problem (Lin & Zhang, 2006; Ravikumar et al., 2009; Meier et al., 2009), but their models rely on sums of univariate or low-dimensional functions and may not be able to capture the complex interactions between multiple covariates. Yamada et al. (2014) propose the HSIC-LASSO approach that leverages kernel learning for feature selection while uncovering non-linear feature interactions. However, it suffers from quadratic scaling in computational complexity with respect to the number of observations.

Neural networks are powerful tools for modeling complex relationships in a wide range of applications, from imaging (Krizhevsky et al., 2017; He et al., 2016) and speech recognition (Graves et al., 2013; Chan et al., 2016) to natural language processing (Young et al., 2018; Devlin et al., 2018) and financial forecasting (Fischer & Krauss, 2018). Their state-of-the-art performance has been achieved through powerful computational resources and the use of large sample sizes. Despite that, high-dimensional data can still lead to overfitting and poor generalization performance for neural networks (Liu et al., 2017).

Recently, there have been novel developments in using regularized neural networks for feature selection or HDDA. A line of research focuses on utilizing the regularized neural networks, specifically employing the group LASSO technique to promote sparsity among input nodes (Liu et al., 2017; Scardapane et al., 2017; Feng & Simon, 2017). These methods treat all outgoing connections from a single input neuron as a group and apply the LASSO penalty to the l_2 norm of weight vectors of each group. Other LASSO-regularized neural networks in feature selection can be found in the work of Li et al. (2016) and Lemhadri et al. (2021). However, LASSO-regularized neural networks tend to over-shrink the weights of relevant variables, leading to the inclusion of many false positives. The adaptive LASSO was employed to alleviate this problem (Dinh & Ho, 2020), yet their results are limited to continuous outcomes and assume that the conditional mean function is exactly a neural network. The work in Yamada et al. (2020) bypassed the l_1 regularization by introducing stochastic gates to the input layer of neural networks. They considered l_0-like regularization based on a continuous relaxation of the Bernoulli distribution. Their method, however, requires a cutoff value for selecting variables with weak signals, and the stochastic gate is unable to completely exclude the non-selected variables during model training and prediction stages.

In this paper, we propose a novel framework for sparse-input neural networks using group concave regularization to overcome the limitations of existing feature selection methods. Although folded concave penalties like MCP and SCAD have been shown to perform well in both theoretical and numerical settings for feature selection and prediction, they have not received the same level of attention as LASSO in the context of machine learning. Our proposed framework aims to draw attention to the underutilized potential of concave penalties for feature selection in neural networks by providing a comprehensive approach for simultaneous feature selection and function estimation in both low- and high-dimensional settings.

The key contributions of this paper are as follows:

- A unified framework for simultaneous feature selection and prediction: We introduce structured sparsity in neural networks by applying concave group penalties, treating all outgoing connections from a single input neuron as a group. An l_2-norm-based concave penalty shrinks entire groups of weights to zero, resulting in a parsimonious neural network that selects only a small subset of input variables.

- A stable optimization algorithm for group concave penalties: We employ composite gradient descent for optimization and explore backward path-wise optimization from the perspective of solution paths in neural networks. This perspective enhances model selection stability and improves the interpretability of optimization paths.

- Empirical validation across diverse data types: Through extensive simulations and real-data experiments, we demonstrate that our method outperforms existing approaches in feature selection consistency and prediction accuracy across continuous, binary, and time-to-event outcomes.

The rest of this article is organized as follows. In Section 2, we formulate the problem of feature selection for a generic non-parametric model and introduce our proposed method. The implementation of the method, including the composite gradient descent algorithm and the backward path-wise optimization, is presented in Section 3. In Section 4, we conduct extensive simulation studies to demonstrate the performance of the proposed method. The application of the method to real-world datasets is presented in Section 5. Lastly, in Section 6, we discuss the results and their implications. The implementation details and supplementary numerical results are provided in the Appendix.

2 Method

2.1 Problem setup

Let $X \in \mathbb{R}^d$ be a d-dimensional random vector and Y be a response variable. We assume the conditional distribution $P_{Y|X}$ depends on a form of $f(X_S)$ with a function $f \in F$ and a subset of variables $S \subseteq \{1, \cdots, d\}$. We are interested in identifying the true set S for important variables and estimating function f so that we can predict Y based on selected variable X_S.

At the population level, we aim to minimize the loss

$$\min_{f \in F, S} \mathbb{E}_{X,Y} \ell(f(X_S), Y)$$

where ℓ is a loss function tailored to a specific problem. In practical settings, the distribution of (X, Y) is often unknown, and instead only an independent and identically distributed (i.i.d.) random sample of size n is available, consisting of pairs of observations $(X_i, Y_i)_{i=1}^n$. Additionally, if the number of variables d is large, an exhaustive search over all possible subsets S becomes computationally infeasible. Furthermore, we do not assume any specific form of the unknown function f and aim to approximate f nonparametrically using neural networks. Thus, our goal is to develop an efficient method that can simultaneously select a variable subset S and approximate the solution f for any given class of functions using a sparse-input neural network.

2.2 Proposed framework

We consider function estimators based on feedforward neural networks. Let $\mathcal{F}_n$ be a class of feed forward neural networks $f_{\mathbf{w}} : \mathbb{R}^d \mapsto \mathbb{R}$ with parameter $\mathbf{w}$. The architecture of a multi-layer perceptron (MLP) can be expressed as a composition of a series of functions

$$f_{\mathbf{w}}(x) = L_D \circ \sigma \circ L_{D-1} \circ \sigma \circ \cdots \circ \sigma \circ L_1 \circ \sigma \circ L_0(x), x \in \mathbb{R}^d,$$

where $\circ$ denotes function composition and $\sigma(x)$ is an activation function defined for each component of x. Additionally,

$$L_i(x) = \mathbf{W}_i x + b_i, i = 0, 1, \ldots, \mathcal{D},$$

where $\mathbf{W}_i \in \mathbb{R}^{d_{i+1} \times d_i}$ is a weight matrix, D is the number of hidden layers, d_i is the width defined as the number of neurons of the i-th layer with $d_0 = d$, and $b_i \in \mathbb{R}^{d_{i+1}}$ is the bias vector in the i-th linear transformation L_i. Note that the vector $\mathbf{w} \in \mathbb{R}^P$ is the column-vector concatenation of all parameters in $\{\mathbf{W}_i, b_i : i = 0, 1, \ldots, \mathcal{D}\}$. We define the empirical loss of $f_{\mathbf{w}}$ as

$$\mathcal{L}_n(\mathbf{w}) = \frac{1}{n} \sum_{i=1}^n \ell(f_{\mathbf{w}}(X_i), Y_i).$$

Ideally, the sparse-input neural network $f_{\mathbf{w}}$ should rely only on the important variables, meaning that $\mathbf{W}_{0,j} = \mathbf{0}$ for $j \notin S$, where $\mathbf{W}_{0,j}$ denotes the jth column vector of $\mathbf{W}_0$. In order to minimize the empirical loss $\mathcal{L}_n(\mathbf{w})$ while inducing sparsity in $\mathbf{W}_0$, we propose to train the neural network by minimizing the following group regularized empirical loss

$$\hat{\mathbf{w}} = \underset{\mathbf{w}\in\mathbb{R}^P}{\arg\min} \left\{ \mathcal{L}_n(\mathbf{w}) + \sum_{j=1}^{d} \rho_\lambda(\|\mathbf{W}_{0,j}\|_2) + \alpha\|\mathbf{w}\|_2^2 \right\}, \tag{1}$$

where $\|\cdot\|_2$ denotes the Euclidean norm of a vector.

The objective function in Eq. (1) comprises three components:

(1) $\mathcal{L}_n(\mathbf{w})$ is the empirical loss function, such as the mean squared error loss for regression tasks, the cross-entropy loss for classification tasks, and the negative log partial likelihood for proportional hazards models. Further details can be found in Appendix A.

(2) ρ_λ is a concave penalty function parameterized by $\lambda \geq 0$. To simultaneously select variables and learn the neural network, we group the outgoing connections from each single input neuron that corresponds to each variable. The concave penalty function ρ_λ is designed to shrink the weight vectors of specific groups to exact zeros, resulting in a neural network that utilizes only a small subset of the original variables.

(3) $\alpha\|\mathbf{w}\|_2^2$, where $\alpha > 0$, represents the ridge regularization term used to prevent overfitting in neural networks. Note that feature selection, employing ρ_λ, depends exclusively on the magnitudes of weights in the input layer. However, it is possible to diminish the influence of ρ_λ by reducing all weights in the input layer while simultaneously allowing larger weights in other layers, without affecting the network's output. The ridge regularization addresses this issue by promoting smaller, well-balanced weights, thereby improving model stability and mitigating overfitting.

It should be noted that when the number of hidden layers $D = 0$, the function $f_{\mathbf{w}}$ reduces to a linear function, and the optimization problem in Eq. (1) becomes the framework of elastic net (Zou & Hastie, 2005), SCAD-L_2 (Zeng & Xie, 2014), and Mnet (Huang et al., 2016), with the choice of ρ_λ to be LASSO, SCAD, and MCP, respectively.

2.3 Concave regularization

There are several commonly used penalty functions that encourage sparsity in the solution, such as LASSO (Tibshirani, 1996), SCAD (Fan & Li, 2001), and MCP (Zhang et al., 2010). When applied to the l_2-norm of the coefficients associated with each group of variables, these penalty functions give rise to group regularization methods, including group LASSO (GLASSO) (Yuan & Lin, 2006), group SCAD (GSCAD) (Guo et al., 2015), and group MCP (GMCP) (Huang et al., 2012). Specifically, LASSO, SCAD, and MCP are defined as follows.

- **LASSO**

$$\rho_\lambda(t) = \lambda|t|.$$

- **SCAD**

$$\rho_\lambda(t) = \begin{cases} \lambda|t| & \text{for } |t| \leq \lambda, \\ -\frac{t^2 - 2a\lambda|t| + \lambda^2}{2(a-1)} & \text{for } \lambda < |t| \leq a\lambda, \\ \frac{(a+1)\lambda^2}{2} & \text{for } |t| > a\lambda, \end{cases}$$

  where $a > 2$ is fixed.

- **MCP**

$$\rho_\lambda(t) = \text{sign}(t)\lambda \int_0^{|t|} \left(1 - \frac{z}{\lambda a}\right)_+ dz,$$

  where $a > 0$ is fixed.

It has been demonstrated, both theoretically and numerically, that the folded concave regularization methods of SCAD and MCP exhibit strong performance in terms of feature selection and prediction (Fan & Li, 2001; Zhang et al., 2010). Unlike the convex penalty LASSO, which tends to over-regularize large terms and provide inconsistent feature selection, concave regularization can reduce LASSO's bias and improve model selection accuracy. The rationale behind the concave penalty lies in the behavior of its derivatives. Specifically, SCAD and MCP initially apply the same level of penalization as LASSO, but gradually reduce the penalization rate until it drops to zero when $t > a\lambda$. Given the benefits of the concave penalization, we propose using the group concave regularization in our framework for simultaneous feature selection and function estimation.

