Efficient Top-K Feature Selection Using Coordinate Descent Method

Lei Xu[1,2*], **Rong Wang**[2*], **Feiping Nie**[1,2†], **Xuelong Li**[2]

[1] School of Computer Science, Northwestern Polytechnical University, Xi'an 710072, P.R. China
[2] School of Artificial Intelligence, OPtics and ElectroNics (iOPEN), Northwestern Polytechnical University, Xi'an 710072, P.R. China

Abstract

Sparse learning based feature selection has been widely investigated in recent years. In this study, we focus on the $l_{2,0}$-norm based feature selection, which is effective for exact top-k feature selection but challenging to optimize. To solve the general $l_{2,0}$-norm constrained problems, we novelly develop a parameter-free optimization framework based on the coordinate descend (CD) method, termed CD-LSR. Specifically, we devise a skillful conversion from the original problem to solving one continuous matrix and one discrete selection matrix. Then the nontrivial $l_{2,0}$-norm constraint can be solved efficiently by solving the selection matrix with CD method. We impose the $l_{2,0}$-norm on a vanilla least square regression (LSR) model for feature selection and optimize it with CD-LSR. Extensive experiments exhibit the efficiency of CD-LSR, as well as the discrimination ability of $l_{2,0}$-norm to identify informative features. More importantly, the versatility of CD-LSR facilitates the applications of the $l_{2,0}$-norm in more sophisticated models. Based on the competitive performance of $l_{2,0}$-norm on the baseline LSR model, the satisfactory performance of its applications is reasonably expected. *The source MATLAB code are available at:* https://github.com/solerxl/Code_For_AAAI_2023.

Introduction

Feature selection techniques aim to select the most informative feature subset from the original data, thereby eliminating irrelevant features and speeding up the learning process (Li et al. 2018, 2020). It has a wide range of real-world applications, such as image processing (Sun, Bebis, and Miller 2004), text mining (Forman and Kirshenbaum 2008), mass spectrometry (Saeys, Inza, and Larrañaga 2007), and so on.

Sparse learning methodologies, which impose sparse regularization on the coefficient matrix to limit the number of non-zero entries, are favored in feature selection (Pang et al. 2019; Chang et al. 2014; Nie et al. 2010). Based on sparse learning techniques, the features corresponding to the non-zero entries will be selected, and the others will be discarded. Among them, the $l_{2,1}$-norm constraint is the most commonly used technique for selecting task-shared features (Nie et al. 2010; He et al. 2012), which is defined as

$$\|\mathbf{W}\|_{2,1} = \sum_{i=1}^{d} \sqrt{\sum_{j=1}^{c} w_{i,j}^2}, \tag{1}$$

where $\mathbf{W} \in \mathcal{R}^{d\times c}$ denotes the coefficient matrix and $w_{i,j}$ denotes the (i,j)-th entry of $\mathbf{W}$. Most previous work performs feature selection by adding the regularization $\lambda\|\mathbf{W}\|_{2,1}$ to the objective function, with λ being the regularization parameter. For example, (Nie et al. 2010) proposed a Robust Feature Selection model to simultaneously overcome outliers issues and solve feature selection problems by imposing $l_{2,1}$-norm on the objective function and the regularization term. Although satisfactory performance was achieved via $l_{2,1}$-norm, its sparsity heavily depends on the parameter λ, which does not have an explicit meaning. Thus, we are not clear about how many features are actually selected. Therefore, we have to spend a long time searching for the appropriate λ to select the exact number of features.

In contrast, the $l_{2,0}$-norm, which is defined as

$$\|\mathbf{W}\|_{2,0} = \sum_{i=1}^{d} \|\sum_{j=1}^{c} w_{i,j}^2\|_0, \tag{2}$$

dispenses with the heavy burden of parameter tuning in feature selection. Specifically, we can select k features by simply imposing the constraint $\|\mathbf{W}\|_{2,0} = k$, where the value of $l_{2,0}$-norm intuitively indicates the exact number of selected features k, such that the parameter tuning process is avoided.

Although $l_{2,0}$-norm is more desirable from the sparsity perspective, it is hard to solve $l_{2,0}$-norm constraint due to its non-convexity. To date, there are mainly two types of algorithms proposed to solve this nontrivial problem, that is, augmented Lagrangian method(ALM) based algorithms and projection gradient descent (PGD) based algorithms. The former introduces the auxiliary variables to solve $l_{2,0}$-norm separately and iteratively updates the original solution variable and the auxiliary variable based on ALM (Bertsekas 1982). For example, Cai et al. (Cai, Nie, and Huang 2013) proposed a robust and pragmatic feature selection method with $l_{2,0}$-norm constraint and solved the model with ALM. Besides, Zhang et al. (Zhang et al. 2021) proposed to transform the $l_{2,0}$-norm constraint into an equivalent 0-1 integer

*These authors contributed equally.
†Corresponding author: Feiping Nie (email: feipingnie@gmail.com).

constraint and solved this problem with ALM by replacing the 0-1 integer constraint with two continuous constraints. In contrast, the PGD based algorithm (Calamai and Moré 1987; Pang et al. 2019) solved the $l_{2,0}$-norm constraint problem by updating the solution with gradient information and projecting the solution to the feasible region at each iteration.

In this paper, we focus on the optimization of general $l_{2,0}$-norm constrained problems. Different from existing $l_{2,0}$-norm-related literature, we propose a novel and efficient coordinate descent based least square regression (CD-LSR) algorithm for this nontrivial problem. We first decompose the constrained variable $\mathbf{W}$ into one discrete selection matrix $\mathbf{S}$ and one weight matrix $\mathbf{V}$, and then update $\mathbf{S}$ and $\mathbf{V}$ iteratively until the objective function value converges. In this way, the $l_{2,0}$-norm constraint is skillfully transformed into the discrete constraint on $\mathbf{S}$, which can be solved efficiently with the coordinate descent (CD) algorithm. We verify the performance of the proposed algorithm on a vanilla LSR model. Extensive experimental results present the superiority of CD-LSR in terms of solution accuracy, fast convergence speed, and classification performance. More importantly, the proposed algorithm is a general optimization framework, which means our algorithm is capable of solving arbitrary $l_{2,0}$-norm constrained models.

