Received March 9, 2020, accepted March 23, 2020, date of publication March 27, 2020, date of current version April 15, 2020.

Digital Object Identifier 10.1109/ACCESS.2020.2983829

Dynamic Graph Regularization and Label Relaxation-Based Sparse Matrix Regression for Two-Dimensional Feature Selection

XIUHONG CHEN[1,2] AND YUN LU[1]

[1]School of Artificial Intelligence and Computer Science, Jiangnan University, Wuxi 214122, China
[2]Jiangsu Key Laboratory of Media Design and Software Technology, Jiangnan University, Wuxi 214122, China

Corresponding author: Xiuhong Chen (xiuhongc@jiangnan.edu.cn)

ABSTRACT Sparse matrix regression (SMR) is a two-dimensional supervised feature selection method that can directly select the features on matrix data. It uses several couples of left and right regression vectors for each classifier and integrates them in formulating the regression function. However, SMR does not consider the local geometry of image samples, and it assumes that the training samples should exactly fit a linear model or a strict binary label matrix by left and right regression matrices. In order to enlarge margins between different classes and preserve the intrinsic geometry structure of samples in the transformed space, we will propose dynamic graph regularization and label relaxation-based SMR (abbreviated as DGRLR-SMR) method for two-dimensional supervised feature selection. First, the label relaxation SMR is established by relaxing the strict binary label matrix into a slack variable matrix via a nonnegative label relaxation matrix by the ε-dragging technique. Second, we construct a dynamic graph matrix learning model, rather than using the heat kernel function to obtain a fixed graph matrix, to capture the discriminative information and the local manifold structure of the image samples. Therefore, the proposed model not only enlarges margins between different classes, but also obtains a sparse transformation matrix and avoids the problem of over-fitting. An optimization algorithm is devised to solve this model, and it has closed-form solutions in each iteration so that it can be implemented easily in real application. Extensive experiments on several data sets demonstrate the superiority of our method.

INDEX TERMS Feature selection, sparse matrix regression, label relaxation, dynamic graph matrix, ε-dragging technique.

I. INTRODUCTION

In many real applications, the data is often of very high dimension and contains some redundant features. Therefore, selecting or extracting some important features from these data is a crucial task in some fields, such as data mining and machine learning [1]. The best representative dimensionality reduction techniques are feature selection (FS) [2] and subspace learning (SL) [3], [4], where feature selection is to select a few relevant features from the high-dimensional data to represent the original data, while subspace learning applies a transformation on the high-dimensional data to obtain a low-dimensional representation of the data. According to whether the label information of the data is used or not, these methods can be divided into supervised, semi-supervised and unsupervised learning. Since feature selection does not change the representation of the original data and maintains its physical meaning, we mainly discuss the supervised feature selection in this work.

Most existing supervised feature selection algorithms are vector-based. For example, Fisher Score [1] is a filtering feature selection algorithm based on linear discriminative analysis (LDA). Liu *et al.* [5] propose a global and local structure preservation feature selection method (GLSPFS). To preserve both the local and global structure of the features as well as samples, Zhu *et al.* [6] give a robust unsupervised spectral feature selection method. Nie *et al.* [7] incorporate sparse constraint in robust linear regression and design a robust feature selection (RFS) algorithm. Liu *et al.* [8] uses $l_{2,1}$-norm as the penalty and give a multi-task feature

The associate editor coordinating the review of this manuscript and approving it for publication was Jianqing Zhu.

selection method. The feature selection approach given by Cai *et al.* [9] has the $l_{2,1}$-norm-based loss function with an explicit $l_{2,0}$-norm equality constraint. Xiang *et al.* [10] present a framework of discriminative least squares regression for feature selection. He *et al.* [11] also study the problem of robust feature extraction based on $l_{2,1}$ regularized correntropy.

However, in many real applications especially in the fields of image processing and video analysis, the data used are often presents in matrix form. Thus, these matrix data are usually scanned into vector data in traditional vector-based methods in advance. Nevertheless, this vectorization of the matrix data may cause some problems. First, vectorized data is often high dimensional, which makes the vector-based methods often suffer from small sample size problem (SSS). Second, vectorization will also ignore the location information of elements in the original matrix and destroy the relative geometric relationship between them. Third, when the image contains some noises (such as the block-wise noisy occlusion), we should treat this occlusion in a whole part. This correlation will be lost when they are treated as flatting or vertical vectors. Therefore, the influence and effects of noisy for vector-based approaches also depends on how to vectorize the image. To address these problems, it is necessary to directly study the problem of feature selection for matrix data. For example, Hou *et al.* [12] measured the relationship between matrix data and the class labels by deploying left and right regression matrices and propose an algorithm named sparse matrix regression (SMR) for two-dimensional supervised feature selection. Yuan *et al.* [13] present a joint sparse matrix regression and nonnegative spectral analysis (JSMRNS) model for image unsupervised feature selection.

On the other hand, in data classification, we expect that margins between different classes in the transformed label space should be as large as possible. Consequently, Leski [14] propose a least squares regression (LSR) model via the squares approximations of the misclassification errors to enlarge the margins between different classes. However, when training samples are transformed into a linear model or a strict binary label matrix, a discriminative transformation matrix may not be learned in practice application. In order to relax the strict binary label matrix, Xiang *et al.* [10] introduced a ε-dragging technique to force the regression targets of different classes moving along with opposite directions. By this technique, class label information will be embedded into the LSR model such that the distances between classes can be enlarged, and a discriminant LSR (DLSR) model for multiclass classification is obtained.

The existing research results show that the manifold learning methods can well preserve the intrinsic geometry structure of samples in the transformed space, but they do not exploit label information to improve the discriminant ability of algorithms. To address this problem, Li *et al.* [15] gave the margin Fisher analysis method to simultaneously preserve both the intrinsic geometry structure and the discriminant structure of the samples by using the label information. Other similar methods also include locality linear discriminant analysis (LLDA) [16], [17], locality sensitive discriminant analysis (LSDA) [18], and local discriminant embedding (LDE) [19], [20]. Recently, Fang *et al.* [21] proposed a regularized label relaxation linear regression (RLR) method, which relaxes the strict binary label matrix into a slack variable matrix by introducing a nonnegative label relaxation matrix and can avoid the problem of over-fitting by constructing the class compactness graph. In all the above manifold learning-based methods, the weights of the adjacency graph characterizing affinity between samples are generally determined in advance by using the heat kernel function. Therefore, the graph matrix is fixed during learning the transformation matrix and is more sensitive to the tuning of the heat kernel parameter. In order to address this problem, many dynamic graph-based methods have been proposed. For example, Liu *et al.* [22] proposed discriminative low-rank preserving projection by incorporating the discriminant analysis and the local neighborhood relationship of the original samples into the low-rank representation. Lai *et al.* [23] present approximate orthogonal sparse embedding for dimensionality reduction. Liu *et al.* [24] combined structural optimal graph with sparse representation for feature extraction. Wen *et al.* [25] combined low-rank representations with adaptive graphs. Meanwhile, Wen *et al.* [26] also applied adaptive graph learning to incomplete multi-view spectral clustering.

Inspired by the SMR and RLR, in this paper, we propose dynamic graph regularization and label relaxation-based SMR (DGRLR-SMR) method for image supervised feature selection. To directly select some important features on the matrix data, DGRLR-SMR uses the sum of squares of differences between the matrix data and the slack label by deploying left and right regression matrices as the loss function. Furthermore, in order to capture the geometric structure and the discriminative information of the samples, we also dynamically learn the graph matrix by following the distribution of samples, rather than using the heat kernel function to learn a fixed graph matrix. Here, we do not need to tune other parameters such as the heat kernel parameter in the graph matrix learning process. Thus, we can couple the graph matrix learning with the low dimensional space learning together and iteratively optimize them to obtain their respective optimal solutions. Therefore, DGRLR-SMR not only relaxes the strict binary label matrix into a slack variable matrix by introducing a nonnegative label relaxation matrix, but also obtains a sparse transformation matrix and avoids the problem of over-fitting. An optimization algorithm is devised to solve DGRLR-SMR, and it has closed-form solutions in each iteration so that it can be implemented easily.

The main contributions of this work follow as: 1) our model can enlarge the margins between different classes as much as possible, which are beneficial to the correct classification; 2) it can also learn the graph matrix dynamically without tuning additional parameter and capture the local geometric structure of training matrix samples; 3) it can select

some more meaningful features by the sparse transformation matrix composed of the left and right regression matrices; 4) we design an alternating iteration algorithm to solve the optimization model, give the closed-form solutions in each iteration and analyze the convergence and complexity of the algorithm; 5) extensive experiments on some datasets demonstrate the superiority of DGRLR-SMR.

The remainder of this paper is organized as follows. In Section II, we review and introduce some related models, including the traditional linear regression (LR), the label relaxation LR (RLR), the bilinear regression (BLR) and the sparse matrix regression (SMR). In Section III, we propose the dynamic graph regularization and label relaxation-based SMR (DGRLR-SMR) method for image supervised feature selection, and present its solving algorithm and algorithmic convergence analyses. In Section IV, we report some experiment results on multiclass classification. Conclusions are drawn in Section V.

II. RELATED WORKS

Throughout this paper, reals are written as lowercase letters. Vectors are denoted by boldface lowercase letters, while matrices are presented as boldface uppercase letters. Given the set of matrix samples $\{\boldsymbol{X}_i \in R^{m\times n} : i = 1, 2, \ldots, l\}$, where m and n are the first and second dimensions of each matrix sample respectively, and l is the number of matrix samples. Suppose that these l matrix samples are from c classes. The associated class label vectors are $\{\boldsymbol{y}_1, \boldsymbol{y}_2, \ldots, \boldsymbol{y}_l\} \subset R^c$, where $\boldsymbol{y}_i = [y_{i1}, y_{i2}, \ldots, y_{ic}]^T \in \{0, 1\}^c$ is the cluster indicator vector of $\boldsymbol{X}_i$, that is, $y_{ii} = 1$ if and only if $\boldsymbol{X}_i$ belongs to the j-th cluster and $y_{ii} = 0$ otherwise. Define $\boldsymbol{1} = [1, 1, \ldots, 1]^T \in R^l$ be a column vector of all ones, $\boldsymbol{I}_k \in R^{k\times k}$ be an identity matrix. Denote $\boldsymbol{x}_i = vec(\boldsymbol{X}_i) \in R^{mn}$ the vector counterparts of the matrix sample $\boldsymbol{X}_i$, i = 1,2,…,l, where $vec(\cdot)$ as an operator which converts a matrix to a vector by collecting the columns. $\boldsymbol{X} = [\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_l] \in R^{mn\times l}$.

A. LABEL RELAXATION LINEAR REGRESSION (LRLR)

The traditional linear regression (LR) is a simple and very effective regression analysis method. The LR model is formulated as follows

$$\min_{\boldsymbol{A}} \left\|\boldsymbol{A}^T\boldsymbol{X} - \boldsymbol{Y}\right\|_F^2 + \alpha \left\|\boldsymbol{A}\right\|_F^2 = \sum_{i=1}^{l}\sum_{k=1}^{c}(\boldsymbol{a}_k^T\boldsymbol{x}_i - y_{ki})^2 + \alpha\sum_{k=1}^{c}\boldsymbol{a}_k^T\boldsymbol{a}_k, \tag{1}$$

where $\boldsymbol{A} = [\boldsymbol{a}_1, \boldsymbol{a}_2, \ldots, \boldsymbol{a}_c] \in R^{mn\times c}$ is a regression coefficients matrix or projection matrix, $\boldsymbol{a}_k$ is the regression coefficient vector or projection vector for the k-th classifier. $\boldsymbol{X} = [\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_l] \in R^{mn\times l}$ is the vector -based sample matrix, and $\boldsymbol{Y} = [\boldsymbol{y}_1, \boldsymbol{y}_2, \ldots, \boldsymbol{y}_l] \in R^{c\times l}$ is the corresponding label matrix. Generally, the linear regression method assumes that the training samples should be exactly transformed into a strict binary label matrix. However, this rigorous assumption cannot learn a discriminative transformation matrix in many real applications. In order to relax this assumption and enlarge the distance between different classes, Fang *et al.* [21] proposed the following label relaxation LR (LRLR) model by the ε-dragging technique

$$\begin{aligned}\min_{\boldsymbol{A}} &\left\|\boldsymbol{U}^T\boldsymbol{X} - (\boldsymbol{Y} + \boldsymbol{B}\odot\boldsymbol{M})\right\|_F^2 + \alpha\left\|\boldsymbol{U}\right\|_F^2,\\ s.t\quad &\boldsymbol{M} \geq 0\end{aligned} \tag{2}$$

where, $\boldsymbol{U} \in R^{mn\times c}$ is a regression coefficient matrix, the operator $\odot$ is the Hadamard product of matrices, $\boldsymbol{M}$ is the label relaxation matrix, and $\boldsymbol{B}$ is a luxury matrix corresponding to label matrix $\boldsymbol{Y}$, the element of which are defined as: if $y_{ki} = 1$, then $b_{ki} = +1$; if $y_{ki} = 0$, then $b_{ki} = -1$. In order to avoid over-fitting problems that may occur due to this label relaxation, they also used a class compact graph regularization term based on the manifold learning instead of $\|\boldsymbol{U}\|_F^2$ in (2).

B. BILINEAR REGRESSION

The above linear regression models are vector-based. If input data are matrix data, we have general bilinear regression (GBR) [27]. In this case, we can replace the regression function $\boldsymbol{a}_k^T\boldsymbol{x}_i$ of linear regression model (1) with a bilinear regression function $\boldsymbol{u}_k^T\boldsymbol{X}_i\boldsymbol{v}_k$, and have the following bilinear regression model

$$\min_{\{\boldsymbol{u}_k\},\{\boldsymbol{v}_k\}}\sum_{i=1}^{l}\sum_{k=1}^{c}(\boldsymbol{u}_k^T\boldsymbol{X}_i\boldsymbol{v}_k - y_{ki})^2 + \alpha\sum_{k=1}^{c}\boldsymbol{a}_k^T\boldsymbol{a}_k, \tag{3}$$

where $\boldsymbol{u}_k \in R^m$ and $\boldsymbol{v}_k \in R^n$ are the left and right regression coefficient vectors for k-th class, respectively. $\boldsymbol{X}_i \in R^{m\times n}$ is the i-th matrix data, and $\boldsymbol{a}_k = vec(\boldsymbol{u}_k\boldsymbol{v}_k^T)$. GBR is the two-dimensional counterpart of the traditional vector-based regression model. Comparing with LR, there are $m + n$ degrees of free variables in $\boldsymbol{a}_k = vec(\boldsymbol{u}_k\boldsymbol{v}_k^T)$, which are too strict in many real applications. Thus, it cannot characterize the original data fully and increase the regression error.