3 Implementation

3.1 Composite gradient descent

The optimization in Eq. (1) is not a convex optimization problem since both empirical loss function $\mathcal{L}_n(\mathbf{w})$ and the penalty function ρ_λ can be non-convex. To obtain the stationary point, we use the composite gradient descent algorithm (Nesterov, 2013). This algorithm is also incorporated in Feng & Simon (2017); Lemhadri et al. (2021) for sparse-input neural networks based on the LASSO regularization. The local convergence of the composite gradient descent algorithm for nonconvex regularization was established in Gong et al. (2013).

Denote $\bar{\mathcal{L}}_{n,\alpha}(\mathbf{w}) = \mathcal{L}_n(\mathbf{w}) + \alpha\|\mathbf{w}\|_2^2$ as the smooth component of the objective function in Eq. (1). The composition gradient iteration for epoch t is given by

$$\mathbf{w}^{t+1} = \arg\min_{\mathbf{w}} \left\{ \frac{1}{2}\|\mathbf{w} - \tilde{\mathbf{w}}^{t+1}\|_2^2 + \gamma \sum_{j=1}^{d} \rho_\lambda(\|\mathbf{W}_{0,j}\|_2) \right\}, \tag{2}$$

where $\tilde{\mathbf{w}}^{t+1} = \mathbf{w}^t - \gamma\nabla\bar{\mathcal{L}}_{n,\alpha}(\mathbf{w}^t)$ is the gradient update only for the smooth component $\bar{\mathcal{L}}_{n,\alpha}(\mathbf{w}^t)$ that can be computed using the standard back-propagation algorithm. Here $\gamma > 0$ is the step size for the update and can be set as a fixed value or determined by employing the backtracking line search method, as described in Nesterov (2013). Let A_j represent the index set of $\mathbf{W}_{0,j}$ within $\mathbf{w}$. We define A as the index set that includes all weights in the input layer, given by $A = \{\bigcup_{j=1}^{d} A_j\}$. By solving Eq. (2), we obtain the iteration form $\mathbf{w}_{A^c}^{t+1} = \tilde{\mathbf{w}}_{A^c}^{t+1}$ and

$$\mathbf{w}_{A_j}^{t+1} = h(\tilde{\mathbf{w}}_{A_j}^{t+1}; \gamma, \lambda), \quad \text{for } j = 1, \cdots, d. \tag{3}$$

Here, A^c refers to the complement of the set A, and the function h represents the thresholding operator, which can be determined by the specific penalty ρ_λ. By taking ρ_λ to be the LASSO, MCP, and SCAD penalty, it can be verified that the GLASSO, GSCAD, and GMCP solutions for the iteration in Eq. (3) have the following form:

- GLASSO
$$h_{\text{GLASSO}}(z; \gamma, \lambda) = S_{\text{g}}(z, \gamma\lambda).$$
- GSCAD
$$h_{\text{GSCAD}}(z; \gamma, \lambda) = \begin{cases} S_{\text{g}}(z, \gamma\lambda), & \text{if } \|z\|_2 \le (\gamma+1)\lambda, \\ \frac{a-1}{a-1-\gamma} S_{\text{g}}(z, \frac{a\gamma\lambda}{a-1}), & \text{if } (\gamma+1)\lambda < \|z\|_2 \le a\lambda, \\ z, & \text{if } \|z\|_2 > a\lambda. \end{cases}$$
- GMCP
$$h_{\text{GMCP}}(z; \gamma, \lambda) = \begin{cases} \frac{a}{a-\gamma} S_{\text{g}}(z, \gamma\lambda), & \text{if } \|z\|_2 \le a\lambda, \\ z, & \text{if } \|z\|_2 > a\lambda, \end{cases}$$

where $S_{\text{g}}(z; \lambda)$ is the group soft-thresholding operator defined as

$$S_{\text{g}}(z; \lambda) = \left(1 - \frac{\lambda}{\|z\|_2}\right)_+ z.$$

Therefore, we can efficiently implement the composite gradient descent by integrating an additional thresholding operation into the input layer. This operation follows the gradient descent step using the smooth component $\bar{\mathcal{L}}_{n,\alpha}(\mathbf{w})$. The calculation for epoch t can be summarized as follows:

(1) Compute the gradient of the smooth component $\nabla\bar{\mathcal{L}}_{n,\alpha}(\mathbf{w}^t)$ using back-propagation.

(2) Perform the gradient update for the smooth component to get an intermediate estimate:

$$\tilde{\mathbf{w}}^{t+1} \leftarrow \mathbf{w}^t - \gamma\nabla\bar{\mathcal{L}}_{n,\alpha}(\mathbf{w}^t).$$

(3) Apply the thresholding operator to obtain the updated weights $\mathbf{w}^{t+1}$:

$$\mathbf{w}^{t+1}_{A^c} \leftarrow \tilde{\mathbf{w}}^{t+1}_{A^c}, \quad \mathbf{w}^{t+1}_{A_j} \leftarrow h\left(\tilde{\mathbf{w}}^{t+1}_{A_j}; \gamma, \lambda\right), \quad \text{for } j = 1, \ldots, d.$$

The final index set of the selected variables is $\hat{S} = \{j : \hat{\mathbf{w}}_{A_j} \neq \mathbf{0}\}$. Note that we consider γ as a scaling factor in the thresholding operator. When $\gamma = 1$ in Step (3), the solutions for GLASSO, GSCAD, and GMCP align with the closed-form results established in Wei & Zhu (2012).

3.2 Backward path-wise optimization

We are interested in learning neural networks not only for a specific value of λ, but also for a range of λs where the networks vary by the number of included variables. Specifically, we consider a range of λ from λ_{min}, where the networks include all or an excessively large number of variables, up to λ_{max}, where all variables are excluded and $|W_0|$ becomes a zero matrix. Since the objective function is not convex and has multiple local minima, the solution of Eq. (1) with random initialization may not vary continuously for $\lambda \in [\lambda_{min}, \lambda_{max}]$, resulting in a highly unstable path of solutions that are regularized by λ.

To address this issue, we consider a path-wise optimization strategy by varying the regularization parameter along a path. In this approach, we use the solution of a particular value of λ as a warm start for the next problem. Regularized linear regression methods (Friedman et al., 2007; 2010; Breheny & Huang, 2011) typically adopt a forward path-wise optimization, starting from a null model with all variables excluded at λ_{max} and working forward with decreasing λs. However, our numerical studies on sparse-input neural networks revealed that starting with a sparse solution as the initial model does not lead to a smooth transition to a dense model. To tackle this problem, we implement a backward path-wise optimization approach, starting from a dense model at the minimum value of λ_{min} and solving toward sparse models up to λ_{max} with all variables excluded from the network. This dense-to-sparse warm start approach is also employed in (Lemhadri et al., 2021) using LASSO regularization.

To further illustrate the importance of using backward path-wise optimization in regularized neural networks, we investigate variables selection and function estimation of a regression model $Y = f(X) + \epsilon$, where $f(X) = \log(|X_1| + 0.1) + X_1X_2 + X_2 + \exp(X_3 + X_4)$ with 4 informative and 16 nuisance variables, and each X_i and ϵ follow the standard normal distribution. More details of the simulations are presented in Section 4. Figure 1 shows the solution paths of GSCAD, GMCP, and GLASSO based on different types of optimization. It is observed that non-pathwise optimization (top-left) leads to fluctuations or variations in the solution path, whereas forward path-wise optimization tends to remain in the same sparse model (GMCP, top-middle) or experience fluctuation solutions (GLASSO, top-right), until transitioning to a full model with a sufficiently small λ. In contrast, backward path-wise optimization (bottom panels) yields smoother solution paths for informative and nuisance variables. Notably, GLASSO (bottom-right) tends to over-shrink the weight vectors of informative variables and include more variables in the model. In contrast, GSCAD (bottom-left) and GMCP (bottom-middle) are designed to prevent over-shrinkage and offer a smooth transition from the full to the null model.

In addition to providing stable and smooth solution paths, backward path-wise optimization is advantageous computationally. In particular:

- The consecutive estimates of weights in the path are close, which reduces the rounds of gradient descent needed for each iteration. Therefore, the bulk of the computational cost occurs at λ_{min}, and a lower number of iterations for the remaining λs results in low computational costs.

- We observe that the excluded variables from previous solutions are rarely included in the following solutions. By pruning the inputs of the neural network along the solution path, further reduction in computation complexity can be achieved as the model becomes sparse. Since the computational cost scales with the number of input features, this approach can significantly speed up computation, particularly for high-dimensional data.

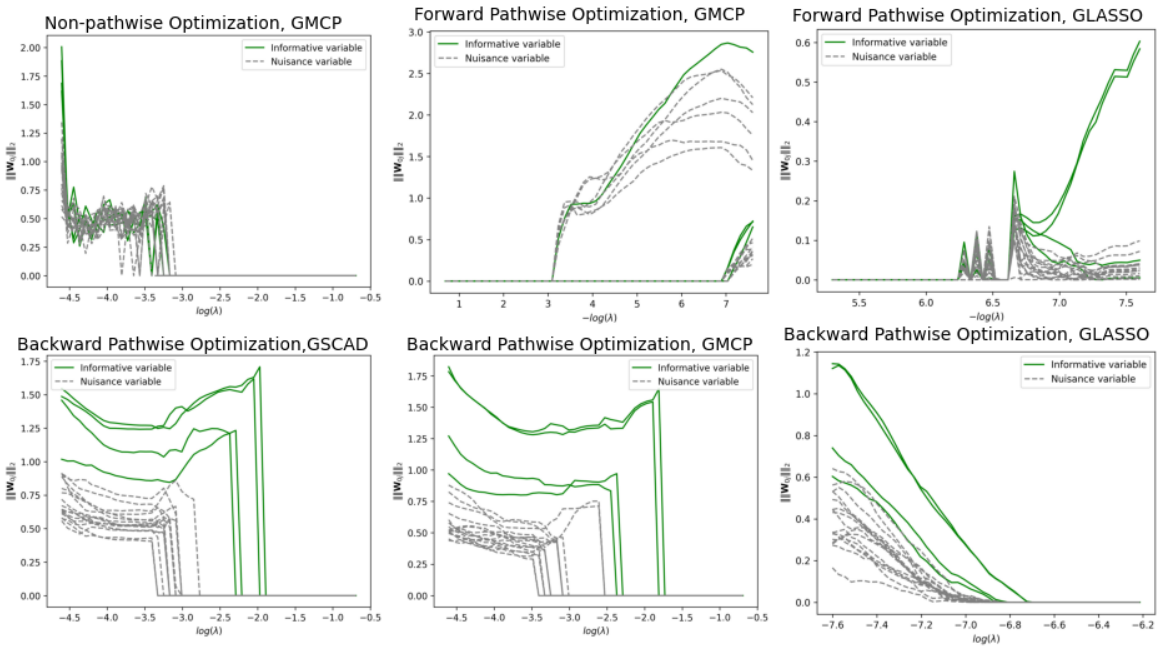

Figure 1: **Solution path of l_2 norm of the weight vector associated with each input node $\|\mathbf{W}_{0j}\|_2$. Top left**: Non-pathwise optimization using GMCP. All the neural network weights are initialized by drawing from $N(0, 0.1)$ for each λ. **Top middle**: forward path-wise optimization using GMCP. It starts from the null model and computes the solution with decreasing λ. Random initialization is used before the selection of the first set of variables. **Top right**:forward path-wise optimization using GLASSO. **Bottom left**: backward path-wise optimization using GSCAD. **Bottom middle**: backward path-wise optimization using GMCP. **Bottom right**: backward path-wise optimization using GLASSO.