The contributions of this paper are as follows:

- We introduce the $l_{2,0}$-norm with a simple LSR model for feature selection, which is simple yet effective with experimentally verified. The $l_{2,0}$-norm constrained model not only exhibits a superior performance, but also gets rid of the tedious tuning cost compared with conventional $l_{2,1}$-norm based models.
- A CD based optimization framework is proposed to solve the general $l_{2,0}$-norm constrained feature selection problems, which is efficient and parameter-free. The proposed algorithm can be applied to arbitrary $l_{2,0}$-norm constrained problems, thus facilitating its extension to more sophisticated models.
- Experimental results on six benchmark datasets validate the effectiveness of the $l_{2,0}$-norm for feature selection, as well as the high efficiency and stability of the proposed optimization framework.

Notations For an arbitrary matrix $\mathbf{Z} \in \mathcal{R}^{m\times n}$, $\mathbf{z}^i$ and $\mathbf{z}_i$ represent the i-th row and i-th column of $\mathbf{Z}$, respectively. The (i,j)-th entry of $\mathbf{Z}$ is written as $z_{i,j}$. The Frobenius norm of the matrix $\mathbf{Z}$ is defined as $\|\mathbf{Z}\|_{\mathrm{F}} = \sqrt{\sum_{i=1}^{m}\|\mathbf{z}^i\|_2^2} = \sqrt{\sum_{i=1}^{m}\sum_{j=1}^{n} z_{i,j}^2}$. When m equals n, the trace operator is denoted as $\mathrm{tr}(\mathbf{Z}) = \sum_{i=1}^{m} z_{i,i}$. $\mathbf{1}_d$ denotes the d-dimensional column vector with all entries being 1. $\mathbf{I}_d$ denotes the d-order identity matrix. Moreover, we define the operator $\mathcal{Q}_d(\cdot)$ as follows:

$$\mathcal{Q}_d(i) = [\underbrace{0,\ldots,0}_{i-1},1,\underbrace{0,\ldots,0}_{d-i}]^{\mathrm{T}}. \tag{3}$$

Namely, $\mathcal{Q}_d(i)$ defines a d-dimensional vector with the i-th entry being 1 and the others being 0.

Related Work

Feature Selection

In general, traditional feature selection algorithms fall into three categories based on how they incorporate with the learning process: filter methods, wrapper methods, and embedded methods. Among them, filter methods (Peng, Long, and Ding 2005; Ding and Peng 2005) select features before the learning process. Therefore, they can be regarded as a data preprocessing step filtering out redundant features. Unlike filter methods, wrapper methods (Hall and Smith 1999; Guyon et al. 2002) are designed as an integral part of the learning algorithms, employing heuristic search strategies to determine some subsets of features and selecting features based on the learning performance. Despite its good performance, wrapper methods need to evaluate the classification performance on each subset of features considered, therefore resulting in expensive computational costs. In contrast to filter methods and wrapper methods in which the feature selection process is significantly different from the training process, embedded methods integrate the feature selection and model training into a unified optimization framework, where feature selection is performed automatically during the training process, so as to achieve both good performance and reasonable computational cost. Given this point, we focus on the embedded methods in this study.

Proposed Method

Problem Formulation

In this paper, we conduct the $l_{2,0}$-norm based feature selection with a simple least square regression model, which is specifically formulated as

$$\min_{\mathbf{W},\mathbf{b}} \|\mathbf{Y} - \mathbf{W}^{\mathrm{T}}\mathbf{X} - \mathbf{b}\mathbf{1}_n^{\mathrm{T}}\|_{\mathrm{F}}^2 \quad s.t.\ \|\mathbf{W}\|_{2,0} = k, \tag{4}$$

where $\mathbf{X} \in \mathcal{R}^{d\times n}$ denotes the d-dimensional data matrix with n samples, $\mathbf{Y} \in \{0,1\}^{c\times n}$ denotes the label matrix with c classes, $\mathbf{W} \in \mathcal{R}^{d\times c}$ denotes the learned coefficient matrix, and $\mathbf{b} \in \mathcal{R}^{c\times 1}$ denotes the bias vector. When $\mathbf{W}$ is learned, the features corresponding to the non-zero rows in $\mathbf{W}$ will be selected.

Optimization

To solve problem (4), we first take the derivative of problem (4) w.r.t. $\mathbf{b}$ and set it to zero, then we have

$$\mathbf{b} = \frac{1}{n}\mathbf{Y}\mathbf{1}_n - \frac{1}{n}\mathbf{W}^{\mathrm{T}}\mathbf{X}\mathbf{1}_n. \tag{5}$$

Substitute Eq. (5) into Eq. (4), problem (4) is converted to

$$\begin{aligned}&\min_{\|\mathbf{W}\|_{2,0}=k} \|\mathbf{Y} - \mathbf{W}^{\mathrm{T}}\mathbf{X} - (\frac{1}{n}\mathbf{Y}\mathbf{1}_n - \frac{1}{n}\mathbf{W}^{\mathrm{T}}\mathbf{X}\mathbf{1}_n)\mathbf{1}_n^{\mathrm{T}}\|_{\mathrm{F}}^2 \\ =&\min_{\|\mathbf{W}\|_{2,0}=k} \|\mathbf{Y}\mathbf{H} - \mathbf{W}^{\mathrm{T}}\mathbf{X}\mathbf{H}\|_{\mathrm{F}}^2,\end{aligned} \tag{6}$$

where $\mathbf{H} = \mathbf{I}_n - \frac{1}{n}\mathbf{1}_n\mathbf{1}_n^{\mathrm{T}}$ is the centering matrix, which possesses the property that $\mathbf{H} = \mathbf{H}\mathbf{H}^{\mathrm{T}}$.

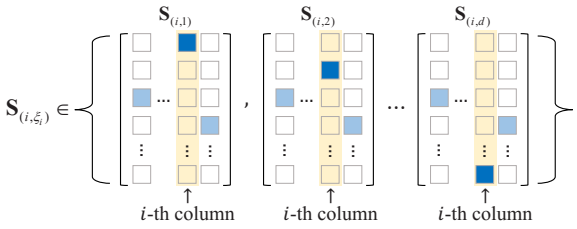


Figure 1: Illustration of coordinate descent method when solving the i-th column in $\mathbf{S}$. The blue blocks represent 1, the blank ones represent 0, and the column highlighted in yellow represents the column to be updated.

After removing the constant term, problem (6) can be further transformed as

$$\min_{\|\mathbf{W}\|_{2,0}=k} \mathrm{tr}(\mathbf{W}^{\mathrm{T}}\mathbf{A}\mathbf{W}) - 2\mathrm{tr}(\mathbf{W}^{\mathrm{T}}\mathbf{B}), \tag{7}$$

where $\mathbf{A} = \mathbf{X}\mathbf{H}\mathbf{X}^{\mathrm{T}}$ and $\mathbf{B} = \mathbf{X}\mathbf{H}\mathbf{Y}^{\mathrm{T}}$.

