Applied Intelligence (2020) 50:2749–2769
https://doi.org/10.1007/s10489-020-01683-7

Distributed learning for supervised multiview feature selection

Min Men[1] · Ping Zhong[1] · Zhi Wang[2] · Qiang Lin[1]

Published online: 21 March 2020
© Springer Science+Business Media, LLC, part of Springer Nature 2020

Abstract
Multiview feature selection technique is specifically designed to reduce the dimensionality of multiview data and has received much attention. Most proposed multiview supervised feature selection methods suffer from the problem of efficiently handling the large-scale and high-dimensional data. To address this, this paper designs an efficient supervised multiview feature selection method for multiclass problems by combining the distributed optimization method in the Alternating Direction Method of Multipliers (ADMM). Specifically, the distributed strategy is reflected in two aspects. On the one hand, a sample-partition based distributed strategy is adopted, which calculates the loss term of each category individually. On the other hand, a view-partition based distributed strategy is used to explore the consistent and characteristic information of views. We adopt the individual regularization on each view and the common loss term which is obtained by fusing different views to jointly share the label matrix. Benefited from the distributed framework, the model can realize a distributed solution for the transformation matrix and reduce the complexity for multiview feature selection. Extensive experiments have demonstrated that the proposed method achieves a great improvement on training time, and the comparable or better performance compared to several state-of-the-art supervised feature selection algorithms.

Keywords Feature selection · Multiview learning · Distributed strategy · ADMM

1 Introduction

In practical application, the collected data are becoming increasingly high-dimensional and large-scale, which will bring the considerable challenges because the computational complexity is dramatically increased when dealing with the big data. Therefore, establishing the fast and effective algorithm becomes more and more important. One of the efficient solutions is to get a low-dimensional representation of the original data so as to enhance the generalization while reducing the probability of over-fitting. Among the dimensionality reduction techniques, feature selection [11, 39] has been proven to be an efficient method. It removes the irrelevant and redundant features, and finds out the optimal subset from the original dataset without altering the semantics of features. According to the availability of label, the feature selection algorithms can be divided into three categories: supervised feature selection algorithms [35, 43], semi-supervised feature selection algorithms [12, 33], and unsupervised feature selection algorithms [31, 42]. The supervised feature selection uses class labels to select the discriminative features. The unsupervised feature selection selects the relevant features by mining the underlying correlation information among the unlabeled samples. The semi-supervised feature selection is designed to handle the dataset with a few available labels and it can be seen as a compromise between the supervised and unsupervised feature selection.

In recent years, the structured sparsity-inducing feature selection (SSFS) methods have demonstrated the powerful performance for specific tasks, such as classification and clustering. SSFS selects the most discriminative features

✉ Ping Zhong
zping@cau.edu.cn

Min Men
menmin0@cau.edu.cn

Zhi Wang
B20193080657@cau.edu.cn

Qiang Lin
997187002@qq.com

1 College of Science, China Agricultural University, Beijing, 100083, China

2 College of Information and Electrical Engineering, China Agricultural University, Beijing, 100083, China

via joint structured sparsity. Lasso [24] is the representative SSFS method for the binary classification problem. Subsequently, many variants of lasso have been developed, such as fused lasso [25] and group lasso [37]. Compared to lasso which is expressed in vector form, many matrix-based SSFS methods are proposed for solving the multiclass problems. For instance, the $L_{2,1}$-norm regularization [16] was used to select features across all data points with joint sparsity. In [2], the $L_{2,0}$-norm was employed on feature selection and achieved the good performance. In [27], the $G_{2,1}$-norm regularization was proposed to exploit the importance of different views and perform feature selection across views. In [29], the G_1-norm regularization was developed to capture the group structures among the heterogeneous features and enforce sparsity at group level. A comprehensive summary of SSFS methods was exhibited in [8].

With the rapid development of data acquisition devices and feature extraction techniques, an object is often described by multiple modalities (views) or comes from multiple sources. For example, in image processing, an image has multiple heterogeneous features via different descriptors, such as HOG [5], LBP [18], SIFT [14], GIST [19] and so on. Compared with the single-view data exhibited in one homogeneous feature space, the multiview data have multiple representations which can provide much richer information about the data. In real life, the multiview data are often defined in a high-dimensional space, which may result in the curse of dimensionality, so the multiview feature selection is a vital research topic. In this paper, we focus on the supervised multiview SSFS methods. Xiao et al. [32] proposed a two-view feature selection method for cross-sensor iris recognition. Wang et al. [29] used the G_1-norm to integrate heterogeneous features beyond two views. There are many other supervised multiview SSFS algorithms [3, 34, 40, 41] that have also been proposed in recent years. They have successfully selected the representative features through exploring the correlation across views and mining the contributions of different views. However, it is noted that these methods usually combine features from multiple representations into a high-dimensional vector and then take this concatenation as the inputs directly, which is usually confronted with the complicated calculation and high computational cost in the process of matrix operations. Moreover, considering the explosion boost of data size, it is crucial to design a powerful model to learn problems with large scale data.

Alternating Direction Method of Multipliers (ADMM) [1], as an efficient algorithm with the superior convergence properties, is successful applied to the distributed optimization problems, such as consensus and sharing. Enlightened by this, we design an efficient supervised multiview feature selection method through the distributed learning strategy. Specifically, to reduce the computational cost of large scale data, we split the original dataset into multiple subsets according to category, and use the capped l_2-norm based loss function on each subset to deal with outliers efficiently. To fully exploit the characteristics and consistency of views, we minimize the individual view-based regularization and the common loss term which is obtained by combining all views together to share a label matrix. By incorporating the distributed strategy into multiview feature selection, the proposed method can not only capture the relationship among views, but also learn the transformation submatrices in a parallel manner, which can greatly reduce the computational complexity.

In summary, the main contributions of this paper are listed below:

1. A distributed learning strategy is incorporated into the multiview feature selection by adopting the sample-partition and view-partition. The sample-partition is beneficial to improve the computing power of the solution algorithm since the loss term is calculated in blocks.
2. The consistency and characteristics of multiple view representations are protected via the common loss term across views and individual regularization on each view.
3. Compared with the traditional supervised multiview feature selection methods which obtain the entire transformation matrix directly, we realize the distributed solution from different view subspace, which can provide a new choose to multiview feature selection.

The structural framework of this paper is as follows. In Section 2, we review the related work on multiview SSFS methods. In Section 3, we introduce the proposed model and the solution procedure in detail. The experimental setting up and results are exhibited in Section 4. Finally, we make a conclusion in Section 5.

2 Related work

The multiview SSFS feature selection algorithms emphasize the correlation of different views and bring a certain boost on feature selection performance. In this section, we review the recent work on the multiview SSFS methods, including supervised, semi-supervised and unsupervised ones.

2.1 Supervised methods

The supervised multiview SSFS methods use class labels to perform feature selection. Xiao et al. [32] proposed a two-view supervised feature selection method for cross-sensor iris recognition. Then, Wang et al. [29] introduced the G_1-norm regularization for handling datasets beyond two views. In [40], the $G_{2,1}$-norm and $L_{2,1}$-norm joint structured sparsity regularization terms were used to enforce the sparsity between features of different views. Cheng et al. [3] added a hypergraph based regularization to enhance the inherent association of data and combined a low-rank constraint with the $L_{2,1}$-norm to perform feature selection. Wang et al. [30] proposed a supervised multiview feature selection method based on the weighted hinge loss (WHMVFS) that can learn the corresponding weight for each view and implement sparsity across views. Yang et al. [34] proposed a multiview feature learning framework with a discriminative regression and learned the weights of different views adaptively. Zhang et al. [41] proposed a self-weighted supervised feature selection method with the $L_{2,1}$-norm regularization to solve an orthogonal linear discriminant analysis problem. Yang et al. [36] proposed a sparse lasso based mutltiview feature selection method for binary classification, which can capture the contribute of different samples and views. Lin et al. [13] proposed a multiview feature selection method by adopting a common penalty for all views and a structured sparsity-inducing norm for each view. Yang et al. [35] proposed a joint local-and-global multiview feature selection method, in which the local neighbor structure and global label-relevant analysis are combined to select the final features. You et al. [38] proposed Multiview Common Component Discriminant Analysis (MvCCDA) to handle view discrepancy, discriminability and nonlinearity in a joint manner. Specifically, it incorporated supervised information and local geometric information to learn a discriminant common subspace.

2.2 Semi-supervised methods

The semi-supervised multiview SSFS methods use a small proportion of labels to perform feature selection. Sindhwani et al. [22] proposed a co-regularization framework for multiview semi-supervised learning. They adopted the least square loss term over the labeled samples and the laplacian regularization over the unlabeled samples. Li et al. [12] constructed a view-based manifold regularization to design a semi-supervised multiview feature selection method. Xue et al. [33] proposed a semi-supervised multiview feature selection algorithm, and it can simultaneously capture individual information in each view and correlations among multiple views by learning a common component. Nie et al. [17] studied a multiview semi-supervised classification framework by modeling the importance of each view with a parameter-free manner. Considering the different roles of labeled and unlabeled samples, Tao et al. [26] added a score for each sample and designed a semi-supervised multiview classification framework with $L_{2,1}$-norm loss function for each view, and the final object function is formulated as the linear weighted combination of all loss functions. Shi et al. [21] designed a novel semi-supervised feature selection framework by utilizing the multiview Hessian regularization to combine the correlated and complementary information of multiview data.

2.3 Unsupervised methods

The unsupervised multiview SSFS algorithms usually construct the similar graph structure and latent subspace to preserve the intrinsic structure of multiview data. Wang et al. [31] proposed an adaptive multiview feature selection (AMFS) method by constructing the view-based Laplacian graphs to preserve the local geometric structure of multiview data. Hou et al. [9] proposed an unsupervised multiview feature selection method which can learn a common similarity matrix among all views to characterize the structures across different views, and the weights of views were also learned. Wang et al. [28] designed a multiview clustering and feature learning framework with the structured sparsity. Their model explored the unsupervised heterogeneous data fusion and clustering analysis by emphasizing structured sparsity across views. Feng et al. [7] proposed an adaptive unsupervised multiview feature selection (AUMFS) method by adding the view-based Laplacian graphs and $L_{2,1}$-norm regularization. Shao et al. [20] processed the multiview data chunk by chunk and proposed an unsupervised feature selection method via the nonnegative matrix factorization (NMF) and graph regularization. Tang et al. [23] embedded the multiview feature selection into an NMF based clustering framework and added weights for measuring the importance of different views.

It is worth noting that some unsupervised multiview SSFS models combine non-negative matrix factorization and clustering framework to conduct feature selection. With the above strategies, these models can learn the transformation submatrices from different view spaces simultaneously [20, 23]. However, for supervised multiview SSFS methods, since all views are combined together to determine the attribute of an object, they usually need to concatenate features in different representation spaces into a high-dimensional vector to calculate the loss term [3, 34, 40, 41]. Under these circumstances, the transformation matrix has to be calculated in an overall way, which will result in a heavy computation when handling high-dimensional data. So, it is challenging for the traditional supervised SSFS models to effectively deal with large scale data.

3 Proposed model

In this section, we first describe the objective function of the proposed method. Then the derivation procedure of solution is given in detail. The computational complexity of the designed algorithm is analyzed in the third part.

3.1 Objective function

In this paper, matrices and vectors are denoted by the boldface uppercase letters and boldface lowercase letters, respectively. Given a multiview dataset $\mathbf{X} \in \mathbb{R}^{n \times d}$ with C categories and V views, where n is the number of objects and d is the dimension of each object. Denote $\mathbf{X}_i = [\mathbf{x}_i^1; \mathbf{x}_i^2; ...; \mathbf{x}_i^{n_i}] \in \mathbb{R}^{n_i \times d} (i = 1, 2, .., C)$ the ith category, where n_i is the number of the ith category, and $\mathbf{x}_i^q$ represents the qth sample of the ith category. For the ith class, let $\mathbf{X}_{iv} \in \mathbb{R}^{n_i \times d_v} (v = 1, 2, ..., V)$ be the subset from the vth view, where d_v represents the dimension of the vth view. Here, we have $n = \sum_{i=1}^{C} n_i$ and $d = \sum_{v=1}^{V} d_v$. Let $\mathbf{Y} \in \{0, 1\}^{n \times C}$ be the label matrix. The label matrix of the ith category is defined by $\mathbf{Y}_i = [\mathbf{y}_i^1; \mathbf{y}_i^2; ...; \mathbf{y}_i^{n_i}] \in \mathbb{R}^{n_i \times C}$. The $L_{2,1}$-norm in this paper is represented as: $\|\mathbf{W}\|_{2,1} = \sum_{p=1}^{m} \|\mathbf{w}^p\|_2$, where $\mathbf{w}^p$ means the pth row of $\mathbf{W}$. In this paper, we attempt to learn a transformation matrix $\mathbf{W} \in \mathbb{R}^{d \times C}$ to help us select features.

We will introduce our model from the sample level and feature level. Firstly, according to the sample-partition based distributed strategy, we split $\mathbf{X}$ and $\mathbf{Y}$ by categories

$$\mathbf{X} = \begin{bmatrix} \mathbf{X}_1 \\ \mathbf{X}_2 \\ \vdots \\ \mathbf{X}_C \end{bmatrix}, \mathbf{Y} = \begin{bmatrix} \mathbf{Y}_1 \\ \mathbf{Y}_2 \\ \vdots \\ \mathbf{Y}_C \end{bmatrix} \tag{1}$$

We attempt to calculate the loss function of each category separately and then combine them as a whole loss term. Specifically, with the training data $\mathbf{X}_i$ and the label matrix $\mathbf{Y}_i$ for the ith category, the traditional least squares loss function with l_2-norm for the ith block of data can be summarized as follows

$$\mathcal{L}_i(\mathbf{W}) = \sum_{q=1}^{n_i} \|\mathbf{x}_i^q \mathbf{W} + \mathbf{b} - \mathbf{y}_i^q\|_2^2 \tag{2}$$

where $b \in \mathbb{R}^d$ is the bias. We can absorb b into $\mathbf{W}$ when the constant value 1 is added as an additional dimension for each data $\mathbf{x}_i^q$. Then the loss term becomes

$$\mathcal{L}_i(\mathbf{W}) = \sum_{q=1}^{n_i} \|\mathbf{x}_i^q \mathbf{W} - \mathbf{y}_i^q\|_2^2 \tag{3}$$

To improve the robustness of the regression model, we adopt the capped l_2-norm based loss function [10]. However, different from [10] which applies the capped l_2-norm to the whole training data to calculate the loss function, the proposed model calculates the loss term on each category separately

$$\mathcal{L}_i(\mathbf{W}) = \sum_{q=1}^{n_i} \min_{\mathbf{W}} \left(\|\mathbf{x}_i^q \mathbf{W} - \mathbf{y}_i^q\|_2, \varepsilon\right), i = 1, 2, ...C \tag{4}$$

where $\varepsilon > 0$ is the thresholding parameter to identify the latent outliers.