3.3 Tuning Parameter Selection

Two tuning parameters are required in our proposed framework: the group penalty coefficient λ and the ridge penalty coefficient α. The former controls the number of selected variables and yields sparser models for larger values of λ, while the latter imposes a penalty on the size of the network weights to prevent overfitting.

In all numerical studies presented in this paper, we adopted a 20% holdout validation set from the training data. The model was trained using the remaining data, and the optimal values for λ and α were selected from a fine grid of values based on their performance on the validation set.

Python code and examples for the proposed group concave regularized neural networks are available at `https://github.com/r08in/GCRNN`.

4 Simulation Studies

We assess the performance of the proposed regularized neural networks in feature selection and prediction through several simulation settings with various types of outcomes. In particular, we consider the concave regularization GMCP and GSCAD for our proposed framework. We name the method of regularized neural networks using GLASSO, GMCP, and GSCAD as GLASSONet, GMCPNet, and GSCADNet, respectively. We compare the proposed group concave regularized estimator GMCPNet and GSCADNet with GLASSONet, neural network (NN) without feature selection ($\lambda = 0$), random (survival) forest (RF), and the STG method proposed in Yamada et al. (2020). We also include the oracle version of NN and RF (Oracle-NN and Oracle-RF) as benchmarks, where true relevant variables are known in advance and used directly in the model fitting process. In our implementation, we replace the gradient update in Step (2) with Adam to improve computational efficiency and achieve faster convergence in practice. Specifically, we set the scaling factor $\gamma = 1$ in the thresholding operator and used a base learning rate of LR $= 0.001$ for Adam. For a fair comparison across all neural network methods, we used a ReLU-activated multi-layer perceptron (MLP) with two hidden layers of 10 and 5 units, respectively. A sensitivity analysis of these choices is provided in Section 4.3, and additional implementation details are can be found in Appendix E.

4.1 Preliminary Simulation Study

In this section, we consider regression models of XOR-type and hierarchical signal structures for continuous outcomes. We generate 500 i.i.d. random training samples according to the following models:

- **XOR-type Signals:** $Y = X_1 + X_2 + \epsilon$,
- **Hierarchical Signals:** $Y = X_1 + X_1X_2 + \epsilon$.

In both models, X_1 and X_2 represent the first two coordinates of the covariate vector $X \in \mathbb{R}^d$, each taking values in $\{\pm 1\}$ with equal probability, while ϵ denotes a standard normal error term. The remaining coordinates, X_i for $i = 3, \ldots, d$, are uninformative variables. We conducted 20 simulations for varying d values ranging from 20 to 500. For each simulation, the performance of the trained model in both prediction and variable selection was evaluated on 500 independently generated random samples. For prediction accuracy, we report the test R^2 scores. For variable selection, we report the false positive rate (FPR)—the percentage of selected but unimportant covariates, defined as FPR $= \frac{|\hat{S} \cap S^c|}{|S^c|} \times 100\%$; and the false negative rate (FNR)—the percentage of important but non-selected covariates, defined as FNR $= \frac{|\hat{S}^c \cap S|}{|S|} \times 100\%$. Recall that S represents the true index sets of important variables and $\hat{S} = \{j : \|\mathbf{W}_{0,j}\|_2 \neq 0\}$ denote the index sets of selected variables. Simulation results are shown in Figure 2.

Our proposed methods, GMCPNet and GSCADNet, demonstrate clear advantages over other approaches, achieving relatively high prediction scores with low FPR and FNR across both models. Notably, neural networks without feature selection tend to overfit as the number of noisy features increases, underscoring the importance of effective feature selection. Additionally, GLASSONet exhibits a tendency to overselect variables, resulting in a high FPR due to the bias introduced by the LASSO penalty.

4.2 High-Dimensional Simulations with Continuous, Binary, and Time-to-Event Outcomes

We evaluate our proposed methods under a more complex nonlinear pattern across various outcome types, considering both low- and high-dimensional scenarios. The data are generated through the following function:

$$f(X) = \log(|X_1| + 0.1) + X_1X_2 + X_2 + \exp(X_3 + X_4),$$

where each component of the covariate vector $X = (X_1, \cdots, X_d)^T \in \mathbb{R}^d$ are generated from independent standard normal distribution. Here $d > 4$ and function $f(X)$ is sparse that only the first four variables are relevant to the outcome. We generate n i.i.d. random samples with continuous outcomes, binary outcomes, and time-to-event outcomes in the following three examples, respectively.

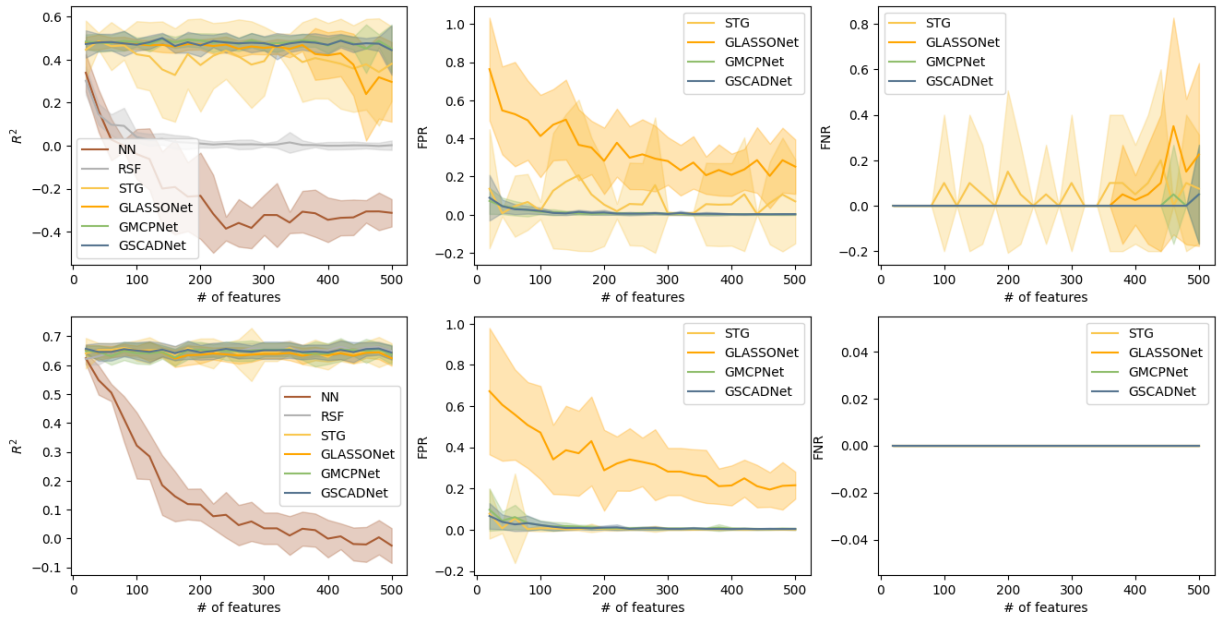

Figure 2: **Top row**: simulation results for the model of XOR-type signals. **Bottom row**: simulation results for the model of hierarchical signals. The R^2 scores, false positive rate (FPR), and false negative rate (FNR) are presented in the left, middle, and right columns, respectively. The central lines are the means while the shaded areas represent standard deviations.

Example 4.1 (Regression Model) *The continuous response Y is generated from a standard regression model with an additive error as follows*

$$Y = f(X) + \epsilon,$$

where ϵ follows a standard normal distribution.

Example 4.2 (Classification Model) *The binary response $Y \in \{0, 1\}$ is generated from a Bernoulli distribution with the following conditional probability*

$$P(Y = 1|X) = \frac{1}{1 + \exp\left(-f(X)\right)}.$$

Example 4.3 (Proportional Hazards Model) *The survival time T follows the proportional hazards model with a hazard function of the form*

$$h(t|X) = h_0(t) \exp\left(f(X)\right), \tag{4}$$

where $h_0(t)$ is the baseline hazard function. Thus, $T = H_0^{-1}\left(-\log(U) \exp f(X)\right)$, where U is a uniform random variable in $[0, 1]$, and H_0 is the baseline cumulative hazard function defined as $H_0(t) = \int_0^t h_0(u)du$. We considered a Weibull hazard function for H_0, with the scale parameter $= 2$ and the shape parameter $= 2$. A proportion $\mathcal{C}$ of the n samples is randomly selected to be censored. The censoring indicator is defined as $\delta_i = 1$ for observed events and $\delta_i = 0$ for censored observations. The observed time for the ith individual is

$$Y_i = T_i \mathbb{I}(\delta_i = 1) + C_i \mathbb{I}(\delta_i = 0),$$

where T_i is the event time and C_i is the censoring time. For censored individuals, C_i is drawn from a uniform distribution $(0, T_i)$, ensuring that censoring precedes the event. In our simulation studies, we consider censoring proportions $\mathcal{C} = 0$, 0.2, and 0.4.

For each example, we consider the low and high dimensional settings in the following scenarios:

1. Low dimension (LD): $d = 20$ and $n = 300$ and 500.
2. High dimension (HD): $d = 1000$ and $n = 500$.

We perform 200 simulations for each scenario. Similar to Section 4.1, the performance of the trained model is evaluated on independently generated n random samples. For prediction accuracy, we report the R^2 score, classification accuracy, and C-index for the regression, classification, and proportional hazards model (PHM), respectively. In addition to FPR and FNR, we also report the model size (MS), which is the average number of selected covariates.

4.2.1 Results

Table 1 presents a summary of the feature selection performance of the four approaches: STG, GLASSONet, GMCPNet, and GSCADNet, across all simulation scenarios. We exclude the results of the STG method for Example 4.3 as it either selects all variables or none of them for the survival outcome. For both LD and HD settings, GMCPNet and GSCADNet consistently outperform the STG and GLASSONet in terms of feature selection. These models exhibit superior performance, achieving model sizes that closely matched the true model, along with low FPR and FNR for most scenarios. While STG performs well in certain LD settings, it tends to over select variables in HD scenarios with a large variability in the model size. On the other hand, GLASSONet is prone to selecting more variables, leading to larger model sizes in both LD and HD settings, which aligns with the inherent nature of the LASSO penalty.