To solve problem (7), we first decompose $\mathbf{W}$ as follows:

$$\mathbf{W} = \mathbf{S}\mathbf{V}, \tag{8}$$

where $\mathbf{S} \in \{0,1\}^{d\times k}$ is a discrete selection matrix and $\mathbf{V} \in \mathcal{R}^{k\times c}$ is composed of the non-zero rows in $\mathbf{W}$. More specifically, each column $\mathbf{s}_i$ of the selection matrix $\mathbf{S}$ is dependent on the index of the i-th selected feature and has the following formulation:

$$\mathbf{s}_i = \mathcal{Q}_d(\xi_i) = [\underbrace{0,\ldots,0}_{\xi_i-1}, 1, \underbrace{0,\ldots,0}_{d-\xi_i}]^{\mathrm{T}} \tag{9}$$

where ξ_i is the i-th non-zero row index in $\mathbf{W}$. Notably, ξ_i also stands for the index of the i-th selected feature.

Substituting Eq. (8) into problem (7), we have the following equivalent problem:

$$\min_{\mathbf{S},\mathbf{V}} \mathrm{tr}(\mathbf{V}^{\mathrm{T}}\mathbf{S}^{\mathrm{T}}\mathbf{A}\mathbf{S}\mathbf{V}) - 2\mathrm{tr}(\mathbf{V}^{\mathrm{T}}\mathbf{S}^{\mathrm{T}}\mathbf{B}). \tag{10}$$

Next, we are going to solve problem (10) by alternatively updating $\mathbf{S}$ and $\mathbf{V}$ and repeating this process iteratively until the objective function value converges. The concrete process are depicted as follows.

The first step is fixing $\mathbf{S}$ and updating $\mathbf{V}$. When $\mathbf{S}$ is fixed, problem (10) can be transformed as

$$\min_{\mathbf{V}} \mathrm{tr}(\mathbf{V}^{\mathrm{T}}\mathbf{S}^{\mathrm{T}}\mathbf{A}\mathbf{S}\mathbf{V}) - 2\mathrm{tr}(\mathbf{V}^{\mathrm{T}}\mathbf{S}^{\mathrm{T}}\mathbf{B}). \tag{11}$$

Taking the derivative of Eq. (11) w.r.t. $\mathbf{V}$ and setting it to zero, we have

$$\mathbf{V} = (\mathbf{S}^{\mathrm{T}}\mathbf{A}\mathbf{S})^{-1}\mathbf{S}^{\mathrm{T}}\mathbf{B}. \tag{12}$$

The second step is fixing $\mathbf{V}$ and updating $\mathbf{S}$, through which we obtain the following subproblem:

$$\min_{\mathbf{S}} \mathrm{tr}(\mathbf{V}^{\mathrm{T}}\mathbf{S}^{\mathrm{T}}\mathbf{A}\mathbf{S}\mathbf{V}) - 2\mathrm{tr}(\mathbf{V}^{\mathrm{T}}\mathbf{S}^{\mathrm{T}}\mathbf{B}). \tag{13}$$

We solve the selection matrix $\mathbf{S}$ by iterating column by column through coordinate descent method (Wright 2015), during which all columns in $\mathbf{S}$ are fixed except the column to be solved being updated to its optima according to Eq. (13).

Algorithm 1: CD-LSR

Input: $\mathbf{X}$, $\mathbf{Y}$, k
Output: $\boldsymbol{\xi}$, $\mathbf{W}$

Calculate $\mathbf{A} = \mathbf{X}\mathbf{H}\mathbf{X}^{\mathrm{T}}$, $\mathbf{B} = \mathbf{X}\mathbf{H}\mathbf{Y}^{\mathrm{T}}$;
Initialize $\mathbf{S}$;
while not convergence **do**
  Update $\mathbf{V} = (\mathbf{S}^{\mathrm{T}}\mathbf{A}\mathbf{S})^{-1}\mathbf{S}^{\mathrm{T}}\mathbf{B}$;
  Calculate $\mathbf{U} = \mathbf{V}\mathbf{V}^{\mathrm{T}}$;
  for $i = 1 : k$ **do**
    $\boldsymbol{\xi} = \boldsymbol{\xi} \backslash \xi_i$; ▷ Exclude ξ_i from $\boldsymbol{\xi}$.
    for $p = 1 : d$ **do**
      if $p \notin \boldsymbol{\xi}$ **then**
        Calculate $\mathcal{F}(\mathbf{S}_{(i,p)})$ based on Eq. (16);
      end if
    end for
    $\xi_i = \arg\min_p \mathcal{F}(\mathbf{S}_{(i,p)})$;
    $\boldsymbol{\xi} = \boldsymbol{\xi} \cup \xi_i$; ▷ Merge ξ_i into $\boldsymbol{\xi}$.
    Update $\mathbf{s}_i$;
  end for
end while
Calculate $\mathbf{W} = \mathbf{S}\mathbf{V}$.

Supposing that we are going to solve the i-th column of $\mathbf{S}$, which has d candidate solutions including $\{\mathcal{Q}_d(1), \mathcal{Q}_d(2), \ldots, \mathcal{Q}_d(d)\}$ with varying index ξ_i, we denote $\mathbf{S}_{(i,p)}$ with $p \in \{1,\ldots,d\}$ as $\{\mathbf{S}_{(i,1)}, \mathbf{S}_{(i,2)}, \ldots, \mathbf{S}_{(i,d)}\}$ varying from different p. For example, $\mathbf{S}_{(i,p)}$ denotes that it is the i-th column to be solved, where only the p-th entry is 1 and the others are 0. It is noteworthy that $\mathbf{S}_{(i,p)}$ and $\mathbf{S}_{(i,q)}$ are identical except the i-the column for any $p \neq q$, as illustrated in Figure 1. Therefore, the problem with respect to $\mathbf{s}_i$ can be reformulated as

$$\begin{aligned} &\min_{\mathbf{S}_{(i,p)}} \mathcal{F}(\mathbf{S}_{(i,p)}) \\ &\text{s.t. } \mathbf{S}_{(i,p)} \in \{\mathbf{S}_{(i,1)}, \mathbf{S}_{(i,2)}, \ldots, \mathbf{S}_{(i,d)}\}, p \notin \boldsymbol{\xi}, \end{aligned} \tag{14}$$

where $\mathcal{F}(\mathbf{S}_{(i,p)}) = \mathrm{tr}(\mathbf{V}^{\mathrm{T}}\mathbf{S}_{(i,p)}^{\mathrm{T}}\mathbf{A}\mathbf{S}_{(i,p)}\mathbf{V}) - 2\mathrm{tr}(\mathbf{V}^{\mathrm{T}}\mathbf{S}_{(i,p)}^{\mathrm{T}}\mathbf{B})$. Hence, when updating $\mathbf{s}_i$, the problem turns into searching for the best $\mathbf{S}_{(i,p)}$ that minimizes $\mathcal{F}(\mathbf{S}_{(i,p)})$, which can be determined by iterating through all possible $\mathbf{S}_{(i,p)}$ and selecting the optimal one. And the problem (13) can be solved by updating each column in $\mathbf{S}$ sequentially.

