Feature Selection Using a Neural Network With Group Lasso Regularization and Controlled Redundancy

Jian Wang, *Member, IEEE*, Huaqing Zhang, Junze Wang, Yifei Pu, and Nikhil R. Pal

***Abstract*— We propose a neural network-based feature selection (FS) scheme that can control the level of redundancy in the selected features by integrating two penalties into a single objective function. The Group Lasso penalty aims to produce sparsity in features in a grouped manner. The redundancy-control penalty, which is defined based on a measure of dependence among features, is utilized to control the level of redundancy among the selected features. Both the penalty terms involve the $L_{2,1}$-norm of weight matrix between the input and hidden layers. These penalty terms are nonsmooth at the origin, and hence, one simple but efficient smoothing technique is employed to overcome this issue. The monotonicity and convergence of the proposed algorithm are specified and proved under suitable assumptions. Then, extensive experiments are conducted on both artificial and real data sets. Empirical results explicitly demonstrate the ability of the proposed FS scheme and its effectiveness in controlling redundancy. The empirical simulations are observed to be consistent with the theoretical results.**

***Index Terms*— Control redundancy, convergence, feature selection (FS), Group Lasso, monotonicity, neural networks.**

I. Introduction

DUE to the massive growth of high dimensional data, feature selection (FS) plays an increasingly crucial role in many applications [1], [2]. Data with high dimensionality commonly contain a large number of redundant and irrelevant features and even some derogatory features. System identification using this kind of data usually suffers from degradation in performance, a significant increase in the memory requirement, and expensive computation. FS is proven to be an effective way to reduce data dimensionality, and it maintains the characteristics of the original data by selecting informative features. It helps to improve the prediction performance, simplify the learned model, and enhance the interpretability of a model [3], [4]. Though there have been many works on FS, it is still necessary to propose more efficient and effective ones to deal with the explosive growth of data.

For a given problem, different features usually make distinct contributions to the final prediction. Thus, they can be classified into different categories. Chakraborty and Pal [6] classified features into four categories: essential features, bad or derogatory features, indifferent features, and redundant features. Essential features are indispensable for solving the problem, and the removal of them leads to a decrease in prediction performance. Bad or derogatory features are harmful to the task and, thus, should be eliminated. Indifferent features have no contribution to the prediction and should be removed as well. Remarkably, redundant features are useful, but they are dependent on each other, so not all of them are necessary to solve the problem. However, the complete removal of redundant features may not be good. To make the system tolerant to measurement error or other noises, it is necessary to keep some redundant features [4], [6]. In this article, while selecting features, we attempt to control the use of redundant features. The designed FS algorithm aims to select essential features, remove bad/derogatory and indifferent features, and control the use of redundant features.

FS has been extensively studied over the past two decades. FS methods are typically classified into three groups: filter, wrapper, and embedded/integrated methods [2]. A filter method [7], [8] selects features based on some general characteristics of the data, regardless of the feedback from the learning model that finally uses these features. Feature ranking is a commonly used filter method, where features with the highest scores are selected according to a certain criterion. Filter methods do not consider the performance of the associated learning models and are mainly used as data preprocessing techniques. For a wrapper method [9], [10], the learning algorithm is regarded as a "black box," and its prediction

Manuscript received August 27, 2018; revised July 24, 2019 and December 5, 2019; accepted February 23, 2020. This work was supported in part by the National Natural Science Foundation of China under Grant 61305075, in part by the Natural Science Foundation of Shandong Province under Grant ZR2015AL014 and Grant ZR201709220208, in part by the Fundamental Research Funds for the Central Universities under Grant 15CX08011A and Grant 18CX02036A, in part by the Science and Technology Support Plan for Youth Innovation of University in Shandong Province under Grant 2019KJH002, and in part by the Major Scientific and Technological Projects of the China National Petroleum Corporation (CNPC) under Grant ZD2019-183-008. *(Jian Wang and Huaqing Zhang contributed equally to this work.) (Corresponding author: Yifei Pu.)*

Jian Wang and Huaqing Zhang are with the College of Science, China University of Petroleum, Qingdao 266580, China (e-mail: wangjiannl@upc.edu.cn; zhhq@upc.edu.cn).

Junze Wang is with the College of Computer and Communication Engineering, China University of Petroleum, Qingdao 266580, China (e-mail: junzewang@outlook.com).

Yifei Pu is with the College of Computer Science, Sichuan University, Chengdu 610065, China (e-mail: puyifei@scu.edu.cn).

Nikhil R. Pal is with the Electronics and Communication Sciences Unit, Indian Statistical Institute, Kolkata 700108, India (e-mail: nrpal59@gmail.com).

This article has supplementary downloadable material available at http://ieeexplore.ieee.org, provided by the authors.

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

Digital Object Identifier 10.1109/TNNLS.2020.2980383

performance is used to assess different feature subsets. This kind of method generally yields better performance than filter methods; nevertheless, the exhaustive subspace search is quite time-consuming and is almost impossible for high-dimensional data sets. Different from the filter and wrapper methods, the embedded/integrated approaches [11], [12] do not separate FS from the learning process. Instead, it incorporates FS into the training process. The algorithm simultaneously selects the most informative features and optimizes the parameters of the associated learning model. Consequently, this kind of method has a lower computational cost than that of the wrapper methods and a better performance than that of the filter methods. In recent years, some new embedded/integrated methods, i.e., sparsity-induced FS methods, have been proposed [13], [14]. They integrate sparsity-inducing regularization techniques into the learning model and are proven to be effective to select useful features, robust to noise, and suitable for mathematical analysis [15].

The abovementioned methods can remove indifferent and derogatory features and select useful features, but they do not constrain the use of redundant features. In real applications, however, plenty of data sets involve a significant number of redundant features. Redundant features are those useful features that are helpful for discrimination but are dependent/correlated with other features. In such a case, neither all of them should be kept nor the redundancy should be completely removed. For example, if we have three highly correlated useful features, x, y, and z, then each of the three contributes more or less equally to the discrimination task. Thus, it may appear that if we keep just one of them, say x, the problem can be solved. This is true if there is no noise, for instance, measurement noise. Thus, to make the system robust to noise, it would be good to keep any two of them and discard the third. Redundant features provide more powerful and robust discriminating ability, especially in the presence of existing measurement errors and other noise. Thus, while selecting features, we need a mechanism to control the level of redundancy in the selected subset. The proposed method in this article is based on this concept which encourages sparsity in the selected features; besides, it enables the user to control the level of redundancy in the selected features.

The main contributions of this article are summarized as follows.

1) We have proposed an integrated scheme for FS along with controlling the extent of redundancy in the selected features using neural networks.
2) Unlike the scheme in [6], the proposed scheme does not need any additional parameters. It uses suitable penalty terms involving the $L_{2,1}$-norm of the weight vectors connecting the input node to all nodes in the first hidden layer.
3) A more general measure of the dependence between features has been proposed. It can assess the correlation/dependence distinctively for low-, medium-, and high-dimensional data sets via an adaptive parameter.
4) We have employed a smoothing technique [22], [43] to deal with the nondifferentiable penalty terms to facilitate the theoretical analysis. The monotonicity and convergence of the proposed algorithm have been rigorously proved.

The remainder of this article is organized as follows. Related works are discussed in Section II. The neural network based on the Group Lasso regularization and controlling of redundancy for FS and its smooth counterpart are presented in Section III. Section IV introduces the main theoretical results. In Section V, the empirical results are discussed in detail. This article is concluded in Section VI. A proof of convergence of the proposed method under some suitable assumptions is included in the Supplementary Material.

II. Related Works

Suppose that the matrix $X \in \mathbb{R}^{P\times N}$, and the ith row vector of X is denoted by $X_i \in \mathbb{R}^{1\times N}$, $i = 1, 2, \ldots, P$. The jth column vector of X is denoted by $X^j \in \mathbb{R}^{P\times 1}$, $j = 1, 2, \ldots, N$. A typical form of $L_{r,q}$-norm regularization term of matrix X can be formulated as follows [15]:

$$\|X\|_{r,q} = \|(\|X_1\|_r, \ldots, \|X_i\|_r, \ldots, \|X_P\|_r)\|_q \tag{1}$$

where $\|\cdot\|_r$ and $\|\cdot\|_q$ represent the L_r- and L_q-norms, separately.

Different types of penalties can be obtained using different choices of r and q. For example, the Frobenius norm can be realized with $r = 2$ and $q = 2$ and Lasso with $r = 1$ and $q = 1$. These penalties treat all elements in X equally regardless of the group structure of elements. They cannot provide a consistent or jointly sparse representation [16]. In contrast, when $r = 2$ and $q = 1$, the Group Lasso ($L_{2,1}$-norm) penalty term exploits the group structure of the parameters. The Group Lasso penalty can induce row or column sparsity, providing the possibility of selecting features. To the best of our knowledge, the $L_{2,1}$-norm of a matrix was first introduced by Ding *et al.* [17]. A variety of FS methods based on $L_{2,1}$-regularization have been proposed in recent years [18]–[20]. The sparse Group Lasso technique [21] was used to implement FS for uncertain data, which showed competitive performance in selecting the subfeatures in a grouped manner. In our previous work [22], we have applied the Group Lasso in pruning hidden layer nodes to improve the generalization performance of neural networks. In [22], we did not use the Group Lasso for FS.

Neural networks have been widely used and proved to be good tools to solve nonlinear learning problems. There are a few methods that achieve FS by integrating sparsity techniques into neural networks. Sun *et al*. [23] introduced a variable selection method using neural networks with Lasso. However, the feature elimination is constrained by the trained network as the network weights are not updated when the Lasso works. Li *et al*. [24] presented a regularization technique for multilayer feedforward neural networks to compress the input layer and eliminate the redundant variables. In [25], a general sparse technique for deep neural architecture was proposed by using Lasso and Group Lasso regularizers without considering an explicit control/removal of redundancy. These methods do not explicitly remove redundant useful features due to their good predictive or discriminative power. Consequently, they

are likely to select redundant features demanding more space and computational time.

The study on the level of redundancy in the selected features has attracted some attention in recent years [26], [27]. An interesting method called "minimum redundancy–maximum relevance" (mRMR) was introduced in [28] and [29]. In [30], a normalized mutual information-based FS (NMIFS) method is proposed, which uses the average normalized mutual information to measure the redundancy among features. Based on the premise of maximum relevance and minimum redundancy, Jiang *et al*. [31] proposed a correlation-based feature weighting (CFW) filter for the Naive Bayes classifiers. A decision-dependent correlation measure was given by Qu *et al*. [32] to remove the redundancy of selected features using mutual information and entropy. Using a trace-based class separability criterion, a redundancy-constrained FS method was suggested by Zhou *et al*. [33]. Xu *et al*. [34] proposed a max-relevance and min-redundancy criterion based on Pearson's correlation coefficient for a semisupervised learning method. Senawi *et al*. [35] suggested a maximum relevance–minimum multicollinearity (MRmMC) method, which measured the relevance between features by a criterion based on conditional variance. Based on ant colony optimization, Tabakhi and Moradi [36] introduced an unsupervised and multivariate filter-based FS method to analyze the relevance and redundancy of features. Feng *et al*. [37] suggested a maximum information and minimum redundancy (MIMR) criterion to select informative and distinctive features. Nie *et al*. [38] introduced an autoweighted FS framework via global redundancy minimization (AGRM) to select the top nonredundant features.