C. SPARSE MATRIX REGRESSION (SMR)

In order to address the problem existing in GBR, Lai *et al.* [23] proposed to use d couples of left projecting vectors $\{\boldsymbol{u}_j^{(k)}\}_{j=1}^d$ and right projecting vectors $\{\boldsymbol{v}_j^{(k)}\}_{j=1}^d$ for the k-th classifier and join them in formulating the regression item. Thus, the loss function for the k-th classifier is

$$\sum_{i=1}^{l}\left(\sum_{j=1}^{d}(\boldsymbol{u}_j^{(k)})^T\boldsymbol{X}_i\boldsymbol{v}_j^{(k)} - y_{ki}\right)^2 = \sum_{i=1}^{l}\left(Tr((\boldsymbol{U}^{(k)})^T\boldsymbol{X}_i\boldsymbol{V}^{(k)}) - y_{ki}\right)^2, \tag{4}$$

where $\boldsymbol{U}^{(k)} = [\boldsymbol{u}_1^{(k)}, \boldsymbol{u}_2^{(k)}, \ldots, \boldsymbol{u}_d^{(k)}] \in R^{m\times d}$, $\boldsymbol{V}^{(k)} = [\boldsymbol{v}_1^{(k)}, \boldsymbol{v}_2^{(k)}, \ldots, \boldsymbol{v}_d^{(k)}] \in R^{n\times d}$.

Denote

$$\begin{aligned}\boldsymbol{p}_k &= vec(\boldsymbol{U}^{(k)}(\boldsymbol{V}^{(k)})^T) \in R^{mn},\\ \boldsymbol{P} &= [\boldsymbol{p}_1, \boldsymbol{p}_2, \ldots, \boldsymbol{p}_c] \in R^{mn\times c},\end{aligned}$$

we have

$$Tr((\boldsymbol{U}^{(k)})^T\boldsymbol{X}_i\boldsymbol{V}^{(k)}) = \left(vec(\boldsymbol{U}^{(k)}(\boldsymbol{V}^{(k)})^T)\right)^T vec(\boldsymbol{X}_i) = (\boldsymbol{p}_k)^T\boldsymbol{x}_i. \tag{5}$$

This indicates that the vector $[Tr((\boldsymbol{U}^{(1)})^T\boldsymbol{X}_i\boldsymbol{V}^{(1)}), \ldots, Tr((\boldsymbol{U}^{(c)})^T\boldsymbol{X}_i\boldsymbol{V}^{(c)})]^T$ is the projection of the vector $\boldsymbol{x}_i = vec(\boldsymbol{X}_i)$ under the transformation matrix $\boldsymbol{P}$. If we replace the regression vector $\boldsymbol{a}_k$ in (1) with the vector $\boldsymbol{p}_k$, the loss function for the k-th classifier in (4) will become the first term in (1). Therefore, the matrix $\boldsymbol{P}$ can be regarded as the linear transformation matrix as in traditional regression model. To achieve feature selection, we expect that the transformation matrix $\boldsymbol{P}$ should have some structure sparsity property. Therefore, the following sparse matrix regression (SMR) model for two-dimensional feature selection is obtained

$$\min_{\{\boldsymbol{U}^{(k)}\},\{\boldsymbol{V}^{(k)}\}} \sum_{k=1}^{c}\sum_{i=1}^{l}\left(Tr((\boldsymbol{U}^{(k)})^T\boldsymbol{X}_i\boldsymbol{V}^{(k)}) - y_{ki}\right)^2 + \alpha\|\boldsymbol{P}\|_{2,1}, \tag{6}$$

with $\boldsymbol{P} = [\boldsymbol{p}_1, \boldsymbol{p}_2, \ldots, \boldsymbol{p}_c] \in R^{mn\times c}$, $\boldsymbol{p}_k = vec(\boldsymbol{U}^{(k)}(\boldsymbol{V}^{(k)})^T) \in R^{mn}$, where $\|\cdot\|_{2,1}$ is the $l_{2,1}$-norm of matrix.

III. THE PROPOSED MODEL AND ALGORITHM

A. MODEL AND FORMULATION

As pointed out in section II, when the training samples are accurately transformed into a strict binary label matrix, the discriminant transformation matrices may not be learned in many real applications. For classification problem, we expect that the margins between different classes should be as large as possible after they are transformed into their label space. To this end, we can use the ε-dragging technique to enlarge these margins and obtain a more discriminative transformation matrix.

Noting that for the i-th training sample $\boldsymbol{X}_i \in R^{m\times n}$, the regression target y_{ki} in (4) or (6) is fixed as $+1$ or 0, which usually cause wrong penalization to the right classifications that are far from $+1$ or 0. To alleviate this situation, we can introduce the nonnegative label relaxation variable m_{ki} and obtain a new discriminative regression target $\bar{y}_{ki} = y_{ki} + b_{ki}m_{ki}$, where b_{ki} is the element in the luxury matrix $\boldsymbol{B} \in R^{c\times l}$ corresponding to the label matrix $\boldsymbol{Y}$. The non-negativity of the label relaxation variable m_{ki} can always increase the absolute value of the new regression target $\bar{y}_{ki}$. As a result, the margins between different classes will be enlarged. By replacing y_{ki} in the loss function of the problem (6) with $\bar{y}_{ki}$, we can get the label relaxation sparse matrix regression as follows

$$\min_{\{\boldsymbol{U}^{(k)}\},\{\boldsymbol{V}^{(k)}\},m_{ki}\geq 0} \sum_{k=1}^{c}\sum_{i=1}^{l}\left(Tr((\boldsymbol{U}^{(k)})^T\boldsymbol{X}_i\boldsymbol{V}^{(k)}) - (y_{ki}+b_{ki}m_{ki})\right)^2 + \alpha\|\boldsymbol{P}\|_{2,1}, \tag{7}$$

with $\boldsymbol{P} = [\boldsymbol{p}_1, \boldsymbol{p}_2, \ldots, \boldsymbol{p}_c] \in R^{mn\times c}$, $\boldsymbol{p}_k = vec(\boldsymbol{U}^{(k)}(\boldsymbol{V}^{(k)})^T) \in R^{mn}$, where $m_{ki} \geq 0$. Since the non-negativity of m_{ki} can preserve and enlarge the distances between different classes in the transformation subspace, we can ensure a more robust performance than the sparse matrix regression (SMR).

On the other hand, it has shown that both the global structure and the local structure of the samples may provide some complementary information to promote the performance of dimensionality reduction. Although the model (7) can reveal the local correlation of elements in matrix data while learning the globality and discrimination of matrix samples, it does not consider the local manifold structure of matrix samples. According to the idea of manifold learning, the samples sharing the same labels should be kept close together in the transformed space. This local structure of the samples can be preserved via learning a graph matrix on low-dimensional space. Intuitively speaking, it can be described by the following model:

$$\min_{\boldsymbol{P}} \sum_{i=1}^{l}\sum_{j=1}^{l}\left\|\boldsymbol{P}^T\boldsymbol{x}_i - \boldsymbol{P}^T\boldsymbol{x}_j\right\|_2^2 w_{ij}, \tag{8}$$

where, $\boldsymbol{x}_i = vec(\boldsymbol{X}_i) \in R^{mn}$, the weight w_{ij} denotes the similarity between the i-th matrix sample $\boldsymbol{X}_i$ and the j-th matrix sample $\boldsymbol{X}_j$. In general, the weight w_{ij} can be defined by the heat kernel function as follows: if the sample $\boldsymbol{X}_i$ and $\boldsymbol{X}_j$ belong to the same class or $\boldsymbol{X}_j$ is one of the k-nearest neighbors of $\boldsymbol{X}_i$, then $w_{ij} = \exp\left(-\|\boldsymbol{X}_i - \boldsymbol{X}_j\|_F^2/(2\sigma^2)\right)$ where σ is a tuning parameter; otherwise, $w_{ij} = 0$.

Many existing learning methods based on the graph weight matrix $\boldsymbol{W}$ need to learn a fixed graph matrix from the original high-dimensional data before learning the transformation matrix $\boldsymbol{P}$. In this way, when the original data is corrupted by noise, an incorrect graph matrix may be obtained. In addition, this strategy need to tune two parameters (i.e. k and σ), which is time-consuming. To address such problems, we can combine the learning of graph matrix with the learning of low-dimensional subspace to obtain the following model

$$\begin{aligned}&\min_{\boldsymbol{P},\boldsymbol{W}} \sum_{i=1}^{l}\sum_{j=1}^{l}\left\|\boldsymbol{P}^T\boldsymbol{x}_i - \boldsymbol{P}^T\boldsymbol{x}_j\right\|_2^2 w_{ij} + \gamma\|\boldsymbol{W}\|_2^2\\ &s.t.\ \boldsymbol{w}_i^T\boldsymbol{1} = 1,\ w_{ii} = 0,\ i = 1, 2, \ldots, l,\\ &\quad w_{ij} \geq 0\ if\ j \in N(i),\ otherwise\ w_{ij} = 0,\end{aligned} \tag{9}$$

where, γ is a parameter,$\boldsymbol{w}_i \in R^l$ is the i-th column of the graph weight matrix $\boldsymbol{W}$, the regularization term $\|\boldsymbol{W}\|_2^2$ is used to avoid the trivial solution, $\boldsymbol{1} = [1, 1, \ldots, 1]^T \in R^l$ is an all-one-element vector, $N(i)$ is the set of all training matrix samples with the same labels as $\boldsymbol{X}_i$, the constraint $\boldsymbol{w}_i^T\boldsymbol{1} = 1$ is used to obtain shift invariant similarity. Thus, the problem (9) can output small value of w_{ii} for distant samples and large value of w_{ii} for close samples. Therefore, the model (9) will learn a dynamical graph weight matrix $\boldsymbol{W}$.

In addition, according to (5), the model (9) can also be rewritten as follows

$$\begin{aligned}\min_{\boldsymbol{P},\boldsymbol{W}} &\sum_{i=1}^{l}\sum_{j=1}^{l}\sum_{k=1}^{c}\left(Tr((\boldsymbol{U}^{(k)})^T\boldsymbol{X}_i\boldsymbol{V}^{(k)})-Tr((\boldsymbol{U}^{(k)})^T\boldsymbol{X}_j\boldsymbol{V}^{(k)})\right)^2\\ &\times w_{ij}+\gamma\left\|\boldsymbol{W}\right\|_2^2\\ s.t.\ &\boldsymbol{w}_i^T\mathbf{1}=1,\ w_{ii}=0,\ i=1,2,\ldots,l,\\ &w_{ij}\geq 0\ if\ j\in N(i),\ otherwise\ w_{ij}=0,\end{aligned}\tag{10}$$

Since both the graph weight matrix $\boldsymbol{W}$ and the transformation matrix $\boldsymbol{P}=[\boldsymbol{p}_1,\boldsymbol{p}_2,\ldots,\boldsymbol{p}_c]\in R^{mn\times c}$ are unknown, (10) may output unreliable models. By integrating (7) with (10), we obtain the following dynamic graph regularized label relaxation sparse matrix regression (DGRLR-SMR) model

$$\begin{aligned}\min_{\{\boldsymbol{U}^{(k)}\},\{\boldsymbol{V}^{(k)}\},\boldsymbol{M},\boldsymbol{W}} &\sum_{k=1}^{c}\sum_{i=1}^{l}\left(Tr((\boldsymbol{U}^{(k)})^T\boldsymbol{X}_i\boldsymbol{V}^{(k)})-(y_{ki}+b_{ki}m_{ki})\right)^2\\ &+\alpha\left\|\boldsymbol{P}\right\|_{2,1}\\ &+\beta\sum_{i=1}^{l}\sum_{j=1}^{l}\sum_{k=1}^{c}\left(Tr((\boldsymbol{U}^{(k)})^T\boldsymbol{X}_i\boldsymbol{V}^{(k)})-Tr((\boldsymbol{U}^{(k)})^T\boldsymbol{X}_j\boldsymbol{V}^{(k)})\right)^2\\ &\times w_{ij}+\gamma\left\|\boldsymbol{W}\right\|_2^2\\ s.t.\ &\boldsymbol{w}_i^T\mathbf{1}=1,\ w_{ii}=0,\ i=1,2,\ldots,l,\\ &w_{ij}\geq 0\ if\ j\in N(i),\ otherwise\ w_{ij}=0,\\ &m_{ki}\geq 0,\ k=1,2,\ldots,c,\ i=1,2,\ldots,l,\end{aligned}\tag{11}$$

with $\boldsymbol{P}=[\boldsymbol{p}_1,\boldsymbol{p}_2,\ldots,\boldsymbol{p}_c]\in R^{mn\times c}$, $\boldsymbol{p}_k=vec(\boldsymbol{U}^{(k)}(\boldsymbol{V}^{(k)})^T)\in R^{mn}$, where α, β and γ are nonnegative parameters. The first and second terms are used to characterize the sparse matrix regression with label relaxation, from which the sparse transformation matrix can be learned while enlarging the margins between different classes, thus improving the classification accuracy. The third and fourth terms can iteratively update the left and right regression matrices and the graph weight matrix, so that the global structure of the samples can be revealed and the local structure of the samples can be preserved while avoiding the trivial solution, thus ensuring that the similar training samples are kept close together in the transformed space.