Note that we have to iterate through all $d - k + 1$ possible candidates to update one single column. If we directly calculate $\mathcal{F}(\mathbf{S}_{(i,p)})$, it will result in quite a high computational complexity to solve the entire variable $\mathbf{S}$. To reduce the computational cost, we first transform $\mathcal{F}(\mathbf{S}_{(i,p)})$ as

$$\begin{aligned} \mathcal{F}(\mathbf{S}_{(i,p)}) &= \mathrm{tr}(\mathbf{V}^{\mathrm{T}}\mathbf{S}_{(i,p)}^{\mathrm{T}}\mathbf{A}\mathbf{S}_{(i,p)}\mathbf{V}) - 2\mathrm{tr}(\mathbf{V}^{\mathrm{T}}\mathbf{S}_{(i,p)}^{\mathrm{T}}\mathbf{B}) \\ &= \mathrm{tr}(\mathbf{E}_{(i,p)}\mathbf{U}) - 2\mathrm{tr}(\mathbf{V}^{\mathrm{T}}\mathbf{S}_{(i,p)}^{\mathrm{T}}\mathbf{B}), \end{aligned} \tag{15}$$

where $\mathbf{E}_{(i,p)} = \mathbf{S}_{(i,p)}^{\mathrm{T}}\mathbf{A}\mathbf{S}_{(i,p)}$ and $\mathbf{U} = \mathbf{V}\mathbf{V}^{\mathrm{T}}$ are symmetric matrices.

The motivations of the above transformation are threefold: Firstly, since $\mathbf{S}$ is a discrete selection matrix, multiplying $\mathbf{S}$ is equivalent to performing column(row) selection on the multiplied matrix. Therefore, we can directly obtain

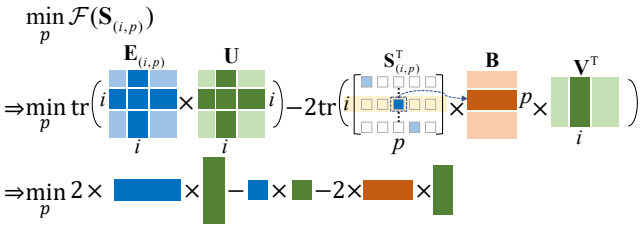


Figure 2: Illustration of the simplification strategy in Eq. (16) by removing irrelevant calculations.

$\mathbf{E}_{(i,p)}$ and $\mathbf{S}_{(i,p)}^{\mathrm{T}}\mathbf{B}$, such that the large matrix multiplication is avoided. Secondly, note that $\mathbf{U}$ can be regarded as a constant value when solving $\mathbf{S}$, for which $\mathbf{U}$ can be precalculated before updating $\mathbf{S}$ at each iteration. Thirdly, there exist too many redundant calculations since we only need to calculate the objective value w.r.t. p. With this in mind, we can further simplify the calculation by eliminating irrelevant terms. Figure 2 illustrates the simplification strategy when solving $\mathbf{s}_i$, where the calculations unrelated to $\mathbf{s}_i$ are removed since they are constant when updating ξ_i.

Consequently, we transform the problem (14) as follows:

$$\begin{aligned} &\min_{p} 2\mathbf{e}_{(i,p)}^{i}\mathbf{u}_i - e_{i,i}u_{i,i} - 2\mathbf{b}^{p}\mathbf{v}^{i\mathrm{T}} \\ &\text{s.t. } p \in \{1,2,\ldots,d\}, p \notin \boldsymbol{\xi}, \end{aligned} \tag{16}$$

where $\mathbf{e}_{(i,p)}^{i}$ stands for the i-th row of $\mathbf{E}_{(i,p)}$ and $e_{i,i}$ stands for the (i,i)-th entry of $\mathbf{E}_{(i,p)}$. Consequently, the complexity of $\mathcal{F}(\mathbf{S}_{(i,p)})$ is reduced to $\mathcal{O}(k+c)$.

We name the proposed algorithm as CD-LSR and summarize the overall procedure of CD-LSR in Algorithm 1.

Algorithm Analysis

The core steps of CD-LSR in each iteration are calculating $\mathbf{V}$, $\mathbf{U}$, and $\mathbf{S}$. Note that the discrete property of $\mathbf{S}$ helps us avoid the matrix multiplication by performing column(row) selection. In this way, the calculations of $\mathbf{V}$, $\mathbf{U}$, and $\mathbf{S}$ take the computational complexity of $\mathcal{O}(k^3+k^2c)$, $\mathcal{O}(k^2c)$ and $\mathcal{O}(k^2d+kdc)$, respectively. Ultimately, the computational complexity of CD-LSR is $\mathcal{O}(tk^3+tk^2c+tk^2d+tkdc)$, where t denotes the iteration number.

Note that CD-LSR is a general optimization framework that can be utilized to solve arbitrary $l_{2,0}$-norm constrained problems. The core of CD-LSR is to decompose $\mathbf{W}$ into the continuous matrix $\mathbf{V}$ and the discrete matrix $\mathbf{S}$ and solve the $l_{2,0}$-norm constraint by optimizing $\mathbf{S}$ with CD method. Moreover, the proposed algorithm does not involve any hyperparameters during the optimization process, which is more efficient than other $l_{2,0}$-norm optimization algorithms.

Experiment

In this study, we design four experiments to verify the effectiveness of CD-LSR from the following aspects: (1) performance on feature selection, (2) solution accuracy, (3) stability, (4) convergence speed, and (5) computational time.

Dataset	Type	# Samp.	# Dim.	# Class
SRBCT	Bioinformatics	83	2308	4
USPS	Handwritten Digit	9298	256	10
UMIST	Handwritten Digit	575	644	20
JAFFE	Human Face	213	676	10
COIL20	Object Image	1440	1024	20
Isolet	Sound	1560	617	2

Table 1: Dataset descriptions.