Denote a diagonal matrix $\mathbf{F}_i$, and define the qth diagonal element of $\mathbf{F}_i$ as

$$(\mathbf{F}_i)_{q,q} = \frac{1}{2} \|\mathbf{x}_i^q \mathbf{W} - \mathbf{y}_i^q\|_2^{-1} \cdot I\left(\|\mathbf{x}_i^q \mathbf{W} - \mathbf{y}_i^q\|_2 \leq \varepsilon\right) \tag{5}$$

where $I(\cdot)$ is an indicative function. It is equal to 1 if $\|\mathbf{x}_i^q \mathbf{W} - \mathbf{y}_i^q\|_2 \leq \varepsilon$, and 0 otherwise. Then the (4) can be converted to

$$\mathcal{L}_i(\mathbf{W}) = \sum_{q=1}^{n_i} (\mathbf{F}_i)_{q,q} \|\mathbf{x}_i^q \mathbf{W} - \mathbf{y}_i^q\|_2^2 \tag{6}$$

which is equivalent to

$$\mathcal{L}_i(\mathbf{W}) = tr\left[(\mathbf{X}_i \mathbf{W} - \mathbf{Y}_i)^T \mathbf{F}_i (\mathbf{X}_i \mathbf{W} - \mathbf{Y}_i)\right] \tag{7}$$

Considering the loss of all C categories, we get the following loss term

$$\min_{\mathbf{W}_v, \mathbf{F}_i} \sum_{i=1}^{C} tr\left[(\mathbf{X}_i \mathbf{W} - \mathbf{Y}_i)^T \mathbf{F}_i (\mathbf{X}_i \mathbf{W} - \mathbf{Y}_i)\right] \tag{8}$$

Secondly, a view-partition based distributed strategy is used for emphasizing the characteristics and consistency of views. We adopt the individual regularization on each view and the common loss term which is obtained by fusing different views to jointly share the label matrix. Specifically, we split $\mathbf{X}_i$ and transformation matrix $\mathbf{W}$ into V views

$$\mathbf{X} = \begin{bmatrix} \mathbf{X}_1^1 & \mathbf{X}_1^2 & \cdots \mathbf{X}_1^V \\ \mathbf{X}_2^1 & \mathbf{X}_2^2 & \cdots \mathbf{X}_2^V \\ \vdots & \vdots & \ddots \vdots \\ \mathbf{X}_C^1 & \mathbf{X}_C^2 & \cdots \mathbf{X}_C^V \end{bmatrix}, \mathbf{W} = \begin{bmatrix} \mathbf{W}_1 \\ \mathbf{W}_2 \\ \vdots \\ \mathbf{W}_V \end{bmatrix} \tag{9}$$

Then, (8) can be rewritten as follows

$$\min_{\mathbf{W}_v, \mathbf{F}_i} \sum_{i=1}^{C} tr\left[\left(\sum_{v=1}^{V} \mathbf{X}_{iv} \mathbf{W}_v - \mathbf{Y}_i\right)^T \mathbf{F}_i \left(\sum_{v=1}^{V} \mathbf{X}_{iv} \mathbf{W}_v - \mathbf{Y}_i\right)\right] \tag{10}$$

With the individual $L_{2,1}$-norm regularization based on each view, our model is built as follows:

$$\min_{\mathbf{W}_v, \mathbf{F}_i} \sum_{i=1}^{C} tr\left[\left(\sum_{v=1}^{V} \mathbf{X}_{iv} \mathbf{W}_v - \mathbf{Y}_i\right)^T \mathbf{F}_i \left(\sum_{v=1}^{V} \mathbf{X}_{iv} \mathbf{W}_v - \mathbf{Y}_i\right)\right] + \lambda \sum_{v=1}^{V} \|\mathbf{W}_v\|_{2,1} \tag{11}$$

where $\lambda > 0$ is the trade-off parameter for balancing the weight of loss term and regularization. In this model, we implement distributed feature learning from two aspects. We first adopt the sample-partition scheme to segment the loss term according to each category, providing the feasibility for the distributed solution. Then, the view-partition scheme with individual regularization and common loss term is incorporated into the model, which is conducive to explore the correlations of views.

3.2 Optimization and solution

The optimization problem (11) is convex with respect to one variable when the other variables are fixed. We will alternatively update $\mathbf{W}_v$ and $\mathbf{F}_i$ via the ADMM [1]. By introducing an auxiliary variable $\mathbf{Z}_{iv} = \mathbf{X}_{iv}\mathbf{W}_v$, we can rewrite (11) in ADMM form

$$\begin{aligned}\min_{\mathbf{W}_v,\mathbf{Z}_{iv},\mathbf{F}_i} &\sum_{i=1}^{C} tr\left[\left(\sum_{v=1}^{V}\mathbf{Z}_{iv}-\mathbf{Y}_i\right)^T \mathbf{F}_i\left(\sum_{v=1}^{V}\mathbf{Z}_{iv}-\mathbf{Y}_i\right)\right]\\ &+\lambda\sum_{v=1}^{V}\|\mathbf{W}_v\|_{2,1}\\ s.t.\quad &\mathbf{Z}_{iv}=\mathbf{X}_{iv}\mathbf{W}_v \end{aligned}\tag{12}$$

The augmented lagrangian of (12) is

$$\begin{aligned}&\mathcal{L}(\mathbf{W}_v,\mathbf{Z}_{iv},\mathbf{F}_i,\mathbf{P}_{iv})\\ &=\sum_{i=1}^{C} tr\left[\left(\sum_{v=1}^{V}\mathbf{Z}_{iv}-\mathbf{Y}_i\right)^T \mathbf{F}_i\left(\sum_{v=1}^{V}\mathbf{Z}_{iv}-\mathbf{Y}_i\right)\right]\\ &+\lambda\sum_{v=1}^{V}\|\mathbf{W}_v\|_{2,1}+\sum_{i=1}^{C}\sum_{v=1}^{V} tr\left[\mathbf{P}_{iv}^T(\mathbf{Z}_{iv}-\mathbf{X}_{iv}\mathbf{W}_v)\right]\\ &+\frac{\rho}{2}\sum_{i=1}^{C}\sum_{v=1}^{V}\|\mathbf{Z}_{iv}-\mathbf{X}_{iv}\mathbf{W}_v\|_F^2\end{aligned}\tag{13}$$

where $\mathbf{P}_{iv}$ is the Lagrangian multiplier and ρ is a non-negative constant to penalize the equality constraints. Since

$$\begin{aligned}&\sum_{i=1}^{C}\sum_{v=1}^{V} tr\left[\mathbf{P}_{iv}^T(\mathbf{Z}_{iv}-\mathbf{X}_{iv}\mathbf{W}_v)\right]\\ &+\frac{\rho}{2}\sum_{i=1}^{C}\sum_{v=1}^{V}\|\mathbf{Z}_{iv}-\mathbf{X}_{iv}\mathbf{W}_v\|_F^2\\ &=\frac{\rho}{2}\sum_{i=1}^{C}\sum_{v=1}^{V}\left(\|\mathbf{Z}_{iv}-\mathbf{X}_{iv}\mathbf{W}_v+\frac{1}{\rho}\mathbf{P}_{iv}\|_F^2-\|\frac{1}{\rho}\mathbf{P}_{iv}\|_F^2\right)\end{aligned}\tag{14}$$

The (13) can be rewritten as

$$\begin{aligned}&\mathcal{L}(\mathbf{W}_v,\mathbf{Z}_{iv},\mathbf{F}_i,\mathbf{P}_{iv})\\ &=\sum_{i=1}^{C} tr\left[\left(\sum_{v=1}^{V}\mathbf{Z}_{iv}-\mathbf{Y}_i\right)^T \mathbf{F}_i\left(\sum_{v=1}^{V}\mathbf{Z}_{iv}-\mathbf{Y}_i\right)\right]\\ &+\lambda\sum_{v=1}^{V}\|\mathbf{W}_v\|_{2,1}+\frac{\rho}{2}\sum_{i=1}^{C}\sum_{v=1}^{V}\\ &\left(\|\mathbf{Z}_{iv}-\mathbf{X}_{iv}\mathbf{W}_v+\frac{1}{\rho}\mathbf{P}_{iv}\|_F^2-\|\frac{1}{\rho}\mathbf{P}_{iv}\|_F^2\right)\end{aligned}\tag{15}$$

According to the distributed optimization learning via ADMM [1], the entire transformation matrix W can be obtained by learning V transformation submatrices $\mathbf{W}_v$. For the vth subproblem, ADMM consists of the iterations

$$\begin{aligned}\mathbf{W}_v^{(k+1)} := \arg\min_{\mathbf{W}_v}&\left(\lambda\|\mathbf{W}_v\|_{2,1}+\frac{\rho}{2}\sum_{i=1}^{C}\|\mathbf{Z}_{iv}^{(k)}\right.\\ &\left.-\mathbf{X}_{iv}\mathbf{W}_v+\frac{1}{\rho}\mathbf{P}_{iv}^{(k)}\|_F^2\right)\end{aligned}\tag{16}$$

$$\begin{aligned}\mathbf{Z}_{iv}^{(k+1)} := \arg\min_{\mathbf{Z}_{iv}}&\left(tr\left[\left(\sum_{v=1}^{V}\mathbf{Z}_{iv}-\mathbf{Y}_i\right)^T\right.\right.\\ &\left.\times\mathbf{F}_i^{(k)}\left(\sum_{v=1}^{V}\mathbf{Z}_{iv}-\mathbf{Y}_i\right)\right]\\ &+\frac{\rho}{2}\sum_{v=1}^{V}\|\mathbf{Z}_{iv}-\mathbf{X}_{iv}\mathbf{W}_v^{(k+1)}\\ &\left.+\frac{1}{\rho}\mathbf{P}_{iv}^{(k)}\|_F^2\right)\end{aligned}\tag{17}$$

$$\begin{aligned}(\mathbf{F}_i)_{q,q}^{(k+1)} &= \frac{1}{2}\left\|\left(\sum_{v=1}^{V}\mathbf{Z}_{iv}^{(k+1)}-\mathbf{Y}_i\right)_q\right\|_2^{-1}\\ &\cdot I\left(\left\|\left(\sum_{v=1}^{V}\mathbf{Z}_{iv}^{(k+1)}-\mathbf{Y}_i\right)_q\right\|_2\le\varepsilon\right)\end{aligned}\tag{18}$$

$$\mathbf{P}_{iv}^{(k+1)} = \mathbf{P}_{iv}^{(k)}+\rho\left(\mathbf{Z}_{iv}^{(k+1)}-\mathbf{X}_{iv}\mathbf{W}_v^{(k+1)}\right)\tag{19}$$

where $\mathbf{W}_v^{(k)},\mathbf{Z}_{iv}^{(k)},\mathbf{F}_i^{(k)},\mathbf{P}_{iv}^{(k)}$ are the values of the kth iteration, $\left\|\left(\sum_{v=1}^{V}\mathbf{Z}_{iv}^{(k+1)}-\mathbf{Y}_i\right)_q\right\|_2$ in (18) means the loss of the qth samples in ith class. The optimization problems (16)-(19) can be inverted into a concise form according to the following theorem.

Theorem 1 *Equations* (16)-(19) *is equivalent to* (20)-(23).

$$\begin{aligned}\mathbf{W}_v^{(k+1)} := \arg\min_{\mathbf{W}_v}&\left(\lambda\|\mathbf{W}_v\|_{2,1}+\frac{\rho}{2}\sum_{i=1}^{C}\|\mathbf{X}_{iv}\mathbf{W}_v\right.\\ &-\left(\mathbf{X}_{iv}\mathbf{W}_v^{(k)}+\overline{\mathbf{Z}_i}^{(k)}-\overline{\mathbf{X}_i\mathbf{W}}^{(k)}\right.\\ &\left.\left.+\frac{1}{\rho}\overline{\mathbf{P}_i}^{(k)}\right)\|_F^2\right)\end{aligned}\tag{20}$$

$$\begin{aligned}\overline{\mathbf{Z}_i}^{(k+1)} := \arg\min_{\overline{\mathbf{Z}_i}}&\left(tr\left[\left(V\overline{\mathbf{Z}_i}-\mathbf{Y}_i\right)^T\mathbf{F}_i^{(k)}\left(V\overline{\mathbf{Z}_i}-\mathbf{Y}_i\right)\right]\right.\\ &\left.+\frac{\rho V}{2}\|\overline{\mathbf{Z}_i}-\overline{\mathbf{X}_i\mathbf{W}}^{(k+1)}+\frac{1}{\rho}\overline{\mathbf{P}_i}^{(k)}\|_F^2\right)\end{aligned}\tag{21}$$

$$\begin{aligned}(\mathbf{F}_i)_{q,q}^{(k+1)} &= \frac{1}{2}\|(V\overline{\mathbf{Z}_i}^{(k+1)}-\mathbf{Y}_i)_q\|_2^{-1}\\ &\cdot I\left(\|(V\overline{\mathbf{Z}_i}^{k+1}-\mathbf{Y}_i)_q\|_2\le\varepsilon\right)\end{aligned}\tag{22}$$

$$\overline{\mathbf{P}_i}^{(k+1)} = \overline{\mathbf{P}_i}^{(k)} + \rho \left(\overline{\mathbf{Z}_i}^{(k+1)} - \overline{\mathbf{X}_i \mathbf{W}}^{(k+1)} \right) \tag{23}$$

where $\overline{\mathbf{Z}_i} = \frac{1}{V} \sum_{v=1}^{V} \mathbf{Z}_{iv}$, $\overline{\mathbf{X}_i \mathbf{W}} = \frac{1}{V} \sum_{v=1}^{V} \mathbf{X}_{iv} \mathbf{W}_v$, *and* $\overline{\mathbf{P}_i} = \frac{1}{V} \sum_{v=1}^{V} \mathbf{P}_{iv}$.