In the optimization problem (11), there are essentially three types of variables, that is, the regression matrices $\{\boldsymbol{U}^{(k)}\}_{k=1}^c$ and $\{\boldsymbol{V}^{(k)}\}_{k=1}^c$, the label relaxation matrix $\boldsymbol{M}$ and the graph weight matrix $\boldsymbol{W}$. The regression matrix variables $\boldsymbol{U}^{(k)}$ and $\boldsymbol{V}^{(k)}$ are coupled in the loss function. On the other hand, all regression matrices $\boldsymbol{U}^{(k)}$ and $\boldsymbol{V}^{(k)}$ are coupled together in formulating matrix $\boldsymbol{P}$, and so the elements of the matrix $\boldsymbol{P}$ are the complex combinations of $\boldsymbol{U}^{(k)}$ and $\boldsymbol{V}^{(k)}$ for $k=1,2,\ldots,c$. In addition, the objection function is also non-smooth. Thus, it is difficult to solve the problem (11) simultaneously. Here, we will design an alternating iteration algorithm to optimize these variables.

B. SOLUTION

Since it is difficult to directly solve the optimization problem (10) or (11), we will design an alternating iteration algorithm to optimize these variables, that is, we will alternatively update one of $\{\boldsymbol{U}^{(k)}\}_{k=1}^c$, $\{\boldsymbol{V}^{(k)}\}_{k=1}^c$, $\boldsymbol{M}$ and $\boldsymbol{W}$ while fixing other variables, respectively.

1) Update $\{\boldsymbol{U}^{(k)}\}_{k=1}^c$ by fixing $\boldsymbol{W}$, $\{\boldsymbol{V}^{(k)}\}_{k=1}^c$ and $\boldsymbol{M}=(m_{ki})$.

After some simple mathematical deductions, the problem (10) or (11) can be rewritten as

$$\begin{aligned}\min_{\{\boldsymbol{U}^{(k)}\}} &\sum_{k=1}^{c}\sum_{i=1}^{l}\left(Tr((\boldsymbol{U}^{(k)})^T\boldsymbol{X}_i\boldsymbol{V}^{(k)})-(y_{ki}+b_{ki}m_{ki})\right)^2\\ &+\alpha Tr(\boldsymbol{P}^T\boldsymbol{D}^{(v)}\boldsymbol{P})+\beta Tr(\boldsymbol{P}^T\boldsymbol{X}\boldsymbol{L}\boldsymbol{X}^T\boldsymbol{P})\end{aligned}\tag{12}$$

where, $\boldsymbol{D}^{(v)}$ is a $mn\times mn$ diagonal matrix with the r-th diagonal element $D_{rr}^{(v)}$ defined by

$$D_{rr}^{(v)}=\begin{cases}\dfrac{1}{2\left\|\boldsymbol{p}^r\right\|_2}, & if\ \boldsymbol{p}^r\neq 0,\\ & r=1,2,\ldots,mn,\\ \delta, & otherwise,\end{cases}\tag{13}$$

δ is a very small positive constant, $\boldsymbol{p}^r$ is the r-th row vector of $\boldsymbol{P}$ in the previous iteration, $r=1,2,\ldots,mn$; the matrix $\boldsymbol{L}=\boldsymbol{S}-\boldsymbol{W}$ is graph Laplacian matrix, $\boldsymbol{S}$ is a diagonal matrix and its diagonal entries are defined as $S_{ii}=\sum_{j=1}^{l}w_{ij}$.

Denote $\hat{\boldsymbol{u}}^{(k)}=vec(\boldsymbol{U}^{(k)})\in R^{md}$, $\boldsymbol{g}_i^{(k)}=vec(\boldsymbol{X}_i\boldsymbol{V}^{(k)})\in R^{md}$ and $\boldsymbol{G}^{(k)}=[\boldsymbol{g}_1^{(k)},\boldsymbol{g}_2^{(k)},\ldots,\boldsymbol{g}_l^{(k)}]\in R^{md\times l}$, then we have

$$\begin{aligned}\boldsymbol{p}_k&=vec(\boldsymbol{U}^{(k)}(\boldsymbol{V}^{(k)})^T)=(\boldsymbol{V}^{(k)}\otimes\boldsymbol{I}_m)\hat{\boldsymbol{u}}^{(k)},\\ \boldsymbol{p}_k^T\boldsymbol{x}_i&=(vec(\boldsymbol{U}^{(k)}(\boldsymbol{V}^{(k)})^T))^T vec(\boldsymbol{X}_i)\\ &=Tr((\boldsymbol{U}^{(k)})^T\boldsymbol{X}_i\boldsymbol{V}^{(k)})=(\hat{\boldsymbol{u}}^{(k)})^T\boldsymbol{g}_i^{(k)},\\ Tr(\boldsymbol{P}^T\boldsymbol{D}^{(v)}\boldsymbol{P})&=\sum_{k=1}^{c}\boldsymbol{p}_k^T\boldsymbol{D}^{(v)}\boldsymbol{p}_k=\sum_{k=1}^{c}(\hat{\boldsymbol{u}}^{(k)})^T\boldsymbol{A}_1^{(k)}\hat{\boldsymbol{u}}^{(k)},\\ Tr(\boldsymbol{P}^T\boldsymbol{X}\boldsymbol{L}\boldsymbol{X}^T\boldsymbol{P})&=\sum_{k=1}^{c}\boldsymbol{p}_k^T\boldsymbol{X}\boldsymbol{L}\boldsymbol{X}^T\boldsymbol{p}_k=\sum_{k=1}^{c}(\hat{\boldsymbol{u}}^{(k)})^T\boldsymbol{B}_1^{(k)}\hat{\boldsymbol{u}}^{(k)},\end{aligned}$$

where, the operator $\otimes$ is the Kronecker product, $\boldsymbol{A}_1^{(k)}=(\boldsymbol{V}^{(k)}\otimes\boldsymbol{I}_m)^T\boldsymbol{D}^{(v)}(\boldsymbol{V}^{(k)}\otimes\boldsymbol{I}_m)$ and $\boldsymbol{B}_1^{(k)}=\boldsymbol{G}^{(k)}\boldsymbol{L}(\boldsymbol{G}^{(k)})^T$. Hence, when $\boldsymbol{D}^{(v)}$ is fixed, we can decompose the problem (12) into the following c independent sub-problems

$$\begin{aligned}\min_{\boldsymbol{U}^{(k)}} &\left\|(\hat{\boldsymbol{u}}^{(k)})^T\boldsymbol{G}^{(k)}-(\boldsymbol{Y}^{(k)}+\boldsymbol{B}^{(k)}\odot\boldsymbol{M}^{(k)})\right\|_2^2\\ &+(\hat{\boldsymbol{u}}^{(k)})^T(\alpha\boldsymbol{A}_1^{(k)}+\beta\boldsymbol{B}_1^{(k)})\hat{\boldsymbol{u}}^{(k)},\\ &k=1,2,\ldots,c.\end{aligned}\tag{14}$$

where, the operator $\odot$ is the Hadamard product; $\boldsymbol{Y}^{(k)}$, $\boldsymbol{B}^{(k)}$ and $\boldsymbol{M}^{(k)}$ are the k-th row of the label matrix $\boldsymbol{Y}$, the luxury matrix $\boldsymbol{B}$ and the label relaxation matrix $\boldsymbol{M}$, respectively.

Taking the derivative of objective function in (14) with respect to $\hat{\boldsymbol{u}}^{(k)}$, and set it to zero, we obtain the following optimal solution for (14)

$$\begin{aligned}\hat{\boldsymbol{u}}^{(k)}=&\left[\boldsymbol{G}^{(k)}(\boldsymbol{G}^{(k)})^T+\alpha\boldsymbol{A}_1^{(k)}+\beta\boldsymbol{B}_1^{(k)}\right]^{-1}\\ &\times\boldsymbol{G}^{(k)}(\boldsymbol{Y}^{(k)}+\boldsymbol{B}^{(k)}\odot\boldsymbol{M}^{(k)})^T,\\ &k=1,2,\ldots,c.\end{aligned}\tag{15}$$

And, we can obtain the left regression matrix $\boldsymbol{U}^{(k)} \in R^{m\times d}$ from $\hat{\boldsymbol{u}}^{(k)}$, $k = 1,2,\ldots,c$.

2) Update $\{\boldsymbol{V}^{(k)}\}_{k=1}^{c}$ by fixing $\boldsymbol{W}$,$\{\boldsymbol{U}^{(k)}\}_{k=1}^{c}$ and $\boldsymbol{M} = (m_{ki})$.

Noting that when $\{\boldsymbol{U}^{(k)}\}_{k=1}^{c}$ is fixed, $\boldsymbol{p}_k = (\boldsymbol{I}_n \otimes \boldsymbol{U}^{(k)})vec((\boldsymbol{V}^{(k)})^T)$, so $\boldsymbol{p}_k$ cannot be formulated by $vec(\boldsymbol{V}^{(k)})$ and so the above deduction cannot be used directly. However, it can be seen from $(\boldsymbol{U}^{(k)}(\boldsymbol{V}^{(k)})^T)^T$=$\boldsymbol{V}^{(k)}(\boldsymbol{U}^{(k)})^T$ that all elements in $\boldsymbol{U}^{(k)}(\boldsymbol{V}^{(k)})^T$ and $\boldsymbol{V}^{(k)}(\boldsymbol{U}^{(k)})^T$ are exactly the same. As a result, the vectors $vec(\boldsymbol{U}^{(k)}(\boldsymbol{V}^{(k)})^T)$ and $vec(\boldsymbol{V}^{(k)}(\boldsymbol{U}^{(k)})^T)$ are different permutation of the same elements. Denote $\boldsymbol{q}_k = vec(\boldsymbol{V}^{(k)}(\boldsymbol{U}^{(k)})^T)$, then the rows of the matrix $\boldsymbol{Q} = [\boldsymbol{q}_1, \boldsymbol{q}_2, \ldots, \boldsymbol{q}_c] \in R^{mn\times c}$ are just different arrangements of the rows of the matrix $\boldsymbol{P}$. Thus, by the definition of the $l_{2,1}$-norm, it holds $\|\boldsymbol{P}\|_{2,1} = \|\boldsymbol{Q}\|_{2,1}$. We define the diagonal matrix $\boldsymbol{D}^{(u)}$ with the r-th diagonal element

$$D_{rr}^{(u)} = \begin{cases} \dfrac{1}{2\left\|\boldsymbol{q}^r\right\|_2}, & if\ \boldsymbol{q}^r \neq 0, \\ \delta, & otherwise, \end{cases} \quad r = 1, 2, \cdots, mn, \tag{16}$$

then the problem (11) can be equivalently reformulated as

$$\min_{\{\boldsymbol{V}^{(k)}\}} \sum_{k=1}^{c}\sum_{i=1}^{l}\left(Tr((\boldsymbol{U}^{(k)})^T\boldsymbol{X}_i\boldsymbol{V}^{(k)}) - (y_{ki} + b_{ki}m_{ki})\right)^2 + \alpha Tr(\boldsymbol{Q}^T\boldsymbol{D}^{(u)}\boldsymbol{Q}) + \beta Tr(\boldsymbol{P}^T\boldsymbol{X}\boldsymbol{L}\boldsymbol{X}^T\boldsymbol{P}) \tag{17}$$

Denote $\hat{\boldsymbol{v}}^{(k)} = vec(\boldsymbol{V}^{(k)}) \in R^{nd}$, $\boldsymbol{h}_i^{(k)} = vec(\boldsymbol{X}_i^T\boldsymbol{U}^{(k)}) \in R^{nd}$ and $\boldsymbol{H}^{(k)} = [\boldsymbol{h}_1^{(k)}, \boldsymbol{h}_2^{(k)}, \ldots, \boldsymbol{h}_l^{(k)}] \in R^{nd\times l}$, we have

$$\begin{aligned} \boldsymbol{q}_k &= (\boldsymbol{U}^{(k)} \otimes \boldsymbol{I}_n)vec(\boldsymbol{V}^{(k)}) = (\boldsymbol{U}^{(k)} \otimes \boldsymbol{I}_n)\hat{\boldsymbol{v}}^{(k)}, \\ \boldsymbol{p}_k^T\boldsymbol{x}_i &= (vec(\boldsymbol{U}^{(k)}(\boldsymbol{V}^{(k)})^T))^T vec(\boldsymbol{X}_i) = Tr((\boldsymbol{U}^{(k)})^T\boldsymbol{X}_i\boldsymbol{V}^{(k)}) \\ &= Tr((\boldsymbol{V}^{(k)})^T\boldsymbol{X}_i^T\boldsymbol{U}^{(k)}) = (\hat{\boldsymbol{v}}^{(k)})^T\boldsymbol{h}_i^{(k)}. \end{aligned}$$

Then, when $\boldsymbol{D}^{(u)}$ is fixed, the problem (17) can also be separated into the following c independent sub-problems

$$\min_{\hat{\boldsymbol{v}}^{(k)}, b_k} \left\|(\hat{\boldsymbol{v}}^{(k)})^T\boldsymbol{H}^{(k)} - (\boldsymbol{Y}^{(k)} + \boldsymbol{B}^{(k)} \odot \boldsymbol{M}^{(k)})\right\|_2^2 + (\hat{\boldsymbol{v}}^{(k)})^T[\alpha\boldsymbol{A}_2^{(k)} + \beta\boldsymbol{B}_2^{(k)}]\hat{\boldsymbol{v}}^{(k)}, \tag{18}$$

where,$\boldsymbol{A}_2^{(k)} = (\boldsymbol{U}^{(k)} \otimes \boldsymbol{I}_n)^T\boldsymbol{D}^{(u)}(\boldsymbol{U}^{(k)} \otimes \boldsymbol{I}_n)$ and $\boldsymbol{B}_2^{(k)} = \boldsymbol{H}^{(k)}\boldsymbol{L}(\boldsymbol{H}^{(k)})^T$, $k = 1,2,\ldots,c$.