Figure 3 displays the distribution of testing prediction scores for the regression, classification, and PHM with a censoring rate of $\mathcal{C} = 0.2$. The complete results of the PHM are presented in Appendix B. GMCPNet and GSCADNet demonstrate comparable performance in both LD and HD settings, achieving similar results to the Oracle-NN and outperforming NN, RF, and even Oracle-RF in most scenarios. STG performs similarly to Oracle-NN in the LD setting of the regression model, but its performance deteriorates in the HD setting and other models. Conversely, while GLASSONet outperforms or is comparable to the Oracle-RF method in the LD settings, it suffers from overfitting in the HD settings by including a large number of false positives in the final model.

It is worth pointing out that the Oracle-NN outperforms the Oracle-RF in every scenario, indicating that neural network-based methods can serve as a viable alternative to tree-based methods when the sample size is sufficiently large relative to the number of predictors.

Overall, the simulation results demonstrate the superior performance of the concave penalty in terms of feature selection and prediction. The proposed GMCPNet and GSCADNet methods exhibit remarkable capabilities in selecting important variables with low FPR and low FNR, while achieving accurate predictions across various models. These methods show promise for tackling the challenges of feature selection and prediction in high-dimensional data.

4.3 Hyperparameter Sensitivity Analysis

We evaluate the robustness of the proposed method by conducting a sensitivity analysis under a high-dimensional regression setting ($d = 1000, n = 500$), as described in Example 1. The analysis examines the effect of three key hyperparameters: the thresholding scaling factor (γ), the learning rate (LR) used in the Adam optimizer, and the network structure. Similar to previous examples, 200 simulations are performed, and the average values of R^2, FPR, and FNR are reported for each configuration. Since GMCPNet exhibits similar performance to GSCADNet, we focus on reporting the results for GSCADNet. The findings are summarized in Figure 4, which highlights the sensitivity to each hyperparameter individually while keeping the other parameters fixed. The fixed hyperparameter choices used in this study ($\gamma = 1$, LR $= 0.001$, and network structure $[10, 5]$, i.e., two hidden layers with 10 and 5 nodes, respectively) are marked on the plots for reference.

For γ, the results show that $\gamma = 1$ and $\gamma = 0.1$ yield similar performance across all metrics, achieving relatively high R^2 while maintaining low FPR and FNR. In contrast, smaller values of γ $(0.001, 0.01)$ tend to under-select relevant features, as indicated by higher FNR and consequently lower R^2 scores. For LR,

Table 1: **Feature selection results of STG, GLASSONet, GMCPNet, and GSCADNet under the regression, classification, and proportional hazards models.** The false positives rate (FPR %), false negatives rate (FNR %), and model size (MS) with standard deviation (SD) in parentheses are displayed.

Model	Method	$n = 300$, $d = 20$		$n = 500$, $d = 20$		$n = 500$, $d = 1000$	
		FPR, FNR	MS (SD)	FPR, FNR	MS (SD)	FPR, FNR	MS (SD)
Regression	STG	7.8, 5.4	5.0 (2.0)	7.2, 2.1	5.1 (1.7)	1.6, 12.1	19.2 (28.0)
	GLASSONet	86.7, 4.4	17.7 (4.7)	96.0, 0.6	19.3 (2.2)	24.3, 29.2	245.0 (98.7)
	GMCPNet	2.2, 4.5	4.2 (1.0)	2.1, 4.2	4.2 (1.0)	0.0, 5.8	4.1 (0.9)
	GSCADNet	2.4, 5.0	4.2 (1.1)	2.0, 3.2	4.2 (0.9)	0.0, 7.1	4.1 (1.0)
Classification	STG	25.3, 16.5	7.4 (6.9)	10.1, 11.0	5.2 (4.8)	3.8, 15.6	40.9 (183.4)
	GLASSONet	89.2, 1.0	18.2 (2.8)	94.7, 0.2	19.1 (2.0)	16.3, 21.5	165.4 (92.9)
	GMCPNet	14.4, 3.9	6.2 (3.6)	9.3, 0.8	5.5 (2.6)	0.3, 16.2	6.5 (4.2)
	GSCADNet	11.6, 5.8	5.6 (2.9)	7.0, 1.0	5.1 (1.9)	0.3, 16.8	6.8 (5.9)
Survival ($\mathcal{C} = 0$)	GLASSONet	97.2, 0.0	19.5 (1.0)	99.2, 0.0	19.9 (0.5)	18.2, 20.0	184.8 (56.2)
	GMCPNet	1.6, 0.4	4.2 (0.6)	0.8, 0.0	4.1 (0.4)	0.0, 1.5	4.1 (0.5)
	GSCADNet	1.9, 0.2	4.3 (0.6)	1.2, 0.0	4.2 (0.5)	0.0, 1.6	4.1 (0.7)
Survival ($\mathcal{C} = 0.2$)	GLASSONet	98.0, 0.1	19.7 (0.8)	99.6, 0.0	19.9 (0.3)	16.8, 18.0	170.6 (49.0)
	GMCPNet	1.9, 0.4	4.3 (0.9)	1.7, 0.0	4.3 (1.0)	0.0, 2.6	4.2 (0.9)
	GSCADNet	1.8, 0.2	4.3 (0.9)	1.7, 0.1	4.3 (0.8)	0.0, 3.5	4.1 (0.7)
Survival ($\mathcal{C} = 0.4$)	GLASSONet	95.0, 0.0	19.2 (1.7)	98.8, 0.0	19.8 (0.5)	15.2, 19.9	154.6 (48.4)
	GMCPNet	5.8, 8.1	4.6 (1.5)	1.2, 0.1	4.2 (0.5)	0.0, 4.2	4.1 (1.0)
	GSCADNet	4.8, 7.5	4.5 (1.3)	1.7, 0.0	4.3 (0.7)	0.0, 4.9	4.2 (1.0)

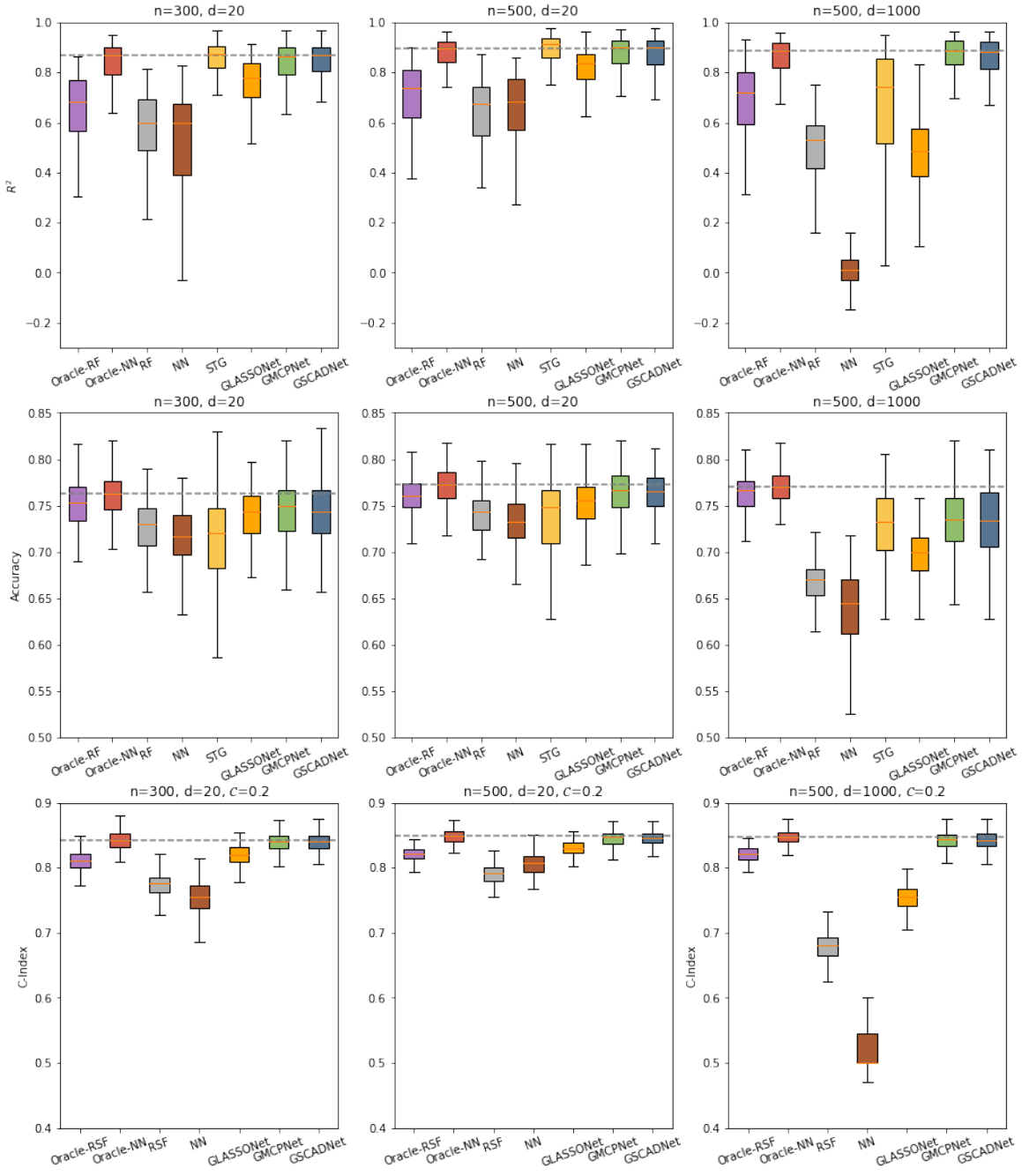

Figure 3: **Top row**: R^2 score of the proposed methods for the regression model outlined in Example 4.1. **Middle row**: Accuracy of the proposed methods for the classification model outlined in Example 4.2. **Bottom row**: C-Index of the proposed methods for the survival model outlined in Example 4.3. The dashed lines represent the median score of the Oracle-NN, used as a benchmark for comparison.

values of 0.001, 0.01, and 0.1 exhibit stable performance with competitive R^2, low FPR, and low FNR. Smaller LR values (0.0001) lead to slower convergence, resulting in lower R^2 and higher FNR, while larger LR values (0.1) slightly improve R^2 and reduce FNR but also increase FPR. For network structures, more complex architectures such as $[10, 10, 5]$ and $[20, 10, 5]$ provide modest improvements in R^2, though they also result in increased computational costs and potentially a higher risk of overfitting. In summary, although we fixed these parameters in our implementation with LR= 0.001, $\gamma = 1$, and network structure $[10, 5]$), our analysis reveals that selection accuracy and prediction performance remain stable for γ between 0.1 and 1, LR between 0.001 and 0.1, and Network structures ranging from [10,5] to more complex architectures like [20,10,5]. These findings suggest that our method is robust to a reasonable range of hyperparameter choices, demonstrating consistent performance across different configurations.