Proof See the Appendix. □

It can be seen from the simplified problems that we only need to calculate a matrix $\overline{\mathbf{Z}_i}$ in the (21) instead of calculating $\mathbf{Z}_{iv} (v = 1, ..., V)$ in (17) for fixed i. The same strategy for computing the matrix $\overline{\mathbf{P}_i}$ rather than $\mathbf{P}_{iv}$ is used in (23).

According to the (20)-(23), the solution of $\mathbf{W}_v$, $\overline{\mathbf{Z}_i}$, $\mathbf{F}_i$ and $\overline{\mathbf{P}_i}$ are presented as follows.

1) Updating $\mathbf{W}_v$, where $v = 1, 2, ..., V$

To update $\mathbf{W}_v$, we fix all other variables. By setting the derivative of (20) w.r.t. $\mathbf{W}_v$ to zero, we have

$$2\lambda \mathbf{D}_v \mathbf{W}_v + \rho \sum_{i=1}^{C} \mathbf{X}_{iv}^T \left(\mathbf{X}_{iv} \mathbf{W}_v - \left(\mathbf{X}_{iv} {\mathbf{W}_v}^{(k)} + \overline{\mathbf{Z}_i}^{(k)} - \overline{\mathbf{X}_i \mathbf{W}}^{(k)} + \frac{1}{\rho} \overline{\mathbf{P}_i}^{(k)} \right) \right) = 0 \tag{24}$$

where $\mathbf{D}_v$ is a diagonal matrix with the hth diagonal element $(\mathbf{D}_v)_{h,h} = \frac{1}{2\|\mathbf{w}_v^h\|_2}$, where $\mathbf{w}_v^h$ means the h row of $\mathbf{W}_v$. Then, we have

$$ {\mathbf{W}_v}^{(k+1)} = \rho \left(2\lambda \mathbf{D}_v + \rho \sum_{i=1}^{C} \mathbf{X}_{iv}^T \mathbf{X}_{iv} \right)^{-1} \times \left(\sum_{i=1}^{C} \mathbf{X}_{iv}^T \left(\mathbf{X}_{iv} {\mathbf{W}_v}^{(k)} + \overline{\mathbf{Z}_i}^{(k)} - \overline{\mathbf{X}_i \mathbf{W}}^{(k)} + \frac{1}{\rho} \overline{\mathbf{P}_i}^{(k)} \right) \right) \tag{25}$$

2) Updating $\overline{\mathbf{Z}_i}$, where $i = 1, 2, ..., C$

By setting the derivative of (21) w.r.t. $\overline{\mathbf{Z}_i}$ to zero with current ${\mathbf{W}_v}^{(k+1)}$, ${\mathbf{F}_i}^{(k)}$, $\overline{\mathbf{P}_i}^{(k)}$, we have

$$2V{\mathbf{F}_i}^{(k)} \left(V\overline{\mathbf{Z}_i} - \mathbf{Y}_i \right) + \rho V \left(\overline{\mathbf{Z}_i} - \overline{\mathbf{X}_i \mathbf{W}}^{(k+1)} + \frac{1}{\rho} \overline{\mathbf{P}_i}^{(k)} \right) = 0 \tag{26}$$

Then, we have

$$\overline{\mathbf{Z}_i}^{(k+1)} = \left(2V{\mathbf{F}_i}^{(k)} + \rho \mathbf{I}_i \right)^{-1} \times \left(2{\mathbf{F}_i}^{(k)} \mathbf{Y}_i + \rho \overline{\mathbf{X}_i \mathbf{W}}^{(k+1)} - \overline{\mathbf{P}_i}^{(k)} \right) \tag{27}$$

where $\mathbf{I}_i \in \mathbb{R}^{n_i \times n_i}$ is an identity matrix.

3) Updating $\mathbf{F}_i$ and $\overline{\mathbf{P}_i}$, where $i = 1, 2, ..., C$

With ${\mathbf{W}_v}^{(k+1)}$ and $\overline{\mathbf{Z}_i}^{(k+1)}$, we calculate ${\mathbf{F}_i}^{(k+1)}$ and $\overline{\mathbf{P}_i}^{(k+1)}$ according to (22) and (23), respectively.

In summary, the complete procedure of our algorithm is shown in Algorithm 1. In each iteration, the transformation submatrices $\mathbf{W}_v$ $(v = 1, ..., V)$ are calculated in parallel with the current $\overline{\mathbf{Z}_i}$ and $\overline{\mathbf{P}_i}$ $(i = 1, ...C)$. We repeat this iteration procedure until it satisfies the convergence condition.

Algorithm 1 Proposed method via ADMM.

```
Input: Training data with $V$ views $\{\mathbf{X}_v \in \mathbb{R}^{n \times d_v}\}_{v=1}^{V}$,
       label matrices $\{\mathbf{Y}_i\}_{i=1}^{C}$, parameter $\lambda$ and $\varepsilon$
Output: $\{\mathbf{W}_v \in \mathbb{R}^{d_v \times C}\}_{v=1}^{V}$
1  Initialize: $\mathbf{W}_v^0 = 0$, $\overline{\mathbf{X}_i \mathbf{W}}^0 = 0$, $\overline{\mathbf{Z}_i}^0 = 0$, $\overline{\mathbf{P}_i}^0 = 0$,
   $\sigma = 10^{-3}$, $\rho = 0.5$, $\max_\rho = 10^3$
2  Initialize $\mathbf{F}_i^0$ as an $n_i \times n_i$ identity matrix.
3  while not converged do
4      Update ${\mathbf{W}_v}^{(k+1)}$ by (25): ${\mathbf{W}_v}^{(k+1)} =$
       $\rho \left( 2\lambda \mathbf{D}_v + \rho \sum_{i=1}^{C} \mathbf{X}_{iv}^T \mathbf{X}_{iv} \right)^{-1} \left( \sum_{i=1}^{C} \mathbf{X}_{iv}^T {\mathbf{Q}_{iv}}^{(k)} \right)$,
       where $\mathbf{Q}_{iv}^{(k)} = \mathbf{X}_{iv} {\mathbf{W}_v}^{(k)} + \overline{\mathbf{Z}_i}^{(k)} - \overline{\mathbf{X}_i \mathbf{W}}^{(k)} + \frac{1}{\rho} \overline{\mathbf{P}_i}^{(k)}$;
5      Update $\overline{\mathbf{X}_i \mathbf{W}}^{(k+1)}$ by
       $\overline{\mathbf{X}_i \mathbf{W}}^{(k+1)} = \frac{1}{V} \sum_{v=1}^{V} \mathbf{X}_{iv} {\mathbf{W}_v}^{(k+1)}$;
6      Update $\overline{\mathbf{Z}_i}^{(k+1)}$ by (27):
       $\overline{\mathbf{Z}_i}^{(k+1)} = \left( 2V\mathbf{F}_i^{(k)} + \rho \mathbf{I}_i \right)^{-1}$
       $\left( 2{\mathbf{F}_i}^{(k)} \mathbf{Y}_i + \rho \overline{\mathbf{X}_i \mathbf{W}}^{(k+1)} - \overline{\mathbf{P}_i}^{(k)} \right)$;
7      Update ${\mathbf{F}_i}^{(k+1)}$ by (22):
       $(\mathbf{F}_i)_{q,q}^{(k+1)} = \frac{1}{2} \|(V\overline{\mathbf{Z}_i}^{(k+1)} - \mathbf{Y}_i)_q\|_2^{-1} \cdot$
       $I\left( \|(V\overline{\mathbf{Z}_i}^{(k+1)} - \mathbf{Y}_i)_q\|_2 \le \varepsilon \right)$;
8      Update $\overline{\mathbf{P}_i}^{(k+1)}$ by (23):
       $\overline{\mathbf{P}_i}^{(k+1)} = \overline{\mathbf{P}_i}^{(k)} + \rho \left( \overline{\mathbf{Z}_i}^{(k+1)} - \overline{\mathbf{X}_i \mathbf{W}}^{(k+1)} \right)$;
9      Update $\rho$ by: $\rho = \min(2\rho, \max_\rho)$;
10     Determine when to stop the iteration:
       $er = \|\overline{\mathbf{Z}_i}^{(k+1)} - \overline{\mathbf{X}_i \mathbf{W}}^{(k+1)}\|_\infty < \sigma$.
11 end
```

3.3 Complexity analysis

As is shown in Algorithm 1, we need to update four variables in each iteration, i.e. $\mathbf{W}_v$, $\mathbf{F}_i$, $\overline{\mathbf{Z}_i}$, $\overline{\mathbf{P}_i}$. In the process of calculating $\mathbf{W}_v$, the complexity of matrix multiplication is $\mathcal{O}(d_v n(d_v + C))$ and the computational complexity of matrix inversion is $\mathcal{O}(d_v^3)$. The computational complexity of $\overline{\mathbf{X}_i \mathbf{W}}$ is $n_i dC$, and the complexity for solving $\overline{\mathbf{Z}_i}$ is $\mathcal{O}(n_i^3 + n_i^2 C + n_i dC)$. The complexity of $\mathbf{F}_i$-update is $\mathcal{O}(n_i C)$. The complexity of updating $\overline{\mathbf{P}_i}$ is included in the $\overline{\mathbf{X}_i \mathbf{W}}$-update. So, the total complexity of each iteration is $\mathcal{O}\left(\sum_{v=1}^{V} (d_v^3 + d_v^2 nC) + 2dnC + nC + \sum_{i=1}^{C} (n_i^3 + n_i^2 C) \right)$.

Instead of computing the entire matrix inversion with size of $n \times n$ or $d \times d$ in the traditional feature selection methods, the matrix inversion of the proposed algorithm is divided into blocks, with size of $n_i \times n_i$ or $d_v \times d_v$. Similarly, the matrix multiplication is also calculated in blocks, such as $\mathbf{X}_{iv}\mathbf{W}_v$ instead of $\mathbf{XW}$. So, the distributed learning strategy can exceedingly cut down the computational cost on multiview feature selection.

4 Experiments

In this section, a series of experiments are conducted on public multiview datesets to compare the efficiency of the proposed method with one single-view supervised feature selection algorithm and five multiview supervised feature selection algorithms.

4.1 Datasets

Eight datasets are employed to validate the performances of competitors. The detailed descriptions about these datasets are as follows. And Table 1 shows the detailed information about the datasets used in experiments.

- Multiple Features (MF) [6][1]: This dataset consists of 2,000 handwritten numerals ('0'–'9') with 10 classes. These digits are shown with six feature sets: 76-D Fourier coefficients of the character shapes, 216-D profile correlations, 64-D Karhunen-Love coefficients, 240-D pixel averages in 2 x 3 windows, 47-D Zernike moments and 6-D morphological features.
- Internet Advertisements Dataset (Ads)[2]: This dataset contains 3,279 (2,821 nonads and 458 ads) instances with five types of representations for each instance: 457-D features from url terms, 495-D features from origurl terms, 472-D features from ancurl terms, 111-D features from alt terms and 19-D features from caption terms.
- COIL$_{20}$ [15][3]: This dataset contains 1,440 grayscale images for 20 objects. This paper extracts four different features via different feature descriptors, including 512-D GIST, 1764-D HOG, 1239-D LBP and 1000-D SIFT.
- Animals with Attributes[4]: This dataset consists of 30,475 images for 50 animals with six feature representations for each image: 2688-D Color Histogram features, 2000-D Local Self-Similarity features, 252-D Pyramid HOG features, 2000-D SIFT features, 2000-D SIFT features and 2000-D SURF features. In this paper, we select the first 20 classes in the alphabetical order (named Animal$_{20}$) as the training data.
- NUS-WIDE-OBJECT (NWO) [4][5]: This dataset contains 30,000 objects for 31 classes with six types of low-level features, including 64-D color histogram, 144-D color correlogram, 73-D edge direction histogram, 128-D wavelet texture, 225-D block-wise color moments and 500-D bag of words based on SIFT descriptions. In this paper, we select the first 5 classes, 10 classes from NUS-WIDE-OBJECT training dataset in the alphabetical order (named NWO1 and NWO2) and adopt all samples (named NWO3) as the training data.
- NUS-WIDE (NW)[5]: This dataset contains 81 classes with five types of low-level features, including 64-D color histogram, 144-D color correlogram, 73-D edge direction histogram, 128-D wavelet texture and 225-D block-wise color moments. We have removed the unlabeled samples and three categories whose number exceeds 30000. The final NW dataset used in this experiments contains 104542 samples with 79 categories.

4.2 Experimental setup

To justify the superiority of the proposed method, we compare it with six state-of-the-art feature selection algorithms: DSML-FS [40], SSD-FS [41], HLR-FS [3], SCM [10], AWD [34] and SFS [13]. The detailed descriptions about these compared methods are as follows.

1. DSML-FS: It is a supervised multimodal feature selection method which adopts the $L_{2,1}$-norm and $G_{2,1}$-norm based joint structured sparsity regularization. There are two trade-off parameters λ_1 and λ_2 in this model.
2. SSD-FS: It is a supervised multiview self-weighted feature selection method established by introducing the sparsity-inducing regularization into a self-weighted orthogonal linear discriminant analysis. A trade-off parameter λ is set in this model.
3. HLR-FS: It is a supervised multiview feature selection method, and adopts the hypergraph based manifold regularization to enhance the inherent association of all points. There are two trade-off parameters λ_1 and λ_2 in this model.
4. SCM: It is a single-view supervised feature selection method which emphasizes the capped-l_2 norm loss and the $L_{2,p}$-norm regularizer minimization simultaneously. This model has two parameters λ and ε, where λ is the trade-off parameter and ε is the threshold parameter defined in the capped l_2-norm based loss term.