Dataset

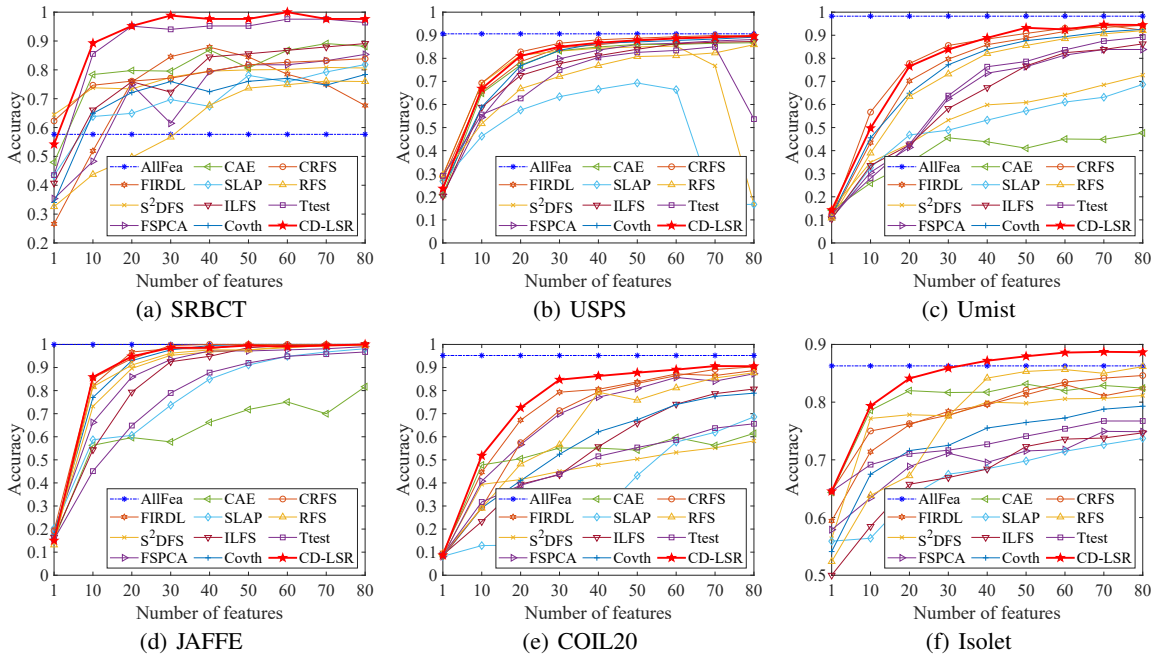
As shown in Table 1, we use six datasets to validate the performance of our method, including one bioinformatics dataset (i.e., SRBCT (Khan et al. 2001)), two handwritten digit datasets (i.e., USPS (Hull 1994) and UMIST (Hou et al. 2014)), one human face datasets (i.e., JAFFE (Lyons, Budynek, and Akamatsu 1999)), one object image dataset (i.e., COIL20 (Nene et al. 1996)), and one sound dataset (i.e., Isolet (Fanty and Cole 1990)). All features are standardized to zero mean and normalized by the standard deviation.

Particularly, SRBCT is a high-dimensional and small-sample-size dataset and USPS is a large-scale dataset, which verify the performance of CD-LSR on high-dimensional features and the scale-up ability on large data, respectively.

Performance on Feature Felection

In this experiment, we apply CD-LSR to feature selection and compare its performance with following methods:

- **Feature Importance Ranking for Deep Learning** [FIRDL] (Wojtas and Chen 2020) devises a dual-net architecture consisting of the "Operator" and the "Selector" to simultaneously identify and rank the features in the optimal feature subset. We refer to the description in the original paper and set network parameters based on the feature dimension of the dataset.
- **Subspace Sparsity Discriminant Feature Selection** [S^2DFS] (Wang et al. 2020) leverages the trace ratio LDA model and selects discriminative features via the structured sparse subspace constraint. We set the dimension of subspace m as the number of selected features.
- **Feature-sparsity constrained PCA** [FSPCA] (Tian et al. 2020) simultaneously performs feature selection and PCA, and directly estimates the row sparsity constrained leading m eigenvectors.
- **Concrete AutoEncoder** [CAE] (Abid, Balin, and Zou 2019) is an end-to-end differentiable deep feature selection method that uses a concrete selector layer as the encoder and a standard neural network as the decoder. We set the parameters as suggested in the original work.
- **Supervised Feature Selection with Local Adaptive Projection** [SLAP] (Chen et al. 2018) is a supervised feature selection method that integrates the similarity learning into the feature selection process. The projected dimension m is set as the number of classes, the neighbor number k is empirically set as 5, and the regularization parameter γ is searched in $\{10^{-3}, 10^{-2}, \ldots 10^{3}\}$.



(a) SRBCT (b) USPS (c) Umist

(d) JAFFE (e) COIL20 (f) Isolet

Figure 3: Classification results on different datasets in terms of different number of selected features.

- **Infinite Latent Feature Selection** [ILFS] (Roffo et al. 2017) is a probabilistic feature selection algorithm that performs the ranking step by considering all the possible subsets of features bypassing the combinatorial problem.
- **Covariance Thresholding** [Covth] (Krauthgamer, Nadler, and Vilenchik 2015) thresholds the data covariance matrix and selects features based on its leading eigenvector. We set the threshold t as $5/\sqrt{n}$.
- **Correntropy Induced Robust Feature Selection** [CRFS] (He et al. 2012) proposes to address the feature selection problem based on the $l_{2,1}$-norm regularized correntropy term. We search the regularization parameter λ in $\{10^{-6}, 10^{-4}, 10^{-2}, \ldots, 10^{4}, 10^{6}\}$.
- **Robust Feature Selection** [RFS] (Nie et al. 2010) is a sparse based method addressing the feature selection problem by solving a joint $l_{2,1}$-norm problem. We search the regularization parameter λ in $\{10^{-6}, 10^{-4}, 10^{-2}, \ldots, 10^{4}, 10^{6}\}$.
- **Ttest** is a commonly used feature selection method introduced by William Sealy Gosset in 1908.

We employ all features (**AllFea**) as baseline and use the linear classifier for classification. We select *Accuracy* as the evaluation metric and the range of the selected feature number is from 1 to 80. For each dataset, we employ the five-fold cross-validation method for parameter tuning and performance evaluation. Since $l_{2,0}$-norm is non-convex, the solution of the proposed method depends on the initialization. Similar to (Pang et al. 2019), we run each method 40 times and report the maximal accuracy as the final performance.