As mentioned earlier, maintaining some redundancy results in easier learning and makes the system robust to measurement errors. Thus, how to control the use of redundant features is an important issue for the learning system. Recently, several works on controlling redundancy have been published. Li and Tang [39] proposed a nonnegative spectral analysis with constrained redundancy for unsupervised FS by jointly leveraging nonnegative spectral clustering and redundancy analysis. Nevertheless, all these works are filter methods. Differently, in [40], an integrated FS framework was proposed, where Pearson's correlation coefficient was used to penalize the correlated features in radial basis function (RBF) networks. Based on an MLP network, Chakraborty and Pal [6] proposed a general scheme to deal with FS with controlled redundancy (FSMLP-CoR). Chung *et al*. [41] suggested an FS method with controlled redundancy using a fuzzy rule-based framework (FRBS-FC). Banerjee and Pal [42] introduced an unsupervised FS scheme with controlled redundancy (UFeSCoR). With proper choices of the parameters, it not only can select a desired number of features but also can control the redundancy among the selected features. FSMLP-CoR, FRBS-FC, and UFeSCoR are all based on the concept of a gate function that sets a gate associated with each feature; thus, they need an additional parameter for each feature. In contrast, the proposed method in this article provides an integrated scheme that directly works on the weights of the neural network without requiring any additional parameters.

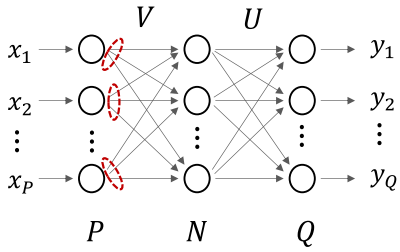

Fig. 1. Architecture of a multilayer perceptron network.

III. Proposed Method

We consider a single-hidden layer backpropagation (BP) neural network, as shown in Fig. 1. Our method can be easily extended to a deep architecture with more than one hidden layer. In Fig. 1, P, N, and Q are the number of nodes in the input, hidden, and output layers, respectively.

Suppose that the training data set is denoted by $\{\mathbf{x}^j, \mathbf{y}^j\}_{j=1}^J$, where $\mathbf{x}^j \in \mathbb{R}^P$ is the feature vector representation of the jth instance and $\mathbf{y}^j \in \mathbb{R}^Q$ is the corresponding target output. We let

$$\mathbf{X} = (x_{i,j})_{P\times J} = (\mathbf{x}^1, \mathbf{x}^2, \ldots, \mathbf{x}^J) = \begin{pmatrix} \mathbf{x}_1 \\ \mathbf{x}_2 \\ \vdots \\ \mathbf{x}_P \end{pmatrix} \tag{2}$$

where $\mathbf{x}_i$ represents the ith $(i = 1, 2, \ldots, P)$ feature of all instances in the training data, and $\mathbf{Y} = (\mathbf{y}^1, \mathbf{y}^2, \ldots, \mathbf{y}^J)$.

Suppose that $\mathbf{V} = (v_{ki})_{N\times P}$ is the weight matrix connecting the input layer to the hidden layer, where v_{ki} is the connecting weight between the ith input node and the kth $(k = 1, 2, \ldots, N)$ hidden node. Let $\mathbf{v}_i = (v_{1i}, v_{2i}, \ldots, v_{Ni})^T$ for $i = 1, 2, \ldots, P$ be the ith column vector of $\mathbf{V}$ and $\mathbf{U} = (u_{lk})_{Q\times N}$ be the weight matrix connecting the hidden and output layers. The lth vector of weight matrix $\mathbf{U}$ is denoted by $\mathbf{u}_l = (u_{l1}, u_{l2}, \ldots, u_{lN})^T$ for $l = 1, 2, \ldots, Q$. For simplicity, we combine the weight matrices $\mathbf{U}$ and $\mathbf{V}$ and rewrite $\mathbf{w} = (\mathbf{u}_1, \ldots, \mathbf{u}_Q, \mathbf{v}_1, \ldots, \mathbf{v}_P)^T \in \mathbb{R}^{N\times(P+Q)}$. Let f and g be the activation functions of hidden and output layers, respectively.

The following vector valued functions are introduced for convenience:

$$G(\mathbf{z}) = (g(z_1), g(z_2), \ldots, g(z_N))^T \quad \forall \mathbf{z} \in \mathbb{R}^N \tag{3}$$

and

$$F(\mathbf{s}) = (f(s_1), f(s_2), \ldots, f(s_Q))^T \quad \forall \mathbf{s} \in \mathbb{R}^Q \tag{4}$$

where $\mathbf{z} = (z_1, z_2, \ldots, z_N)$ is any vector in $\mathbb{R}^N$, $g(z_k)(k = 1, 2, \ldots, N)$ is the output of the activation function g on z_k, and $G(\mathbf{z})$ is the output of the activation function g on the vector $\mathbf{z}$. Similarly, $\mathbf{s} = (s_1, s_2, \ldots, s_Q)$ is any vector in $\mathbb{R}^Q$, $f(s_l)$ $(l = 1, 2, \ldots, Q)$ is the output of the activation function f on s_l, and $F(\mathbf{s})$ is the output of the activation function f on vector $\mathbf{s}$.

The empirical error function of the neural network is defined as

$$E(\mathbf{w}) = \left\| F(\mathbf{U}G\left(\sum_{i=1}^{P} \mathbf{v}_i \mathbf{x}_i\right) - \mathbf{Y} \right\|_F^2 \tag{5}$$

where $\mathbf{w}$ represents the weights of the network. As mentioned earlier, the vector $\mathbf{v}_i = (v_{1i}, v_{2i}, \ldots, v_{Ni})^T (\ i = 1, 2, \ldots, P)$ is the ith column vector of the weight matrix $\mathbf{V}$, i.e., $\mathbf{v}_i$ is the weight vector connecting the ith input node to all the hidden nodes. $\sum_{i=1}^{P} \mathbf{v}_i \mathbf{x}_i$ denotes the input matrix of the hidden layer and $G(\cdot)$ is the output of hidden layer. Multiplied by $\mathbf{U}$, $\mathbf{U}G(\cdot)$ denotes the input of the output layer, and $F(\mathbf{U}G(\cdot))$ is the activation output of the constructed neural network.

A. Group Lasso and Controlled Redundancy BP

Lasso or Group Lasso adds a penalty to the square error in such a manner that if more variables take part in the model, the penalty increases. This penalty trades off with the square error. Thus, the Lasso/Group Lasso forces coefficients of derogatory/useless features to almost zero as those features do not affect the square error and setting their weights to zero reduces penalty. Consequently, the total objective function, including the penalty imposed by Lasso/Group Lasso, is reduced.

Here, a Group Lasso penalty for the input layer is used to achieve sparsity of features. Another penalty term is designed to control the redundancy among the selected features. The total cost with Group Lasso penalty and redundancy control is proposed as follows:

$$E_T = E(\mathbf{w}) + \lambda E_{\text{PN}} + \mu E_C \tag{6}$$

where $E(\mathbf{w})$ is the original error function (5), E_{PN} is the Group Lasso penalty exerted on the weights connecting the input layer and the first hidden layer, and E_C represents the penalty term for redundancy based on the correlations among features. We note that the variables of all three components of the cost function are all or part of the weight parameters, $\mathbf{w}$, instead of the training samples, $\mathbf{X}$ and $\mathbf{Y}$. The Group Lasso penalty E_{PN} is defined as

$$E_{\text{PN}} = \sum_{i=1}^{P} ||\mathbf{v}_i|| \tag{7}$$

where $\mathbf{v}_i = (v_{1i}, v_{2i}, \ldots, v_{Ni})^T (\ i = 1, 2, \ldots, P)$ is the weight vector connecting the ith feature to all the hidden nodes and $\|\mathbf{v}_i\|$ is the Group Lasso regularization imposed on $\mathbf{v}_i$. Here, we employ $\|\cdot\|$ instead of $\|\cdot\|_2$ to represent the L_2-norm for simplicity and hereinafter for the remainder of this article. The penalty for redundancy, E_C, is defined as

$$E_C = \frac{1}{P(P-1)} \sum_{i=1}^{P} ||\mathbf{v}_i|| \sum_{k=1, k\neq i}^{P} ||\mathbf{v}_k|| \cdot \text{dep}(x_i, x_k) \tag{8}$$

where $\text{dep}(x_i, x_k) \geq 0$ is a measure of dependence between the ith feature, x_i, and the kth feature, x_k, and $\|\mathbf{v}_i\|$ and $\|\mathbf{v}_k\|$ are the regularizations imposed on the input weight vectors that correspond to the ith and kth features, respectively. Then, the formula $||\mathbf{v}_i|| \sum_{k=1, k\neq i}^{P} ||\mathbf{v}_k|| \cdot \text{dep}(x_i, x_k)$ indicates the dependence between the ith feature and all the other features. If the ith feature is important, then $||\mathbf{v}_i||$ is expected to be high. In this case, if $\text{dep}(x_i, x_k)$ is high, the penalty will prevent the growing of $||\mathbf{v}_k||$. Here, E_C represents a measure of total dependence/redundancy in the selected feature set.

1) Feature Sparsity: As mentioned earlier, the Group Lasso penalty in (7) is used to determine the importance of features and encourage sparsity. But how does it distinguish the useful and harmful features?

For the ith feature, all the connecting weights work together in a grouped manner. If it is a harmful or derogatory feature, the corresponding part in the original error, $E(\mathbf{w})$, is large. To minimize the total cost, the penalty, E_{PN}, is consequently minimized. The L_2-norm of its connecting weight vector is decreased to (or close to) zero. If the norm of the weight vector corresponding to this feature satisfies $(\|\mathbf{v}_i\|)/(N) < \theta$ (where N is the number of hidden layer nodes), it is not considered a useful feature and is discarded. Conversely, a useful or good feature contributes significantly to minimize the original error; hence, its connecting weights are generally larger. As a result, the L_2-norm of its weight vector is not decreased and is likely to satisfy $(\|\mathbf{v}_i\|)/(N) \geq \theta$. Such a feature is judged to be a useful one and is selected. The parameter, λ, determines the influence of the penalty term E_{PN} on the whole scheme. A small λ makes the penalty term contribute little to the error, and consequently, most features may be selected. While with a large λ, sparsity is considered important to the system, and only very useful features are selected.