[1]http://archive.ics.uci.edu/ml/datasets/Multiple+Features

[2]https://archive.ics.uci.edu/ml/datasets/internet+advertisements

[3]http://www.cs.columbia.edu/CAVE/software/softlib/COIL-20.php

[4]https://cvml.ist.ac.at/AwA/

[5]https://lms.comp.nus.edu.sg/research/NUS-WIDE.htm

Table 1 Multiple view representations of different datasets

Views	MF	Ads	COIL_{20}	Animal_{20}	NWO1/2/3	NW
1	FOU(76)	URL(457)	GIST(512)	CH(2688)	CH(64)	CH(64)
2	FAC(216)	ORIGURL(495)	HOG(1764)	LSS(2000)	CORR(144)	CORR(144)
3	KAR(64)	ANCURL(472)	LBP(1239)	PHOG(252)	EDH(73)	EDH(73)
4	PIX(240)	ALT(111)	SIFT(1000)	REGIFT(2000)	WT(128)	WT(128)
5	ZER(47)	CAPTION(19)		SIFT(2000)	CM(225)	CM(225)
6	MOR(6)			SURF(2000)	SIFT(500)	
dimension	649	1554	4515	10940	1134	634
classes	10	2	20	20	5/10/31	79
samples	2000	3279	1440	14112	3415/5789/30000	104542

5. AWD: It is a supervised multiview feature selection method established by employing the discriminative regression and adapted-weighting coefficients on each view. A trade-off parameter λ is set in this model.
6. SFS: It is a supervised multiview feature selection method with the sharing category via the ADMM. A trade-off parameter λ is set in this model.

In the multi-view feature selection process, KNN and SVM classifiers are commonly used for the subsequent classification in feature selection. Thus, we adopt the KNN and SVM classifiers in the experiments. For SVM classifier, we apply the Gaussian kernel and linear kernel. The Gaussian kernel value in SVM classifier ranges $\{10^{-3}, ..., 10^{1}\}$. To avoid the experimental deviation and overfitting problem, we embed an inner 5-fold cross validation into the outer 5-fold cross validation for searching the optimal classifier parameters. The whole dataset is first divided into five equal parts. One of them is used as the testing set and the remaining four sets are merged as the training set. Then, an inner 5-fold cross validation is employed on the training set to find the optimal parameters which are then used for the testing process. For all feature selection methods, the trade-off parameter λ, λ_1 and λ_2 are all searched in the range of $\{10^{-3}, 10^{-2}, ..., 10^{2}, 10^{3}\}$. The experiments are implemented in MATLAB 2016b with Intel(R) Core(TM) i7-8700, CPU 3.20 GHz 3.19 GHz and RAM 8 GB.

4.3 Experimental results and analysis

Firstly, we conduct experiments on five small scale datasets including Multiple Features, Ads, COIL_{20}, NWO1 and NWO2. Both KNN and SVM are adopted to test the performance of compared methods. For each dataset, the detailed classification accuracies[6] in the different top-K feature cases are listed in Tables 2, 3, 4, 5, 6, 7, 8, 9, 10 and 11, respectively. From the results of these tables, we can make some conclusions: (1) The accuracy is not positively related to the number of selected features, i.e. the classification accuracy does not always increase when the number of selected features increases. (2) The proposed method obtains the best results on more than half of the experiments in terms of KNN classifier. On the SVM classifier, our algorithm achieves the best results in almost half of the experiments.

In order to analyze the computational complexity of different algorithms, the training times for getting the transformation matrix $\mathbf{W}$ on five datasets are shown in Fig. 1. We can see that the proposed method and SFS exceed the other five algorithms on four datasets (MF, Ads, NWO1 and NWO2). Specifically, compared with DSML-FS and SCM, the proposed method takes less than a quarter of their training time. Since the proposed method includes the optimization of the outlier ratio parameter ε, it is slightly lower than SFS. However, in terms of the classification accuracy, the proposed method outperforms SFS.

Next, to verify the superiority of the proposed model when there are a large number of training samples or features, we further conduct experiments on Animal_{20}, NWO3 and NW datasets. Since the SVM classifier takes too long time on large-scale and high-dimensional datasets, we only use KNN classifier in this part. The detailed results including classification accuracy, optimal parameter combination and the training time are shown in Table 12 and Fig. 2. Particularly, the symbol "–" in Table 12 represents an algorithm running for more than 60 hours or exceeding the memory of the computer. For the HLR-FS algorithm, it needs to calculate the singular value decomposition (SVD) and matrix inversion, which requires high computational cost when dealing with the high-dimensional and large-scale datasets. The SSD-FS algorithm also has to compute the SVD based on the feature dimensionality, so it is hard to handle the high-dimensional datasets well. In the solution procedure of SCM algorithm, an $n \times n$ based matrix

[6] Ads is an unbalanced dataset, classification accuracy obtained by SVM may not work. So we use F1 scores as the evaluation criterion on Ads dataset when adopting the SVM classifier.

Table 2 Classification accuracies of different feature selection methods on MF dataset (KNN)

Selected	DSML-FS	HLR-FS	SSD-FS	SCM	AWD	SFS	Ours
Features	(λ_1, λ_2)	(λ_1, λ_2)	λ	(λ, ε)	λ	λ	(λ, ε)
20	0.9765±0.0096	0.9720± 0.0076	0.8500± 0.0112	**0.9800 ± 0.0095**	0.9690 ±0.0110	0.9435±0.0143	0.9750±0.0075
	(0.1, 0.001)	(0.001, 10)	100	(10, 0.1)	0.01	0.1	(1, 0.01)
40	**0.9860 ± 0.0104**	0.9825± 0.0079	0.9090± 0.0102	0.9850± 0.0073	0.9800± 0.0124	0.9785±0.0105	0.9845±0.0082
	(1, 0.001)	(0.01, 10)	100	(1, 0.2)	0.01	1	(0.1, 0)
60	0.9835± 0.0045	0.9845± 0.0065	0.9490± 0.0108	0.9845± 0.0086	0.9795± 0.0145	**0.9870 ± 0.0076**	0.9855± 0.0062
	(1, 0.001)	(0.001, 10)	10	(10, 0.01)	0.01	1	(0.01, 0)
80	0.9840± 0.0113	**0.9885 ± 0.0074**	0.9595 ±0.0168	0.9860± 0.0063	0.9830± 0.0099	0.9875 ± 0.0079	0.9870 ±0.0069
	(0.1, 0.001)	(0.001, 10)	10	(10, 0.01)	0.01	1	(0.01, 0.01)
100	0.9850± 0.0106	**0.9860 ± 0.0072**	0.9775± 0.0105	0.9850 ±0.0090	0.9810± 0.0080	0.9855±0.0054	**0.9860 ± 0.0063**
	(0.001, 1)	(0.001, 0.1)	10	(10, 0.1)	0.01	0.001	(1, 0)
120	0.9850± 0.0092	0.9850± 0.0059	0.9815 ±0.0091	0.9845± 0.0082	0.9860±0.0074	0.9860± 0.0065	**0.9865 ± 0.0038**
	(0.1, 0.001)	(0.01, 10)	10	(1, 0.4)	0.01	0.1	(1, 0.01)
140	0.9870± 0.0078	0.9865± 0.0065	0.9835± 0.0080	0.9855± 0.0069	0.9850± 0.0088	0.9850 ± 0.0085	**0.9875 ± 0.0059**
	(0.001, 1)	(0.001, 0.1)	10	(1, 0.01)	0.1	0.1	(0.1, 0.2)

Table 3 Classification accuracies of different feature selection methods on Ads dataset (KNN)

Selected	DSML-FS	HLR-FS	SSD-FS	SCM	AWD	SFS	Ours
Features	(λ_1, λ_2)	(λ_1, λ_2)	λ	(λ, ε)	λ	λ	(λ, ε)
20	0.8603±0.0008	0.8603±0.0023	0.8600±0.0006	0.8603± 0.0008	0.8600± 0.0006	0.8600 ± 0.0016	**0.8606 ± 0.0017**
	(100, 100)	(0.1, 1)	0.01	(10, 0.01)	10	100	(10, 0.2)
40	0.8606±0.0013	0.8597± 0.0010	0.8600±0.0006	0.8603± 0.0013	0.8603± 0.0018	0.8600 ± 0.0007	**0.8609 ± 0.0013**
	(0.1, 0.001)	(0.1, 1000)	0.1	(10, 0.5)	100	1	(0.01, 0.05)
60	0.8603± 0.0013	0.8600± 0.0006	0.8600±0.0006	0.8603± 0.0013	0.8606± 0.0014	0.8603 ± 0.0017	**0.8609 ± 0.0013**
	(1, 10)	(0.1, 1000)	0.1	(10, 0.5)	1	10	(0.001, 0.05)
80	0.8606± 0.008	**0.8609 ± 0.0013**	0.8600±0.0006	0.8600± 0.0006	0.8594± 0.0007	0.8588 ± 0.0020	0.8606± 0.0008
	(0.001, 0.001)	(0.1, 1)	0.1	(100, 0.01)	1000	0.01	(1000, 0)
100	0.8597± 0.0001	**0.8612 ± 0.0019**	0.8600±0.0006	0.8600±0.0006	0.8594± 0.0029	0.8588 ± 0.0020	0.8609± 0.0013
	(1,)	(0.1, 1)	10	(10, 0.5)	1	10	(0.001, 0.05)
120	0.8597± 0.0001	0.8603± 0.0033	0.8600±0.0006	0.8600± 0.0006	0.8594±0.0017	0.8591 ± 0.0017	**0.8606 ± 0.0008**
	(1, 1)	(0.1, 1)	10	(10, 0.5)	10	10	(100, 0)
140	0.8597± 0.0011	0.8603± 0.0023	0.8600±0.0006	0.8603± 0.0008	0.8594± 0.0017	0.8588 ± 0.0017	**0.8606 ± 0.0023**
	(0.001, 100)	(0.1, 100)	10	(1000, 0.2)	10	10	(0.001, 0.01)

Table 4 Classification accuracies of different feature selection methods on COIL$_{20}$ dataset (KNN)

Selected Features	DSML-FS (λ_1, λ_2)	HLR-FS (λ_1, λ_2)	SSD-FS λ	SCM (λ, ε)	AWD λ	SFS λ	Ours (λ, ε)
20	0.9939± 0.0049	0.9736± 0.0074	0.9875± 0.0077	0.9841± 0.0048	0.8140± 0.232	0.9505± 0.0165	**0.9950 ± 0.0014**
	(0.001, 0.001)	(0.001, 0.001)	10	(0.001, 0.4)	1	0.1	(0.001, 0)
40	0.9951± 0.0019	0.9835± 0.0079	0.9945± 0.0040	0.9945± 0.0030	0.9271± 0.0141	0.9513± 0.0086	**1.0000 ± 0.0000**
	(0.001, 0.001)	(0.001, 0.001)	10	(0.001, 0.4)	10	0.1	(0.001, 0.01)
60	0.9931± 0.0032	0.9923± 0.0058	0.9958± 0.0029	0.9979± 0.0020	0.9570± 0.0050	0.9580 ± 0.0247	**1.0000 ± 0.0000**
	(0.001, 0.001)	(0.001, 0.001)	100	(100, 0.01)	10	0.1	(0.001, 0.01)
80	0.9937± 0.0047	0.9937± 0.0047	0.9959± 0.0044	0.9987± 0.0018	0.9708± 0.0046	0.9577± 0.0089	**1.0000 ± 0.0000**
	(0.01, 0.001)	(0.001, 0.001)	0.001	(1000, 0.1)	10	1	(0.001, 0.01)
100	0.9924± 0.0057	0.9965± 0.0035	0.9950± 0.0060	0.9993 ±0.0016	0.9785± 0.0058	0.9595±0.0091	**1.0000 ± 0.0000**
	(0.01, 0.001)	(0.001, 0.001)	0.1	(10, 0.01)	10	1	(0.001, 0.01)
120	0.9916 ±0.0021	0.9959± 0.0044	0.9972± 0.0030	0.9993±0.0016	0.9806± 0.0030	0.9672±0.0132	**1.0000 ± 0.0000**
	(1, 0.001)	(0.001, 0.001)	10	(100, 0.1)	10	0.1	(0.001, 0.01)
140	0.9909 ± 0.0041	0.9965± 0.0036	0.9986± 0.0032	0.9993± 0.0016	0.9812± 0.0024	0.9708± 0.0118	**1.0000 ± 0.0000**
	(1, 0.1)	(0.001, 0.001)	10	(100, 0.1)	10	0.1	(0.01, 0)

Table 5 Classification accuracies of different feature selection methods on NWO1 dataset (KNN)

Selected Features	DSML-FS (λ_1, λ_2)	HLR-FS (λ_1, λ_2)	SSD-FS λ	SCM (λ, ε)	AWD λ	SFS λ	Ours (λ, ε)
20	0.6176± 0.0293	0.5997± 0.0157	0.5982± 0.0110	0.6085± 0.0170	0.5821± 0.0144	0.5710 ± 0.0015	**0.6293 ± 0.0164**
	(1, 001)	(1000, 1000)	0.01	(10, 0.01)	0.001	10	(0.01, 0)
40	**0.6439 ± 0.0142**	0.6217 ±0.0143	0.6094 ±0.0174	0.6404 ±00196	0.6296± 0.0156	0.5859±0.0185	0.6398±0.0123
	(1, 1)	(0.01, 0.1)	0.001	(10, 0.01)	0.001	10	(0.1, 0.05)
60	0.6551± 0.0153	0.6428± 0.0087	0.6111± 0.0130	**0.6577 ± 0.0244**	0.6281± 0.0205	0.6243 ± 0.0218	0.6480± 0.0126
	(0.001, 0.001)	(0.001, 0.1)	0.001	(1, 0.01)	0.001	1	(0.01, 0)
80	0.6577± 0.0139	0.6542± 0.0127	0.6070± 0.0168	**0.6586 ± 0.0215**	0.6228± 0.0106	0.6524 ± 0.0209	0.6559±0.0114
	(1, 1)	(0.001, 0.1)	0.001	(10, 0.01)	0.001	1	(0.1, 0)
100	**0.6738 ± 0.0179**	0.6553± 0.0110	0.6047± 0.0092	0.6615± 0.0170	0.6363± 0.0150	0.6589 ± 0.0198	0.6656± 0.0205
	(1,1)	(0.001,0.1)	0.001	(1,0.01)	0.001	1	(0.1,0.01)
120	**0.6776 ± 0.0157**	0.6477± 0.0220	0.6199± 0.0100	0.6688± 0.0134	0.6512± 0.0077	0.6612±0.0207	0.6647± 0.0163
	(1, 1)	(0.001, 1)	0.001	(1, 0.01)	0.001	1	(0.01, 0.01)
140	0.6764± 0.0198	0.6539± 0.0156	0.6246± 0.0154	0.6767± 0.0187	0.6451± 0.0109	**0.6779 ± 0.0234**	0.6717± 0.0192
	(1, 1)	(0.001, 0.1)	1	(1, 0.1)	0.001	1	(0.01, 0)