The optimal solutions should be

$$\hat{\boldsymbol{v}}^{(k)} = \left[\boldsymbol{H}^{(k)}(\boldsymbol{H}^{(k)})^T + \alpha\boldsymbol{A}_2^{(k)} + \beta\boldsymbol{B}_2^{(k)}\right]^{-1} \times \boldsymbol{H}^{(k)}(\boldsymbol{Y}^{(k)} + \boldsymbol{B}^{(k)} \odot \boldsymbol{M}^{(k)})^T, \quad k = 1, 2, \ldots, c. \tag{19}$$

And, we can get the right regression matrix $\boldsymbol{V}^{(k)} \in R^{n\times d}$ from $\hat{\boldsymbol{v}}^{(k)}$.

3) Update $\boldsymbol{M} = (m_{ki})$ by fixing $\boldsymbol{W}$,$\{\boldsymbol{U}^{(k)}\}_{k=1}^{c}$ and $\{\boldsymbol{V}^{(k)}\}_{k=1}^{c}$.

In this case, the problem (11) is equivalent to the following optimization problem

$$\min_{m_{ki}\geq 0} \sum_{k=1}^{c}\sum_{i=1}^{l}\left(b_{ki}m_{ki} - (Tr((\boldsymbol{U}^{(k)})^T\boldsymbol{X}_i\boldsymbol{V}^{(k)}) - y_{ki})\right)^2. \tag{20}$$

It can also be decomposed into the cl independent sub-problems as follows

$$\min_{m_{ki}\geq 0} \left(b_{ki}m_{ki} - (Tr((\boldsymbol{U}^{(k)})^T\boldsymbol{X}_i\boldsymbol{V}^{(k)}) - y_{ki})\right)^2, \quad k = 1, 2, \ldots, c,\ i = 1, 2, \ldots, l.$$

The optimal solution of $\boldsymbol{M} = (m_{ki})$ is

$$m_{ki} = \max\{b_{ki}(Tr((\boldsymbol{U}^{(k)})^T\boldsymbol{X}_i\boldsymbol{V}^{(k)}) - y_{ki}), 0\}, \quad k = 1, 2, \ldots, c,\ i = 1, 2, \ldots, l. \tag{21}$$

4) Update $\boldsymbol{W}$ by fixing $\{\boldsymbol{U}^{(k)}\}_{k=1}^{c}$, $\{\boldsymbol{V}^{(k)}\}_{k=1}^{c}$ and $\boldsymbol{M} = (m_{ki})$.

Here, the problem (11) will be transformed into the following minimization problem

$$\begin{aligned} &\min_{\boldsymbol{W}} \sum_{k=1}^{c}\sum_{i=1}^{l}\sum_{j=1}^{l} z_{ij}^{(k)}w_{ij} + \frac{\gamma}{\beta}\sum_{i=1}^{l}\sum_{j=1}^{l} w_{ij}^2 \\ &s.t.\ \boldsymbol{w}_i^T\boldsymbol{1} = 1,\ w_{ii} = 0,\ i = 1, 2, \ldots, l, \\ &\quad w_{ij} \geq 0\ if\ j \in N(i),\ otherwise\ w_{ij} = 0,\ i = 1, 2, \ldots, l. \end{aligned} \tag{22}$$

where, $z_{ij}^{(k)} = \left\|Tr((\boldsymbol{U}^{(k)})^T\boldsymbol{X}_i\boldsymbol{V}^{(k)}) - Tr((\boldsymbol{U}^{(k)})^T\boldsymbol{X}_j\boldsymbol{V}^{(k)})\right\|_2^2$, $j = 1,2,\ldots,l$, $i = 1,2,\ldots,l$.

This quadratic programming problem can be solved by the dual method. The optimal solution of the problem (22) is obtained by

$$w_{ij} = \max\left\{\frac{1}{|N(i)|} - \frac{\beta}{2\gamma}\sum_{k=1}^{c}\left(z_{ij}^{(k)} - \frac{1}{|N(i)|}\sum_{j\in N(i)} z_{ij}^{(k)}\right), 0\right\}, \quad j \in N(i) \tag{23}$$

For the training matrix sample $\boldsymbol{X}_i$, if it comes from the k-th class, then $|N(i)| = n_k$, where n_k is the number of the training matrix samples in the k-th class, and $\sum_{k=1}^{c} n_k = l$. Thus, the graph weight matrix $\boldsymbol{W}$ obtained by (23) should be a diagonal block matrix $\boldsymbol{W}$=$diag(\boldsymbol{W}^{(1)}, \boldsymbol{W}^{(2)},\ldots, \boldsymbol{W}^{(c)})$, where $\boldsymbol{W}^{(k)}$ is a $n_k \times n_k$ matrix defined by (23), $k = 1,2,\ldots,c$.

We iteratively update $\{\boldsymbol{U}^{(k)}\}_{k=1}^{c}$, $\{\boldsymbol{V}^{(k)}\}_{k=1}^{c}$, $\boldsymbol{M}$ and $\boldsymbol{W}$ by solving the above problems until a convergence criterion is satisfied. The whole iterative procedure for solving the model (11) is summarized in the following algorithm DGRLR-SMR.

After obtaining the final transformation matrix $\boldsymbol{P}$ by the algorithm DGRLR-SMR, we use the l_2-norm of the row vectors of $\boldsymbol{P}$ to evaluate the importance of each feature. Thus, we can either select a fixed number of the most important elements or set a threshold and select the element whose importance is larger than this value. In this way, we can preserve the selected rows in $\boldsymbol{P}$ and change the other rows to zero vectors to implement feature selection of the samples.

Algorithm 1 DGRLR-SMR

Input: Data matrixes $\boldsymbol{X}_i$ for $i = 1,2,\ldots, l$, labels matrix $\boldsymbol{Y} = [\boldsymbol{y}_1, \boldsymbol{y}_2, \ldots, \boldsymbol{y}_l] \in R^{c\times l}$; tuning parameters α, β, γ; the number of couples $d \in \{1, 2, \ldots\}$

Initialization: $\boldsymbol{V}=[\boldsymbol{I}_{d\times d}, \boldsymbol{0}_{d\times(n-d)}]^T$ and $\boldsymbol{M}=\boldsymbol{I}_{c\times l}$; $\boldsymbol{L}=\boldsymbol{I}_{l\times l}$; $\boldsymbol{D}^{(v)} =\boldsymbol{I}_{mn\times mn}$

Repeat

Update $\hat{\boldsymbol{u}}^{(k)}$ by (15) and obtain $\boldsymbol{U}^{(k)}$, $k = 1,2,\ldots,c$;

Update $\boldsymbol{D}^{(u)}$ by (16);

Update $\hat{\boldsymbol{v}}^{(k)}$ by (19) and obtain $\boldsymbol{V}^{(k)}$, $k = 1,2,\ldots,c$;

Update $\boldsymbol{D}^{(v)}$ by (13);

Update $\boldsymbol{M}$ by (21);

Update $\boldsymbol{W}$ by (23);

Calculate the Laplacian matrix $\boldsymbol{L}=\boldsymbol{S}$ - $(\boldsymbol{W}+\boldsymbol{W}^T)/2$;

Until convergence

Output: $\boldsymbol{P} = [\boldsymbol{p}_1, \boldsymbol{p}_2, \ldots, \boldsymbol{p}_c], \boldsymbol{p}_k = vec(\boldsymbol{U}^{(k)}(\boldsymbol{V}^{(k)})^T)$.

When we determine the features, we can use the 1-nearst neighborhood classifier (i.e. NN-classifier) to perform classification on the data with selected features.

C. CONVERGENCE ANALYSIS AND COMPUTATIONAL COMPLEXITY

In this section, we will prove the convergence of the proposed algorithm, and then analyze the complexity.

Theorem 1. The objective function value in the problem (10) or (11) monotonically decreases until algorithm converges.

Proof. Suppose that after the t-th iteration, we have obtained $\{\boldsymbol{U}_t^{(k)}\}_{k=1}^c$, $\{\boldsymbol{V}_t^{(k)}\}_{k=1}^c$, $\boldsymbol{M}_t$, $\boldsymbol{W}_t$, $\boldsymbol{D}_t^{(u)}$ and $\boldsymbol{D}_t^{(v)}$. Denote the value of the objective function during t-th iteration as $f(\{\boldsymbol{U}_t^{(k)}\}_{k=1}^c, \{\boldsymbol{V}_t^{(k)}\}_{k=1}^c, \boldsymbol{M}_t, \boldsymbol{W}_t)$.

By Algorithm, variables $\{\boldsymbol{U}^{(k)}\}_{k=1}^c$ are updated by $\{\boldsymbol{U}_{t+1}^{(k)}\}_{k=1}^c = \arg\min_{\{\boldsymbol{U}^{(k)}\}_{k=1}^c} f(\{\boldsymbol{U}^{(k)}\}_{k=1}^c, \{\boldsymbol{V}_t^{(k)}\}_{k=1}^c, \boldsymbol{M}_t, \boldsymbol{W}_t)$ with fixing $\{\boldsymbol{V}_t^{(k)}\}_{k=1}^c$, $\boldsymbol{M}_t$ and $\boldsymbol{W}_t$. Then, $\{\boldsymbol{U}_{t+1}^{(k)}\}_{k=1}^c$ satisfy the following inequality

$$\begin{aligned}
&\sum_{k=1}^{c}\sum_{i=1}^{l}\left(Tr((\boldsymbol{U}_{t+1}^{(k)})^T\boldsymbol{X}_i\boldsymbol{V}_t^{(k)}) - (y_{ki}+b_{ki}(m_{ki})_t\right)^2 \\
&\quad + \alpha Tr((\boldsymbol{P}_{t+1,t}^T\boldsymbol{D}_t^{(v)}\boldsymbol{P}_{t+1,t}) + \beta Tr(\boldsymbol{P}_{t+1,t}^T\boldsymbol{X}\boldsymbol{L}_t\boldsymbol{X}^T\boldsymbol{P}_{t+1,t}) \\
&\le \sum_{k=1}^{c}\sum_{i=1}^{l}\left(Tr((\boldsymbol{U}_t^{(k)})^T\boldsymbol{X}_i\boldsymbol{V}_t^{(k)}) - (y_{ki}+b_{ki}(m_{ki})_t\right)^2 \\
&\quad + \alpha Tr((\boldsymbol{P}_{t,t}^T\boldsymbol{D}_t^{(v)}\boldsymbol{P}_{t,t}) + \beta Tr(\boldsymbol{P}_{t,t}^T\boldsymbol{X}\boldsymbol{L}_t\boldsymbol{X}^T\boldsymbol{P}_{t,t}) \qquad (24)
\end{aligned}$$

where

$$\begin{aligned}
\boldsymbol{P}_{t+1,t} &= [(\boldsymbol{p}_1)_{t+1,t}, (\boldsymbol{p}_2)_{t+1,t}, \ldots, (\boldsymbol{p}_c)_{t+1,t}], \\
(\boldsymbol{p}_k)_{t+1,t} &= vec(\boldsymbol{U}_{t+1}^{(k)}(\boldsymbol{V}_t^{(k)})^T), \\
\boldsymbol{P}_{t,t} &= [(\boldsymbol{p}_1)_{t,t}, (\boldsymbol{p}_2)_{t,t}, \ldots, (\boldsymbol{p}_c)_{t,t}], \\
(\boldsymbol{p}_k)_{t,t} &= vec(\boldsymbol{U}_t^{(k)}(\boldsymbol{V}_t^{(k)})^T),
\end{aligned}$$

and $\boldsymbol{L}_t = \boldsymbol{S}_t - \boldsymbol{W}_t$. By the definition of $\boldsymbol{D}_t^{(v)}$, it holds

$$\begin{aligned}
&Tr((\boldsymbol{P}_{t+1,t}^T\boldsymbol{D}_t^{(v)}\boldsymbol{P}_{t+1,t}) \\
&\quad = \|\boldsymbol{P}_{t+1,t}\|_{2,1} + \sum_{r=1}^{mn}\left(\frac{\|(\boldsymbol{P}_{t+1,t})^r\|_2^2}{2\|(\boldsymbol{P}_{t,t})^r\|_2} - \|(\boldsymbol{P}_{t+1,t})^r\|_2\right) \qquad (25)
\end{aligned}$$

$$\begin{aligned}
&Tr((\boldsymbol{P}_{t,t}^T\boldsymbol{D}_t^{(v)}\boldsymbol{P}_{t,t}) \\
&\quad = \|\boldsymbol{P}_{t,t}\|_{2,1} + \sum_{r=1}^{mn}\left(\frac{\|(\boldsymbol{P}_{t,t})^r\|_2^2}{2\|(\boldsymbol{P}_{t,t})^r\|_2} - \|(\boldsymbol{P}_{t,t})^r\|_2\right). \qquad (26)
\end{aligned}$$

Combining (24), (25) and (26) with the inequality $\frac{\|\boldsymbol{a}\|_2^2}{2\|\boldsymbol{b}\|_2} - \|\boldsymbol{a}\|_2 \ge \frac{\|\boldsymbol{b}\|_2^2}{2\|\boldsymbol{b}\|_2} - \|\boldsymbol{b}\|_2$, it yields the following result

$$\begin{aligned}
&\sum_{k=1}^{c}\sum_{i=1}^{l}\left(Tr((\boldsymbol{U}_{t+1}^{(k)})^T\boldsymbol{X}_i\boldsymbol{V}_t^{(k)}) - (y_{ki}+b_{ki}(m_{ki})_t\right)^2 \\
&\quad + \alpha\|\boldsymbol{P}_{t+1,t}\|_{2,1} + \beta Tr(\boldsymbol{P}_{t+1,t}^T\boldsymbol{X}\boldsymbol{L}_t\boldsymbol{X}^T\boldsymbol{P}_{t+1,t}) \\
&\le \sum_{k=1}^{c}\sum_{i=1}^{l}\left(Tr((\boldsymbol{U}_t^{(k)})^T\boldsymbol{X}_i\boldsymbol{V}_t^{(k)}) - (y_{ki}+b_{ki}(m_{ki})_t\right)^2 \\
&\quad + \alpha\|\boldsymbol{P}_{t,t}\|_{2,1} + \beta Tr(\boldsymbol{P}_{t,t}^T\boldsymbol{X}\boldsymbol{L}_t\boldsymbol{X}^T\boldsymbol{P}_{t,t}),
\end{aligned}$$

that is,

$$\begin{aligned}
&f(\{\boldsymbol{U}_{t+1}^{(k)}\}_{k=1}^c, \{\boldsymbol{V}_t^{(k)}\}_{k=1}^c, \boldsymbol{M}_t, \boldsymbol{W}_t) \\
&\qquad \le f(\{\boldsymbol{U}_t^{(k)}\}_{k=1}^c, \{\boldsymbol{V}_t^{(k)}\}_{k=1}^c, \boldsymbol{M}_t, \boldsymbol{W}_t). \qquad (27)
\end{aligned}$$