2) Controlling of Redundancy: The penalty term, E_{PN}, helps to select informative features, but it does not consider the level of redundancy among the selected features. Given a problem, suppose that features x_1, x_2, and x_3 are all useful ones, and x_2 and x_3 are highly correlated with feature x_1. Then, the feature subset $s = \{x_1, x_2, x_3\}$ is a redundant subset. Because we can use only one of them to solve the problem, the other two are not necessary. However, the minimum number of required features is not the best choice; a feature subset with some redundancy is more robust to measurement error. In the previous example, suppose that we choose only x_1 and discard x_2, x_3. In this case, if there is an error occurring in the measurement of x_1, then the prediction will be affected. While if we choose x_1 and x_2 simultaneously, some measurement error in one of the variables is likely to be tolerated. Thus, to select a useful set of features, we should control the use of redundant features.

The regularization, E_C, in (8) is designed to penalize the use of many correlated/dependent features. The controlling coefficient μ is a constant that determines the level of redundancy control. $\text{dep}(x_i, x_k)$ is a measure of dependence between features x_i and x_k. It can be any linear or nonlinear measure of dependence. Here, we use Pearson's correlation coefficient as the measure of dependence. Note that other measures are applicable as well, such as the nonlinear measure and mutual information. Here, we let $\text{dep}(x_i, x_k) = \rho^{2s}(x_i, x_k)$, $s \in \{1, 2, \ldots\}$; then, the redundancy control term, E_C, takes

the form as

$$E_C = \frac{1}{P(P-1)} \sum_{i=1}^{P} ||\mathbf{v}_i|| \sum_{k=1,k\neq i}^{P} ||\mathbf{v}_k|| \cdot \rho^{2s}(x_i, x_k). \quad (9)$$

Similar to (8), $\|\mathbf{v}_i\|$ and $\|\mathbf{v}_k\|$ are the regularizations imposed on the ith and the kth features, respectively. The measure $\rho^{2s}(x_i, x_k)$ is a measure of dependence that can be adjusted depending on the dimension of the data. The parameter "s" in (9) is usually fixed at 1, but, as mentioned earlier, it can be varied depending on the data set.

For simplicity, let us consider $s = 1$. Then, the dependence measure between the ith and kth features, $\rho^2(x_i, x_k)$, is calculated using the square of Pearson's correlation coefficient between the vectors $\mathbf{x}_i$ and $\mathbf{x}_k$. Suppose that x_1 is a useful feature and has been selected. Let x_2 be another useful feature that has no/low correlation to x_1; then, the dependence, $\text{dep}(x_1, x_2) = \rho^2(x_1, x_2)$, contributes little to E_C. Thus, feature x_2 is not/less penalized by E_C, and x_2 can be selected, if required. Suppose that x_3 is a useful feature and is highly correlated (positively or negatively) to x_1 with correlation value C ($|C|$ close to *one*). Then, the value of the dependence measure, $\text{dep}(x_1, x_3)$, is high, and it contributes more to E_C. Thus, the selection of x_3 is prevented. With different values of the penalty coefficient, μ, we can select different numbers of such redundant features. In this way, we can control the level of redundancy in the selected features.

To summarize, whether a feature is selected or not depends on both the sparsity penalty and the redundancy control penalty. For an indifferent or derogatory feature, for instance, feature x_1, its connecting weights will be drastically punished to or close to zero by the sparsity penalty term E_{PN} due to its little, even no, contribution to the minimization of the error (5). The norm of its connecting weight vector, $\|\mathbf{v}_1\|$, thereby becomes very small and is likely to satisfy $(\|\mathbf{v}_1\|)/(N) < \theta$. Consequently, the feature x_1 is discarded. For a useful feature, for example, feature x_2, the sparsity penalty term imposes little punishment on its connecting weights due to its considerable contribution to minimize the original error. However, it is still possible for this feature to be removed, for example, if feature x_2 is highly correlated with a selected feature, then its connecting weights may be severely punished by the redundancy penalty term E_C. Consequently, it may satisfy $(\|\mathbf{v}_2\|)/(N) < \theta$ and be deleted. However, whether a dependent feature would be discarded or not also depends on the level of controlling redundancy, which is monitored by the parameter μ. If the feature x_2 is a discriminatory one and is not correlated with the other useful features, it has a high possibility to be selected. In this case, the connecting weights corresponding to x_2 are penalized significantly neither by the sparsity penalty term E_{PN} nor by the redundancy penalty term E_C. As a result, its associated norm $\|\mathbf{v}_2\|$ is likely to satisfy $(\|\mathbf{v}_2\|)/(N) \geq \theta$. Then, x_2 is selected in this case. Note that, the use of the threshold θ does not affect the differentiability of the BP algorithm since θ is not a part of the objective function and θ is used for selecting features, which is performed after training.

B. Smoothing Group Lasso and Controlled Redundancy BP

Clearly, the Group Lasso penalty term and the redundancy control term are nondifferentiable at the origin, which may lead to numerical problems when the norm of the weight vector is close to zero. Smoothing techniques are effective to solve this problem [43].

For any finite dimensional vector $\mathbf{z}$, we introduce the following smooth function [22], [43]:

$$h(\mathbf{z}) = \begin{cases} \|\mathbf{z}\|, & \|\mathbf{z}\| \geq \alpha \\ \dfrac{\|\mathbf{z}\|^2}{2\alpha} + \dfrac{\alpha}{2}, & \|\mathbf{z}\| < \alpha \end{cases} \quad (10)$$

where $h(\cdot)$ represents a polynomial function to approximate the Group Lasso penalty, and it is easy to see that $h(\cdot)$ is a differentiable smoothing function; $\mathbf{z}$ represents a finite dimensional vector, and the smoothing parameter $0 < \alpha \leq 1$ is a fixed positive constant. With decreasing α, $h(\cdot)$ approaches the original Group Lasso penalty and produces satisfactory approximation performance.

Accordingly, $\|\mathbf{v}_i\|$ in the penalty term E_{PN} in (7) and in E_C in (8) can be approximated by the smoothing function in (10). The penalty term E_{PN} is now rewritten as

$$E_{\text{PN}} = \sum_{i=1}^{P} h(\mathbf{v}_i) \quad (11)$$

where $\mathbf{v}_i$ is the weight vector connecting the ith input node to all the hidden nodes, and $h(\mathbf{v}_i)$ is the approximation of $\|\mathbf{v}_i\|$. E_C is rewritten as

$$E_C = \frac{1}{P(P-1)} \sum_{i=1}^{P} h(\mathbf{v}_i) \sum_{k=1,k\neq i}^{P} h(\mathbf{v}_k)\text{dep}(x_i, x_k) \quad (12)$$

where $\mathbf{v}_k$ is the weight vector connecting the kth input node to all the hidden nodes. $h(\mathbf{v}_i)$ and $h(\mathbf{v}_k)$ are the approximations of $\|\mathbf{v}_i\|$ and $\|\mathbf{v}_k\|$, respectively. Then, the weight updating formulas with smoothing penalty term become

$$\frac{\partial E_{\text{PN}}}{\partial \mathbf{v}_i} = \begin{cases} \dfrac{\mathbf{v}_i}{||\mathbf{v}_i||}, & ||\mathbf{v}_i|| \geq \alpha \\ \dfrac{\mathbf{v}_i}{\alpha}, & ||\mathbf{v}_i|| < \alpha \end{cases} \quad (13)$$

$$\frac{\partial E_C}{\partial \mathbf{v}_i} = \frac{1}{P(P-1)} \frac{\partial \sum_{i=1}^{P} h(\mathbf{v}_i) \sum_{k=1,k\neq i}^{P} h(\mathbf{v}_k)\text{dep}(x_i, x_k)}{\partial \mathbf{v}_i} = \begin{cases} \dfrac{2}{P(P-1)} \dfrac{\mathbf{v}_i}{||\mathbf{v}_i||} \sum_{k=1,k\neq i}^{P} h(\mathbf{v}_k)\text{dep}(x_i, x_k), & ||\mathbf{v}_i|| \geq \alpha \\ \dfrac{2}{P(P-1)} \dfrac{\mathbf{v}_i}{\alpha} \sum_{k=1,k\neq i}^{P} h(\mathbf{v}_k)\text{dep}(x_i, x_k), & ||\mathbf{v}_i|| < \alpha. \end{cases} \quad (14)$$

The total cost function (6) is then rewritten as

$$E_T(\mathbf{w}) = E(\mathbf{w}) + \lambda \sum_{i=1}^{P} h(\mathbf{v}_i) + \mu \frac{1}{P(P-1)} \sum_{i=1}^{P} h(\mathbf{v}_i) \sum_{k=1,k\neq i}^{P} h(\mathbf{v}_k)\text{dep}(x_i, x_k). \quad (15)$$

In this article, the gradient descent method is used to minimize $E_T(\mathbf{w})$. The updating rules for the weights at the mth iteration are

$$\mathbf{u}_l^{m+1} = \mathbf{u}_l^m - \eta \frac{\partial E_T}{\partial \mathbf{u}_l} \tag{16}$$

$$\mathbf{v}_i^{m+1} = \mathbf{v}_i^m - \eta \frac{\partial E_T}{\partial \mathbf{v}_i}. \tag{17}$$

Thus, we can get the combined weight updating formula as follows:

$$\mathbf{w}^{m+1} = \mathbf{w}^m - \eta \frac{\partial E_T(\mathbf{w}^m)}{\partial \mathbf{w}^m}, \quad m \in \mathbb{N}. \tag{18}$$

For simplicity, we let $E_{T\mathbf{w}}(\mathbf{w}^m) = (\partial E_T(\mathbf{w}^m))/(\partial \mathbf{w}^m)$ in the rest of this article.

IV. Main Theoretical Results

To analyze the algorithm using smoothing Group Lasso penalty and redundancy control, the following four assumptions are needed:

- *A1:* The activation functions f and g are continuous differentiable on $\mathbb{R}$, and f, g, f' and g' are uniformly bounded on $\mathbb{R}$, that is, there exists a positive constant $C \in \mathbb{R}$ such that $\sup_{x\in\mathbb{R}}\{|f(x)|, |g(x)|, |f'(x)|, |g'(x)|\} \le C$.
- *A2:* The learning rate $\eta > 0$ and satisfies that $\eta < \frac{1}{C_8}$, where C_8 is given in (43) in the Supplementary Material.
- *A3:* The weight sequence $\{\mathbf{w}^m\}$ $(m \in \mathbb{N})$ is uniformly bounded in $\mathbb{R}^{N\times(P+Q)}$.
- *A4:* The stationary points of (15) are at most countably infinite.

Similar assumptions are also made in [44].