Table 6 Classification accuracies of different feature selection methods on NWO2 dataset (KNN)

Selected	DSML-FS	HLR-FS	SSD-FS	SCM	AWD	SFS	Ours
Features	(λ_1, λ_2)	(λ_1, λ_2)	λ	(λ, ε)	λ	λ	(λ, ε)
20	0.4310± 0.0117	0.4066± 0.0126	0.3709± 0.0106	0.4289± 0.0099	0.3994± 0.0101	0.3724 ± 0.0124	**0.4382 ± 0.0094**
	(1, 0.001)	(0.01, 1)	1	(10, 0.01)	10	10	(0.01, 0.05)
40	**0.4790 ± 0.4916**	0.4522± 0.0094	0.4300± 0.0175	0.4723± 0.0147	0.4507± 0.0150	0.4270 ± 0.0080	0.4669± 0.0144
	(1, 0.001)	(0.001, 1)	100	(1, 0.2)	0.001	0.01	(0.1, 0.05)
60	**0.4916 ± 0.0220**	0.4745± 0.0174	0.4439± 0.0175	0.4842± 0.0164	0.4688± 0.0119	0.4488 ± 0.0141	0.4849±0.0150
	(1, 0.1)	(0.001, 1)	100	(1, 0.01)	0.001	0.1	(0.1, 0.01)
80	**0.4958 ± 0.0184**	0.4821± 0.0196	0.4424± 0.0178	0.4842± 0.0086	0.4716± 0.0180	0.4654±0.0101	0.4883± 0.0118
	(1, 0.001)	(0.001, 1)	1	(1, 0.01)	0.001	0.001	(0.1, 0.05)
100	0.4980± 0.0182	0.4849± 0.0138	0.4424± 0.0081	0.4934± 0.0190	0.4799± 0.0104	0.4731 ± 0.0113	**0.5006 ± 0.0118**
	(1, 0.001)	(0.001, 1)	1	(1, 0.01)	0.001	1	(0.01 ,0.05)
120	0.4980± 0.0221	0.4909± 0.0156	0.4496± 0.0066	0.4904± 0.0153	0.4828± 0.0102	0.4826± 0.0116	**0.5030 ± 0.0223**
	(1, 0.001)	(0.01, 1)	0.001	(1, 0.1)	0.001	0.1	(0.01, 0.05)
140	0.5056±0.0172	0.4870± 0.0092	0.4621± 0.0163	0.4994± 0.0152	0.4849± 0.0092	0.4883 ± 0.0158	**0.5065 ± 0.0152**
	(1, 1)	(0.001, 0.1)	1000	(1, 0.3)	0.001	0.1	(0.01, 0.01)

Table 7 Classification accuracies of different feature selection methods on MF dataset (SVM)

Selected	DSML-FS	HLR-FS	SSD-FS	SCM	AWD	SFS	Ours
Features	(λ_1, λ_2)	(λ_1, λ_2)	λ	(λ, ε)	λ	λ	(λ, ε)
20	**0.9840 ± 0.0651**	0.9690± 0.0911	0.8230± 0.0197	0.9795± 0.0570	0.9735± 0.0088	0.9455± 0.0101	0.9830± 0.0693
	(10,0.001)	(0.01,1)	1	(10,0.01)	0.01	0.1	(1,0.05)
40	**0.9885 ± 0.0602**	0.9850± 0.0883	0.9105± 0.0185	0.9845± 0.0596	0.9825± 0.0061	0.9835± 0.0052	0.9875± 0.0353
	(1,0.001)	(0.01,10)	100	(10,0.4)	0.01	1	(0.1,0.1)
60	**0.9885 ± 0.0518**	0.9865± 0.0762	0.9660± 0.0188	0.9865± 0.0454	0.9855± 0.0062	0.9870 ±0.0080	0.9875± 0.0637
	(1,1)	(0.01,10)	10	(1,0.1)	0.01	1	(0.01.0)
80	0.9880± 0.0778	0.9890± 0.0627	0.9765± 0.0135	0.9865± 0.0894	0.9865± 0.0080	0.9875 ±0.0059	**0.9900 ± 0.0306**
	(0.1,0.1)	(0.01,10)	10	(1,0.01)	0.01	0.001	(0.01,0.05)
100	**0.9885 ± 0.0848**	**0.9885 ± 0.0627**	0.9815± 0.0065	0.9855± 0.0974	0.9865± 0.0072	0.9870 ±0.0078	0.9880± 0.0480
	(1,0.001)	(0.01,10)	10	(1,0.1)	0.01	0.1	(0.01,0)
120	0.9875± 0.0586	0.9870± 0.0758	0.9795± 0.0078	0.9870± 0.0647	0.9855± 0.0089	**0.9880 ± 0.0087**	**0.9880 ± 0.0622**
	(1,0.1)	(0.01,1)	10	(1,0.3)	0.1	0.1	(1,0)
140	09885± 0.6021	0.9870± 0.0647	0.9805± 0.0105	0.9880± 0.0647	0.9860± 0.0078	0.9880± 0.0087	**0.9890 ± 0.0720**
	(0.1,0.001)	(0.01,10)	10	(1,0.4)	0.1	1	(0.1,0)

Table 8 F1 scores of different feature selection methods on Ads dataset (SVM)

Selected Features	DSML-FS (λ_1, λ_2)	HLR-FS (λ_1, λ_2)	SSD-FS λ	SCM (λ, ε)	AWD λ	SFS λ	Ours (λ, ε)
20	0.8647± 0.0270	0.9060±0.0054	0.9022± 0.0044	0.8839±0.0112	0.8407± 0.0302	0.8061± 0.0285	**0.9211 ± 0.0020**
	(1000,0.001)	(0.1,0.1)	100	(0.001,0.01)	0.1	0.1	(1000,0.01)
40	0.8487± 0.8317	0.8587±0.0200	0.8847± 0.0111	0.8353±0.0079	0.8349± 0.0242	0.7871± 0.0160	**0.8951 ± 0.0165**
	(10,1)	(0.1,0.1)	10	(0.001,0.01)	1000	0.001	(1000,0.01)
60	0.8317± 0.0243	0.8424±0.0273	0.8642± 0.0140	0.7783±0.0080	0.8317± 0.0242	0.7496± 0.0512	**0.8837 ± 0.0190**
	(0.001,1000)	(0.1,0.1)	10	(10,0.2)	1000	0.1	(1000,0.01)
80	0.8309± 0.0243	0.8094±0.0329	0.8532± 0.0124	0.7229±0.0283	0.8308 ±0.0244	0.7369 ± 0.0172	**0.8761 ± 0.0214**
	(10,100)	(1,10)	10	(10,0.3)	100	100	(1000,0.01)
100	0.8307± 0.0245	0.7947±0.0371	0.8177± 0.0031	0.7051±0.0110	0.8307± 0.0243	0.7429± 0.0526	**0.8674 ± 0.0257**
	(0.001,1000)	(1,10)	100	(10,0.3)	1000	0.1	(1000,0.01)
120	0.8307± 0.0243	0.7890±0.0363	0.7833± 0.0173	0.7017±0.0099	0.8302± 0.0248	0.7341± 0.0359	**0.8674 ± 0.0257**
	(0.001,1000)	(0.1,0.1)	10	(10,0.3)	1000	0.1	(1000,0.01)
140	0.8303± 0.0249	0.7818±0.0363	0.7874± 0.0074	0.6808± 0.0064	0.8299±0.0254	0.7304± 0.0283	**0.8665 ± 0.0267**
	(100,1000)	(0.1,0.1)	100	(10,0.3)	1000	0.1	(1000,0.01)

Table 9 Classification accuracies of different feature selection methods on COIL_{20} dataset (SVM)

Selected Features	DSML-FS (λ_1, λ_2)	HLR-FS (λ_1, λ_2)	SSD-FS λ	SCM (λ, ε)	AWD λ	SFS λ	Ours (λ, ε)
20	0.8715± 0.0141	0.7953± 0.0191	0.8223± 0.0074	0.8813± 0.0167	0.5950 ±0.0121	0.8273± 0.0137	**0.8797 ± 0.0168**
	(1,1)	(0.001,0.001)	0.01	(1,0.01)	1	0.1	(0.01,0)
40	**0.9742 ± 0.0047**	0.8831± 0.0146	0.9230± 0.0197	0.9619± 0.0100	0.8327± 0.0110	0.9184± 0.0154	0.9617± 0.0115
	(1,0.1)	(0.001,0.001)	10	(1,0.01)	10	1	(0.01,0.01)
60	**0.9951 ± 0.0040**	0.9172± 0.0104	0.9690±0.0107	0.9881±0.0065	0.9269± 0.0157	0.9577± 0.0105	0.9812± 0.0072
	(0.1,0.001)	(0.001,0.001)	100	(1,0.01)	10	1	(0.001,0.05)
80	**0.9959 ± 0.0055**	0.9438± 0.0102	0.9876±0.0077	0.9923± 0.0078	0.9473± 0.0078	0.9721± 0.0105	0.9823 ±0.0064
	(0.01,0.001)	(0.001,0.001)	100	(1,0.01)	1	0.1	(0.001,0.01)
100	**0.9972 ± 0.0038**	0.9605± 0.0076	0.9931± 0.0067	0.9945± 0.0029	0.9628± 0.0155	0.9770± 0.0036	0.9959± 0.0014
	(0.01,0.001)	(0.001,0.001)	100	(0.1,0.1)	1	1	(0.01,0)
120	0.9958± 0.0039	0.9694 0.0091	0.9960± 0.0043	0.9951± 0.0019	0.9660± 0.0071	0.9824± 0.0134	**0.9979 ± 0.0019**
	(1,0.001)	(0.001,0.001)	1	(0.1,0.1)	1	0.1	(0.01,0.05)
140	0.9958± 0.0047	0.9730 0.0069	**0.9987 ± 0.0030**	0.9979± 0.0019	0.9730± 0.0047	0.9860± 0.0059	0.9979± 0.0019
	(1,0.001)	(0.001,0.001)	1	(0.001,0.01)	10	0.1	(0.001,0)

Table 10 Classification accuracies of different feature selection methods on NWO1 dataset (SVM)

Selected Features	DSML-FS (λ_1, λ_2)	HLR-FS (λ_1, λ_2)	SSD-FS λ	SCM (λ, ε)	AWD λ	SFS λ	Ours (λ, ε)
20	0.6409±0.00616	0.6125± 0.0055	0.5874± 0.0191	**0.6410 ± 0.0113**	0.5804± 0.0053	0.5599± 0.0106	0.6357± 0.0012
	(1,0.001)	(0.001,0.1)	0.001	(1,0.1)	0.01	10	(0.01,0.01)
40	**0.7024 ± 0.02003**	0.6755± 0.0088	0.6492± 0.0142	0.6943± 0.0141	0.6694± 0.0100	0.6539± 0.0109	0.6866± 0.0013
	(1,0.001)	(0.01,0.1)	1000	(1,0.1)	0.001	0.001	(0.01,0.01)
60	**0.7335 ± 0.0152**	0.7070± 0.1240	0.6744± 0.0154	0.7142± 0.0202	0.6963± 0.0125	0.7019± 0.0147	0.7095± 0.0012
	(1,0.001)	(0.01,0.1)	0.001	(1,0.1)	0.001	1	(0.01,0.05)
80	**0.7461 ± 0.0149**	0.7226± 0.0127	0.6817± 0.0116	0.7303± 0.0101	0.7089± 0.0195	0.7373± 0.0166	0.7247± 0.0017
	(1,0.001)	(0.01,0.1)	1000	(1,0.1)	0.001	1	(0.01,0.01)
100	**0.7587 ± 0.0175**	0.7326± 0.0166	0.6855± 0.0149	0.7432± 0.0182	0.7289± 0.0195	0.7455± 0.0109	0.7420± 0.0158
	(1,0.1)	(0.01,0.1)	0.001	(1,0.1)	0.01	1	(0.01,0.05)
120	**0.7631 ± 0.0123**	0.7449± 0.0151	0.6981± 0.0184	0.7543± 0.0232	0.7353± 0.0140	0.7508± 0.0086	0.7449± 0.0174
	(1,0.001)	(0.01,0.1)	0.001	(1,0.1)	0.001	1	(0.01,0.05)
140	**0.7698 ± 0.0165**	0.7505± 0.0134	0.7113± 0.0159	0.7581± 0.0162	0.7394± 0.0097	0.7531± 0.0130	0.7537± 0.0182
	(1,1)	(0.01,0.1)	0.001	(1,0.1)	0.001	0.001	(0.01,0)

Table 11 Classification accuracies of different feature selection methods on NWO2 dataset (SVM)