Similarly, we can also get the following inequality when fixing $\{\boldsymbol{U}_{t+1}^{(k)}\}_{k=1}^c$, $\boldsymbol{M}_t$ and $\boldsymbol{W}_t$

$$\begin{aligned}
&\sum_{k=1}^{c}\sum_{i=1}^{l}\left(Tr((\boldsymbol{U}_{t+1}^{(k)})^T\boldsymbol{X}_i\boldsymbol{V}_{t+1}^{(k)}) - (y_{ki}+b_{ki}(m_{ki})_t\right)^2 \\
&\quad + \alpha\|\boldsymbol{Q}_{t+1,t+1}\|_{2,1} + \beta Tr(\boldsymbol{P}_{t+1,t+1}^T\boldsymbol{X}\boldsymbol{L}_t\boldsymbol{X}^T\boldsymbol{P}_{t+1,t+1}) \\
&\le \sum_{k=1}^{c}\sum_{i=1}^{l}\left(Tr((\boldsymbol{U}_{t+1}^{(k)})^T\boldsymbol{X}_i\boldsymbol{V}_t^{(k)}) - (y_{ki}+b_{ki}(m_{ki})_t\right)^2 \\
&\quad + \alpha\|\boldsymbol{Q}_{t+1,t}\|_{2,1} + \beta Tr(\boldsymbol{P}_{t+1,t}^T\boldsymbol{X}\boldsymbol{L}_t\boldsymbol{X}^T\boldsymbol{P}_{t+1,t})
\end{aligned}$$

Since $\|\boldsymbol{Q}_{t+1,t+1}\|_{2,1} = \|\boldsymbol{P}_{t+1,t+1}\|_{2,1}$ and $\|\boldsymbol{Q}_{t+1,t}\|_{2,1} = \|\boldsymbol{P}_{t+1,t}\|_{2,1}$, we have

$$\begin{aligned}
&f(\{\boldsymbol{U}_{t+1}^{(k)}\}_{k=1}^c, \{\boldsymbol{V}_{t+1}^{(k)}\}_{k=1}^c, \boldsymbol{M}_t, \boldsymbol{W}_t) \\
&\qquad \le f(\{\boldsymbol{U}_{t+1}^{(k)}\}_{k=1}^c, \{\boldsymbol{V}_t^{(k)}\}_{k=1}^c, \boldsymbol{M}_t, \boldsymbol{W}_t). \qquad (28)
\end{aligned}$$

When fixing $\{\boldsymbol{U}_{t+1}^{(k)}\}_{k=1}^c$, $\{\boldsymbol{V}_{t+1}^{(k)}\}_{k=1}^c$ and $\boldsymbol{W}_t$, we obtain

$$\begin{aligned}
&\sum_{k=1}^{c}\sum_{i=1}^{l}\left(Tr((\boldsymbol{U}_{t+1}^{(k)})^T\boldsymbol{X}_i\boldsymbol{V}_{t+1}^{(k)}) - (y_{ki}+b_{ki}(m_{ki})_{t+1}\right)^2 \\
&\le \sum_{k=1}^{c}\sum_{i=1}^{l}\left(Tr((\boldsymbol{U}_{t+1}^{(k)})^T\boldsymbol{X}_i\boldsymbol{V}_{t+1}^{(k)}) - (y_{ki}+b_{ki}(m_{ki})_t\right)^2.
\end{aligned}$$

Thus, the following inequality holds

$$f(\{\boldsymbol{U}_{t+1}^{(k)}\}_{k=1}^{c}, \{\boldsymbol{V}_{t+1}^{(k)}\}_{k=1}^{c}, \boldsymbol{M}_{t+1}, \boldsymbol{W}_t) \leq f(\{\boldsymbol{U}_{t+1}^{(k)}\}_{k=1}^{c}, \{\boldsymbol{V}_{t+1}^{(k)}\}_{k=1}^{c}, \boldsymbol{M}_t, \boldsymbol{W}_t) \quad (29)$$

And when $\{\boldsymbol{U}_{t+1}^{(k)}\}_{k=1}^{c}, \{\boldsymbol{V}_{t+1}^{(k)}\}_{k=1}^{c}$ and $\boldsymbol{M}_{t+1}$ are fixed, it yields by (22)

$$\beta \sum_{k=1}^{c}\sum_{i=1}^{l}\sum_{j=1}^{l}(z_{ij}^{(k)})_{t+1}(w_{ij})_{t+1} + \gamma \|\boldsymbol{W}_{t+1}\|_F^2 \leq \beta \sum_{k=1}^{c}\sum_{i=1}^{l}\sum_{j=1}^{l}(z_{ij}^{(k)})_{t+1}(w_{ij})_t + \gamma \|\boldsymbol{W}_t\|_F^2,$$

where

$$(z_{ij}^{(k)})_{t+1} = \left\| Tr((\boldsymbol{U}_{t+1}^{(k)})^T \boldsymbol{X}_i \boldsymbol{V}_{t+1}^{(k)}) - Tr((\boldsymbol{U}_{t+1}^{(k)})^T \boldsymbol{X}_j \boldsymbol{V}_{t+1}^{(k)}) \right\|_2^2.$$

So, we can get

$$f(\{\boldsymbol{U}_{t+1}^{(k)}\}_{k=1}^{c}, \{\boldsymbol{V}_{t+1}^{(k)}\}_{k=1}^{c}, \boldsymbol{M}_{t+1}, \boldsymbol{W}_{t+1}) \leq f(\{\boldsymbol{U}_{t+1}^{(k)}\}_{k=1}^{c}, \{\boldsymbol{V}_{t+1}^{(k)}\}_{k=1}^{c}, \boldsymbol{M}_{t+1}, \boldsymbol{W}_t). \quad (30)$$

Therefore, we obtain from (27), (28), (29) and (30)

$$f(\{\boldsymbol{U}_{t+1}^{(k)}\}_{k=1}^{c}, \{\boldsymbol{V}_{t+1}^{(k)}\}_{k=1}^{c}, \boldsymbol{M}_{t+1}, \boldsymbol{W}_{t+1}) \leq f(\{\boldsymbol{U}_t^{(k)}\}_{k=1}^{c}, \{\boldsymbol{V}_t^{(k)}\}_{k=1}^{c}, \boldsymbol{M}_t, \boldsymbol{W}_t),$$

which indicates that the objective value of (11) monotonically decrease in each iteration.

On the other hand, since the objective value of (11) is lower bounded, the sequence of objective function values obtained by algorithm is convergent.

Next, we briefly analyze the computational complexity of the iterative optimization procedure of the proposed DGRLR-SMR algorithm. In each iteration of the algorithm, the time cost focuses on updating $\{\boldsymbol{U}^k\}_{k=1}^{c}$, $\{\boldsymbol{V}^k\}_{k=1}^{c}$, $\boldsymbol{M}$ and $\boldsymbol{W}$. Since $1 \leq d \leq \min(m, n)$ holds [12], the computational complexity of updating $\{\boldsymbol{U}^k\}_{k=1}^{c}$, $\{\boldsymbol{V}^k\}_{k=1}^{c}$, $\boldsymbol{M}$ and $\boldsymbol{W}$ is, respectively,

$$O(mndlc + m^3d^3c + m^2d^2lc + mdl^2c),$$
$$O(mndlc + n^3d^3c + n^2d^2lc + ndl^2c),$$
$$O(mn(m+n)dlc)$$

and $O(l^2c)$ from (15), (19), (21) and (23). Therefore, the total computational complexity of algorithm is

$$O(((m^3 + n^3)d^3c + (m+n)(mn+l)dlc)T),$$

where T is the total number of iterations.

IV. EXPERIMENTS AND ANALYSIS

In this section, we compare DGRLR-SMR with some other existing feature selection approaches on several benchmark datasets to test and verify its effectiveness.

TABLE 1. Characters of different data sets.

Data	Size	Scale(m×n)	Class	Num. selected feature
ORL	400	32×32	40	50,100,…,300
BioID	1460	32×32	22	50,100,…,300
UMIST	575	28×23	20	40,80,…,240
JAFFE	213	32×32	10	20,40,…,120
USPS	2000	16×16	10	20,40,…,120
COIL-20	1440	32×32	20	20,40,…,120

FIGURE 1. Some images from six public data sets.

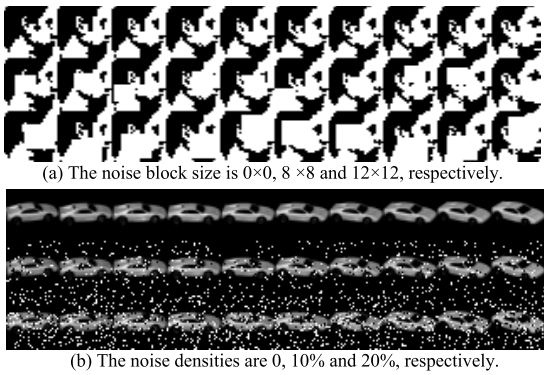


FIGURE 2. Some noisy images from (a) UMIST and (b) COIL-20.

A. DATASETS AND EVALUATION METRICS

In our experiments, six public datasets are employed to show the performance of different feature selection methods. These different datasets are four face image datasets, including UMIST [16], ORL [29], BioID [30] and JAFFE [31], one handwritten digit dataset USPS [32] and one object image dataset COIL-20 [33]. The detailed statistical characters of all datasets are listed in Table 1.

Fig. 1 shows some of the images on each dataset. And, Fig. 2 shows some of "clean" images and corrupted images from COIL-20 and UMIST data sets, respectively.

In order to verify the effectiveness of the proposed method, we will compare it with seven existing methods. These methods include Fisher score (Fisher - Scor) [1], global and local structure preservation framework for feature selection

(GLSPFS) [5], local and global structure preservation for robust unsupervised spectral feature selection(LGSP) [6], discriminative least squares regressions(DLSR) [10], sparse matrix regression (SMR) [12], regularized label relaxation linear regression (RLR) [21] and general bilinear regression (GBR) [27].

In DGRLR-SMR, the regularization parameters α, β and γ are fine-tuned by searching in the range of $\{10^{-4}, 10^{-3}, 10^{-2}, \ldots, 10^{2}, 10^{3}, 10^{4}\}$ via using cross validation method.

To test the quality of the selected features, we employ the following four different kinds of evaluation metrics. The first metric is the classification accuracy achieved by classifier using the selected features. The second one is the redundancy rate (RED) contained in the selected features. The third one is the variance of classifier using the selected features. The forth one is the normalized mutual information (NMI) as an evaluation indicator. Intuitively, an ideal feature selection approach should select features with high classification accuracy, few redundancies, low variance and high mutuality.

The classification accuracy serves as the evaluation metric for all the experiments, which is defined as

$$Accuracy = \frac{\sum_{i=1}^{n} \delta(y_i, \hat{y}_i)}{n}, \tag{31}$$

where y_i is the ground truth of each label, $\hat{y}_i$ is the corresponding predicted label, and n is the number of test samples. The function $\delta(\cdot, \cdot)$ measures the two input arguments, and outputs "1" if the two arguments are equal; or outputs "0" otherwise.

The redundancy is a popular evaluation metric for feature selection. Denote $\mathscr{F}$ the set of selected features, $\mathbf{X}_{\mathscr{F}}$ the data represented by the features in $\mathscr{F}$. The redundancy rate of $\mathscr{F}$ [34] is defined as follows

$$RED(\mathscr{F}) = \frac{1}{|\mathscr{F}|\,(|\mathscr{F}| - 1)} \sum_{\boldsymbol{s}_i, \boldsymbol{s}_j \in F, i>j}^{corr_{i,j}}, \tag{32}$$

where $|\mathscr{F}|$ is the cardinality of $\mathscr{F}$, i.e., the number of selected feature, and $corr_{i,j}$ is the Pearson's correlation coefficient between two features $\mathbf{s}_i$ and $\mathbf{s}_j$, computed by using the data points in $\mathbf{X}_{\mathscr{F}}$. This measurement assesses the averaged correlation among all feature pairs in $\mathscr{F}$, and a large value indicates that many selected features are correlated and thus high redundancy is expected to exist in $\mathscr{F}$.

The variance is also an important feature selection evaluation metric used to reflect algorithm stability. The variance is defined as follows

$$var(\boldsymbol{x}) = E_{\boldsymbol{D}}[(f(\boldsymbol{x}; \boldsymbol{D}) - \bar{f}(\boldsymbol{x}))^2], \tag{33}$$

where $f(\boldsymbol{x};\boldsymbol{D})$ is the predictive output of function f to $\boldsymbol{x}$ learned in training set $\boldsymbol{D}$, and $\bar{f}(\boldsymbol{x})$ is expectation prediction of algorithm, i.e. $\bar{f}(\boldsymbol{x}) = E_{\boldsymbol{D}}[f(\boldsymbol{x}; \boldsymbol{D})]$.

Normalized mutual information (NMI) [35] is also a common metric for feature selection calculated by

$$NMI = \frac{MI(C, C')}{\max(\Gamma(C), \Gamma(C'))}, \tag{34}$$

where C is a set of the truth labels and C' is a set of predicted labels. $MI(C, C')$ is the mutual information metric, $\Gamma(C)$ and $\Gamma(C')$ are the entropies of C and C', respectively. A larger NMI means better performance.

In all experiments, we use the 1-nearest neighborhood (1-NN) classifier [1] to perform classification on the testing data with selected features. We randomly select a proper number of samples from each class as training samples and the remaining as testing samples, and perform feature selection on training samples. To avoid any bias, we report 10 runs to randomly select the training samples in all experiments.