The results are shown in Figure 3, from which we can find that our method ranks top two in all datasets, indicating that our method is able to select more informative features than other competing methods.

Solution Accuracy Evaluation

Next, we compare the solution accuracy of CD-LSR on solving problem (4) with following $l_{2,0}$-norm-related methods:

- **0-1 Integer Programmed Feature Selection** [IPFS] (Zhang et al. 2021) first transforms the $l_{2,0}$-norm constraint into an equivalent 0-1 integer constraint, then solves this problem with the augmented Lagrangian method (ALM) by replacing the 0-1 integer constraint with two continuous constraints. According to the original paper, we set the penalty parameters as $\mu_1 = \mu_2 = \mu_3 = \mu^*$, and update μ iteratively with $\mu^* \leftarrow \rho\mu^*$, where ρ is set to 1.05 as suggested in the original paper.
- **Projection Gradient Descent** [PGD] (Pang et al. 2019) iteratively updates the solution with the gradient information and then projects the solution into the feasible region. The step length in PGD is searched in $\{10^{-8}, 10^{-7}, 10^{-6}, \ldots, 10^{1}, 10^{2}\}$.
- **Robust and Pragmatic Feature Selection** [RPFS] (Cai, Nie, and Huang 2013) solves the $l_{2,0}$-norm constraint with the ALM. For RPFS, the quadratic penalty parameter μ and the step parameter ρ are searched in $\{10^{-6}, 10^{-5}, 10^{-4}, \ldots, 10^{-1}, 10^{0}\}$ and $\{1, 1.01, 1.02, \ldots, 1.1\}$, respectively.

We employ five-fold cross-validation to select optimal parameters for PGD and RPFS. After that, all methods are evaluated on the entire dataset with the optimal parameter. For each dataset, we chose the number of features from 1 to 10 to evaluate the solution accuracy. We run all methods

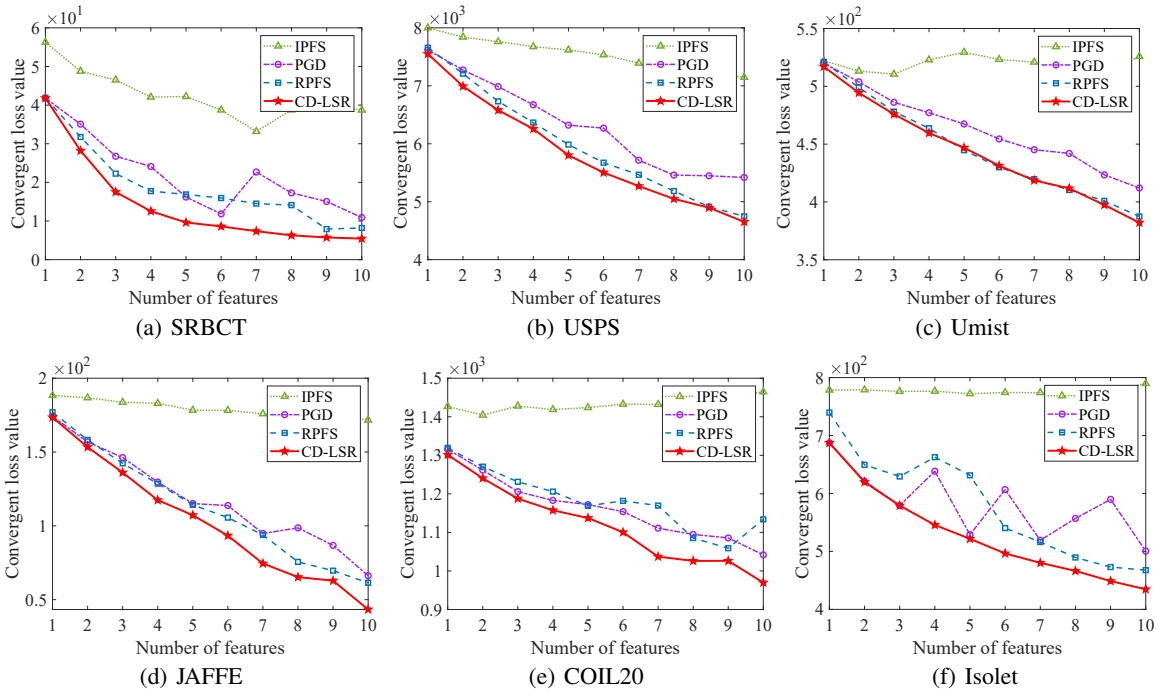


Figure 4: Convergent objective function value in terms of different number of selected features.

40 times respectively. In each trial, the solutions of all methods are initialized with the same random matrix. Similar to (Pang et al. 2019), we select the lowest convergent loss as their final result, as shown in Figure 4. We can see that CD-LSR achieves the best result compared with other methods.

Convergence Analysis

Since the $l_{2,0}$-norm constraint is not convex, we can only find the locally optimal solution. Different initial solutions will lead to different locally optimal solutions, for which, in this part, we evaluate the stability and the convergence speed of CD-LSR. For the economy of space, we select two datasets (i.e., SRBCT and JAFFE) and set feature numbers as $\{1, 5, 10\}$. We keep the same experimental setting as described in the "**Solution Accuracy Evaluation**" section.

We first compare the mean values and standard deviations of the converged objective function with other comparison methods, as shown in Table 2. We can see that CD-LSR remains the best in all cases. Meanwhile, CD-LSR remains relatively low standard deviations in most cases, which verifies the stability of the proposed method.