Selected Features	DSML-FS (λ_1, λ_2)	HLR-FS (λ_1, λ_2)	SSD-FS λ	SCM (λ, ε)	AWD λ	SFS λ	Ours (λ, ε)
20	**0.3939 ± 0.0073**	0.3548± 0.0042	0.2698± 0.0056	0.3797± 0.0094	0.3470± 0.3685	0.3519± 0.0089	0.3571± 0.0063
	(10,0.001)	(0.01,10)	1000	(10,0.2)	10	10	(10,0)
40	0.4077± 0.0076	0.3812± 0.0068	0.3265± 0.0080	**0.4089 ± 0.0107**	0.3685± 0.0053	0.3793± 0.0040	0.3970± 0.0055
	(10,1)	(0.01,10)	0.001	(10,0.01)	10	10	(10,0.01)
60	0.4168± 0.0064	0.4068± 0.0083	0.3572± 0.0099	**0.4180 ± 0.0067**	0.3828± 0.0075	0.3956± 0.0104	0.4173± 0.4391
	(1,0.1)	(0.001,1)	0.001	(10,0.01)	10	10	(0.1,0.05)
80	0.4301± 0.0053	0.4189± 0.0135	0.3774± 0.0119	0.4232± 0.0061	0.4094± 0.0072	0.4023± 0.0080	**0.4391 ± 0.0063**
	(1,1)	(0.01,1)	0.001	(1,0.5)	0.001	10	(0.1,0.05)
100	0.4484± 0.0098	0.4414± 0.0079	0.4101 0.0141	0.4376± 0.0054	0.4182± 0.0081	0.4262± 0.0042	**0.4500 ± 0.0089**
	(1,0.001)	(0.01,1)	1000	(1,0.4)	0.001	1	(0.1,0.01)
120	0.4572± 0.0066	0.4541± 0.0108	0.4289± 0.0086	0.4543± 0.0072	0.4313± 0.0052	0.4469±0.0042	**0.4583 ± 0.0119**
	(1,0.1)	(0.01,1)	0.001	(1,0.4)	1	1	(0.1,0.01)
140	**0.4692 ± 0.0067**	0.4616± 0.0105	0.4445± 0.0071	0.4595± 0.0072	0.4453± 0.0058	0.4626± 0.0044	0.4676± 0.0080
	(1,0.001)	(0.01,1)	0.001	(1,0.4)	1	1	(0.01,0)

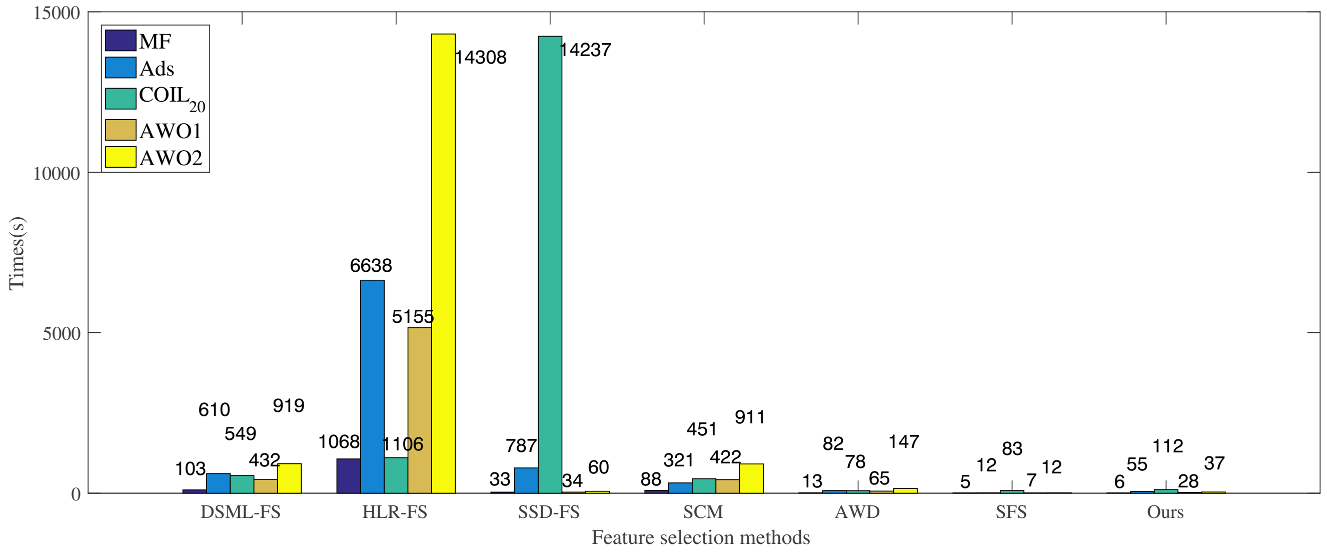

Fig. 1 Comparisons of the training time for getting the transformation matrix **W** by DSML-FS, HLR-FS, SSD-FS, SCM, AWD, SFS and Ours on five datasets

inversion needs to be calculated, which is easy to face memory overflow problems when handling the large-scale datasets. For DSML-FS and AWD algorithms, they need to calculate the matrix multiplication with size of $n \times n$, which requires high computer memory when dealing with large scale dataset.

From the results in Table 12 and Fig. 2, we can get the following conclusions: (1) The proposed method and SFS require less training time. The HLR-FS, SSD-FS and SCM method run too much longer than the other methods on Animal$_{20}$ or NWO3 dataset. This suggests that these three methods have obvious inferiority when solving the high-dimensional or large-scale problems. On NW dataset, except the proposed method and SFS, the other methods can not obtain the classification accuracies because of the memory overflow. (2) The DSML-FS method obtains the best accuracies on Animal$_{20}$ and NWO3 datasets and our method ranks about 2% less than the best ones. However, in terms of time complexity, the training time of our method is 19 times faster than that of DSML-FS method on Animal$_{20}$ dataset and 295 times faster than that of DSML-FS method on NWO3 dataset, which demonstrates that the proposed method brings great superiority on reducing the computational cost. (3) Compared with SFS, the proposed method is weaker in the training time, but the accuracies on these big datasets are generally higher than those of SFS.

4.4 Friedman statistical analysis

Based on the experimental results in Tables 2-11, Friedman statistical hypothesis tests are adopted to determine whether the performances of these algorithms are different. Firstly, based on the Tables 2-6, we calculate the variable

$$\tau_{\chi^2} = \frac{12N}{k(k+1)}\left(\sum_{i=1}^{k} r_i^2 - \frac{k(k+1)^2}{4}\right) = 107.21 \qquad (28)$$

where r_i represents the average rank of the ith method listed in Table 13, $N = 35$ is the number of cases and $k = 7$ is the number of methods. τ_{χ^2} obeys the χ^2 distribution with

Table 12 Classification accuracies of the compared methods on Animal$_{20}$, NWO3 and NW datasets (KNN)

Data Sets	DSML-FS (λ_1, λ_2)	HLR-FS (λ_1, λ_2)	SSD-FS λ	SCM (λ, ε)	AWD λ	SFS λ	Ours (λ, ε)
Animal$_{20}$	**0.2874 ± 0.0118**	–	–	0.2580±0.0120	0.2591± 0.0035	0.2696± 0.0094	0.2662± 0.0045
	(1, 1)	–	–	(1, 0.1)	0.1	0.001	(0.1, 0.01)
NWO3	**0.3602 ± 0.0047**	–	0.3118±0.0045	–	0.3408± 0.0047	0.3392±0.0054	0.3439± 0.0040
	(1, 1)	–	0.001	–	0.1	0.001	(0.01, 0.01)
NW	–	–	–	–	–	0.1671±0.0023	**0.1885 ± 0.0025**
	–	–	–	–	–	10	(10,0.01)

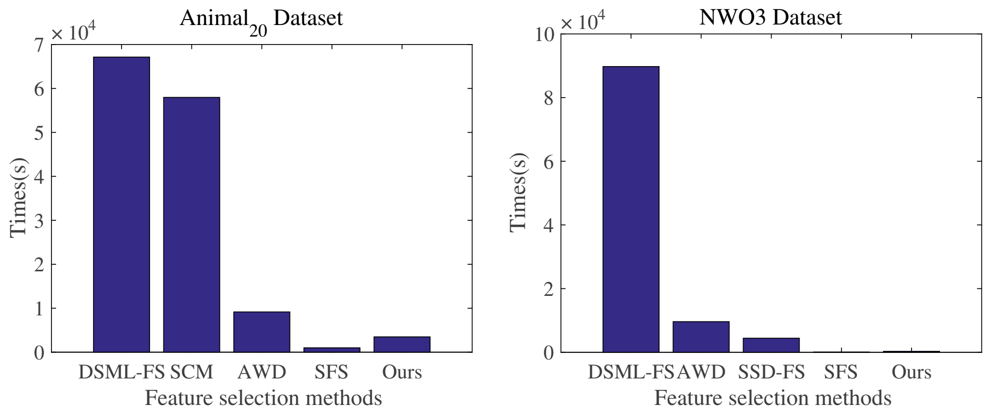

Fig. 2 Comparisons of the training time for getting the transformation matrix **W** by different feature selection methods on NWO3 and Animal$_{20}$ datasets

$k-1$ degrees of freedom. The Friedman statistics variable is then calculated by

$$\tau_F = \frac{(N-1)\tau_{\chi^2}}{N(k-1)-\tau_{\chi^2}} = 50.1 \tag{29}$$

where τ_F obeys the F distribution with $k-1$ and $(k-1)(N-1)$ degrees of freedom. When the significance level $\alpha = 0.05$, the critical value $F_\alpha(6, 204) = 2.14$ and it is less than $\tau_F = 50.1$, which means the performances of these algorithms are significantly different. In order to further distinguish the performance difference among these algorithms, we compute the critical difference (CD) value by adopting Nemenyi test

$$CD = 2.949\sqrt{\frac{k(k+1)}{6N}} = 1.523 \tag{30}$$

Since the differences between the proposed method and the compared methods are as follows

$d(\text{DSML-FS} - \text{Ours}) = 1.143 < 1.523$,
$d(\text{HLR-FS} - \text{Ours}) = 2.143 > 1.523$
$d(\text{SSD-FS} - \text{Ours}) = 3.943 > 1.523$,
$d(\text{SCM} - \text{Ours}) = 1.172 < 1.523$
$d(\text{SFS} - \text{Ours}) = 3.7 > 1.523$,
$d(\text{AWD} - \text{Ours}) = 3.7 > 1.523$

where $d(i-j)$ means the difference between the algorithm i and j. We can then conclude that the proposed method outperforms the HLR-FS, SSD-FS, SFS and AWD algorithms because the differences are bigger than the critical difference. There is no significant difference between DSML-FS, SCM and the proposed method.

Secondly, based on the Tables 7-11, we calculate the variable

$$\tau_{\chi^2} = \frac{12N}{k(k+1)}\left(\sum_{i=1}^{k} r_i^2 - \frac{k(k+1)^2}{4}\right) = 98.963 \tag{31}$$

The Friedman statistics variable is then calculated by

$$\tau_F = \frac{(N-1)\tau_{\chi^2}}{N(k-1)-\tau_{\chi^2}} = 30.303 \tag{32}$$

The critical value $F_{0.05}(6, 204) = 2.14$ is less than $\tau_F = 30.303$, which means the performances of these algorithms are significantly different. With the $CD = 2.949\sqrt{\frac{k(k+1)}{6N}} = 1.523$, the differences between the proposed method and the compared methods are as follows

$d(\text{DSML-F}S - \text{Ours}) = 0.057 < 1.523$,
$d(\text{HLR-FS} - \text{Ours}) = 2.386 > 1.523$
$d(\text{SSD-FS} - \text{Ours}) = 3.357 > 1.523$,
$d(\text{SCM} - \text{Ours}) = 1.514 < 1.523$
$d(\text{SFS} - \text{Ours}) = 3.529 > 1.523$,
$d(\text{AWD} - \text{Ours}) = 2.757 > 1.523$

We can then conclude that the proposed method outperforms the HLR-FS, SSD-FS, SFS and AWD algorithms because the differences are bigger than the critical difference. There is no significant difference between DSML-FS, SCM and the proposed method. The SVM based statistical analysis results are the same as that of KNN.

4.5 Verification experiment analysis

To verify how the capped l_2-norm based loss function effects the performance of experimental results, we further

Table 13 The average rank of different feature selection methods

Methods	DSML-FS	HLR-FS	SSD-FS	SCM	AWD	SFS	Ours
KNN	2.885	3.885	5.685	2.914	5.442	5.442	1.742
SVM	2.014	4.457	5.428	3.585	5.600	4.828	2.071