B. CLASSIFICATION ACCURACY, REDUNDANCY RATE, VARIANCE AND NORMALIZED MUTUAL INFORMATION

First, we compare the classification accuracy of different methods. For the given six data sets, 5 image samples are randomly selected from each class as training samples, and the remaining as test samples. Since different data sets have different scales, the number of selected features in the experiments is different for different data sets. The total number of selected features for each data set is listed in Table 1.

Since the aim of feature selection is to find a compact representation, the number of selected features is generally limited to a small range, instead of displaying all features. Each feature selection algorithm is first performed on the training data set to determine the selected feature. Then, a classifier is trained with training samples containing only the selected features. And, the learned classifier is used to classify the testing samples with the selected features. Repeat this process 10 times, and the average classification accuracy is the results as shown in Fig. 3.

Fig. 3 depicts the case where the classification accuracies of different feature selection methods vary with the increase of the number of selected features. From this, we can get the following observations. For all data sets, with more selected features, the classification accuracies of all methods monotonously increase and can achieve higher values. And, DGRLR-SMR is superior to all other feature selection methods on all data sets in most cases. For example, on the USPS data set, DGRLR-SMR is improved by about 7% on average compared with the best results of other feature selection methods. On the ORL data set, DGRLR-SMR gets about 6% improvements in average. In addition, for different methods, the classification accuracy corresponding to the number of different training samples per class is shown in Fig. 4. Obviously, regardless of the number training samples in each class, DGRLR-SMR algorithm has higher classification accuracy than all other comparison algorithms. Note that in our experiments, we have observed that DGRLR-SMR algorithm is superior to other comparison methods on all given data sets, and these methods show similar performance relationships on these different data sets. Therefore, for the sake of simplification and saving page space, we only select some representative datasets in future experiments and give the experimental results of corresponding evaluation metric on these selected data sets.

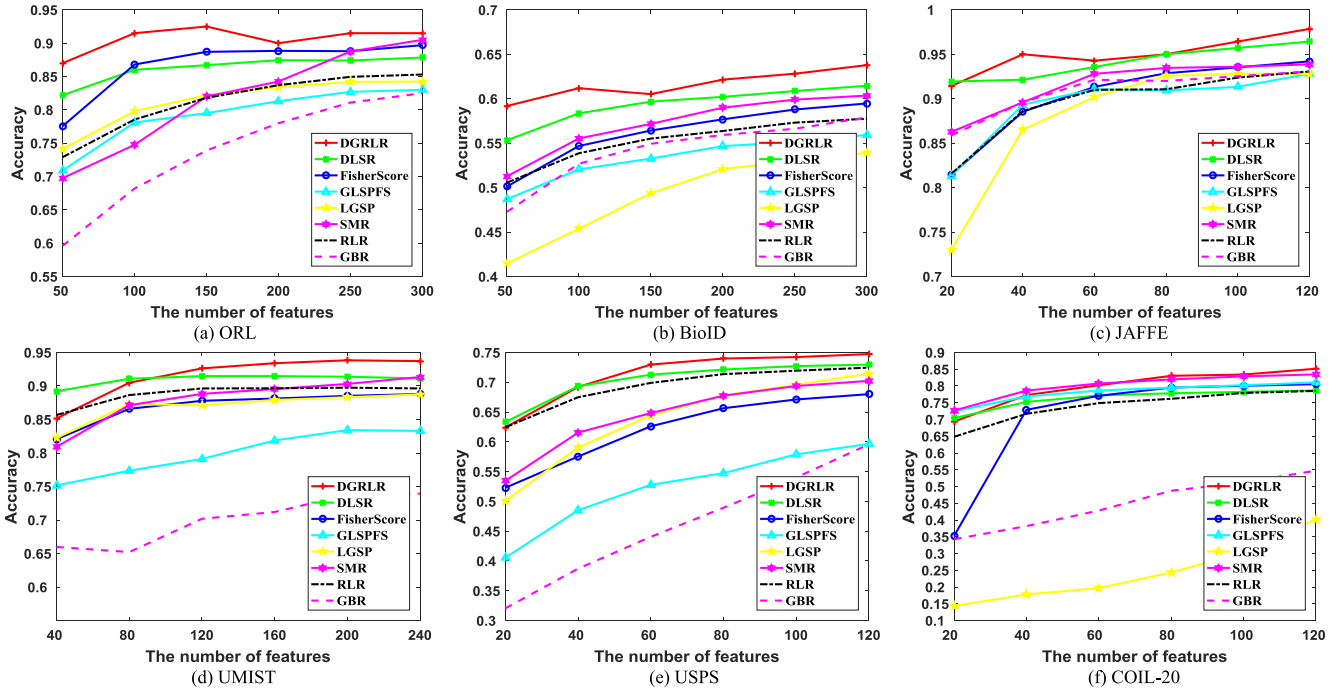


FIGURE 3. Classification accuracies of different methods on six data sets with different numbers of selected features.

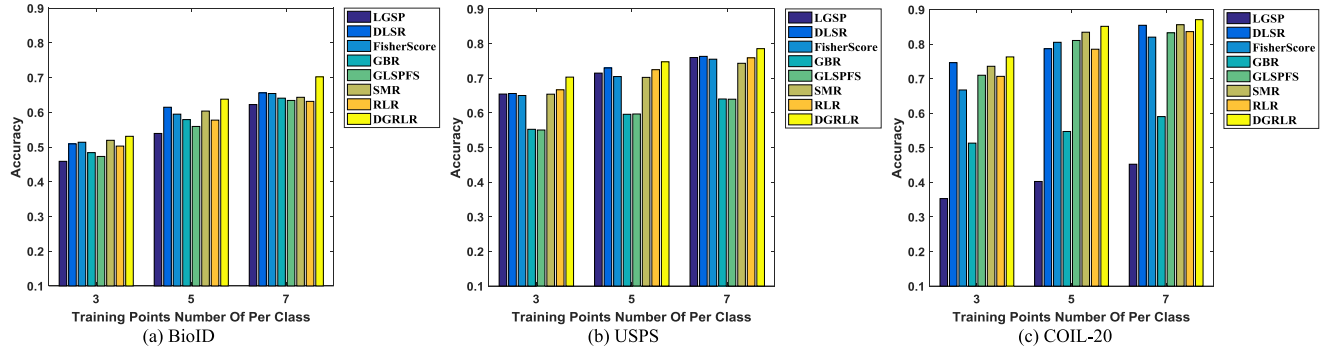


FIGURE 4. Classification accuracies of different methods on the BioID, the USPS and the COIL-20 data sets with different number of training samples per class.

We can also compare DGRLR-SMR algorithm with other seven algorithms on the noisy data sets. Here, we only consider the classification performance of the proposed method on noisy UMIST and COIL-20 data sets with various levels of contiguous occlusions and random pixel corruptions as follows:

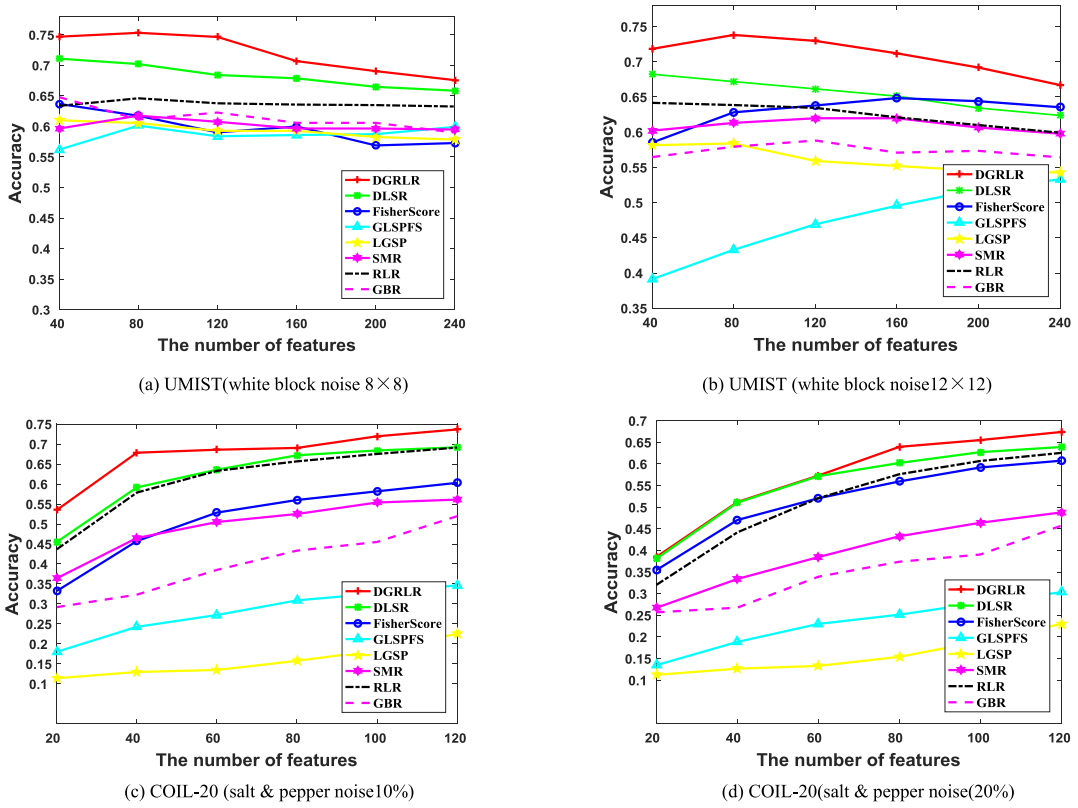
- *Contiguous Occlusions*: The block occlusions are randomly added to different location in each training image sample from the UMIST data set with the white block size of 8×8 and 12×12, respectively.
- *Random Pixel Corruptions:* Pixels are randomly selected from each training image sample in the COIL-20 data set and corrupted by salt & pepper noise. The rates of corrupted pixels are 10% and 20%, respectively.

Fig. 2 shows some of the corrupted images from UMIST and COIL-20 data sets. The classification accuracies of different methods are also shown in Fig. 5.

It can be seen from the Fig. 5 that on noisy UMIST and COIL-20 data sets, the classification accuracy of DGRLR-SMR is still higher than that of other methods. In other words, DGRLR-SMR outperforms all the other methods.

As we can see from Fig. 5(a) and (b), despite the increasing size of white block, DGRLR-SMR is still able to achieve better classification performance than other algorithms. Moreover, even if the density of salt and pepper noise increases, it can still keep best performance compared with other methods (see Fig. 5(c) and (d)), which also confirms the robustness of the DGRLR-SMR to noises. One of the main reasons is that DGRLR-SMR algorithm enlarges the margins between different classes in the projection space, so that it has good robustness.

Next, we will calculate the RED values between selected features obtained by different feature selection methods. For different training data sets, the number of selected



(a) UMIST(white block noise 8×8)

(b) UMIST (white block noise12×12)

(c) COIL-20 (salt & pepper noise10%)

(d) COIL-20(salt & pepper noise(20%)

FIGURE 5. Classification accuracies of different methods on two corrupted data sets.

features is also different. Denote $\mathscr{F}$ the most important s features selected by the related methods, and then calculate the RED value by (32). For each feature selection methods with different the number of selected features s, the average of 10 independent RED values is listed in Table 2. Here, we only report the results on the three representative data sets BioID UMIST and USPS, and the smallest values are boldfaced.

As the smaller the RED value is, the better the performance of feature selection is, so it can be seen from Table 2 that compared with other feature selection methods, the feature redundancies of DGRLR-SMR are the smallest, which also shows the effectiveness of DGRLR-SMR.

Then, we can compare the stability of all feature selection algorithms by their variance. The variance of different feature selection methods is calculated by the equation (33). For each feature selection method with the different number of selected features s, we calculate the average of 10 independent variance values. The results are listed in Table 3. Here, we only report the results on three representative data sets BioID, UMIST and USPS, and the smallest values are boldfaced.

As we all know, the smaller the variance is, the more stable the algorithm is, and thus the better its robustness is. The results in Table 3 show that compared with other feature selection algorithms, the variances of our DGRLR- SMR method are the smallest, which also indicates that our method is the most stable and robust.

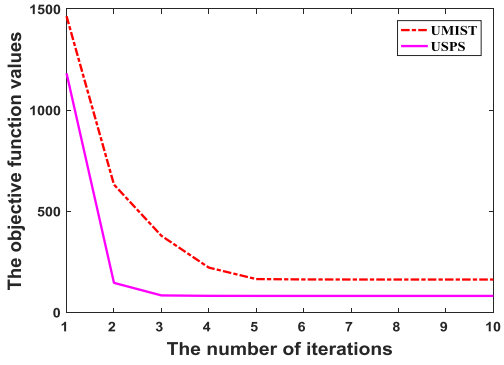


FIGURE 6. The convergences of DGRLR-SMR algorithm.

Finally, we will calculate the NMI values between selected features obtained by different feature selection methods. The value of NMI is calculated by (34). For each feature selection method with different the number of selected features s, the average of 10 independent NMI values is listed in Table 4. Here, we only report the results on three data sets BioID, UMIST and USPS, and the best values are boldfaced.

As the higher the NMI value is, the better the performance of feature selection is. Therefore, it can be seen from Table 4 that compared with other feature selection algorithms, the NMIs of DGRLR-SMR are relatively higher, which also indicates the high performance of DGRLR-SMR.

TABLE 2. Feature redundancies (RED) of different methods on BioID, UMIST and USPS data sets.