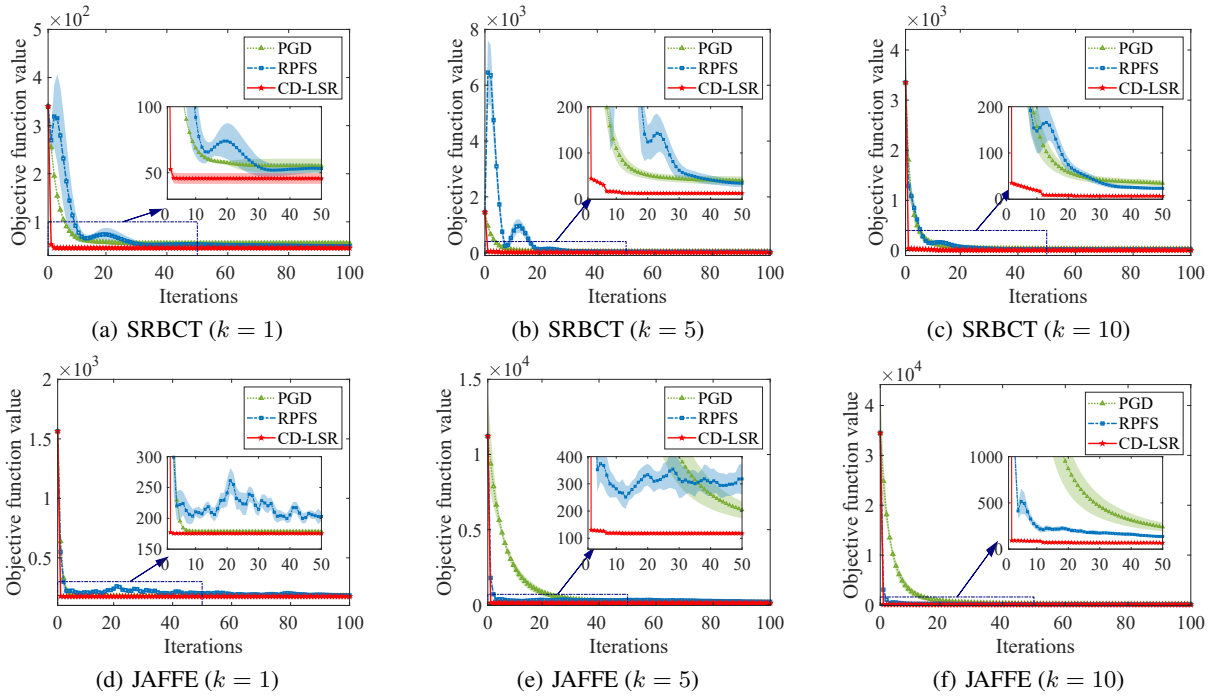
Moreover, we plot the loss curve of each algorithm in Figure 5. Since IPFS transforms the $l_{2,0}$-norm into two continuous constraints, the solution in each iteration does not rigorously satisfy $\|\mathbf{W}\|_{2,0} = k$. From our perspective, it does not make much sense to observe the convergence of IPFS on the original $l_{2,0}$-norm constraint, for which we omit IPFS in Figure 5. We can see that CD-LSR method converges quickly within ten iterations in most cases. The introduction of auxiliary variables in ALM interferes with the searching process for the feasible solution, for which RPFS starts with more drastic fluctuations than other methods.

Dataset	k	IPFS	PGD	RPFS	CD-LSR
SRBCT	1	57.05±0.9	55.30±5.4	49.08±3.5	**45.71±4.1**
	5	51.66±4.2	38.60±9.0	29.81±9.2	**12.91±2.1**
	10	49.61±5.9	31.23±8.4	12.28±2.5	**6.994±1.0**
JAFFE	1	188.4±0.0	178.7±3.2	180.0±2.7	**175.4±1.9**
	5	182.9±1.1	131.9±7.0	124.2±4.9	**118.0±7.1**
	10	189.4±11.9	94.98±8.5	74.07±4.8	**68.36±8.5**

Table 2: Mean and Std of the converged objective function value after 40 random runs. The best and the second-best results are in bold face and underlined, respectively.

Computational Time Comparison

In this part, we compare the computational time of different optimization methods. For a comprehensive comparison, we conduct the experiments on one high-dimensional dataset (SRBCT), one large-scale dataset (USPS), and one regular dataset (JAFFE). We set $k = 5$ and run all methods 40 times, with the average computational time recorded as their final result. We regard the objective function value converges as long as the rate of change of the objective function value is less than 0.01%. For a fair comparison, for all competing algorithms, the optimization process will be terminated if the convergence condition is achieved. We also set the maximum number of iterations to 1000 so as to prevent an excessively long optimization process. All algorithms are test on a Windows machine with 3.4GHz, i7-6700CPU and 32 GB RAM memory. The mean computational time for ten runs of



(a) SRBCT ($k = 1$) (b) SRBCT ($k = 5$) (c) SRBCT ($k = 10$)

(d) JAFFE ($k = 1$) (e) JAFFE ($k = 5$) (f) JAFFE ($k = 10$)

Figure 5: Error bar figures of the objective function values in each iteration, where the curves and the error bars denote the mean loss values and the standard deviations of 40 random runs, respectively.

Method	SRBCT (**high** dim)	JAFFE (regular)	USPS (**large** size)
IPFS	**1.761±0.02**	0.477±0.03	36.80±4.62
PGD	7.539±0.03	0.757±0.03	624.9±16.1
RPFS	58.73±0.56	5.335±0.08	**3.114±0.31**
CD-LSR	5.913±0.07	**0.342±0.01**	3.376±0.66

Table 3: Computational time (CPU time in second) comparison on different datasets. The best and the second-best results are in bold face and underlined, respectively.

different methods are listed in Table 3.

From Table 3, we can see that although other algorithms achieve varying degrees of success on some datasets, our method is the only one that maintains the top-2 speed in all datasets, indicating that our method is scalable to high-dimensional data and large-scale data. Moreover, it is essential to reiterate that our method is parameter-free, which dispenses with the heavy burden of parameter tuning and is more efficient in practical situations.

Discussion

The major contribution of this study is to propose a simple yet efficient optimization framework for the $l_{2,0}$-norm constraint, which possesses a desirable sparsity property for feature selection but is difficult to optimize. The proposed CD-LSR algorithm is applicable to arbitrary $l_{2,0}$-norm constrained problems, therefore facilitating the $l_{2,0}$-norm to be extended to more complex models. Moreover, different from existing hyperparameters-involved optimization algorithms, our algorithm is entirely parameter-free, which will be more efficient in practical situations.

However, our study is subject to two limitations. First, the computational cost of our method heavily depends on the selected feature number k. The coordinate descent may take a long time to optimize if k is getting too large. Nevertheless, the experimental results demonstrate the fast convergence speed of our algorithm, which will alleviate the time cost on large k to some extent. Another limitation is that $l_{2,0}$-norm requires the dedicated number k. Although it is widely accepted to set k in advance (Zhang et al. 2021; Wang et al. 2020), if the users have no preliminary knowledge about the feature space, they cannot presume an appropriate k for feature selection, which will result in a sub-optimal result. Unfortunately, this problem is not the focus of this study and will be discussed and addressed in our future study.

Conclusion

In this paper, we derive a novel and efficient coordinate descent based algorithm to solve the general $l_{2,0}$-norm constrained problem. Extensive experiments verify the stability and the efficiency of the proposed algorithm. The experimental results also validate that our method is able to select more informative features compared with the other state-of-the-art methods for classification.