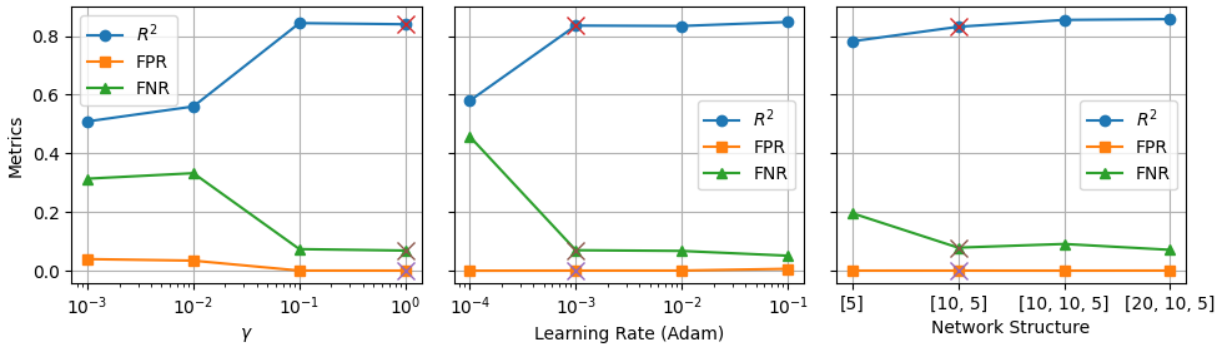

Figure 4: Sensitivity analysis of GSCADNet to hyperparameters: **Left**: γ, the scaling factor for the thresholding operator. **Middle**: Learning rate (LR) in the Adam optimizer. **Right**: Network structure. The network structure $[l_1, l_2, \ldots, l_k]$ represents the number of nodes in each hidden layer. The fixed choices used in our numerical study ($\gamma = 1$, LR $= 0.001$, and network structure $[10, 5]$) are marked on the plots with an"x" symbol for each metric (R^2, FPR, and FNR).

5 Real Data Example

5.1 Survival Analysis on CALGB-90401 dataset

We utilize the data from the CALGB-90401 study, a double-blinded phase III clinical trial that compares docetaxel and prednisone with or without bevacizumab in men with metastatic castration-resistant prostate cancer (mCRPC) to illustrate the performance of our proposed method. The CALGB-90401 data consists of 498,801 single-nucleotide polymorphisms (SNPs) that are processed from blood samples from patients. We assume a dominant model for SNPs and thus each of the SNPs is considered as a binary variable. Since our interest is studying the DNA damage repair genes, we only consider 625 SNPs based on an updated literature search (Mateo et al., 2015; Wyatt et al., 2016; Beltran et al., 2011; Mosquera et al., 2013; Robinson et al., 2015; Abida et al., 2019; De Laere et al., 2017). We also include the eight clinical variables that have been identified as prognostic markers of overall survival in patients with mCRPC (Halabi et al., 2014): opioid analgesic use (PAIN), ECOG performance status, albumin (ALB), disease site (defined as lymph node only, bone metastases with no visceral involvement, or any visceral metastases), LDH greater than the upper limit of normal (LDH.High), hemoglobin (HGB), PSA, and alkaline phosphatase (ALKPHOS). The final dataset contains $d = 635$ variables, $n = 631$ patients and a censoring rate $C = 6.8\%$.

We consider the PHM in the form of Eq. (4) for our proposed methods to identify clinical variables or SNPs that can predict the primary outcome of overall survival in these patients. To evaluate the feature selection and prediction performance of the methods, we randomly split the dataset 100 times into training sets (n=526) and testing sets (n=105) using a 5:1 allocation ratio. We apply the methods to each of the training sets and calculate the time-dependent area under the receiver operating characteristic curve (tAUC) on the corresponding testing sets. The tAUC assesses the discriminative ability of the predicted model and is computed using the Uno method (Uno et al., 2007). The results of the 100 random splits are presented in

Figure 5. Our proposed method, GSCADNet, outperforms the others in survival prediction (left panel). It is worth noting that the NN method, which lacks feature selection, tends to overfit in high-dimensional data and performs poorly. Although these three regularized methods of sparse-input neural networks perform similarly in survival prediction, GLASSONet has a tendency to over-select variables and the proposed GMCPNet and GSCADNet select a relatively smaller set of variables without compromising prediction performance (middle panel). The right panel of Figure 5 demonstrates that GSCADNet successfully selects most of the key clinical variables and detects some of the important SNPs in predicting overall survival.

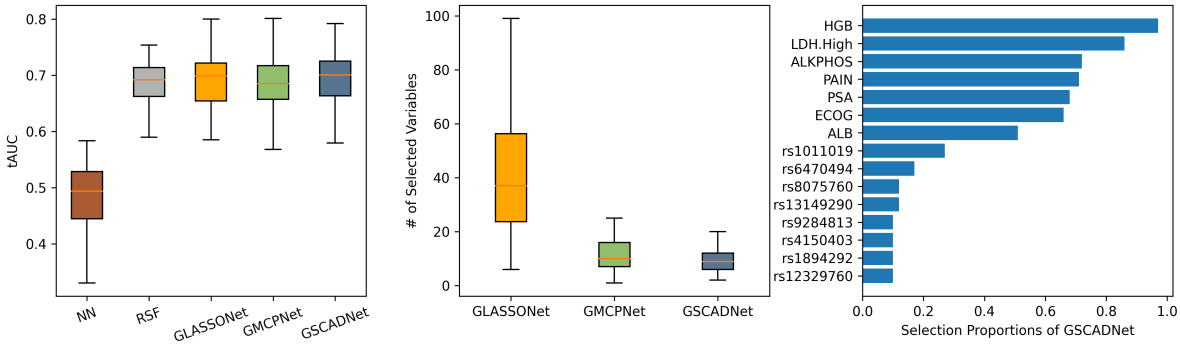

Figure 5: **Left**: Boxplots of tAUC from testing set over 100 random splits. **Middle**: the number of selected variables for GLASSONet, GMCPNet, and GSCADNet. **Right**: Variables selected by GSCADNet with selection proportion$\geq 10\%$ over 100 random splits.

5.2 Classification on High-Dimensional MNIST

We aim to visualize variable selection in a high-dimensional binary classification setting using the MNIST dataset. The MNIST dataset is a well-known benchmark dataset in computer vision, consisting of grayscale images of handwritten digits from 0 to 9. In this study, we focus on distinguishing digits 7 and 8, which share structural similarities that make the classification task nontrivial. While other digit pairs may also exhibit visual similarity, this choice provides a meaningful evaluation of feature selection methods in identifying relevant pixels for classification. We evaluate our proposed methods GMCPNet and GSCADNet, along with existing methods GLASSONet, STG, NN, and RF, based on their feature selection and classification accuracy.

The MNIST dataset consists of grayscale images with 28×28 pixels, resulting in 784 variables. To create a high-dimensional, low-sample setting, we construct a training dataset by selecting 250 images of 7s and 8s each, yielding $d = 784$ features and $n = 500$ samples. Importantly, the class labels depend primarily on the central pixels, meaning an effective feature selection method should correctly identify and focus on these relevant regions. To ensure the feature space is not inherently sparse, we introduce i.i.d. standard Gaussian noise to the images. The trained models are evaluated on the testing dataset with 2002 images. We repeated the process of random sampling and model fitting 100 times, and the feature (pixel) selection and classification results are shown in Figure 6. We observe that GLASSONet, GMCPNet, GSCADNet all achieve median accuracies greater than 91%, outperforming the other methods. While the heatmaps of feature selection show that GLASSONet, GMCPNet, GSCADNet consistently select relevant pixels in high frequencies, GLASSONet tends to over select variables and GMCPNet and GSCADNet choose irrelevant pixels in much lower frequencies (indicated by dark red colors).

6 Discussion

In this paper, we have proposed a novel framework that utilizes group concave regularization for feature selection and function estimation in complex modeling, specifically designed for sparse-input neural networks. Unlike the convex penalty LASSO, the concave regularization methods such as MCP and SCAD gradu-

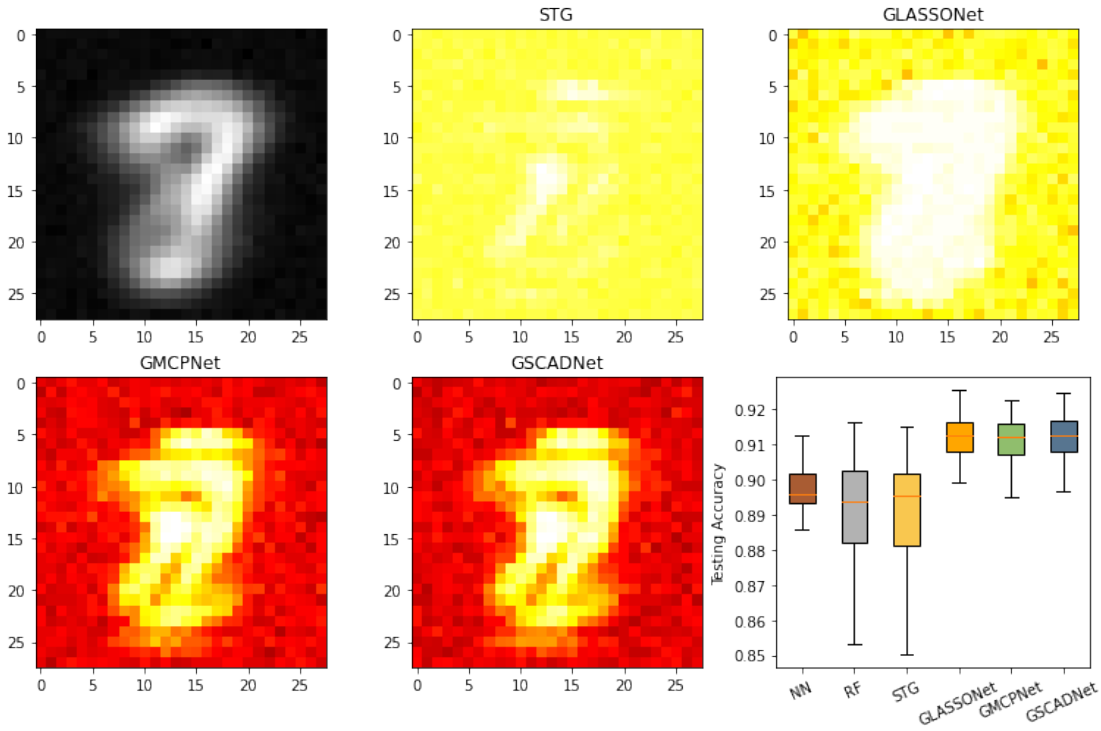

Figure 6: **Comparing feature selection and classification performance by STG, GLASSONet, GMCPNet, and GSCADNet. Top left**: the image that takes the average of all images in the training set and shows relevant pixels in grayscale. **Bottom right**: testing accuracy for the classification of 7s and 8s in a high-dimensional, low-sample MNIST setting with training dataset size $d = 784$ and $n = 500$. **Other panels**: heatmaps depicting the selection frequencies of each pixel across 100 repetitions for each method. Lighter colors indicate higher selection frequencies, with white highest and darker colors lowest.

ally reduce the penalization rate for large terms, preventing over-shrinkage and improving model selection accuracy. Our optimization algorithm, based on the composite gradient descent, is simple to implement, requiring only an additional thresholding operation after the regular gradient descent step on the smooth component. Furthermore, we incorporate backward path-wise optimization to efficiently navigate the optimization landscape across a fine grid of tuning parameters, generating a smooth solution path from dense to sparse models. This path-wise optimization approach improves stability and computational efficiency, potentially enhancing the applicability of our framework for sparse-input neural networks.