Theorem 1 (Monotonicity): Suppose that the cost function is defined by (15), and the weight sequence $\{\mathbf{w}^m\}$ $(m \in \mathbb{N})$ is generated by the iterative updating formulas (16) and (17). If the assumptions A1–A3 are valid, then the cost function holds that

$$E_T(\mathbf{w}^{m+1}) \le E_T(\mathbf{w}^m), \quad m = 0, 1, \ldots \tag{19}$$

Theorem 2 (Convergence): Suppose that the assumptions A1–A3 are valid, then the weight sequence $\{\mathbf{w}^m\}$ that is generated by (16) and (17) satisfies the following weak convergence:

$$\lim_{m\to\infty} \|E_{T\mathbf{w}}(\mathbf{w}^m)\| = 0, \quad m \in \mathbb{N}. \tag{20}$$

In addition, if the assumption A4 is also valid, there holds the strong convergence: $\exists\ \mathbf{w}^* \in \mathbb{R}^{N\times(P+Q)}$ such that

$$\lim_{m\to\infty} \mathbf{w}^m = \mathbf{w}^*, \quad m \in \mathbb{N}. \tag{21}$$

Remark: The proofs of these theorems are done in a manner similar to the proofs in [45] and included in the Supplementary Materials. The contributions in [45] mainly focus on structure optimization (pruning less useful hidden nodes) of the network by imposing Lasso (not Group Lasso) penalty on the hidden outputs.

V. Experimental Results

In this section, we experimentally evaluate the performance of the proposed method and some existing FS methods on both synthetic and real-world data sets. The theoretical analysis is validated as well.

TABLE I
Description of All Data Sets Used

Data set	Data set Size	Dimensionality	Classes
Iris	150	4	3
Thyroid	215	5	3
Phoneme	5404	5	2
Liver	345	6	2
Vertebral	310	6	3
Noisy Iris	150	8	3
Noisy Balance	525	8	3
Synthetic	300	7	3
Ecoli	336	7	8
Appendicitis	106	7	2
Pima	768	8	2
Glass	214	9	6
WBC	683	9	2
Wine	178	13	3
Heart	270	13	2
Vehicle	846	18	4
SPECT Heart	267	22	2
Wdbc	569	30	2
Spectf Heart	267	44	2
Spambase	4597	57	2
Sonar	208	60	2
ORL	400	1024	40
Colon	62	2000	2
SRBCT	83	2308	4
ARP	130	2400	10
PIE	210	2420	10

A. Data Sets

Twenty-six classification data sets are used in our experiments. Detailed information about these data sets is listed in Table I. These data sets are divided into two groups: group one, consists of the first 21 data sets in Table I; group one represents low- and medium-dimensional data sets. Group two includes five high-dimensional data sets. Each feature in each data set is normalized to [0, 1] by the Max–Min scaling, i.e., we normalize a feature x_j to x'_j using the relation

$$x'_j = \frac{x_j - x_{\min}}{x_{\max} - x_{\min}}. \tag{22}$$

B. Experimental Settings

Here, single-hidden-layer backpropagation networks are used for learning. In general, more than one hidden layer can be adopted. The gradient descent method is employed for training. We take Hyperbolic Tangent and Log sigmoid functions as activation functions for the hidden and output layers, respectively. There are five parameters to be determined, as shown in the upper half of Table II. To investigate the effects of different parameters, we first report results using different choices of parameters. However, for comparison of results, the parameters N, η, λ, and μ are determined by cross-validation. The dimension-dependent choice of θ is obtained by the following formula:

$$\theta = \max_{i=1,2,\ldots,P} \left\{ 0.1 * \frac{\|\mathbf{v}_i\|}{N} \right\} \tag{23}$$

TABLE II
DESCRIPTION OF USED SYMBOLS

Parameter	Meaning
η	the learning rate
N	the number of hidden layer nodes
λ	the regularization coefficient
μ	the redundancy control coefficient
θ	the threshold of eliminating a feature
Other symbols	Meaning
AccTr	the average training accuracy
AccTe	the average test accuracy
AvgSel	the average number of selected features
CCorr	the correlation between features and class labels
MaxCorr	the maximum correlation in selected features

where N is the number of hidden nodes. The algorithms are implemented in MATLAB R2017a on a workstation with an Intel Xeon E5-2620 v3 processor and 128-GB RAM. Parallel computing is applied to improve efficiency. The maximum number of iterations is set to be 5000 for all experiments. To demonstrate the effectiveness of the proposed method, we use a tenfold cross-validation scheme, where each validation is a three-stage procedure, as shown in Fig. 7 in the Supplementary Material.

Remark 1: In the rest of this article, all the simulations are performed based on the algorithm, as shown in Fig. 7 in the Supplementary Material except for those in Sections V-C, V-D, and V-F. In Section V-C, with a pair of fixed parameters (λ, μ), the results of a typical run are plotted to compare the difference between the two versions of the Group Lasso algorithm, with and without the smoothing technique. For the two algorithms, we use identical initial weights during training for a fair comparison. In Section V-D, a particular training strategy is designed to clearly demonstrate the redundancy controlling ability of the proposed method [see (15)–(18)], where we repeat the experiments more times, 1000 times, on the Iris data set. Similarly, with a pair of fixed parameters, Section V-F shows the robustness of the proposed method on noisy data sets. The remaining experiments reported in Sections V-E, V-G, and V-H are all done by using the algorithm in Fig. 7 in the Supplementary Material. They separately depict different characteristics of the presented method from different aspects.

Remark 2: For the high-dimensional data sets, we use $\text{dep}(x_i, x_k) = \rho^4\ (x_i, x_k)$, i.e., $s = 2$, instead of $\rho^2\ (x_i, x_k)$ in the redundancy control term (8), because a high-dimensional data set has many features and is likely to have more pairs of dependent features. But because of the normalization by $P(P-1)$, the sum of squares of correlations may not make a significant impact for these data sets. The use of $\rho^4\ (x_i, x_k)$ can help to strengthen the relative impact of high correlation compared with low correlation among features due to $\rho(x_i, x_k) \in [0, 1]$. Thereby, the algorithm implements a relatively large penalty on the pairs of features with high correlations. Let $\rho_1 > \rho_2$, $\rho_2 > 0.5$, $\rho_1 = \rho_2 + \delta$, $\delta > 0$, and $\rho_1^2 - \rho_2^2 = (\rho_2 + \delta)^2 - \rho_2^2 = \delta(\delta + 2\rho_2)$. Since $\rho_2 > 0.5$, $\delta(\delta + 2\rho_2) > \delta$. Thus, with high values of ρ, the relative impact of the penalty using ρ^2 will be higher compared with the use of $abs(\rho)$. Similarly, relative impact of penalty will be higher with use of ρ^4 compared with the use of ρ^2. This is particularly true with $\rho > 0.71$ (i.e., $\rho^2 > 0.5$). We note that this change would not influence the theoretical results. For our theoretical analysis, we use $s = 1$, i.e., ρ^2.

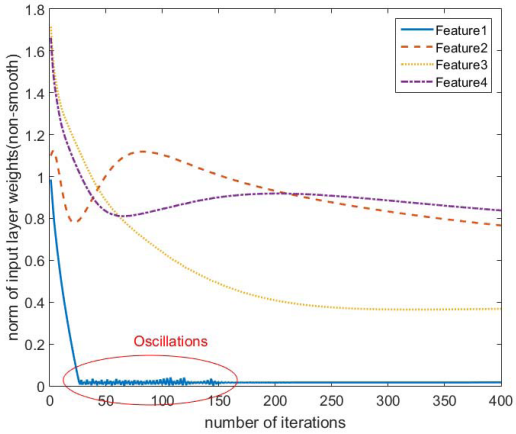

Fig. 2. Norm of the connecting weight vectors of input nodes for the Iris data set with $\lambda = 1$ and $\mu = 0.4$ of the nonsmoothing penalty term.

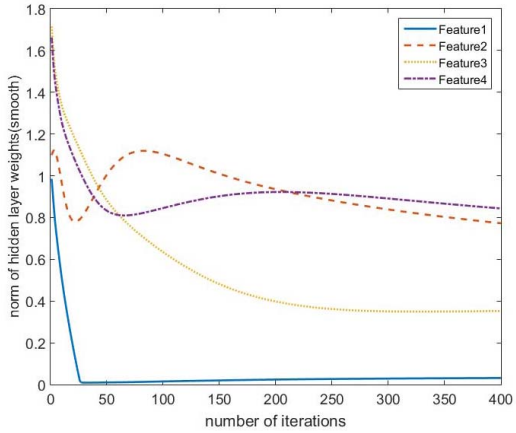

Fig. 3. Norm of the connecting weight vectors of input nodes for the Iris data set with $\lambda = 1$ and $\mu = 0.4$ of the smoothing penalty term.

C. Performance Comparison With and Without Smoothing Technique for the Penalty Terms

We have implemented the proposed method using smoothing (SGLC) and nonsmoothing (GLC) penalty terms. Fig. 2 shows the results of nonsmoothing method on the Iris data set with $\lambda = 1$ and $\mu = 0.4$. The Iris data set (https://archive.ics.uci.edu/ml/data sets/Iris/) is a well-known benchmark classification data set. It has 150 samples each with four features (sepal length, sepal width, petal length, and petal width) and three categories. We can see that numerical oscillations occur when iteration is less than 200. This is because the function (6) is nondifferential at the origin. Fig. 3 shows the results when we use the smooth function (10). We can see that the oscillations disappear due to formulas (13) and (14). In spite of this, the curves in Fig. 3 are similar with the ones in Fig. 2. In both cases, the norm of the

TABLE III
COMPARISON OF RESULTS BETWEEN SGL AND SGLC ON IRIS DATA SET

Methods	SGL	SGLC	SGL	SGLC	SGL	SGLC	SGL	SGLC	SGL	SGLC
η	0.2	0.2	0.2	0.2	0.2	0.2	0.4	0.4	0.8	0.8
λ	0.008	0.008	0.512	0.512	2.048	2.048	1	1	1	1
μ	-	1	-	1	-	1	-	0.001	-	1
s_1	0	0	0	0	0	0	0	0	0	0
s_2	0	0	0	0	0	0	0	0	0	0
s_3	140	21	29	54	20	112	81	67	112	100
s_4	463	51	403	276	777	785	361	407	365	737
$s_{1,2}$	0	0	0	0	0	0	0	0	0	0
$s_{1,3}$	0	0	0	0	0	0	0	0	0	0
$s_{1,4}$	0	0	0	0	0	0	0	0	0	0
$s_{2,3}$	6	284	0	161	0	8	7	5	1	12
$s_{2,4}$	8	644	0	509	41	92	41	38	14	150
$s_{3,4}$	280	0	507	0	113	0	300	291	354	0
$s_{1,2,3}$	0	0	0	0	0	0	0	0	0	0
$s_{1,2,4}$	0	0	0	0	0	0	0	0	1	0
$s_{1,3,4}$	0	0	0	0	0	0	0	0	0	0
$s_{2,3,4}$	102	0	61	0	49	3	210	192	153	1
$s_{1,2,3,4}$	1	0	0	0	0	0	0	0	0	0
Feat1	1	0	0	0	0	0	0	0	1	0
Feat2	117	928	61	670	90	103	258	235	169	163
Feat3	529	305	597	215	182	123	598	555	620	113
Feat4	854	695	971	785	980	880	912	928	887	888
AvgSel	2.23	1.97	2.46	1.76	1.26	1.11	1.94	1.86	1.95	1.27
AccTr	0.96	0.96	0.96	0.96	0.96	0.96	0.97	0.97	0.97	0.96
AccTe	0.95	0.94	0.95	0.94	0.95	0.95	0.95	0.95	0.95	0.95

weights connecting to feature 1 goes to zero, and this feature is removed. For the other three features, the norms of the connecting weights approach to larger fixed values, and they are selected. Although both cases exhibit similar performance with regard to FS, the modified smoothing penalties can effectively avoid oscillations during the training and ease the theoretical analysis. Therefore, for experiments in the rest of this article, we use the smoothing penalty terms.