Datasets	s	FisherScor	GLSPFS	LGSP	DLSR	GBR	SMR	RLR	DGRLR-SMR
BioID	50	0.2428	0.1704	0.3678	0.1630	0.2605	0.2774	0.1556	**0.1509**
	100	0.2455	0.1663	0.3659	0.1702	0.2486	0.2741	0.1582	**0.1557**
	150	0.2509	0.1661	0.3636	0.1771	0.2468	0.2785	0.1638	**0.1612**
	200	0.2545	0.1680	0.3642	0.1809	0.2489	0.2764	0.1676	**0.1665**
	250	0.2582	0.1727	0.3633	0.1855	0.2507	0.2728	0.1716	**0.1713**
	300	0.2600	0.1782	0.3587	0.1907	0.2520	0.2704	0.1775	**0.1760**
Datasets	s	**FisherScor**	**GLSPFS**	**LGSP**	**DLSR**	**GBR**	**SMR**	**RLR**	**DGRLR-SMR**
UMIST	40	0.1482	0.1371	0.1885	0.1247	0.1418	0.0738	0.0752	**0.0714**
	80	0.1252	0.1081	0.1372	0.1052	0.1257	0.0830	0.0750	**0.0709**
	120	0.1100	0.0990	0.1163	0.0935	0.1207	0.0916	0.0732	**0.0714**
	160	0.1042	0.0957	0.1043	0.0841	0.1236	0.0951	0.0745	**0.0700**
	200	0.1004	0.0933	0.0970	0.0804	0.1209	0.0914	0.0751	**0.0702**
	240	0.0973	0.0912	0.0925	0.0783	0.1194	0.0889	0.0770	**0.0717**
Datasets	s	**FisherScor**	**GLSPFS**	**LGSP**	**DLSR**	**GBR**	**SMR**	**RLR**	**DGRLR-SMR**
USPS	20	0.1748	0.2244	0.1439	0.0725	0.1544	0.1061	0.0824	**0.0720**
	40	0.1274	0.1765	0.1056	0.0727	0.1550	0.0932	0.0739	**0.0694**
	60	0.1084	0.1586	0.0887	0.0737	0.1307	0.0846	0.0696	**0.0664**
	80	0.0966	0.1464	0.0829	0.0744	0.1155	0.0840	0.0687	**0.0636**
	100	0.0901	0.1343	0.0779	0.0778	0.1041	0.0822	0.0679	**0.0667**
	120	0.0846	0.1245	0.0760	0.0917	0.0910	0.0794	0.0656	**0.0626**

Based on the above experimental results and analysis, we can get some reasons why the proposed algorithms are superior to other comparison methods are as follows. Firstly, our methods take matrix as input and preserve the location information of elements in the original matrix. However, some other methods, such as FisherScore, LGSP, DLSR and RLR, are all vector-based. All these methods ignore the location information of pixels in image and convert directly the original matrix data into vectors before feature selection. As a result, they disrupt the location relationship between pixels in the original image, and only select features with discriminative power. And, for noisy images, the vector-based methods may select pixel points from the occlusion of noise. With the increase of selected feature number, more and more useless features (i.e. features from occlusion) are selected by the vector-based methods.

Secondly, several other contrast algorithms, such as SMR and Fisher Score algorithms, assume that the training samples should be accurately transform into a linear model or a strict binary label matrix. However, this assumption is too strict to accurately learn discriminant transformed matrix in the practical application. Our DGRLR-SMR algorithm adopts ε-dragging technology to enlarge the margin between different classes and get more discriminant transformation matrix. Thus the classification performance of the proposed algorithm is superior to other comparison algorithms.

Thirdly, instead of the fixed graph matrix defined by the heat kernel function in the manifold learning, our method can dynamically learn the graph matrix, which makes the similar samples to be close enough in the transformed space. By iteratively updating the graph sparse matrix and transformation matrix, the selected features can reveal the global structure of the samples and preserve their local structure. However, other some comparison algorithms, such as FisherScore, SMR and RLR, ignore the local relationship of similar samples, and do not fully consider the spatial structure and global structure of samples. Therefore, the performance of these methods is not as good as that of the proposed DGRLR-SMR algorithm.

C. CONVERGENCE

In order to prove the convergence of the proposed method, we further consider the variation of the objective function values of the proposed algorithm. Because the algorithm has similar convergence on different data sets, we only report the experimental results on UMIST and USPS data sets. Fig. 6 shows that the behavior of the objective function values of our proposed optimization algorithm with respect to the increase of the iterations. As can be seen from Fig. 6, the proposed algorithm to optimize the proposed objective functions in (10) or (11) monotonically decreases the objective function values until algorithm achieves converges. And, it needs a few iterations (i.e., less than 20 iterations). This also shows from another perspective that the proposed methods are efficient and effective.

D. PARAMETERS SETTINGS

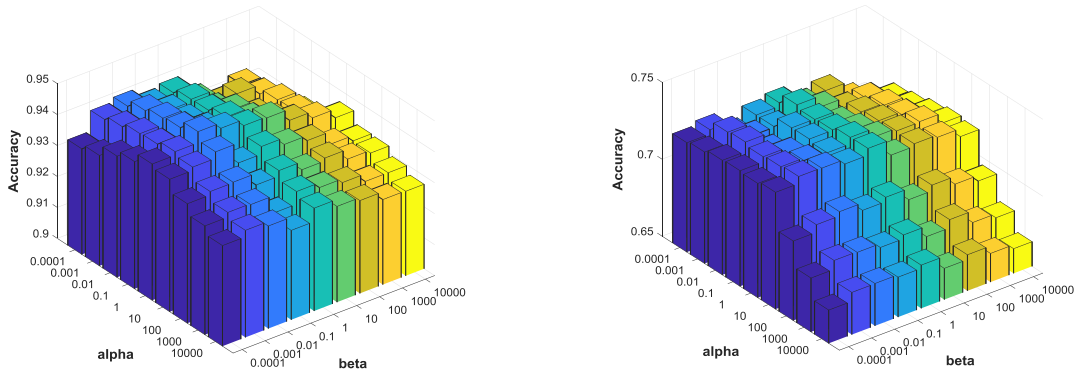
In this sub-section, we will analyze the parameters settings in DGRLR-SMR. In our proposed model (10) or (11), there

TABLE 3. The variances of different methods on BioID, UMIST and USPS data sets.

Datasets	*s*	FisherScor	GLSPFS	LGSP	DLSR	GBR	SMR	RLR	DGRLR-SMR
BioID	50	0.0007	0.0024	0.0007	0.0008	0.0009	0.0013	0.0008	**0.0002**
	100	0.0007	0.0023	0.0004	0.0007	0.0013	0.0005	0.0013	**0.0001**
	150	0.0007	0.0021	0.0007	0.0004	0.0011	0.0009	0.0009	**0.0004**
	200	0.0005	0.0016	0.0017	0.0006	0.0011	0.0009	0.0008	**0.0005**
	250	0.0006	0.0014	0.0005	0.0005	0.0009	0.0020	0.0008	**0.0005**
	300	0.0007	0.0013	0.0008	0.0003	0.0008	0.0021	0.0008	**0.0003**
Datasets	***s***	**FisherScor**	**GLSPFS**	**LGSP**	**DLSR**	**GBR**	**SMR**	**RLR**	**DGRLR-SMR**
UMIST	40	0.0009	0.0015	0.0007	0.0006	0.6299	0.0019	0.0010	**0.0004**
	80	0.0008	0.0010	0.0002	0.0007	0.4766	0.0014	0.0006	**0.0002**
	120	0.0009	0.0006	0.0004	0.0007	0.2442	0.0011	0.0007	**0.0004**
	160	0.0010	0.0008	0.0006	0.0007	0.2265	0.0014	0.0004	**0.0003**
	200	0.0011	0.0007	0.0005	0.0009	0.0644	0.0016	0.0004	**0.0003**
	240	0.0011	0.0007	0.0004	0.0006	0.9992	0.0013	0.0005	**0.0003**
Datasets	***s***	**FisherScor**	**GLSPFS**	**LGSP**	**DLSR**	**GBR**	**SMR**	**RLR**	**DGRLR-SMR**
USPS	20	0.0011	0.0020	0.0009	0.0006	0.0004	0.0015	0.0020	**0.0005**
	40	0.0006	0.0021	0.0006	0.0004	0.0005	0.0004	0.0015	**0.0003**
	60	0.0005	0.0006	0.0003	0.0003	0.0004	0.0002	0.0009	**0.0003**
	80	0.0005	0.0005	0.0005	0.0004	0.0007	0.0004	0.0006	**0.0003**
	100	0.0003	0.0005	0.0007	0.0003	0.0005	0.0004	0.0005	**0.0003**
	120	0.0005	0.0004	0.0011	0.0004	0.0005	0.0004	0.0005	**0.0004**

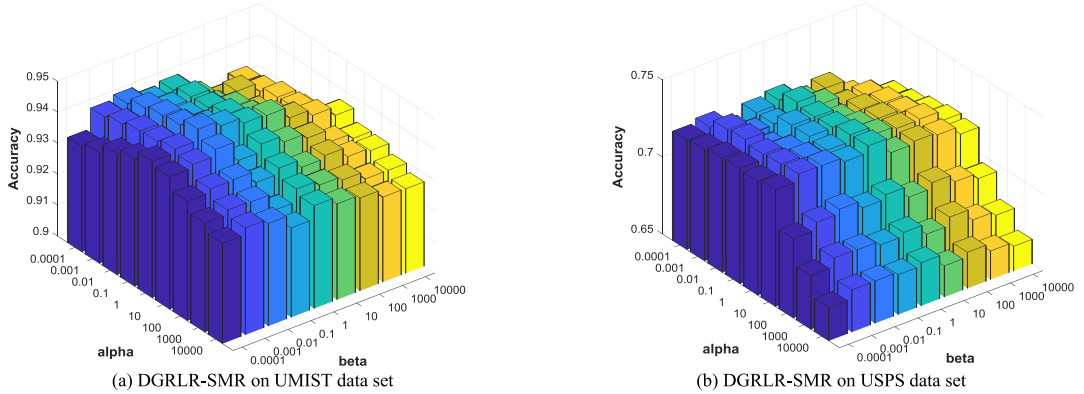
TABLE 4. The NMI of different methods on BioID, UMIST and USPS data sets.

Datasets	*s*	FisherScor	GLSPFS	LGSP	DLSR	GBR	SMR	RLR	DGRLR-SMR
BioID	50	0.4441	0.4162	0.4040	0.4915	0.4707	0.4323	0.4545	**0.4937**
	100	0.4797	0.4627	0.4308	0.5167	0.4787	0.4343	0.4639	**0.5289**
	150	0.4960	0.4786	0.4445	0.5127	0.4974	0.4573	0.4773	**0.5365**
	200	0.4968	0.4909	0.4476	0.5224	0.5077	0.4551	0.4894	**0.5301**
	250	0.5115	0.4945	0.4600	0.5279	0.5051	0.4572	0.4983	**0.5296**
	300	0.5134	0.4951	0.4730	0.5294	0.5140	0.4539	0.5043	**0.5366**
Datasets	***s***	**FisherScor**	**GLSPFS**	**LGSP**	**DLSR**	**GBR**	**SMR**	**RLR**	**DGRLR-SMR**
UMIST	40	0.5504	0.5863	0.5880	**0.7002**	0.5637	0.5715	0.6725	0.6975
	80	0.5306	0.6043	0.5842	0.5874	0.5432	0.5900	0.6589	**0.6904**
	120	0.5542	0.6220	0.5610	0.5074	0.5678	0.6193	0.6463	**0.6994**
	160	0.5572	0.6160	0.5734	0.4695	0.5726	0.6350	0.6405	**0.7005**
	200	0.5709	0.6261	0.5722	0.4821	0.5815	0.6458	0.6528	**0.6685**
	240	0.5742	0.6287	0.5739	0.4758	0.5740	0.6380	0.6452	**0.6598**
Datasets	***s***	**FisherScor**	**GLSPFS**	**LGSP**	**DLSR**	**GBR**	**SMR**	**RLR**	**DGRLR-SMR**
USPS	20	0.4043	0.2052	0.3586	**0.4807**	0.2461	0.3868	0.4131	0.4230
	40	0.4644	0.2278	0.3968	0.5081	0.2813	0.4406	0.5011	**0.5159**
	60	0.4959	0.2502	0.4623	0.5419	0.3023	0.4720	0.5438	**0.5498**
	80	0.5290	0.2833	0.4856	0.5556	0.3134	0.5089	0.5426	**0.5646**
	100	0.5442	0.2913	0.5223	0.5676	0.3503	0.5441	0.5298	**0.5719**
	120	0.5563	0.3484	0.5404	0.5666	0.4049	0.5489	0.5485	**0.5695**



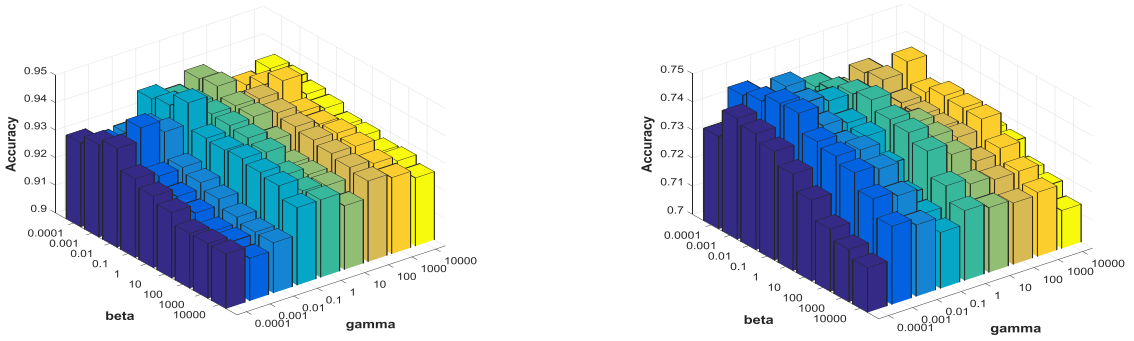
(a) DGRLR-SMR on UMIST data set (b) DGRLR-SMR on USPS data set

FIGURE 7. ACA results of proposed method with different parameter combinations of α and β.



(a) DGRLR-SMR on UMIST data set (b) DGRLR-SMR on USPS data set

FIGURE 8. ACA results of proposed method with different parameter combinations of α and γ.