Among numerous feature selection methods, penalized regression has gained substantial popularity. However, many of these methods rely on the assumption and application of linear theory, which may not capture the complex relationships between covariates and the outcome of interest. In biomedical research, for instance, researchers often normalize data and employ penalized techniques under a linear model for feature selection. However, relying solely on data transformation risks overlooking intricate biological relationships and fails to address the dynamic nature of on-treatment biomarkers. Moreover, advancements in molecular and imaging technologies have introduced challenges in understanding the non-linear relationships between high-dimensional biomarkers and clinical outcomes. Novel approaches are urgently needed to tackle these complexities, leading to an improved understanding of non-linear relationships and optimizing patient treatment and care.

The runtime of our proposed method over a solution path of λs (with a fixed α) can be comparable to or even shorter than training a single model with a fixed λ, such as the NN method without feature selection ($\lambda = 0$). To illustrate this, we examine the algorithm complexity of the NN method, which can be approximated as $\mathcal{O}(ndT)$, where T denotes the number of epochs for learning the neural network. In contrast, training our proposed method over a solution path of m λs has a complexity of $\mathcal{O}(n\bar{d}T'm)$, where $\bar{d}$ represents the averaged number of inputs along the solution path with dimension pruning, and T' is the number of epochs for each λ in the path. In our simulation with the HD scenario ($d = 1000$), we set $T = 5000$, $T' = 200$, and $m = 50$. Assuming the number of inputs decreases equally along the solution path from the full model to the null model, we have $\bar{d} = d/2 = 500$. Thus, $ndT = n\bar{d}T'm$ indicates that solving for an entire path of our proposed method requires a similar computation as training a single model. In real applications, especially in high-dimensional scenarios, the dimensionality usually drops quickly along the solution path. Therefore, $\bar{d}$ can be much smaller than $d/2$, and thus solving for a whole solution path can be more computationally efficient. It is worth pointing out that we set T' to be small for the first parameter $\lambda_{\min}$ as well in the HD setting, to avoid overfitting of an initial dense model.

One limitation of the proposed method arises in ultra-high dimensional scenarios where the number of variables reaches hundreds of millions. Directly applying the proposed sparse-input neural networks in such cases can lead to an exceedingly complex optimization landscape, making it computationally infeasible. A potential solution is to employ a pre-screening step to reduce dimensionality before applying the proposed approach (Fan & Lv, 2010).

Another limitation of the proposed group-regularized method is its focus on individual feature selection. This limitation becomes particularly relevant when dealing with covariates exhibiting grouping structures, such as a group of indicator variables representing a multilevel categorical covariate, or scientifically meaningful groups based on prior knowledge. A potential future research direction could involve redefining the groups within the proposed framework. This could be achieved by considering all outgoing connections from a group of input neurons as a single group, enabling group selection and accommodating the presence of grouping structures.

In conclusion, our study demonstrates the advantages of employing group concave regularization for sparse-input neural networks. The findings highlight its effectiveness in consistently selecting relevant variables and accurately modeling complex non-linear relationships between covariates and outcomes across both low- and high-dimensional settings. The proposed approach holds promising potential to enhance modeling strategies and find wide-ranging applications, particularly in diseases characterized by non-linear biomarkers, such as oncology and infectious diseases. A key future direction is the development of theoretical properties of our method, including selection consistency and estimation performance guarantees, to further strengthen its theoretical foundation and applicability.

Acknowledgments

This research was supported in part by the National Institutes of Health Grants R01CA256157, R01CA249279, R21CA263950, U01CA287008, the United States Army Medical Research Materiel Command grant Award Number HT9425-23-1-0393, the Food and Drug Administration (FDA) Award 1U01FD007857-01 of the U.S. Department of Health and Human Services (HHS), and the Prostate Cancer Foundation Challenge Award.

A Empirical Loss Function

The empirical loss functions $\mathcal{L}_n(\mathbf{w})$ for regression, classification, and survival models in Examples 4.1-4.3 are defined as follows:

- Mean squared error loss for regression tasks. This loss function measures the average squared difference between the true values Y_i and the predictions $f_{\mathbf{w}}(X_i)$:

$$\mathcal{L}_n(\mathbf{w}) = \frac{1}{n}\sum_{i=1}^{n}(Y_i - f_{\mathbf{w}}(X_i))^2.$$

- Cross-entropy loss for classification tasks. It is widely used in classification problems and quantifies the dissimilarity between the true labels Y_i and the predicted probabilities $\hat{Y}_i$ of class 1. The predicted probability $\hat{Y}_i$ is obtained by applying the sigmoid function to $f_{\mathbf{w}}(X_i)$:

$$\mathcal{L}_n(\mathbf{w}) = -\frac{1}{n}\sum_{i=1}^{n}\left[Y_i \log(\hat{Y}_i) + (1 - Y_i)\log(1 - \hat{Y}_i)\right].$$

- Negative log partial likelihood for proportional hazards models. It is derived from survival analysis and aims to maximize the likelihood of observing events while considering censoring information. It incorporates the event indicator δ_i, which is 1 if the event of interest occurs at time Y_i and 0 if the observation is right-censored. The negative log partial likelihood is defined as:

$$\mathcal{L}_n(\mathbf{w}) = -\frac{1}{n}\sum_{i=1}^{n}\left\{\delta_i f_{\mathbf{w}}(X_i) - \delta_i \log\sum_{j\in R_i}\exp(f_{\mathbf{w}}(X_j))\right\}.$$

  Here, $R_i = \{j : Y_j \geq Y_i\}$ represents the risk set just before time Y_i. The negative log partial likelihood is specifically used in the proportional hazards model.

B Complete Results for Survival Model

Figure 7 shows that larger variations in C-index are associated with larger censoring rates overall. GMCPNet and GSCADNet achieve comparable results to Oracle-NN while surpassing all other methods, including Oracle-RSF.

C Simulation with Correlated Variables

The simulation study in Section 4 focuses on independent covariates. However, in real-world applications, particularly in high-dimensional settings, the presence of correlations among covariates is common and presents a challenge for feature selection. In this section, we assess the effectiveness of the proposed method using simulated data that incorporates correlated variables.

To be more specific, we extend the high-dimensional scenario described in Section 4 by generating a correlated covariate vector, denoted as $\mathbf{X} \sim N(0, \Sigma)$. The correlation structure is defined using a power decay pattern,

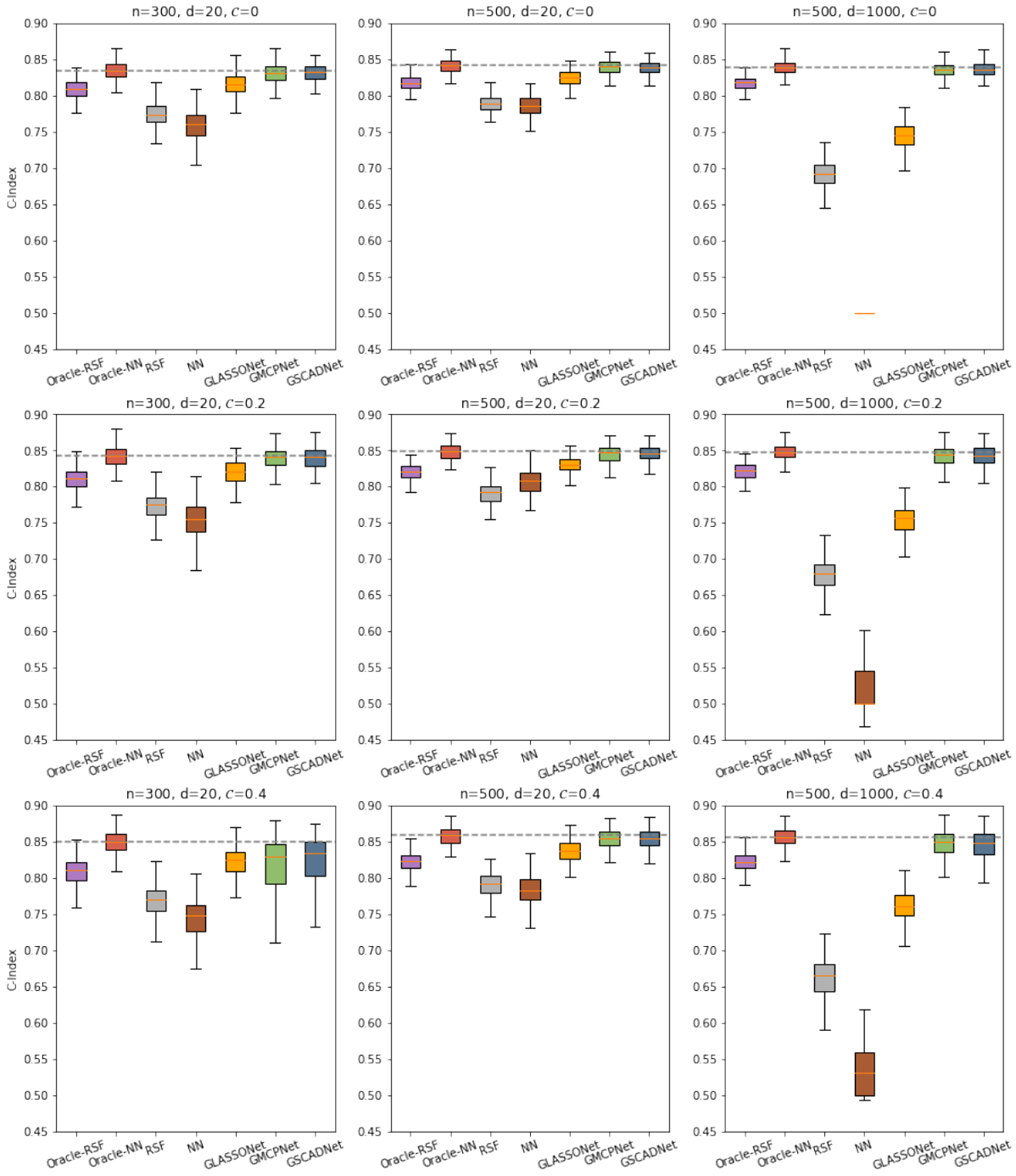

Figure 7: **C-Index of the proposed methods for the survival model outlined in Example 4.3.** The dashed line represents the median C-Index of the Oracle-NN, used as a benchmark for comparison.

where $\Sigma_{ij} = 0.5^{|i-j|}$. This modification allows us to examine the performance of our method in the presence of correlation among the covariates. Comparing the results of feature selection for independent covariates in Table 1 to the outcomes presented in Table 2, it becomes evident that STG and GLSSONet exhibit larger variations in selected model sizes, along with higher false negative rates (FNR) and false positive rates (FPR) in the regression model. This behavior can be attributed to the presence of correlated features. In contrast, the proposed GMCPNet and GSCADNet methods effectively identify relevant variables while maintaining relatively low false positive and negative rates across all models. Furthermore, Figure 8 demonstrates that both GMCPNet and GSCADNet perform comparably to the Oracle-NN method in the regression and survival models, while outperforming other non-oracle approaches in the classification model. These findings indicate that the proposed methods exhibit robustness against correlations among covariates in terms of feature selection and model prediction.