D. Performance Comparison With and Without Redundancy Control

To demonstrate the effectiveness of the proposed method (SGLC), we compared it with the method using Smoothing Group Lasso penalty without controlling redundancy (SGL) [44]. Table III summarizes their FS results and classification performance on Iris data set. Here we have repeated the experiment 1000 times with different initial values of weights and different partitions (80% training and 20% test) of the data sets. We use s_i to represent the case when only the ith feature is chosen, $s_{i,j}$ to represent the selection of two features, i.e., the ith and the jth features, and $s_{i,j,k}$ to represent the selection of the ith, the jth and the kth features. Similarly, $s_{i,j,k,l}$ represents the case when the ith, jth, kth, and lth features are chosen simultaneously. Feat1, Feat2, Feat3, and Feat4 show the frequencies with which the corresponding features are selected. We have experimented with $\eta \in [0.1, 1]$, $\lambda \in [0, 3]$, and $\mu \in [0, 1]$. Here, we list part of them for observation.

TABLE IV
CORRELATION MATRIX FOR IRIS DATA SET

Features	1	2	3	4	CCor
1	1.00	-0.11	0.87	0.82	0.78
2	-0.11	1.00	-0.42	-0.36	-0.42
3	0.87	-0.42	1.00	**0.96**	**0.95**
4	0.82	-0.36	**0.96**	1.00	**0.96**

Table IV shows the correlation structure of Iris data set. From Table III, we can see that the features 3, 4, combination $s_{3,4}$, and $s_{2,3,4}$ are selected a large number of times when $\eta = 0.2$ and $\lambda = 0.008$ for SGL. It is well known that features 3 and 4 are the most important features in this data set for classification, as shown in Table IV by the class correlation (CCor). Feature 1 has a stronger effect on classification than that by feature 2. However, feature 1 has higher correlations to features 3 and 4 than those for feature 2, as shown in Table IV, which makes the feature combination $s_{2,3,4}$ more discriminatory than $s_{1,3,4}$. Feature combinations $s_{2,3}$, $s_{2,4}$, and $s_{1,2,3,4}$ are selected occasionally. However, features 3 and 4 are the most discriminatory features and have a correlation of 0.96, so they are not required together. In this case, the subset selected by SGL has a high level of redundancy. In contrast, with the same η and λ, feature combinations $s_{3,4}$, $s_{2,3,4}$, and $s_{1,2,3,4}$ are almost eliminated by the SGLC method when $\mu = 1$. This is due to the high correlation between features 3 and 4, which leads to a large E_C, thus penalized drastically. Meanwhile, the selection frequencies of $s_{2,3}$ and $s_{2,4}$ increase

by 278 and 636, respectively, due to their low correlation. In this case, feature 1 is totally eliminated. Each of features 3 and 4 is selected a lesser number of times. While feature 2 is selected 811 times more than that for SGL. Compared with the SGL method, the SGLC method selects on average 0.26 features less but produces similar classification performance and tends to select subsets with lower redundancy.

With the increase in λ, when $\eta = 0.2$ and $\lambda = 0.512$, feature combinations $s_{2,3}$, $s_{2,4}$, and $s_{1,2,3,4}$ are eliminated by the SGL method. The selected frequency of combination $s_{2,3,4}$ is decreased because feature 2 has a lower correlation with class labels. The selected frequency of combination $s_{3,4}$ is significantly increased (by 227). Meanwhile, the selected frequencies of features 3 and 4 are decreased by 111 and 60, respectively. This is because the dependence among features is not considered in the SGL method. With $\mu = 1$, the SGLC method removes $s_{3,4}$ and $s_{2,3,4}$ due to the high correlation between the features 3 and 4. Remarkably, the selected frequencies for feature combinations $s_{2,3}$ and $s_{2,4}$ are increased to 161 and 509 from zero, respectively. The training accuracy (AccTr) is not changed, and the test accuracy (AccTe) is slightly decreased. Similar results can be observed when $\lambda = 2.048$, where the SGLC method selects on average less number of features than that by the SGL method but has the same classification performance. When $\eta = 0.4$ and $\lambda = 1$, we can see that the SGL method selects more features than the case with $\eta = 0.2$ and $\lambda = 2.048$. The selection frequencies of combinations $s_{3,4}$ and $s_{2,3,4}$ are increased by 187 and 161, respectively. When $\mu = 0.001$, we can see that the selected frequencies of combinations $s_{3,4}$ and $s_{2,3,4}$ are decreased by 9 and 18, respectively. The average number of selected features practically remains the same and so is the classification performance. This demonstrates the ability of redundancy control term to select appropriate features even with a very small coefficient to penalize redundancy.

E. Controlling Redundancy

Table V summarizes the results on a set of popular data sets with different μ. The parameters N, λ, and η are determined using cross-validation by the algorithm in Fig. 7 in the Supplementary Material. Table V shows the results when $\lambda = 0.002$ because we have found that it is the best choice for most of the data sets. The last column, N, is the number of the hidden nodes selected by the cross-validation for each data set. For the convenience of observation, we list part of the results of different μ's to demonstrate the effects of FS and redundancy control for different data sets. Based on the sensitivity of the parameter μ, the results in Table V are classified into two categories.

1) First Category: Data sets in the first category are sensitive to the parameter μ. With an increase of μ, the number of selected features decreases rapidly. This category consists of Thyroid, Vertebral, Ecoli, Iris, Liver, and Vehicle data sets. With the increase in μ, fewer features are selected, and the maximum correlation value among selected features is decreased. However, the generalization performance is maintained or marginally decreased.

TABLE V
SELECTION OF FEATURES FOR ALL DATA SETS WITH DIMENSION <100 WHEN $\lambda = 0.002$

	Dataset(η)	μ	AccTr	AccTe	AvgSel	MaxCorr	N
1	Thyroid (0.1)	0	98.14	96.73	5.0	0.72	5
		0.04	97.73	97.19	4.1	0.70	
		0.08	95.30	93.46	2.9	0.48	
		0.12	95.19	93.90	2.0	0.41	
	Vertebral (0.2)	0	87.53	85.48	5.2	0.66	5
		0.04	87.78	87.42	3.2	0.52	
		0.08	87.78	87.10	3.0	0.52	
		2	72.97	71.61	1.8	0.06	
	Ecoli (0.1)	0	85.88	83.88	6.9	0.81	7
		0.04	84.62	80.04	6.2	0.60	
		0.08	84.00	81.47	5.1	0.45	
		1	76.32	75.55	2.3	0.17	
	Iris (0.1)	0	97.63	96.00	3.1	0.96	3
		0.04	97.33	96.00	2.9	0.90	
		0.08	96.15	95.33	1.5	0.32	
		0.12	95.92	96.00	1.0	0	
	Liver (0.1)	0	70.53	67.53	5.8	0.74	5
		0.04	70.76	66.36	5.7	0.74	
		0.08	68.73	66.08	5.0	0.66	
		3	61.54	63.50	2.4	0.22	
	Vehicle (0.2)	0	81.22	78.37	17.3	1.00	7
		0.04	81.32	77.20	16.3	0.99	
		0.08	81.22	77.90	15.3	0.98	
		1	78.37	75.31	10.0	0.91	
2	Glass (0.5)	0	78.92	64.39	8.9	0.78	9
		1	77.10	64.50	6.0	0.49	
		2	73.32	67.23	5.3	0.49	
		4	66.36	60.63	3.3	0.44	
	Wine (0.1)	0	100.00	98.30	10.1	0.79	9
		1	99.69	98.33	8.2	0.70	
		2	99.00	98.30	7.0	0.64	
		3	95.44	92.12	4.2	0.38	
	WBC (0.1)	0	97.61	96.92	7.3	0.90	9
		1	97.52	96.63	4.2	0.73	
		3	96.00	94.58	2.7	0.56	
		4	92.39	92.25	1.4	0.14	
	SPECT Heart (0.1)	0	92.97	81.60	20.9	0.76	9
		1	90.93	80.85	14.1	0.69	
		20	88.72	83.87	9.9	0.58	
		100	83.23	79.76	4.5	0.27	
	Sonar (0.2)	0	100.00	79.83	40.5	0.92	11
		1	99.89	80.33	33.6	0.91	
		20	96.21	83.17	11.6	0.68	
		100	85.04	75.10	4.7	0.34	
	SRBCT (0.008)	0	97.99	81.29	2304.1	0.95	20
		6E+03	100.00	93.64	156.9	0.63	
		1.2E+04	100.00	93.96	118.5	0.66	
		2.2E+04	100.00	98.57	99.2	0.59	

Table VI shows the correlation matrix for Thyroid data set. Detailed results for different μ's when $\lambda = 0.002$ are summarized in Table VII, where the columns F1, F2, F3, F4, and F5 represent the selection frequencies of the five features, respectively. When $\mu = 0$, all five features are selected due to the small value of λ. When $\mu = 0.04$, on average, 0.9

TABLE VI
CORRELATION MATRIX FOR THYROID DATA SET

Features	1	2	3	4	5	CCorr
1	1.00	-0.49	-0.54	0.29	0.30	0.11
2	-0.49	1.00	0.72	-0.42	-0.41	-0.13
3	-0.54	0.72	1.00	-0.24	-0.23	0.08
4	0.29	-0.42	-0.24	1.00	0.50	0.56
5	0.30	-0.41	-0.23	0.50	1.00	0.52

TABLE VII
SELECTION OF FEATURES FOR DIFFERENT μ'S ON THYROID DATA SET WITH $\lambda = 0.002$

μ	F1	F2	F3	F4	F5	AccTe	AvgSel	MaxCorr
0	100	100	100	100	100	96.73	5.0	0.72
0.04	80	100	90	40	100	97.19	4.1	0.70
0.08	70	100	0	20	100	93.46	2.9	0.48
0.12	0	100	0	0	100	93.90	2.0	0.41

features are removed, while the average test accuracy exhibits a slight increase (0.46%). It shows a better generalization ability. Feature 4 is sometimes removed because of its higher dependence with other features. Accordingly, MaxCorr is decreased by 0.02 on average. When $\mu = 0.08$, more features are discarded with 3.27% decrease in test accuracy. Feature 3 makes the least contribution to classification (in terms of class-correlation, as shown in Table VI) and is completely discarded in this case. But why feature 4 is removed, while feature 5 is selected? Compared with feature 5, feature 4 has a slightly higher contribution to classification and a lower correlation with feature 1. However, it has a higher correlation with feature 2, which is always selected. Therefore, when its most correlated feature, i.e., feature 5, is selected, it is penalized to be removed. This is verified when $\mu = 0.12$, where feature 4 is totally discarded. Along with feature 5, feature 2 is always selected because it has a higher contribution to classification than that of features 1 and 3, and it has lower correlations with the selected features. When $\mu = 0.12$, feature 1 is removed as well due to its little contribution to classification and higher correlation with feature 2. Consequently, MaxCorr is decreased by 0.31 on average. In this case, on average, three features are removed with a decrease of 2.83% in test accuracy.