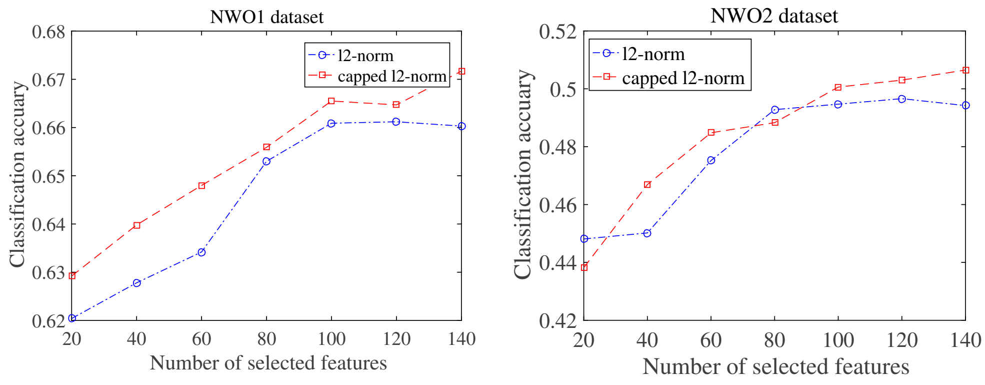

Fig. 3 Classification accuracies of (33) and Ours on NWO1 and NWO2 datasets

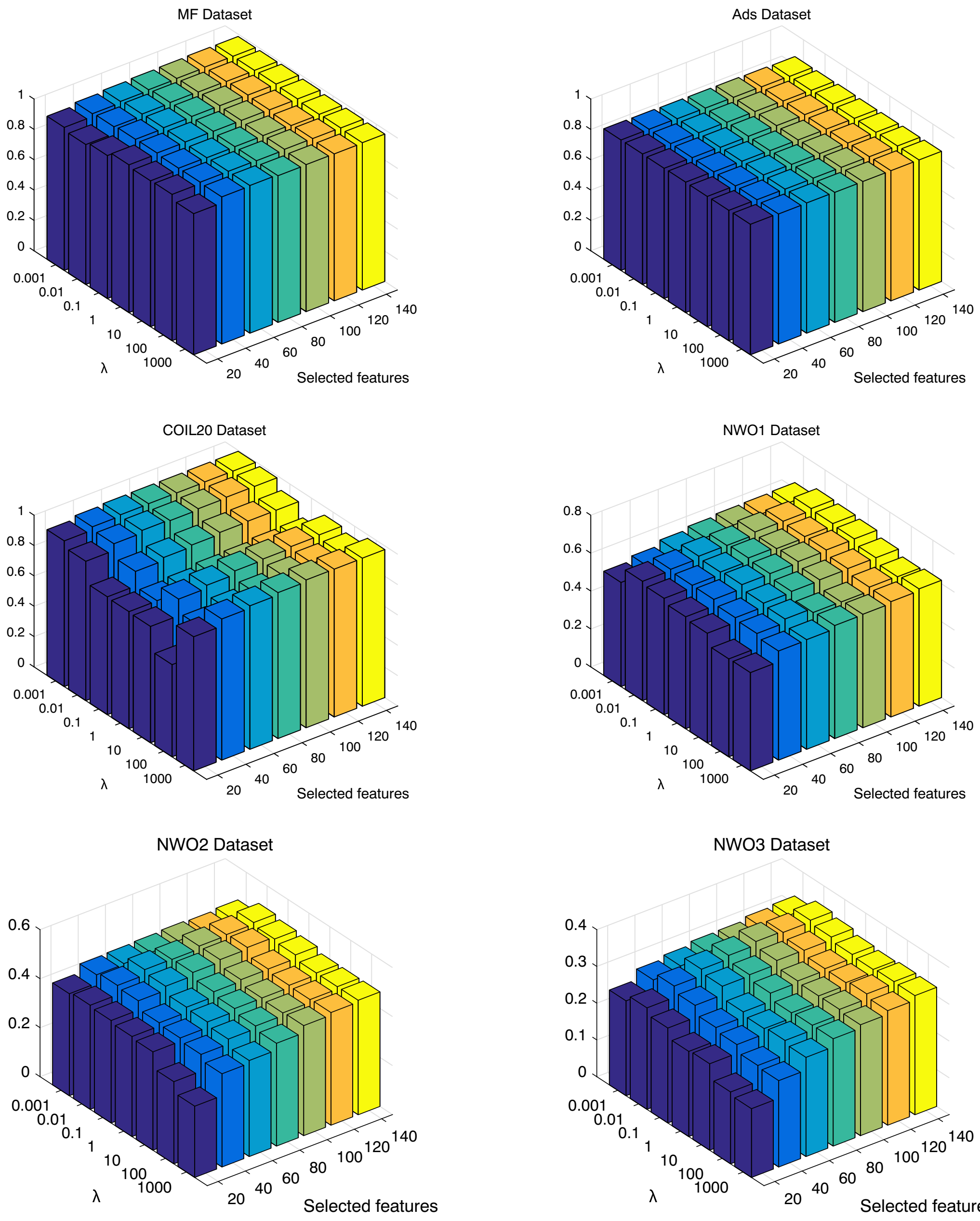

Fig. 4 Classification accuracies of the proposed method with respect to λ under different number of selected features, where λ is tuned from {0.001, 0.01, 0.1, 1, 10, 100, 1000}

add some experiments on NWO1 and NWO2 datasets with KNN classifier to compare the performance between the capped l_2-norm and traditional l_2-norm. By replacing the capped l_2-norm with l_2-norm, the proposed model can be changed as follows.

$$\min_{\mathbf{W}_v} \sum_{i=1}^{C} \sum_{v=1}^{V} \|\mathbf{X}_{iv}\mathbf{W}_v - \mathbf{Y}_i\|_F^2 + \lambda \sum_{v=1}^{V} \|\mathbf{W}_v\|_{2,1} \tag{33}$$

The KNN based classification accuracies of the (33) and ours are shown in Fig. 3. We can see that the classification accuracies of the capped l_2-norm based loss term get 2% higher than traditional l_2-norm on these two datasets, which validates that the proposed model has robustness to noise and is helpful to subsequent classification task.

4.6 Parameter analysis

In this part, the parameter sensitivity of the proposed model is analyzed. We research the influence of λ and the ratio of outliers on experimental results, respectively. The ratio of outliers determines the value of threshold ε. The value of another parameter is fixed when we investigate the effect of one parameter.

Firstly, we vary λ in the range $\{0.001, 0.01, 0.1, 1, 10, 100, 1000\}$ with fixed ratio of outliers. The classification accuracies by KNN with respect to λ on six datasets are shown in Fig. 4. We can see that the classification accuracies are not sensitive to the value of λ on MF and Ads datasets, but they have a certain degree of fluctuation on the other datasets. Particularly, the better results are usually achieved in the range $\{0.001, 0.01, 0.1\}$.

Secondly, considering that there may be potential outliers in the datasets, we set the ratio of outliers to all data points as $\{0, 0.01, 0.05, 0.1, 0.2\}$ and fix the value of λ to exploit how the accuracies change when removing different number of latent outliers. The classification accuracies by KNN with respect to the ratio of outliers on four datasets are shown in Fig. 5. We can observe that the proposed model is less sensitive with respect to the ratio of outliers on MF and Ads datasets but more sensitive to the ratio of outliers on NWO1 and NWO2 datasets, which may suggest that there exists some noises and outliers in these two datasets. From the results on NWO1 and NWO2 datasets, we can see that as the ratio of outliers increases, the accuracies first increase and then decrease in most cases. The reason may be that our model is removing outliers and noises when the ratio gradually rises. But when the ratio reaches a certain value, the accuracies start to decrease because too much data points are removed. In summary, the feature selection performance obtains a certain improvement when removing a proportion of potential outliers. The better results are usually obtained in the proportion of $\{0.01, 0.05\}$.

4.7 Convergence analysis

To verify the convergence performance of Algorithm 1, we conduct some experiments on eight datasets. For each

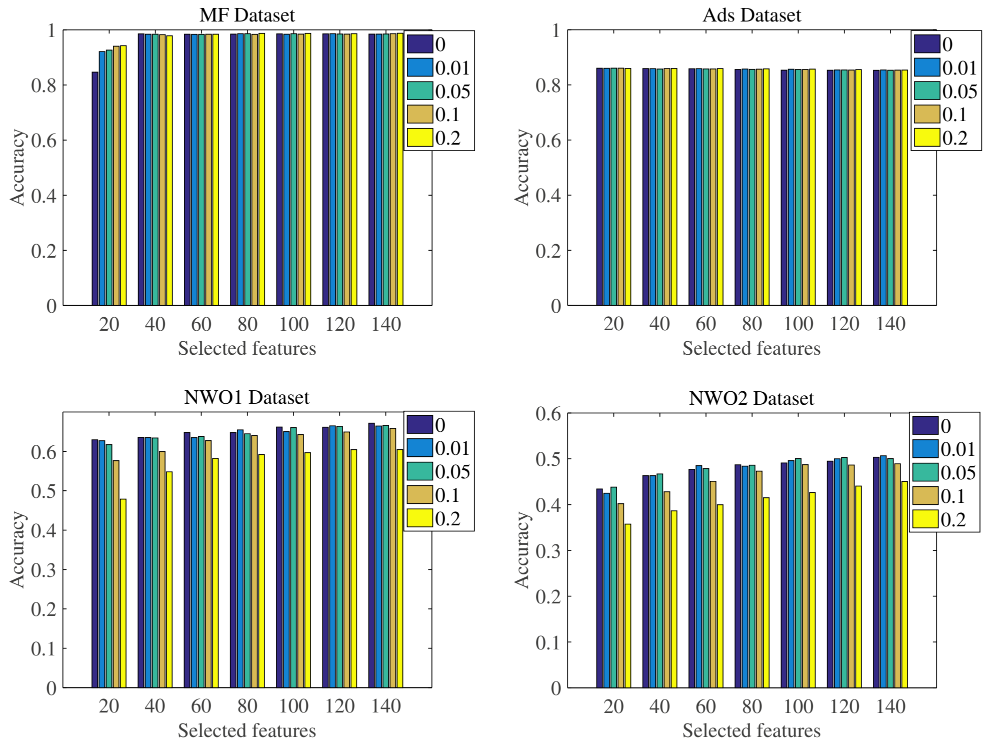

Fig. 5 Classification accuracies of the proposed method with respect to the ratio of outliers under different number of selected features, where the ratio of outliers is tuned from $\{0, 0.01, 0.05, 0.1, 0.2\}$

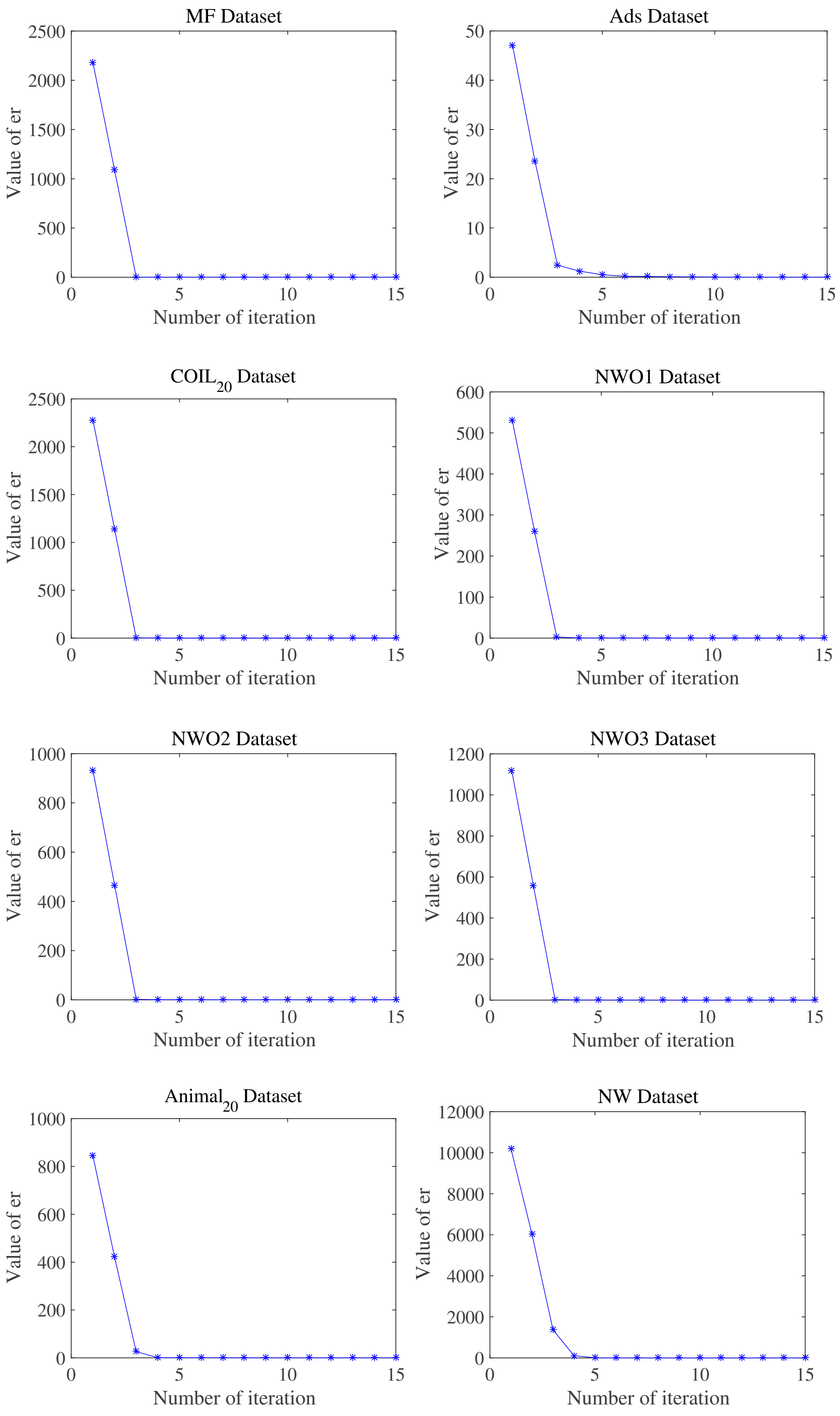

Fig. 6 Convergence curve of Algorithm 1

dataset, the parameters λ and ε are fixed as the optimal combination. Figure 6 shows how the convergence criterion term *er* in Algorithm 1 changes as the number of iteration increases. We can see that the value of *er* gradually reduces and then tends to zero, which means that the proposed method can satisfy the convergence condition within 15 iterations.

5 Conclusion

In this paper, an efficient multiview feature selection framework is designed via the distributed learning strategy. The proposed method adopts the sample-partition based distributed strategy to compute the loss term on each category and uses the view-partition based distributed strategy by minimizing the individual regularization of each view and common loss term across views. The capped l_2-norm based loss term instead of traditional l_2-norm is used to improve the robustness to outliers. Based on the distributed strategy, the proposed method has great superiority on reducing the computational complexity and fastening the running time of feature selection. The comparison experiments show the effectiveness of the proposed model in terms of the training time and classification accuracy.

In our model, the relationship of different views is taken into account via the distributed multiview learning. In the future work, we will try to introduce a set of weight parameters on different views to design a more fitting model for multiview learning.

Acknowledgments The authors would like to thank the reviewers for their valuable comments and suggestions to improve the quality of this paper in advance.

Appendix

Theorem 2 *Equations* (16)-(19) *is equivalent to* (20)-(23).

Proof 1) For fixed i, the solution in (17) is obtained by updating V matrices $\mathbf{Z}_{iv} \in \mathbb{R}^{n_i \times C} (v = 1, 2...V)$ with totally $n_i CV$ variables. Now, we will carry it out by solving the (21) with only $n_i C$ variables.