Table 2: **Feature selection results of STG, GLASSONet, GMCPNet, and GSCADNet using correlated features in high-dimensional scenario ($n = 500, d = 1000$).** The False positives rate (FPR %), False negatives rate (FNR %), and model size (MS) with standard deviation (SD) in parentheses are displayed.

Method	Regression		Classification		Survival ($\mathcal{C} = 0.2$)	
	FPR, FNR	MS (SD)	FPR, FNR	MS (SD)	FPR, FNR	MS (SD)
STG	8.4, 16.6	86.8(132.6)	1.5, 21.0	18.6(121.1)	-, -	-(-)
GLASSONet	28.8, 26.6	290.0(144.6)	19.3, 22.4	195.7(116.4)	16.0, 1.9	163.1(51.4)
GMCPNet	0.1, 13.4	4.0(1.4)	0.2, 13.9	5.5(4.5)	0.0, 0.0	4.1(0.4)
GSCADNet	0.1, 13.2	4.0(1.2)	0.1, 11.8	4.8(2.9)	0.0, 0.0	4.1(0.6)

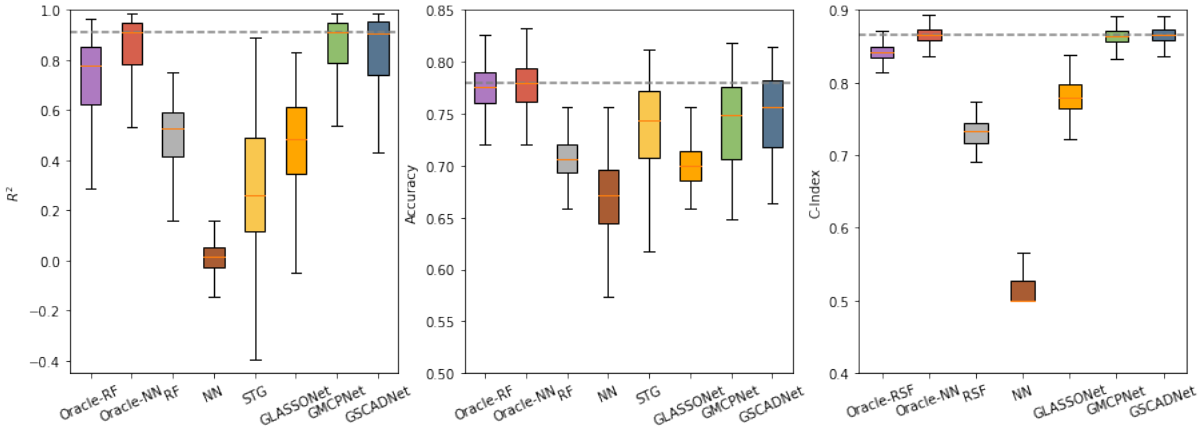

Figure 8: **Prediction scores of the proposed methods for the regression, classification, and survival models ($\mathcal{C} = 0.2$) using correlated features in high-dimensional scenario ($n = 500, d = 1000$).** The dashed lines represent the median score of the Oracle-NN, used as a benchmark for comparison.

D Understanding the Impact of LASSO Bias on Feature Selection

Our numerical study demonstrates that GLASSONet tends to select many noisy variables due to the bias introduced by the LASSO penalty. In contrast, the group-concave regularization used in our proposed framework (e.g., GMCPNet and CSCADNet) reduces this bias and improves feature selection accuracy. To further investigate the impact of LASSO's bias on feature selection, we propose a modified version of GLASSONet, applying a relaxed LASSO approach and terming it relaxed-GLASSONet.

The relaxed-GLASSONet method follows a two-stage procedure: for each group LASSO parameter λ, we first select features using GLASSONet. Then, we refit a standard neural network with only the selected features by setting $\lambda = 0$, thereby reducing bias during model fitting. The final model is selected based on its predictive performance on a validation set. Our goal is to explore whether the relaxed-GLASSONet can mitigate the feature overselection observed in the LASSO-regularized approach by removing the bias, and ultimately enhance prediction performance.

We apply the relaxed-GLASSONet method to synthetic data generated from the XOR-type signal regression model, repeating the simulation described in Section 4.1. We compare relaxed-GLASSONet with GLASSONet, as well as our proposed GMCPNet and GSCADNet methods, with results presented in Figure 9. Our results indicate that relaxed-GLASSONet selects significantly fewer false positives across varying feature counts, thereby improving prediction accuracy. Notably, relaxed-GLASSONet performs comparably to GMCPNet and GLASSONet in low-dimensional settings, but its performance declines as dimensionality increases. These findings confirm that the LASSO penalty tends to over-select features to compensate for its inherent bias. This overselection can be mitigated by reducing bias through model refitting with only the selected features, leading to more accurate feature selection.

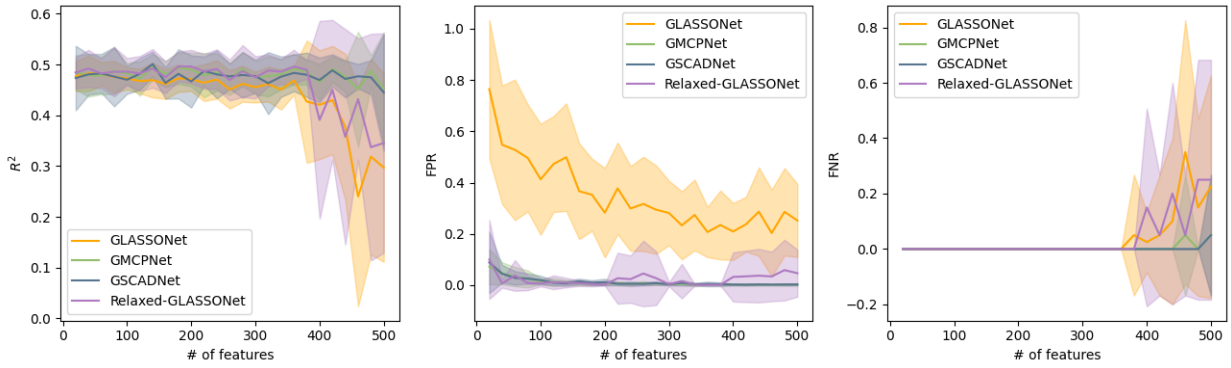

Figure 9: **Simulation results of relaxed-GLASSONet for XOR-Type signal model.** The R^2 scores, false positive rate (FPR), and false negative rate (FNR) are presented in the left, middle, and right panels, respectively. The central lines are the means while the shaded areas represent standard deviations.

E Implementation Details

E.1 Simulation studies

To ensure a fair comparison among all the neural-net-based methods, we adopted a ReLu-activated Multi-Layer Perceptron (MLP) with two hidden layers consisting of 10 and 5 units, respectively. The network weights were initialized by sampling from a Gaussian distribution with mean 0 and standard deviation 0.1, while the bias terms were set to 0 following the Xavier initialization technique (Glorot & Bengio, 2010). The optimization of the neural networks was performed using the Adam optimizer with a base learning rate (LR) of 0.001.

For all the methods falling within the framework of Equation (1) in the paper, we selected the optimal values of λ and α from a two-dimensional grid, with λ and α ranging over 50 and 10 evenly spaced values on a logarithmic scale, respectively. The selection was based on their performance on the validation set, which consisted of 20% of the training set. The parameter search ranges are displayed in Table 3. We set $\lambda = 0$ for NN and Oracle-NN to deactivate feature selection. For GLASSONet, GMCPNet, and GSCADNet, the number of epochs at λ_{min} was set to 2000 for the low-dimensional (LD) scenario and 200 for the high-dimensional (HD) scenario. For all other values of λ, the number of epochs was set to 200 for both LD and HD settings. The number of epochs for NN was consistently fixed at 5000.

We employed Random Forest (RF) with 1000 decision trees for the model fitting process. We implemented the STG method as described in Yamada et al. (2020) that the LR and regularization parameter λ were optimized via Optuna with 500 trials, using 10% of the training set as a validation set. The number of epochs was 2000 for each trial.

Table 3: **List of the search range for the tuning parameters used in our simulation.**

Param	Search range	
	LD	HD
λ	[1e-3, 0.5]	[1e-2, 0.5]
α	[1e-3, 0.1]	[1e-2, 0.1]
LR (STG)	[1e-4,0.1]	[1e-4, 0.1]
λ (STG)	[1e-3, 10]	[1e-2, 100]
λ (LASSONet)	[5e-4, 2e-3]	[5e-4, 2e-3]

E.2 Real Data Example

In the analysis of real data examples, the implementation details remain the same as the high dimension (HD) scenario in the simulation studies, with the following modifications:

- For the survival analysis on the CALGB-90401 dataset, we utilized the MLP with two hidden layers, each consisting of 10 nodes. In hyperparameter tuning, we explored 100 values of λ ranging from 0.01 to 0.1 for GMCPNet and GSCADNet. Additionally, we increase the number of candidates for α to 50.
- In the classification task on the MNIST dataset, we adjust the search range of α to [1e-3, 0.1].

The data from CALGB 90401 is available from the NCTN Data Archive at https://nctn-data-archive.nci.nih.gov/. The MNIST dataset is retrieved using their official source.

References

Wassim Abida, Joanna Cyrta, Glenn Heller, Davide Prandi, Joshua Armenia, Ilsa Coleman, Marcin Cieslik, Matteo Benelli, Dan Robinson, Eliezer M Van Allen, et al. Genomic correlates of clinical outcome in advanced prostate cancer. *Proceedings of the National Academy of Sciences*, 116(23):11428–11436, 2019.

Himisha Beltran, David S Rickman, Kyung Park, Sung Suk Chae, Andrea Sboner, Theresa Y MacDonald, Yuwei Wang, Karen L Sheikh, Stéphane Terry, Scott T Tagawa, et al. Molecular characterization of neuroendocrine prostate cancer and identification of new drug targets. *Cancer discovery*, 1(6):487–495, 2011.