Similar results can be observed for other data sets in this category. For example, for the Vertebral data set, when $\mu = 0.08$, on average, half of the features are removed, while the test accuracy increases by 1.62% than the case when $\mu = 0$. For Ecoli, when $\mu = 0.08$, the maximum correlation and test accuracy are decreased by 0.36 and 2.41%, respectively. For Iris, when $\mu = 0$, a feature subset with a high correlation of 0.96 is selected. When $\mu = 0.12$, only one feature is selected with no dependence, while there is no decline in test performance. All these results show the effectiveness of the SGLC method in FS and redundancy control on these data sets.

2) Second Category: The data sets in this category are a bit slow in reacting to the change in μ. The relatively larger dimension may be a possible factor. With suitable value of μ, the proposed method is still effective in controlling of the redundancy among the selected features.

TABLE VIII
SELECTION OF FEATURES FOR DIFFERENT μ'S ON WBC DATA SET WITH $\lambda = 0.002$

μ	F1	F2	F3	F4	F5	F6	F7	F8	F9
0	100	100	95	100	25	100	100	100	10
1	100	100	5	5	0	100	10	100	0
3	70	80	0	0	0	100	0	20	0
4	20	20	0	0	0	100	0	0	0

TABLE IX
CORRELATION MATRIX FOR WBC DATA SET

	1	2	3	4	5	6	7	8	9	CCorr
1	1.0	0.64	0.65	0.49	0.52	0.59	0.55	0.53	0.35	0.71
2	0.64	1.0	**0.91**	0.71	0.75	0.69	0.76	0.72	0.46	0.82
3	0.65	**0.91**	1.0	0.69	0.72	0.71	0.74	0.72	0.44	0.82
4	0.49	0.71	0.69	1.0	0.59	0.67	0.67	0.6	0.42	0.71
5	0.52	0.75	0.72	0.59	1.0	0.59	0.62	0.63	0.48	0.69
6	0.59	0.69	0.71	0.67	0.59	1.0	0.68	0.58	0.34	0.82
7	0.55	0.76	0.74	0.67	0.62	0.68	1.0	0.67	0.35	0.76
8	0.53	0.72	0.72	0.6	0.63	0.58	0.67	1.0	0.43	0.72
9	0.35	0.46	0.44	0.42	0.48	0.34	0.35	0.43	1.0	0.42

Some additional results for the WBC data set are shown in Table VIII, where F1–F9 represent the selection frequencies of the nine features, respectively. The correlation structure of this data set is displayed in Table IX. When $\lambda = 0.002$ and $\mu = 0$, on average, 7.3 features are selected, and the selected set contains some highly correlated features. Features 5 and 9 are mostly deleted due to their low correlation to class labels, as shown in Table IX. When $\mu = 1$, features 5 and 9 are always removed, while features 3, 4, and 7 are mostly removed. Feature 3 has the largest correlation to feature 2 and, thus, is removed. Features 4 and 7 have a higher correlation with the selected features, especially with feature 2. Thus, they are removed as well. In this case, compared with the original data set when $\mu = 0$, the maximum correlation is decreased by 0.17, and the test accuracy is slightly dropped by 0.29%, as shown in Table V. When $\mu = 3$, more than $2/3rd$ features are discarded. Features 3, 4, 5, 7, and 9 are always deleted for their higher correlation with the selected features. The maximum correlation of selected features is decreased by 0.34 compared with the originally selected subset, and AccTe is decreased by 2.34%. When $\mu = 4$, on average, 1.4 features are selected. Feature 8 is completely dropped for it has a correlation of 0.72 with feature 2, and this is the highest correlation in the selected subset. Consequently, the maximum correlation is significantly decreased by 0.76, while the test performance is dropped by about 4.67%.

The results in Table V indicate that the proposed method is effective on the Sonar data set as well. When $\mu = 0$, almost $\frac{1}{3}$rd features are removed. A subset with a maximum correlation of 0.92 is selected. When $\mu = 1$, almost half of the features are discarded, while the test performance has a

TABLE X
CORRELATION MATRIX FOR THE NOISY IRIS DATA SET

Features	1	2	3	4	5	6	7	8	CCorr
1	1.00	-0.11	0.87	0.82	0.82	0.82	0.05	0.56	0.78
2	-0.11	1.00	-0.42	-0.36	-0.36	-0.36	0.01	-0.26	-0.42
3	0.87	-0.42	1.00	0.96	0.96	0.96	0.04	0.65	0.95
4	0.82	-0.36	0.96	1.00	1.00	1.00	0.05	0.67	0.96
5	0.82	-0.36	0.96	1.00	1.00	1.00	0.05	0.67	0.96
6	0.82	-0.36	0.96	1.00	1.00	1.00	0.05	0.67	0.96
7	0.05	0.01	0.04	0.05	0.05	0.05	1.00	0.04	0.03
8	0.56	-0.26	0.65	0.67	0.67	0.67	0.04	1.00	0.64

0.5% increase compared with the case $\mu = 0$. The maximum correlation of the selected features is dropped only by 0.01. This indicates some redundant features may be present in the selected set. After more than $\frac{4}{5}$th features are deleted, with $\mu = 20$, the maximum correlation is significantly dropped by 0.24. Remarkably, the test accuracy increased by 3.34% compared with the case with $\mu = 0$. This demonstrates that the removal of some redundant features produces better generalization ability. About 92% features are removed when $\mu = 100$. As a result, the test accuracy is decreased by 4.73%. This suggests that some useful features with low dependence might have been removed when $\mu = 100$. For other data sets in this category, the SGLC method also works well, as shown in Table V.

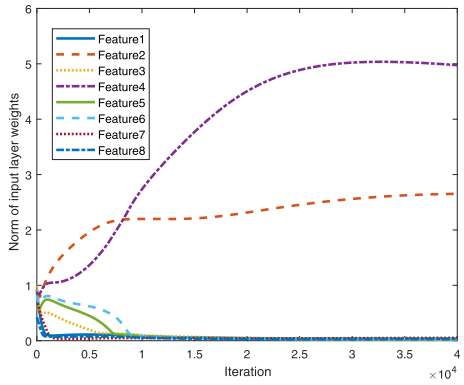

Fig. 4. Norm of the connecting weight vectors of input nodes for the Noisy Iris data set when $\eta = 0.002$, $\lambda = 0.5$, and $\mu = 0.5$.

F. Noisy Data Sets

To show the effectiveness of SGLC for noisy data, the results on two noisy data sets are discussed in this section. For all tests, nine hidden nodes are used for both data sets.

The Noisy Iris data set is created by adding four dummy features, namely, features 5–8, to the original Iris data set. Feature 5 is created by 0.8 times feature 4. Feature 6 is created by 1.2 times feature 4. Feature 7 is a random noise in [0, 1]. Feature 8 is created by feature 4 multiplied by random numbers in [0, 1]. The correlation matrix of all features in the Noisy Iris data set is shown in Table X. Obviously, there is high redundancy in this data set.

Fig. 4 shows the norm of the connecting weight vectors of the input nodes for the Noisy Iris data set for the SGLC method when $\eta = 0.002$, $\lambda = 0.5$, and $\mu = 0.5$. From Fig. 4, we can see that the connecting weight norms of features 4 and 2 are high, while the connecting weight norms of other features are penalized to zero. In this case, features 4 and 2 are selected. This is because feature 4 is important for classification, and feature 2 has a low correlation to feature 4 and has some negative correlation with class labels, as shown in Table X. The connecting weight norms of features 1 and 8 are penalized to be very close to zero around iteration 1200, where both are removed. This happens because they have a higher correlation with feature 4. Features 3, 5, and 6 contribute to classification; nevertheless, they are highly correlated with feature 4. None of them should be selected simultaneously along with the selection of feature 4. Therefore, their connecting weight norms are penalized to almost zero between iteration from 6000 to 8000. The random noise, feature 7, has a low correlation to feature 4. However, it has no contribution to minimize the cost function. As a result, the norm of its connecting weight vector is penalized to zero around iteration 1300. In this case, the maximum correlation coefficient between the selected features is 0.36, namely, the absolute value of the correlation coefficient between the features 2 and 4, and the test accuracy obtained is 95.24%. These results demonstrate that the proposed method is robust to random noise and multiplicative noise. The detailed generation of the Noisy Balance data set and the analysis of the results of the data set are given in the Supplementary Material.