(a) DGRLR-SMR on UMIST data set (b) DGRLR-SMR on USPS data set

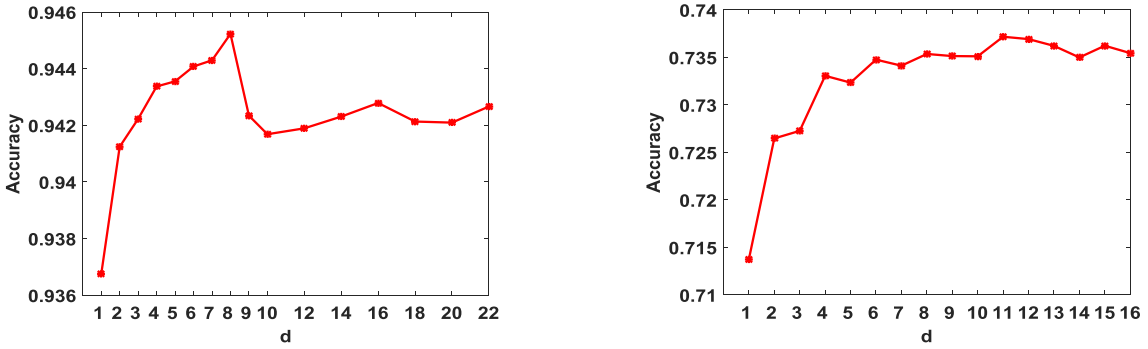
FIGURE 9. ACA results of proposed method with different parameter combinations of β and γ.

are two types of parameters to dominate their performance: three regularization parameters α, β and γ; the number of regression vectors d. The parameter α is used to adjust the sparsity of the transformation matrix $\mathbf{P}$, β is used to control trade-off between the loss function and the dynamic graph local preserving, and γ is used to avoid over-fitting when the dynamic graph is found. Therefore, parameters α, β and γ balance the effectiveness of regression and feature selection.

Because it is still a challenge to search a proper parameter combination for specific data set, we will run the proposed our algorithm on UMIST and USPS data sets with different parameters combinations by two dimensional grid searches, respectively, and empirically determine and choose the best one. Fig. 7–9 show the variation of the average classification accuracies (ACA) of the proposed method with different combinations of parameters on UMIST and USPS data sets respectively. Obviously, with the increase of α and β (or α

TABLE 5. The values of parameters in different methods on BioID, UMIST and USPS datasets.

Datasets	Parameters	FisherScor	GLSPFS	LGSP	GBR	DLSR	RLR	SMR	DGRLR-SMR
BoiID	α	/	1	0.01	/	1	1	10	0.01
	β	/	10	0.1	/	1000	/	/	0.001
	γ	/	/	100	/	/	/	/	0.1
	d	/	/	/	/	/	/	15	15
Datasets	**Parameters**	**FisherScor**	**GLSPFS**	**LGSP**	**GBR**	**DLSR**	**RLR**	**SMR**	**DGRLR-SMR**
UMIST	α	/	1	1	/	1	0.1	1000	1
	β	/	1	100	/	1000	/	/	0.01
	γ	/	/	1	/	/	/	/	0.1
	d	/	/	/	/	/	/	11	8
Datasets	**Parameters**	**FisherScor**	**GLSPFS**	**LGSP**	**GBR**	**DLSR**	**RLR**	**SMR**	**DGRLR-SMR**
USPS	α	/	1	1	/	1	0.1	1	10
	β	/	1	10	/	1000	/	/	0.1
	γ	/	/	0.1	/	/	/	/	0.1
	d	/	/	/	/	/	/	7	11



(a) DGRLR-SMR on UMIST data set (b) DGRLR-SMR on USPS data set

FIGURE 10. ACA of our proposed method with different numbers of regression vectors *d*.

and γ, or β and γ), the classification accuracy of our method on UMIST and USPS data sets will increase first and then decreases. Table 5 lists the specific values of three parameters α, β and γ when each method achieves the highest classification accuracy.

Another parameter d in our model is the number of left and right regression vectors, which could play an important role in balancing the capacity of learning and generalization for the regression models. To verify the effectiveness of multiple regression vectors, we will calculate the classification accuracies of DGRLR-SMR with fixed $\{\alpha, \beta, \gamma\}$ and different d on the UMIST and USPS data sets. In our experiments, it assumes that d varies within the range $[1, \min(m, n)]$. The variation curve of classification accuracy of our method is shown in Fig. 10. Obviously, with the increase of parameter d, classification accuracy does not always increase consistently. For the UMIST data set, when the number of regression vectors d is small (e.g., less than 8), the classification accuracy will increase with the increase of d. After the value of d increases to a certain value (e.g., 8), the accuracy will fluctuate and decrease. For the USPS data set, when the number of regression vectors d is less than 11, the classification accuracy will fluctuate with the increase of d. After the value of d increases to 11, the accuracy will fluctuate and decrease. Therefore, it is not that the more the number of left and right regression vectors, the better the classification performance. As a result, we should choose the appropriate value of d to make the algorithm achieve the best performance in real applications. The value of parameter d that each method achieves its highest classification accuracy in our experiments is listed in Table 5.

V. CONCLUSION

In this paper, we have proposed the dynamic graph regularized label relaxation sparse matrix regression model (DGRLR-SMR), which joins the dynamic graph learning with the label relaxation sparse matrix regression to perform feature selection for two-dimensional matrix data. First, comparing with the traditional vector-based supervised methods for feature selection, DGRLR-SMR can directly use the matrix data as input data and select the discriminative features on the matrix data. Thus, the proposed method considers the location information of elements in the original image data. Secondly, we also dynamically learn the graph matrix to uncover the discriminative information and the local geometric structure of the image samples. Thus, DGRLR-SMR not only relaxes the strict binary label matrix into a slack variable matrix by introducing a nonnegative label relaxation

matrix, but also obtains a sparse transformed matrix (generated by several pairs of left and right regression matrices) and avoids the problem of over-fitting. Therefore, we can couple the graph matrix learning with the low dimensional space learning together to iteratively optimize them so that achieving their individually optimal result. We have analyzed the computational complexity and convergence of our proposed algorithm. Numerical experiments results on several data sets indicate that this algorithm outperform the state-of-the-art algorithms in term of classification accuracy, the redundancy rate, the variance and the normalized mutual information.

REFERENCES

[1] D. G. Stork, P. E. Hart, and R. O. Duda, *Pattern Classification*. Hoboken, NJ, USA: Wiley, 2000.
[2] A. Jain and D. Zongker, "Feature selection: Evaluation, application, and small sample performance," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 19, no. 2, pp. 153–158, Feb. 1997.
[3] Y. Fu, S. Yan, and T. S. Huang, "Correlation metric for generalized feature extraction," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 30, no. 12, pp. 2229–2235, Dec. 2008.
[4] J. Yang, A. F. Frangi, J.-Y. Yang, D. Zhang, and Z. Jin, "KPCA plus LDA: A complete kernel Fisher discriminant framework for feature extraction and recognition," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 27, no. 2, pp. 230–244, Feb. 2005.
[5] X. Liu, L. Wang, J. Zhang, J. Yin, and H. Liu, "Global and local structure preservation for feature selection," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 25, no. 6, pp. 1083–1095, Jun. 2014.
[6] X. Zhu, S. Zhang, R. Hu, Y. Zhu, and J. Song, "Local and global structure preservation for robust unsupervised spectral feature selection," *IEEE Trans. Knowl. Data Eng.*, vol. 30, no. 3, pp. 517–529, Mar. 2018.
[7] F. Nie, H. Huang, X. Cai, and C. Ding, "Efficient and robust feature selection via joint $\ell 2$, 1-norms minimization," in *Proc. NIPS*, 2010, pp. 1813–1821.
[8] J. Liu, S. Ji, and J. Ye, "Multi-task feature learning via efficient -norm minimization," in *Proc. 21th Conf. Uncertainty Artif. Intell.*, Montreal, QC, Canada, Jun. 2009, pp. 339–348.
[9] X. Cai, F. Nie, and H. Huang, "Exact top-k feature selection via -norm constraint," in *Proc. 23th Int. Joint Conf. Artif. Intell.*, 2013, pp. 1240–1246.
[10] S. Xiang, F. Nie, G. Meng, C. Pan, and C. Zhang, "Discriminative least squares regression for multiclass classification and feature selection," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 23, no. 11, pp. 1738–1754, Nov. 2012.
[11] R. He, T. Tan, L. Wang, and W.S. Zheng, "$\ell 2$, 1 Regularized correntropy for robust feature selection," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2012, pp. 2504–2511.
[12] C. Hou, Y. Jiao, F. Nie, T. Luo, and Z.-H. Zhou, "2D feature selection by sparse matrix regression," *IEEE Trans. Image Process.*, vol. 26, no. 9, pp. 4255–4268, Sep. 2017.
[13] H. Yuan, J. Li, L. L. Lai, and Y. Y. Tang, "Joint sparse matrix regression and nonnegative spectral analysis for two-dimensional unsupervised feature selection," *Pattern Recognit.*, vol. 89, pp. 119–133, May 2019.
[14] J. Łęski, "Ho–Kashyap classifier with generalization control," *Pattern Recognit. Lett.*, vol. 24, no. 14, pp. 2281–2290, Oct. 2003.
[15] X. Li, S. Lin, S. Yan, and D. Xu, "Discriminant locally linear embedding with high-order tensor data," *IEEE Trans. Syst., Man, Cybern., B (Cybern.)*, vol. 38, no. 2, pp. 342–352, Apr. 2008.
[16] S. Yan, D. Xu, B. Zhang, H.-J. Zhang, Q. Yang, and S. Lin, "Graph embedding and extensions: A general framework for dimensionality reduction," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 29, no. 1, pp. 40–51, Jan. 2007.
[17] Z. Fan, Y. Xu, and D. Zhang, "Local linear discriminant analysis framework using sample neighbors," *IEEE Trans. Neural Netw.*, vol. 22, no. 7, pp. 1119–1132, Jul. 2011.
[18] M. Sugiyama, "Dimensionality reduction of multimodal labeled data by local Fisher discriminant analysis," *J. Mach. Learn. Res.*, vol. 8, pp. 1027–1061, May 2007.
[19] D. Cai, X. He, K. Zhou, J. Han, and H. Bao, "Locality sensitive discriminant analysis," in *Proc. 20th Int. Joint Conf. Artif. Intell.*, Jan. 2007, pp. 708–713.
[20] H.-T. Chen, H.-W. Chang, and T.-L. Liu, "Local discriminant embedding and its variants," in *Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2005, pp. 846–853.
[21] X. Fang, Y. Xu, X. Li, Z. Lai, W. K. Wong, and B. Fang, "Regularized label relaxation linear regression," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 29, no. 4, pp. 1006–1018, Apr. 2018.
[22] Z. Liu, J. Wang, G. Liu, and L. Zhang, "Discriminative low-rank preserving projection for dimensionality reduction," *Appl. Soft Comput.*, vol. 85, Dec. 2019, Art. no. 105768.
[23] Z. Lai, W. K. Wong, Y. Xu, J. Yang, and D. Zhang, "Approximate orthogonal sparse embedding for dimensionality reduction," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 27, no. 4, pp. 723–735, Apr. 2016.
[24] Z. Liu, Z. Lai, W. Ou, K. Zhang, and R. Zheng, "Structured optimal graph based sparse feature extraction for semi-supervised learning," *Signal Process.*, vol. 170, May 2020, Art. no. 107456.
[25] J. Wen, Y. Xu, and H. Liu, "Incomplete multiview spectral clustering with adaptive graph learning," *IEEE Trans. Cybern.*, vol. 50, no. 4, pp. 1418–1429, Apr. 2020.
[26] J. Wen, X. Fang, Y. Xu, C. Tian, and L. Fei, "Low-rank representation with adaptive graph regularization," *Neural Netw.*, vol. 108, pp. 83–96, Dec. 2018.
[27] K. Gabriel, "Generalised bilinear regression," *Biometrika*, vol. 85, no. 3, pp. 689–700, Sep. 1998.
[28] C. Hou, F. Nie, D. Yi, and Y. Wu, "Efficient image classification via multiple rank regression," *IEEE Trans. Image Process.*, vol. 22, no. 1, pp. 340–352, Jan. 2013.
[29] P. N. Belhumeur, J. P. Hespanha, and D. J. Kriegman, "Eigenfaces vs. Fisherfaces: Recognition using class specific linear projection," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 19, no. 7, pp. 711–720, Jul. 1997.
[30] O. Jesorsky, K. J. Kirchberg, and R. W. Frischholz, "Robust face detection using the Hausdorff distance," in *Proc. Int. Conf. Audio-Video-Based Biometric Person Authentication*, 2001, pp. 90–95.
[31] M. J. Lyons, J. Budynek, and S. Akamatsu, "Automatic classification of single facial images," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 21, no. 12, pp. 1357–1362, Dec. 1999.
[32] J. J. Hull, "A database for handwritten text recognition research," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 16, no. 5, pp. 550–554, May 1994.
[33] S. Nene, S. Nayar, and H. Murase, "Columbia object image library (COIL-20)," Dept. Comput. Sci., Columbia Univ., New York, NY, USA, Tech. Rep. CUCS-006-96, 1996.
[34] Z. Zhao, F. Morstatter, S. Sharma, S. Alelyani, A. Anand, and H. Liu, "Advancing feature selection research-ASU feature selection repository," School Comput. Informat., Decis. Syst. Eng., Arizona State Univ., Tempe, AZ, USA, Tech. Rep. TR-10-007, 2010.
[35] K. Fan, "On a theorem of Weyl concerning eigenvalues of linear transformations," *Proc. Nat. Acad. Sci. USA*, vol. 35, no. 11, pp. 652–655, 1949.

XIUHONG CHEN received the Ph.D. degree in applied mathematics from the East China University of Science and Technology, in 2000. From 2001 to 2006, he was a Postdoctoral Fellow with the Department of Mathematics, Nanjing University, and the School of Computer Science, Nanjing University of Science and Technology. He is currently a Professor with the School of Digital Media, Jiangnan University, Wuxi, China. His research interests include image processing, pattern recognition and intelligence computation, object detection and tracking, optimization theory and method, and so on.

YUN LU received the bachelor's degree in software engineering from the Changshu Institute of Technology. She is currently pursuing the master's degree with the School of Digital Media, Jiangnan University. Her current research interests include pattern recognition and image processing.