Acknowledgments

This work was supported by the National Natural Science Foundation of China under Grant 62176212.

References

Abid, A.; Balin, M. F.; and Zou, J. Y. 2019. Concrete Autoencoders for Differentiable Feature Selection and Reconstruction. In *Proc. ICML*, 444–453.

Bertsekas, D. P. 1982. *Constrained optimization and Lagrange multiplier methods*. Academic press.

Cai, X.; Nie, F.; and Huang, H. 2013. Exact Top-k Feature Selection via $\ell_{2,0}$-Norm Constraint. In *Proc. IJCAI*, 1240–1246.

Calamai, P. H.; and Moré, J. J. 1987. Projected gradient methods for linearly constrained problems. *Math. Program.*, 39(1): 93–116.

Chang, X.; Nie, F.; Yang, Y.; and Huang, H. 2014. A Convex Formulation for Semi-Supervised Multi-Label Feature Selection. In *Proc. AAAI*, 1171–1177.

Chen, X.; Yuan, G.; Wang, W.; Nie, F.; Chang, X.; and Huang, J. Z. 2018. Local Adaptive Projection Framework for Feature Selection of Labeled and Unlabeled Data. *IEEE Trans. Neural Networks Learn. Syst.*, 29(12): 6362–6373.

Ding, C. H. Q.; and Peng, H. 2005. Minimum Redundancy Feature Selection from Microarray Gene Expression Data. *J. Bioinform. Comput. Biol.*, 3(2): 185–206.

Fanty, M. A.; and Cole, R. A. 1990. Spoken Letter Recognition. In *Proc. NIPS*, 220–226.

Forman, G.; and Kirshenbaum, E. 2008. Extremely fast text feature extraction for classification and indexing. In *Proc. CIKM*, 1221–1230.

Guyon, I.; Weston, J.; Barnhill, S.; and Vapnik, V. 2002. Gene Selection for Cancer Classification using Support Vector Machines. *Mach. Learn.*, 46(1-3): 389–422.

Hall, M. A.; and Smith, L. A. 1999. Feature Selection for Machine Learning: Comparing a Correlation-Based Filter Approach to the Wrapper. In *Proc. FLAIRS*, 235–239.

He, R.; Tan, T.; Wang, L.; and Zheng, W. 2012. $\ell_{2,1}$ Regularized correntropy for robust feature selection. In *Proc. CVPR*, 2504–2511.

Hou, C.; Nie, F.; Li, X.; Yi, D.; and Wu, Y. 2014. Joint Embedding Learning and Sparse Regression: A Framework for Unsupervised Feature Selection. *IEEE Trans. Cybern.*, 44(6): 793–804.

Hull, J. J. 1994. A Database for Handwritten Text Recognition Research. *IEEE Trans. Pattern Anal. Mach. Intell.*, 16(5): 550–554.

Khan, J.; Wei, J. S.; Ringner, M.; Saal, L. H.; Ladanyi, M.; Westermann, F.; Berthold, F.; Schwab, M.; Antonescu, C. R.; Peterson, C.; et al. 2001. Classification and diagnostic prediction of cancers using gene expression profiling and artificial neural networks. *Nature medicine*, 7(6): 673–679.

Krauthgamer, R.; Nadler, B.; and Vilenchik, D. 2015. Do semidefinite relaxations solve sparse PCA up to the information limit? *The Annals of Statistics*, 43(3): 1300 – 1322.

Li, J.; Cheng, K.; Wang, S.; Morstatter, F.; Trevino, R. P.; Tang, J.; and Liu, H. 2018. Feature Selection. *ACM Comput. Surv.*, 50: 1–45.

Li, X.; Zhang, H.; Zhang, R.; and Nie, F. 2020. Discriminative and Uncorrelated Feature Selection With Constrained Spectral Analysis in Unsupervised Learning. *IEEE Trans. Image Process.*, 29: 2139–2149.

Lyons, M. J.; Budynek, J.; and Akamatsu, S. 1999. Automatic Classification of Single Facial Images. *IEEE Trans. Pattern Anal. Mach. Intell.*, 21(12): 1357–1362.

Nene, S. A.; Nayar, S. K.; Murase, H.; et al. 1996. Columbia object image library (coil-100). *Tech. Rep. CUCS-005-96*.

Nie, F.; Huang, H.; Cai, X.; and Ding, C. 2010. Efficient and Robust Feature Selection via Joint $\ell_{2,1}$-Norms Minimization. In *Proc. NIPS*, 1813–1821.

Pang, T.; Nie, F.; Han, J.; and Li, X. 2019. Efficient Feature Selection via $\ell_{2,0}$-norm Constrained Sparse Regression. *IEEE Trans. Knowl. Data Eng.*, 31(5): 880–893.

Peng, H.; Long, F.; and Ding, C. H. Q. 2005. Feature Selection Based on Mutual Information: Criteria of Max-Dependency, Max-Relevance, and Min-Redundancy. *IEEE Trans. Pattern Anal. Mach. Intell.*, 27(8): 1226–1238.

Roffo, G.; Melzi, S.; Castellani, U.; and Vinciarelli, A. 2017. Infinite Latent Feature Selection: A Probabilistic Latent Graph-Based Ranking Approach. In *Proc. ICCV*, 1398–1406.

Saeys, Y.; Inza, I.; and Larrañaga, P. 2007. A review of feature selection techniques in bioinformatics. *Bioinform.*, 23(19): 2507–2517.

Sun, Z.; Bebis, G.; and Miller, R. 2004. Object detection using feature subset selection. *Pattern Recognit.*, 37(11): 2165–2176.

Tian, L.; Nie, F.; Wang, R.; and Li, X. 2020. Learning Feature Sparse Principal Subspace. In *Proc. NIPS*, 14997–15008.

Wang, Z.; Nie, F.; Tian, L.; Wang, R.; and Li, X. 2020. Discriminative Feature Selection via A Structured Sparse Subspace Learning Module. In *Proc. IJCAI*, 3009–3015.

Wojtas, M.; and Chen, K. 2020. Feature Importance Ranking for Deep Learning. In *Proc. NIPS*, 5105–5114.

Wright, S. J. 2015. Coordinate descent algorithms. *Math. Program.*, 151(1): 3–34.

Zhang, X.; Fan, M.; Wang, D.; Zhou, P.; and Tao, D. 2021. Top-k Feature Selection Framework Using Robust 0-1 Integer Programming. *IEEE Trans. Neural Netw. Learn. Syst.*, 32(7): 3005–3019.