Table XI shows the detailed selection results for SGLC method on the Noisy Iris and Noisy Balance data sets. For both data sets, we use $\eta = 0.001$. The first column in Table XI shows the results on the Noisy Iris data set when $\lambda = 0.5$ and $\mu = 0.5$. Feature combinations $s_{2,6}$ and $s_{2,4,6}$ are mostly selected. However, feature 6 has a high correlation to feature 4, so a significant redundancy exists in $s_{2,4,6}$. In addition, each of the redundant feature combinations $s_{2,3,4}$, $s_{2,3,4,5}$, and $s_{2,3,5}$ is selected ten times as well. This is because the parameter μ is small. When μ increases to 0.7, we can see from the second column that the selection of redundant features is decreased and $s_{2,4}$ is the most frequently selected set. Highly redundant combinations $s_{2,4,6}$ and $s_{2,3,4,5}$ are removed. This is due to the increase of the penalty for redundancy. For the Noisy Balance data set, the third column of Table XI shows the results when $\lambda = 0.5$ and $\mu = 0.5$. In this case, feature combinations $s_{1,2,3,5}$, $s_{1,2,3,4}$, $s_{1,2,3,4,5}$, and $s_{1,2,3,6}$ are usually selected. Here, $s_{1,2,3,4,5}$ is a redundant feature combination due

TABLE XI
SUMMARY OF RESULTS FOR NOISY IRIS AND NOISY BALANCE DATA SETS

Noisy Iris		Noisy Balance	
η=0.001	η=0.001	η=0.001	η=0.001
λ=0.5	λ=0.5	λ=0.5	λ=0.5
μ=0.5	μ= 0.7	μ=0.5	μ=0.7
$s_{2,6}$=20			
$s_{2,4,6}$=20			
$s_{2,3}$=10	$s_{2,4}$=50		
s_5=10	$s_{2,6}$=20	$s_{1,2,3,5}$=40	$s_{1,2,3,5} = 50$
$s_{2,4}$=10	$s_{2,3,4}$=10	$s_{1,2,3,4}$=30	$s_{1,2,3,6}$=30
$s_{2,3,4}$=10	$s_{2,5}$=10	$s_{1,2,3,4,5}$=20	$s_{1,2,3,4}$=20
$s_{2,3,4,5}$=10	$s_{2,3,6}$=10	$s_{1,2,3,6}$=10	s_{others}=0
$s_{2,3,6}$=10	s_{others}=0	s_{others}=0	
s_{others}=0			
AvgSel=2.5	AvgSel=2.2	AvgSel=4.2	AvgSel=4
AccTr=0.96	AccTr= 0.97	AccTr=0.96	AccTr=0.95
AccTe=0.96	AccTe=0.95	AccTe=0.96	AccTe=0.96
MaxCorr=1	MaxCorr=0.96	MaxCorr=1	MaxCorr=0

TABLE XII
COMPARISON OF RESULTS OF THE PROPOSED METHOD WITH THOSE OF RCFS AND mFSMLP-Cor (RESULTS OF RCFS AND mFSMLP-Cor ARE DIRECTLY COLLECTED FROM [6])

Datasets	RCFS		mFSMLP-CoR		SGLC		
	AccTe	Max-Corr	AccTe	Max-Corr	AccTe	Avg-Corr	Max Corr
Iris	95.4	0.96	94.8	**0.42**	**96.0**	**0.42**	0.96
Synthetic	99.7	**0.46**	**100**	**0.46**	**100**	**0.46**	**0.46**
Thyroid	87.9	0.72	90.7	0.42	**94.4**	**0.41**	**0.41**
WBC	94.6	0.90	95.3	0.69	**96.0**	**0.64**	0.69
Wine	94.8	0.87	97.6	0.64	**97.8**	**0.62**	0.64
Glass	45.5	0.54	62.9	**0.48**	**66.8**	0.49	0.49
Sonar	75.4	0.91	77.6	0.63	**78.4**	**0.62**	0.79
SPECT Heart	78.1	0.63	80.1	**0.51**	**82.0**	**0.51**	0.63
Ecoli	78.6	0.40	79.8	0.40	**80.0**	**0.35**	0.81
Colon	75.6	0.99	80.6	**0.47**	**82.1**	**0.47**	0.55
SRBCT	92.0	0.97	95.0	0.68	**95.3**	**0.59**	0.76
ARP	69.9	0.99	88.5	**0.62**	**89.7**	**0.62**	0.70
PIE	93.1	0.98	95.7	0.59	**95.8**	**0.48**	0.55
ORL	62.5	0.99	78.5	**0.67**	**80.0**	**0.67**	0.69

to the complete correlation between features 4 and 5. When we increase the value of μ to 0.7, we can see that $s_{1,2,3,4,5}$ is eliminated. The MaxCorr is decreased from 1 to 0, which means that no redundant feature is selected.

G. Comparison With Other Methods

We now compare the proposed method with the other two approaches with redundancy handling. The redundancy constrained FS (RCFS) [33] selects features with trace-based class separability and constraining redundancy. It selects the user-provided number of features. The mFSMLP-CoR [6] is an improved version of FSMLP-CoR with a more effective learning process. Detailed comparisons with RCFS, mFSMLP-CoR, and the proposed method on different data sets are given in Table XII. The results of RCFS and mFSMLP-CoR are directly collected from [6]. The number of hidden nodes for SGLC is determined by cross-validation, as shown in Table V. In Step 1, the determined numbers of hidden nodes for Colon, ARP, PIE, and ORL data sets are 300, 400, 400, and 600, respectively. For the backpropagation process in Step 3, the determined numbers of hidden nodes for Colon, ARP, PIE, and ORL data sets are 50, 100, 100, and 400, respectively. To make a fair comparison, the results in this table are achieved by using the same number of selected features for all three methods. The other optimal values of parameters in SGLC are determined by the cross-validation as in Fig. 7 in the Supplementary Material. The results in bold represent the best ones. The mFSMLP-CoR reported the maximum absolute pairwise correlations over the tenfold experiment. However, the global maximum over the tenfold may be misleading. For example, in nine of the ten folds, the maximum correlation could be very low and in just one fold; it could be high resulting in the value of the max-correlation very high. This may give a false perception of the results. Hence, along with the maximum correlation, we also report the average of the maximum absolute correlations over the tenfold. From the table, we can see that the average maximum correlation, which indicates the level of redundancy in the selected feature subset for the proposed method, is quite low. Moreover, most of the corresponding test results are higher than those based on the other two methods using the same number of selected features. To check the statistical significance of our results, we have conducted the one-tailed Wilcoxon signed ranked test. While comparing SGLC with mFSML-Cor at $\alpha = 0.05$ (P = 0.0008), SGLC performs significantly better than RCFS at $\alpha = 0.05$ in terms of test accuracy. Similarly, when compared with RCFS, SGLC is found to be significantly better performing than RCFS at $\alpha = 0.05$ ($P = 0.0005$). These show that when we select a fixed number features, the proposed method can make a better compromise between redundancy and accuracy to select more appropriate features yielding better classification performance compared with the other two methods.

We have also compared the performance of the proposed method with two methods of FS in conjunction with a fuzzy rule-based classifier. The first method uses a fuzzy rule-based framework to select features, and the classifier is also a fuzzy rule-based classifier [41]. In Table XIII, this method is denoted as FRBS-FC. The other FS method is the mRMR method [28], but the classifier is also a fuzzy rule-based classification system. In Table XIII, we have reported the classification performance of the two FS methods directly from [41, Table III]. In order to make a fair comparison, for each data set, we have noted the number of features used for the mRMR method in Table III of [41] and picked the same number of features (rounded) for our method, SGLC. For the mRMR [28] method, we directly adopt the results given in [41], namely, mRMR-FC in Table XIII. For Appendicitis, SGLC achieves 3.5% higher accuracy than the mRMR-FC method while using the same number of selected features. Compared with the FRBS-FC method, for this data set, SGLC

TABLE XIII
COMPARISON RESULTS WITH FRBS-FC AND mRMR. THE FRBS-FC AND mRMR RESULTS ARE OBTAINED FROM [41]

Datasets	FRBS-FC		mRMR-FC		SGLC		
	AvgSel	AccTe	AvgSel	AccTe	AvgSel	AccTe	Avg MaxCorr
Iris(4)	1.0	**96.0**	1	**96.0**	1	**96.0**	**0.0**
Appendicitis(7)	2.1	83.4	2	84.9	2	**88.4**	**0.58**
Glass(9)	5.3	64.7	5	64.9	5	**65.0**	**0.48**
Wdbc(30)	1.95	94.6	2	91.1	2	**94.7**	**0.36**
Phoneme(5)	2.1	75.7	2	77.1	2	**77.2**	**0.03**
Pima(8)	2.2	73.8	2	74.8	2	**76.3**	**0.22**
Heart(13)	4.2	76.6	4	81.9	4	**83.3**	**0.31**
Spectf Heart(44)	2.2	78.8	2	79.9	2	**80.2**	**0.44**
Spambase(57)	5.2	85.1	5	86.6	5	**87.1**	**0.16**

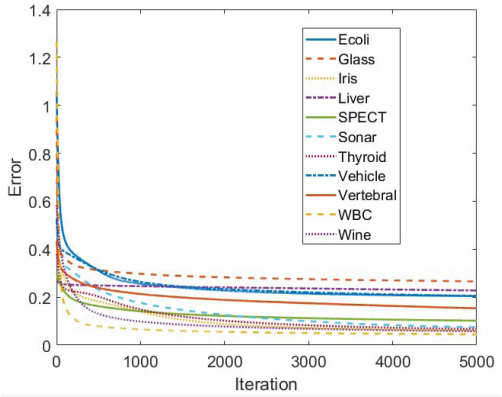

Fig. 5. Training error curves to verify monotonicity.

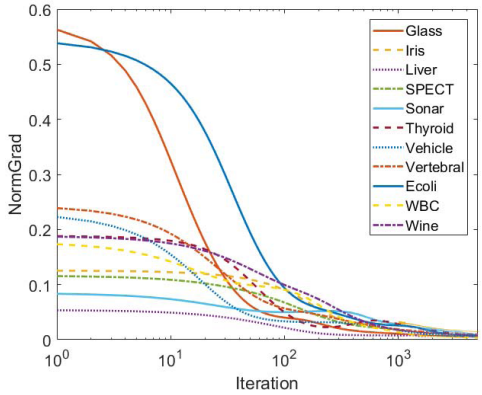

Fig. 6. Gradient norm curves to verify the convergence of the algorithm.

uses fewer features (on average) but generates 5% higher performance. Similar results can be observed for the Glass and Wdbc data sets. SGLC obtains better generalization ability than the other two methods. For the other five data sets also, we can see that SGLC obtains better test accuracies with less or the same number of features than the other two methods, and all the corresponding average maximum absolute correlation coefficients are quite small. This indicates that these data sets contain some redundant features, and the SGLC method can pick more discriminative features by monitoring the use of redundant features.

H. Validation of Theoretical Results

Fig. 7 in the Supplementary Material shows the average training error curves obtained by the algorithm in Fig. 7 on different data sets. It reveals that all error curves decrease monotonically to stable minimum values with the increase of iteration. Thus, the monotonicity of Theorem 1 is validated. We note that, for some simulations, if the condition A2 is not satisfied, the monotonicity property may not valid, yet the algorithm will not face any problem to yield a useful solution. Fig. 6 demonstrates the performance curves of gradients on different data sets. It reveals that all the norms of gradients tend to zero with the increase of iteration. The convergence of the algorithm in Theorem 2 is verified as well.

VI. CONCLUSION

In this article, we have proposed an integrated FS scheme with control on the level of redundancy. The Group Lasso regularization is exerted on the weights connecting the input and hidden layers of a neural network to produce group sparsity and select useful features. Most data sets, however, usually contain some redundant features. Keeping all discriminatory redundant features will increase the cost and complexity of designing the system. On the other hand, the removal of all redundant features is also not good. A system with some redundancy brings about an easier learning process and is more robust to measurement and other noise. Hence, another essential penalty based on correlation measure is designed to control the level of redundancy in the selected features. However, both penalties are nondifferential at the origin. This not only leads to difficulties in the theoretical analysis but also generates oscillations in the experiments. A smoothing technique is used to overcome this drawback. On this basis, the theoretical analysis for monotonicity and convergence is presented. Although we have used the Pearson Correlation coefficient as a measure of dependence, in a more general setting, other measures of dependence, such as mutual information, can also be used without any changes in the theory and learning algorithm.