Let $\overline{\mathbf{Z}_i} = \frac{1}{V}\sum_{v=1}^{V} \mathbf{Z}_{iv}$, the (17) can be rewritten as

$$\min_{\overline{\mathbf{Z}_i}, \mathbf{Z}_{iv}} \; tr\left[\left(V\overline{\mathbf{Z}_i} - \mathbf{Y}_i\right)^T \mathbf{F}_i{}^{(k)} \left(V\overline{\mathbf{Z}_i} - \mathbf{Y}_i\right)\right] + \frac{\rho}{2}\sum_{v=1}^{V} \|\mathbf{Z}_{iv} - \mathbf{X}_{iv}\mathbf{W}_v{}^{(k+1)} + \frac{1}{\rho}\mathbf{P}_{iv}{}^{(k)}\|_F^2$$
$$s.t. \;\; \overline{\mathbf{Z}_i} = \frac{1}{V}\sum_{v=1}^{V} \mathbf{Z}_{iv} \tag{34}$$

Minimizing over $\mathbf{Z}_{iv}(v = 1, 2, ..., V)$ with $\overline{\mathbf{Z}_i}$ fixed has the solution

$$\mathbf{Z}_{iv} - \mathbf{X}_{iv}\mathbf{W}_v{}^{(k+1)} + \frac{1}{\rho}\mathbf{P}_{iv}{}^{(k)} = 0 \tag{35}$$

Let $\overline{\mathbf{X}_i\mathbf{W}} = \frac{1}{V}\sum_{v=1}^{V} \mathbf{X}_{iv}\mathbf{W}_v$, $\overline{\mathbf{P}_i} = \frac{1}{V}\sum_{v=1}^{V} \mathbf{P}_{iv}$. Then, we have

$$\overline{\mathbf{Z}_i} - \overline{\mathbf{X}_i\mathbf{W}}^{(k+1)} + \frac{1}{\rho}\overline{\mathbf{P}_i}^{(k)} = 0 \tag{36}$$

Combining (35) and (36), we have

$$\mathbf{Z}_{iv} = \mathbf{X}_{iv}\mathbf{W}_v{}^{(k+1)} - \frac{1}{\rho}\mathbf{P}_{iv}{}^{(k)} + \overline{\mathbf{Z}_i} - \overline{\mathbf{X}_i\mathbf{W}}^{(k+1)} + \frac{1}{\rho}\overline{\mathbf{P}_i}^{(k)} \tag{37}$$

Substituting (37) into (34),we get the following equation.

$$\min_{\overline{\mathbf{Z}_i}} \; tr\left[\left(V\overline{\mathbf{Z}_i} - \mathbf{Y}_i\right)^T \mathbf{F}_i{}^{(k)} \left(V\overline{\mathbf{Z}_i} - \mathbf{Y}_i\right)\right] + \frac{\rho V}{2}\|\overline{\mathbf{Z}_i} - \overline{\mathbf{X}_i\mathbf{W}}^{(k+1)} + \frac{1}{\rho}\overline{\mathbf{P}_i}^{(k)}\|_F^2 \tag{38}$$

Based on the above derivation, we can see that solving (17) is equivalent to solving (21).

2) Similarly, for fixed i, the $\mathbf{P}_{iv}$-update in (19) can be carried out by solving the (23) with only $n_i C$ variables. Replacing $\mathbf{Z}_{iv}$ in (37) with $\mathbf{Z}_{iv}{}^{(k+1)}$ obtains

$$\mathbf{Z}_{iv}{}^{(k+1)} = \mathbf{X}_{iv}\mathbf{W}_v{}^{(k+1)} - \frac{1}{\rho}\mathbf{P}_{iv}{}^{(k)} + \overline{\mathbf{Z}_i}^{(k+1)} - \overline{\mathbf{X}_i\mathbf{W}}^{(k+1)} + \frac{1}{\rho}\overline{\mathbf{P}_i}^{(k)} \tag{39}$$

Substituting (39) into (19) gives

$$\mathbf{P}_{iv}{}^{(k+1)} = \rho\left(\overline{\mathbf{Z}_i}^{(k+1)} - \overline{\mathbf{X}_i\mathbf{W}}^{(k+1)}\right) + \overline{\mathbf{P}_i}^{(k)} \tag{40}$$

(40) shows that the variables $\mathbf{P}_{iv}{}^{(k+1)}(v = 1, 2, ..., V)$ are all equal for fixed i. So we can get $\mathbf{P}_{iv}{}^{(k+1)} = \frac{1}{V}\sum_{v=1}^{V} \mathbf{P}_{iv}{}^{(k+1)} = \overline{\mathbf{P}_i}^{(k+1)}$. Then, we have

$$\overline{\mathbf{P}_i}^{k+1} = \rho\left(\overline{\mathbf{Z}_i}^{(k+1)} - \overline{\mathbf{X}_i\mathbf{W}}^{(k+1)}\right) + \overline{\mathbf{P}_i}^{(k)} \tag{41}$$

Based on the above derivation, we can see that solving (19) is equivalent to solving (23).

3) By replacing $\mathbf{Z}_{iv}$ in (37) with $\mathbf{Z}_{iv}{}^{(k)}$ and substituting $\mathbf{Z}_{iv}{}^{(k)}$ into (16), we have

$$\mathbf{W}_v{}^{(k+1)} := \arg\min_{\mathbf{W}_v}\left(\lambda\|\mathbf{W}_v\|_{2,1} + \frac{\rho}{2}\sum_{i=1}^{C}\|\mathbf{X}_{iv}\mathbf{W}_v - \left(\mathbf{X}_{iv}\mathbf{W}_v{}^{(k)} + \overline{\mathbf{Z}_i}^{(k)} - \overline{\mathbf{X}_i\mathbf{W}}^{(k)} + \frac{1}{\rho}\overline{\mathbf{P}_i}^{(k)}\right)\|_F^2\right) \tag{42}$$

Based on the above derivation, we can see that solving (16) is equivalent to solving (20).

4) According to $\overline{\mathbf{Z}_i} = \frac{1}{V}\sum_{v=1}^{V}\mathbf{Z}_{iv}$, we can see that solving (18) is equivalent to solving (22). $\square$

References

1. Boyd S, Parikh N, Chu E, Peleato B, Eckstein J (2011) Distributed optimization and statistical learning via the alternating direction method of multipliers. Found Trends Mach Learn 3(1): 1–122
2. Cai X, Nie F, Huang H (2013) Exact top-k feature selection via $l_{2,0}$-norm constraint. In: Proceedings of the Twenty-third International Joint Conference on Artificial Intelligence, pp 1240–1246
3. Cheng X, Zhu Y, Song J, Wen G, He W (2017) A novel low-rank hypergraph feature selection for multiview classification. Neurocomputing 253:115–121
4. Chua TS, Tang J, Hong R, Li H, Luo Z, Zheng Y (2009) NUS-WIDE: A Real-World Web Image Database from National University of Singapore. In: Proceedings of the ACM International Conference on Image and Video Retrieval, pp 368–375
5. Dalal N, Triggs B (2005) Histograms of oriented gradients for human detection. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp 886–893
6. Dua D, Graff C (2019) UCI Machine learning repository, University of California, School of Information and Computer Science. http://archive.ics.uci.edu/ml, Irvine
7. Feng Y, Xiao J, Zhuang Y, Liu X (2012) Adaptive unsupervised multi-view feature selection for visual concept recognition. In: Proceedings of the 11th Asian Conference on Computer Vision, pp. 343–357
8. Gui J, Sun Z, Ji S (2017) Feature selection based on structured sparsity: a comprehensive study. IEEE Trans Neural Netw Learn Syst 28(7):1490–1507
9. Hou C, Nie F, Tao H, Yi D (2017) Multi-view unsupervised feature selection with adaptive similarity and view weight. IEEE Trans Knowl Data Eng 29(9):1998–2011
10. Lan G, Hou C, Nie F, Luo T, Yi D (2018) Robust feature selection via simultaneous sapped norm and sparse regularizer minimization. Neurocomputing 283:228–240
11. Lee S, Park YT, d'Auriol BJ (2012) A novel feature selection method based on normalized mutual information. Appl Intell 37(1):100–120
12. Li Y, Shi X, Du C, Liu Y, Wen Y (2016) Manifold regularized multi-view feature selection for social image annotation. Neurocomputing 204:135–141
13. Lin Q, Xue Y, Wen J, Zhong P (2019) A sharing multi-view feature selection method via Alternating Direction Method of Multipliers. Neurocomputing 333:124–134
14. Lowe DG (2004) Distinctive image features from scale-invariant keypoints. Int J Comput Vis 60(2):91–110
15. Nene SA, Nayar SK, Murase H (1996) Columbia object image library (COIL-20)
16. Nie F, Huang H, Cai X, Ding C (2010) Efficient and robust feature selection via joint $l_{2,1}$-norms minimization. In: Proceedings of the Advances in Neural Information Processing Systems, pp 1813–1821
17. Nie F, Li J, Li X (2017) Convex multiview semi-supervised classification. IEEE Trans Image Process 26(12):5718–5729
18. Ojala T, Pietikainen M, Maenpaa T (2002) Multiresolution gray-scale and rotation invariant texture classification with local binary patterns. IEEE Trans Pattern Anal Machine Intell 24(7):971–987
19. Oliva A, Torralba A (2001) Modeling the shape of the scene: a holistic representation of the spatial envelope. Int J Comput Vis 42(3):145–175
20. Shao W, He L, Lu C, Wei X, Yu P (2016) Online unsupervised multi-view feature selection. In: Proceedings of the IEEE 16th International Conference on Data Mining, pp 1203–1208
21. Shi C, Duan C, Gu Z, Tian Q, An G, Zhao R (2019) Semi-supervised feature selection analysis with structured multiview sparse regularization. Neurocomputing 330:412–424
22. Sindhwani V, Niyogi P, Belkin M (2005) A co-regularization approach to semi-supervised learning with multiple views. In: Proceedings of ICML workshop on Learning With Multiple Views, pp 74–79
23. Tang C, Chen J, Liu X, Li M, Wang P, Wang M, Lu P (2018) Consensus learning guided multiview unsupervised feature selection. Knowl-Based Syst 160:49–60
24. Tibshirani R (1996) Regression shrinkage and selection via the lasso. J R Stat Soc Series B Stat Methodol 58(1):267–288
25. Tibshirani R, Saunders M, Rosset S, Zhu J, Knight K (2005) Sparsity and smoothness via the fused lasso. J R Stat Soc Ser B Stat Methodol 67(1):91–108
26. Tao H, Hou C, Nie F, Zhu J, Yi D (2017) Scalable multi-view semi-supervised classification via adaptive regression. IEEE Trans Image Process 26(9):4283–4296
27. Wang H, Nie F, Huang H (2011) Identifying quantitative trait loci via group-sparse multitask regression and feature selection: an imaging genetics study of the ADNI cohort. Bioinformatics 28(2):229–237
28. Wang H, Nie F, Huang H (2013) Multi-view clustering and feature learning via structured sparsity. In: Proceedings of the International Conference on Machine Learning, pp 352–360
29. Wang H, Nie F, Huang H, Ding C (2013) Heterogeneous visual features fusion via sparse multimodal machine. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp 3097–3102
30. Wang N, Xue Y, Lin Q, Zhong P (2019) Structured sparse multi-view feature selection based on weighted hinge loss. Multimed Tools Appl 78(11):15455–15481
31. Wang Z, Feng Y, Qi T, Yang X, Zhang J (2016) Adaptive multi-view feature selection for human motion retrieval. IET Signal Process 120:691–701
32. Xiao L, Sun Z, He R, Tan T (2013) Coupled feature selection for cross-sensor iris recognition. In: Proceedings of the IEEE Sixth International Conference on Biometrics: Theory, Applications and Systems (BTAS), pp 1–6
33. Xue X, Nie F, Wang S, Stantic B (2017) Multiview correlated feature learning by uncovering shared component. In: Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence
34. Yang M, Deng C, Nie F (2019) Adaptive-weighting discriminative regression for multiview classification. Pattern Recogn 88:236–245
35. Yang W, Gao Y, Cao L, Yang M, Shi Y (2014) mPadal: a joint local-and-global multi-view feature selection method for activity recognition. Appl Intell 41(3):776–790
36. Yang W, Gao Y, Shi Y, Cao L (2015) MRM-Lasso: A sparse multiview feature selection method via low-rank analysis. IEEE Trans Neural Netw Learn Syst 26(11):2801–2815
37. Yuan M, Lin Y (2006) Model selection and estimation in regression with grouped variables. J R Stat Soc Ser B Stat Methodol 68(1):49–67

38. You X, Xu J, Yuan W, Jing XY, Tao D, Zhang T (2019) Multi-view common component discriminant analysis for cross-view classification. Pattern Recogn 92:37–51
39. Zhang L, Lu X (2018) New fast feature selection methods based on multiple support vector data description. Appl Intell 48(7):1776–1790
40. Zhang Q, Tian Y, Yang Y, Pan C (2015) Automatic spatial spectral feature selection for hyperspectral image via discriminative sparse multimodal learning. IEEE Trans Geosci Remote Sens 53(1):261–279
41. Zhang R, Nie F, Li X (2017) Self-weighted supervised discriminative feature selection. IEEE Trans Neural Netw Learn Syst 29(8):3913–3918
42. Zhang Y, Li HG, Wang Q, Peng C (2019) A filter-based bare-bone particle swarm optimization algorithm for unsupervised feature selection. Appl Intell 49(8):2889–2898
43. Zhong J, Wang N, Lin Q, Zhong P (2019) Weighted feature selection via discriminative sparse multi-view learning. Knowl-Based Syst 178:132–148

Publisher's note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.

Min Men has received the B.S. degree from China Agricultural University in 2018. She is currently studying for the master degree in the college of science, China Agricultural University, China. Her research interest is machine learning.

Ping Zhong is a professor and Ph.D supervisor in College of Science, China Agricultural University, Beijing, China. She has published many papers. Her research interests include machine learning, support vector machines, and steganalysis.

Zhi Wang has received the B.S. degree from Zhengzhou University in 2016. And then, she has received the M.S. degree from Dalian Maritime University in 2019. She is currently pursuing the Ph.D. degree at China Agricultural University. Her current research interest is machine learning.

Qiang Lin was born in 1990 in China. He has received the B.S. degree and M.S. degree from China Agricultural University in 2012 and 2015. Now he is a Ph.D. student of China Agricultural University. His research direction is large-scale machine learning.