Himisha Beltran, Alexander W Wyatt, Edmund C Chedgy, Adam Donoghue, Matti Annala, Evan W Warner, Kevin Beja, Michael Sigouros, Fan Mo, Ladan Fazli, et al. Impact of therapy on genomics and transcriptomics in high-risk prostate cancer treated with neoadjuvant docetaxel and androgen deprivation therapy-molecular analysis high-risk pca after neoadjuvant therapy. *Clinical Cancer Research*, 23(22):6802–6811, 2017.

Patrick Breheny and Jian Huang. Coordinate descent algorithms for nonconvex penalized regression, with applications to biological feature selection. *The annals of applied statistics*, 5(1):232, 2011.

William Chan, Navdeep Jaitly, Quoc Le, and Oriol Vinyals. Listen, attend and spell: A neural network for large vocabulary conversational speech recognition. In *2016 IEEE international conference on acoustics, speech and signal processing (ICASSP)*, pp. 4960–4964. IEEE, 2016.

Bram De Laere, Pieter-Jan van Dam, Tom Whitington, Markus Mayrhofer, Emanuela Henao Diaz, Gert Van den Eynden, Jean Vandebroek, Jurgen Del-Favero, Steven Van Laere, Luc Dirix, et al. Comprehensive profiling of the androgen receptor in liquid biopsies from castration-resistant prostate cancer reveals novel intra-ar structural variation and splice variant expression patterns. *European urology*, 72(2):192–200, 2017.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018.

Vu Dinh and Lam Si Tung Ho. Consistent feature selection for neural networks via adaptive group lasso. *arXiv preprint arXiv:2006.00334*, 2020.

David L Donoho et al. High-dimensional data analysis: The curses and blessings of dimensionality. *AMS math challenges lecture*, 1(2000):32, 2000.

Jianqing Fan and Runze Li. Variable selection via nonconcave penalized likelihood and its oracle properties. *Journal of the American statistical Association*, 96(456):1348–1360, 2001.

Jianqing Fan and Jinchi Lv. A selective overview of variable selection in high dimensional feature space. *Statistica Sinica*, 20(1):101, 2010.

Jean Feng and Noah Simon. Sparse-input neural networks for high-dimensional nonparametric regression and classification. *arXiv preprint arXiv:1711.07592*, 2017.

Thomas Fischer and Christopher Krauss. Deep learning with long short-term memory networks for financial market predictions. *European journal of operational research*, 270(2):654–669, 2018.

Jerome Friedman, Trevor Hastie, Holger Höfling, and Robert Tibshirani. Pathwise coordinate optimization. *The annals of applied statistics*, 1(2):302–332, 2007.

Jerome Friedman, Trevor Hastie, and Rob Tibshirani. Regularization paths for generalized linear models via coordinate descent. *Journal of statistical software*, 33(1):1, 2010.

Xavier Glorot and Yoshua Bengio. Understanding the difficulty of training deep feedforward neural networks. In *Proceedings of the thirteenth international conference on artificial intelligence and statistics*, pp. 249–256. JMLR Workshop and Conference Proceedings, 2010.

Pinghua Gong, Changshui Zhang, Zhaosong Lu, Jianhua Huang, and Jieping Ye. A general iterative shrinkage and thresholding algorithm for non-convex regularized optimization problems. In *international conference on machine learning*, pp. 37–45. PMLR, 2013.

Alex Graves, Abdel-rahman Mohamed, and Geoffrey Hinton. Speech recognition with deep recurrent neural networks. In *2013 IEEE international conference on acoustics, speech and signal processing*, pp. 6645–6649. Ieee, 2013.

Quanquan Gu, Zhenhui Li, and Jiawei Han. Generalized fisher score for feature selection. *arXiv preprint arXiv:1202.3725*, 2012.

Xiao Guo, Hai Zhang, Yao Wang, and Jiang-Lun Wu. Model selection and estimation in high dimensional regression models with group scad. *Statistics & Probability Letters*, 103:86–92, 2015.

Isabelle Guyon and André Elisseeff. An introduction to variable and feature selection. *Journal of machine learning research*, 3(Mar):1157–1182, 2003.

Susan Halabi, Chen-Yen Lin, W Kevin Kelly, Karim S Fizazi, Judd W Moul, Ellen B Kaplan, Michael J Morris, and Eric J Small. Updated prognostic model for predicting overall survival in first-line chemotherapy for patients with metastatic castration-resistant prostate cancer. *Journal of Clinical Oncology*, 32(7): 671, 2014.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 770–778, 2016.

Daniel L Hertz, Kouros Owzar, Sherrie Lessans, Claudia Wing, Chen Jiang, William Kevin Kelly, Jai Patel, Susan Halabi, Yoichi Furukawa, Heather E Wheeler, et al. Pharmacogenetic discovery in calgb (alliance) 90401 and mechanistic validation of a vac14 polymorphism that increases risk of docetaxel-induced neuropathyvac14 snp predicts docetaxel-induced neuropathy. *Clinical Cancer Research*, 22(19): 4890–4900, 2016.

Jian Huang, Patrick Breheny, and Shuangge Ma. A selective review of group selection in high-dimensional models. *Statistical science: a review journal of the Institute of Mathematical Statistics*, 27(4), 2012.

Jian Huang, Patrick Breheny, Sangin Lee, Shuangge Ma, and Cun-Hui Zhang. The mnet method for variable selection. *Statistica Sinica*, pp. 903–923, 2016.

Inaki Inza, Pedro Larranaga, Rosa Blanco, and Antonio J Cerrolaza. Filter versus wrapper gene selection approaches in dna microarray domains. *Artificial intelligence in medicine*, 31(2):91–103, 2004.

Sangjin Kim and Susan Halabi. High dimensional variable selection with error control. *BioMed research international*, 2016, 2016.

Ron Kohavi and George H John. Wrappers for feature subset selection. *Artificial intelligence*, 97(1-2): 273–324, 1997.

Daphne Koller and Mehran Sahami. Toward optimal feature selection. Technical report, Stanford InfoLab, 1996.

Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. *Communications of the ACM*, 60(6):84–90, 2017.

Ismael Lemhadri, Feng Ruan, Louis Abraham, and Robert Tibshirani. Lassonet: A neural network with feature sparsity. *The Journal of Machine Learning Research*, 22(1):5633–5661, 2021.

Yifeng Li, Chih-Yu Chen, and Wyeth W Wasserman. Deep feature selection: theory and application to identify enhancers and promoters. *Journal of Computational Biology*, 23(5):322–336, 2016.

Yi Lin and Hao Helen Zhang. Component selection and smoothing in multivariate nonparametric regression. *The Annals of Statistics*, 34(5):2272 – 2297, 2006. doi: 10.1214/009053606000000722. URL `https://doi.org/10.1214/009053606000000722`.

Bo Liu, Ying Wei, Yu Zhang, and Qiang Yang. Deep neural networks for high dimension, low sample size data. In *IJCAI*, pp. 2287–2293, 2017.

Joaquin Mateo, Suzanne Carreira, Shahneen Sandhu, Susana Miranda, Helen Mossop, Raquel Perez-Lopez, Daniel Nava Rodrigues, Dan Robinson, Aurelius Omlin, Nina Tunariu, et al. Dna-repair defects and olaparib in metastatic prostate cancer. *New England Journal of Medicine*, 373(18):1697–1708, 2015.

Lukas Meier, Sara Van de Geer, and Peter Bühlmann. High-dimensional additive modeling. *The Annals of Statistics*, 37(6B):3779–3821, 2009.

Juan Miguel Mosquera, Himisha Beltran, Kyung Park, Theresa Y MacDonald, Brian D Robinson, Scott T Tagawa, Sven Perner, Tarek A Bismar, Andreas Erbersdobler, Rajiv Dhir, et al. Concurrent aurka and mycn gene amplifications are harbingers of lethal treatmentrelated neuroendocrine prostate cancer. *Neoplasia*, 15(1):1–IN4, 2013.

Yu Nesterov. Gradient methods for minimizing composite functions. *Mathematical Programming*, 140(1): 125–161, 2013.

Pradeep Ravikumar, John Lafferty, Han Liu, and Larry Wasserman. Sparse additive models. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 71(5):1009–1030, 2009.

Dan Robinson, Eliezer M Van Allen, Yi-Mi Wu, Nikolaus Schultz, Robert J Lonigro, Juan-Miguel Mosquera, Bruce Montgomery, Mary-Ellen Taplin, Colin C Pritchard, Gerhardt Attard, et al. Integrative clinical genomics of advanced prostate cancer. *Cell*, 161(5):1215–1228, 2015.

Simone Scardapane, Danilo Comminiello, Amir Hussain, and Aurelio Uncini. Group sparse regularization for deep neural networks. *Neurocomputing*, 241:81–89, 2017.

Jiliang Tang, Salem Alelyani, and Huan Liu. Feature selection for classification: A review. *Data classification: Algorithms and applications*, pp. 37, 2014.

Robert Tibshirani. Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society. Series B (Methodological)*, pp. 267–288, 1996.

Hajime Uno, Tianxi Cai, Lu Tian, and Lee-Jen Wei. Evaluating prediction rules for t-year survivors with censored regression models. *Journal of the American Statistical Association*, 102(478):527–537, 2007.

Peter M Visscher, Matthew A Brown, Mark I McCarthy, and Jian Yang. Five years of gwas discovery. *The American Journal of Human Genetics*, 90(1):7–24, 2012.

Fengrong Wei and Hongxiao Zhu. Group coordinate descent algorithms for nonconvex penalized regression. *Computational statistics & data analysis*, 56(2):316–326, 2012.

Alexander W Wyatt, Arun A Azad, Stanislav V Volik, Matti Annala, Kevin Beja, Brian McConeghy, Anne Haegert, Evan W Warner, Fan Mo, Sonal Brahmbhatt, et al. Genomic alterations in cell-free dna and enzalutamide resistance in castration-resistant prostate cancer. *JAMA oncology*, 2(12):1598–1606, 2016.

Makoto Yamada, Wittawat Jitkrittum, Leonid Sigal, Eric P Xing, and Masashi Sugiyama. High-dimensional feature selection by feature-wise kernelized lasso. *Neural computation*, 26(1):185–207, 2014.

Yutaro Yamada, Ofir Lindenbaum, Sahand Negahban, and Yuval Kluger. Feature selection using stochastic gates. In *International Conference on Machine Learning*, pp. 10648–10659. PMLR, 2020.

Tom Young, Devamanyu Hazarika, Soujanya Poria, and Erik Cambria. Recent trends in deep learning based natural language processing. *ieee Computational intelligenCe magazine*, 13(3):55–75, 2018.

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

Lingmin Zeng and Jun Xie. Group variable selection via scad-l 2. *Statistics*, 48(1):49–66, 2014.

Cun-Hui Zhang et al. Nearly unbiased variable selection under minimax concave penalty. *The Annals of statistics*, 38(2):894–942, 2010.

Hui Zou. The adaptive lasso and its oracle properties. *Journal of the American statistical association*, 101 (476):1418–1429, 2006.

Hui Zou and Trevor Hastie. Regularization and variable selection via the elastic net. *Journal of the royal statistical society: series B (statistical methodology)*, 67(2):301–320, 2005.