Twenty-six data sets, which cover low-, medium-, and high-dimensional data sets, are used to test the proposed method. A three-stage scheme is employed to evaluate the effectiveness of FS. The experimental results demonstrate that the proposed

method can remove bad/indifferent features and simultaneously control the level of redundancy in the selected features. Comparative analysis shows a competitive performance with a lower or comparable level of redundancy in the selected features. These empirical results are found to be consistent with the theoretical results.

REFERENCES

[1] Y. Saeys, I. Inza, and P. Larranaga, "A review of feature selection techniques in bioinformatics," *Bioinformatics*, vol. 23, no. 19, pp. 2507–2517, Oct. 2007.

[2] J. Li *et al.*, "Feature selection: A data perspective," *ACM Comput. Surv.*, vol. 50, no. 6, pp. 1–45, 2017.

[3] S. Li and D. Wei, "Extremely high-dimensional feature selection via feature generating samplings," *IEEE Trans. Cybern.*, vol. 44, no. 6, pp. 737–747, Jun. 2014.

[4] I. Guyon and A. Elisseeff, "An introduction to variable and feature selection," *J. Mach. Learn. Res.*, vol. 3, pp. 1157–1182, Jan. 2003.

[5] H. Yin, "ViSOM–a novel method for multivariate data projection and structure visualization," *IEEE Trans. Neural Netw.*, vol. 13, no. 1, pp. 237–243, 2002.

[6] R. Chakraborty and N. R. Pal, "Feature selection using a neural framework with controlled redundancy," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 26, no. 1, pp. 35–50, Jan. 2015.

[7] S. Solorio-Fernández, J. F. Martínez-Trinidad, and J. A. Carrasco-Ochoa, "A new unsupervised spectral feature selection method for mixed data: A filter approach," *Pattern Recognit.*, vol. 72, pp. 314–326, Dec. 2017.

[8] M. Liu and D. Zhang, "Feature selection with effective distance," *Neurocomputing*, vol. 215, pp. 100–109, Nov. 2016.

[9] R. Ruiz, J. C. Riquelme, and J. S. Aguilar-Ruiz, "Incremental wrapper-based gene selection from microarray data for cancer classification," *Pattern Recognit.*, vol. 39, no. 12, pp. 2383–2392, Dec. 2006.

[10] M. Monirul Kabir, M. Monirul Islam, and K. Murase, "A new wrapper feature selection approach using neural network," *Neurocomputing*, vol. 73, nos. 16–18, pp. 3273–3283, Oct. 2010.

[11] L. Jiang, L. Zhang, L. Yu, and D. Wang, "Class-specific attribute weighted naive Bayes," *Pattern Recognit.*, vol. 88, pp. 321–330, Apr. 2019.

[12] K. Nag and N. R. Pal, "A multiobjective genetic programming-based ensemble for simultaneous feature selection and classification," *IEEE Trans. Cybern.*, vol. 46, no. 2, pp. 499–510, Feb. 2016.

[13] H. Yan and J. Yang, "Sparse discriminative feature selection," *Pattern Recognit.*, vol. 48, no. 5, pp. 1827–1835, May 2015.

[14] Z. Zhang, L. Bai, Y. Liang, and E. Hancock, "Joint hypergraph learning and sparse regression for feature selection," *Pattern Recognit.*, vol. 63, pp. 291–309, Mar. 2017.

[15] J. Gui, Z. Sun, S. Ji, D. Tao, and T. Tan, "Feature selection based on structured sparsity: A comprehensive study," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 28, no. 7, pp. 1490–1507, Jul. 2017.

[16] J. Wen, Z. Lai, W. K. Wong, J. Cui, and M. Wan, "Optimal feature selection for robust classification via l2,1-norms regularization," in *Proc. 22nd Int. Conf. Pattern Recognit.*, Aug. 2014, pp. 517–521.

[17] C. Ding, D. Zhou, X. He, and H. Zha, "R1-PCA: Rotational invariant L1-norm principal component analysis for robust subspace factorization," in *Proc. 23rd Int. Conf. Mach. Learn. (ICML)*. New York, NY, USA: ACM, 2006, pp. 281–288.

[18] F. Nie, H. Huang, X. Cai, and C. Ding, "Efficient and robust feature selection via joint $\ell_{2,1}$-norms minimization," in *Proc. Adv. Neural Inf. Process. Syst.*, 2010, pp. 1813–1821.

[19] D. Han and J. Kim, "Unified simultaneous clustering and feature selection for unlabeled and labeled data," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 29, no. 12, pp. 6083–6098, Dec. 2018.

[20] J. Wen, Z. Lai, Y. Zhan, and J. Cui, "The l2,1-norm-based unsupervised optimal feature selection with applications to action recognition," *Pattern Recognit.*, vol. 60, pp. 515–530, Dec. 2016.

[21] Z. Xie and Y. Xu, "Sparse group LASSO based uncertain feature selection," *Int. J. Mach. Learn. Cybern.*, vol. 5, no. 2, pp. 201–210, Apr. 2014.

[22] J. Wang, C. Xu, X. Yang, and J. M. Zurada, "A novel pruning algorithm for smoothing feedforward neural networks based on group lasso method," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 29, no. 5, pp. 2012–2024, May 2018.

[23] K. Sun, S.-H. Huang, D. S.-H. Wong, and S.-S. Jang, "Design and application of a variable selection method for multilayer perceptron neural network with LASSO," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 28, no. 6, pp. 1386–1396, Jun. 2017.

[24] F. Li, J. M. Zurada, Y. Liu, and W. Wu, "Input layer regularization of multilayer feedforward neural networks," *IEEE Access*, vol. 5, pp. 10979–10985, 2017.

[25] S. Scardapane, D. Comminiello, A. Hussain, and A. Uncini, "Group sparse regularization for deep neural networks," *Neurocomputing*, vol. 241, pp. 81–89, Jun. 2017.

[26] M. A. Hall, "Correlation-based feature selection for machine learning," Ph.D. dissertation, Dept. Comput. Sci., Univ. Waikato, Hamilton, New Zealand, 1999.

[27] L. Jiang, C. Li, S. Wang, and L. Zhang, "Deep feature weighting for naive bayes and its application to text classification," *Eng. Appl. Artif. Intell.*, vol. 52, pp. 26–39, Jun. 2016.

[28] C. Ding and H. Peng, "Minimum redundancy feature selection from microarray gene expression data," in *Proc. IEEE Bioinf. Conf.*, 2003, pp. 523–528.

[29] H. Peng, F. Long, and C. Ding, "Feature selection based on mutual information criteria of max-dependency, max-relevance, and min-redundancy," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 27, no. 8, pp. 1226–1238, Aug. 2005.

[30] P. A. Estevez, M. Tesmer, C. A. Perez, and J. M. Zurada, "Normalized mutual information feature selection," *IEEE Trans. Neural Netw.*, vol. 20, no. 2, pp. 189–201, Feb. 2009.

[31] L. Jiang, L. Zhang, C. Li, and J. Wu, "A correlation-based feature weighting filter for naive Bayes," *IEEE Trans. Knowl. Data Eng.*, vol. 31, no. 2, pp. 201–213, Feb. 2019.

[32] G. Qu, S. Hariri, and M. Yousif, "A new dependency and correlation analysis for features," *IEEE Trans. Knowl. Data Eng.*, vol. 17, no. 9, pp. 1199–1207, Sep. 2005.

[33] L. Zhou, L. Wang, and C. Shen, "Feature selection with redundancy-constrained class separability," *IEEE Trans. Neural Netw.*, vol. 21, no. 5, pp. 853–858, May 2010.

[34] J. Xu, B. Tang, H. He, and H. Man, "Semisupervised feature selection based on relevance and redundancy criteria," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 28, no. 9, pp. 1974–1984, Sep. 2017.

[35] A. Senawi, H.-L. Wei, and S. A. Billings, "A new maximum relevance-minimum multicollinearity (MRmMC) method for feature selection and ranking," *Pattern Recognit.*, vol. 67, pp. 47–61, Jul. 2017.

[36] S. Tabakhi and P. Moradi, "Relevance–redundancy feature selection based on ant colony optimization," *Pattern Recognit.*, vol. 48, no. 9, pp. 2798–2811, Sep. 2015.

[37] J. Feng, L. Jiao, F. Liu, T. Sun, and X. Zhang, "Unsupervised feature selection based on maximum information and minimum redundancy for hyperspectral images," *Pattern Recognit.*, vol. 51, pp. 295–309, Mar. 2016.

[38] F. Nie, S. Yang, R. Zhang, and X. Li, "A general framework for auto-weighted feature selection via global redundancy minimization," *IEEE Trans. Image Process.*, vol. 28, no. 5, pp. 2428–2438, May 2019.

[39] Z. Li and J. Tang, "Unsupervised feature selection via nonnegative spectral analysis and redundancy control," *IEEE Trans. Image Process.*, vol. 24, no. 12, pp. 5343–5355, Dec. 2015.

[40] N. R. Pal and M. Malpani, "Redundancy-constrained feature selection with radial basis function networks," in *Proc. Int. Joint Conf. Neural Netw. (IJCNN)*, Brisbane, QLD, Australia, Jun. 2012, pp. 1–8.

[41] I.-F. Chung, Y.-C. Chen, and N. R. Pal, "Feature selection with controlled redundancy in a fuzzy rule based framework," *IEEE Trans. Fuzzy Syst.*, vol. 26, no. 2, pp. 734–748, Apr. 2018.

[42] M. Banerjee and N. R. Pal, "Unsupervised feature selection with controlled redundancy (UFeSCoR)," *IEEE Trans. Knowl. Data Eng.*, vol. 27, no. 12, pp. 3390–3403, Dec. 2015.

[43] M. Forti, P. Nistri, and M. Quincampoix, "Generalized neural network for nonsmooth nonlinear programming problems," *IEEE Trans. Circuits Syst. I, Reg. Papers*, vol. 51, no. 9, pp. 1741–1754, Sep. 2004.

[44] H. Zhang, J. Wang, Z. Sun, J. M. Zurada, and N. R. Pal, "Feature selection for neural networks using group lasso regularization," *IEEE Trans. Knowl. Data Eng.*, vol. 32, no. 4, pp. 659–673, Apr. 2020, doi: 10.1109/TKDE.2019.2893266.

[45] J. Wang, Y. Wen, Z. Ye, L. Jian, and H. Chen, "Convergence analysis of BP neural networks via sparse response regularization," *Appl. Soft Comput.*, vol. 61, pp. 354–363, Dec. 2017.

[46] E. Mizutani and S. E. Dreyfus, "On complexity analysis of supervised MLP-learning for algorithmic comparisons," in *Proc. Int. Joint Conf. Neural Networks. (IJCNN)*, 2001, pp. 347–352.