Yokohama Publishers
ISSN 1349-8169 ONLINE JOURNAL

Pacific Journal of Optimization
Vol. 18, No. 1, pp. 155–176, 2022

A SUBSPACE ELIMINATION STRATEGY FOR ACCELERATING SUPPORT MATRIX MACHINE*

Renxiu Feng, Peiwei Zhong and Yitian Xu†

Abstract: Support matrix machine (SMM) is an effective method for classification problems with matrix-form. However, it is time-consuming to solve it, since the complexity increases quickly with the size of matrix variable. Currently, screening methods can efficiently improve the computational speed of SVM models, however they are not applicable to SMM because the SMM has nuclear norm regularization and cannot be expressed as a vector norm. To deal with this issue, in this paper, we introduce a support matrix machine based on squared hinge loss (L_2-SMM), which employs a smoothed substitution formula. It allows the solver to be accelerated without significantly affecting performance. Additionally, we construct a subspace elimination strategy for L_2-SMM (SES-L_2-SMM) to expedite the speed of its training process. To the best of our knowledge, this is the first attempt to introduce a subspace elimination strategy to SMMs models. At each step, an active subspace is chosen so that we can solve a lower-dimensional optimization problem. Later, we use the alternating direction method of multipliers (ADMM) algorithm to resolve the problem. Extensive comparative experiments on multiple real-world datasets demonstrate the efficacy of SES-L_2-SMM.

Key words: *support matrix machine, subspace elimination strategy, nuclear norm regularization, ADMM*

Mathematics Subject Classification: *74P99, 90C06, 90C25*

1 Introduction

Support vector machine (SVM), due to the solid theoretical foundation of statistics and great generalization performance [31, 32], has been successfully applied in various fields, such as object recognition [26, 28], biomedicine [14], text classification [20], etc [17, 19, 38]. Though the conventional SVM can get small generalization error in the case of normal binary classification, it does not perform well when coping with the problem of matrix-form data, where one matrix (second-order array) is constructed as one sample. In many machine learning problems, the samples can be directly represented as matrices, like digital images. To fit the SVM models, we must reshape the matrix-form features of each sample into a vector. In such a reshaping process, the structural information embedded in the feature matrix will be ignored. Another disadvantage of vectorization is the dimensionality curse problem [5]. Taking the digital images as an example, if each sample is a gray-scale image with 32×32 pixels, the dimension of each vectorization sample is exceedingly large as 1024, which may cause the over-fitting problem for small samples.

*This work was supported in part by National Natural Science Foundation of China (No. 12071475, 11671010) and Beijing Natural Science Foundation (No. 4172035).

†Corresponding author

To address the above mentioned issues, various modifications of SVM for matrix-form data classification have been presented. Wolf et al. [35] designed rank-k SVM by regularizing the regression matrix into the sum of k rank-one orthogonal matrices, thus catching the holistic feature of matrix data. The bilinear SVM (BSVM) [27] decomposed regression matrix into the product of two low-rank matrices. Although the above algorithms can hold the structural information of the rows and columns in input matrices, the rank of the regression matrix needs to be determined beforehand, which makes it difficult to adjust the parameters to the desired result. By using the nuclear norm of the matrix as the convex substitution of the matrix rank, one [16] could automatically figure out the rank of the regression matrix. There are other improved models [6,8,18,39]. Specifically, Luo et al. [18] represented an innovative support matrix machine (named as SMM) through the use of hinge loss, Frobenius norm, and nuclear norm, which can obtain structural information from the input matrices. By comparison with other matrix classification algorithms, SMM is more suitable for image classification and electroencephalogram (EEG) classification. The alternating direction method of multipliers (ADMM) algorithm can effectively solve SMM. Nevertheless, in numerous practical applications, the scale of matrix variables in the era of big data is increasing. It is still a challenging task to solve the problem of large-scale matrix variables because the computational cost increases quickly with the size of the matrix variables. Consequently, for large-scale matrix variables datasets, degrading the training cost of SMM is a matter worthy of research.

Ghaoui et al. [10] developed a safe screening method to optimize Lasso, which distinguishes features that are homologous to zero coefficients and eliminates them before solving the problem. Thus we only need to solve a smaller problem and accelerate the computing speed. So far, there are many improved algorithms [2,7,22,23,29]. Furthermore, safe screening rule was first introduced into SVM by Ogawa et al. [24]. It aims to identify non-support vectors and preassign them beforehand. Motivated by that idea, there are also various improvement algorithms [25, 33] and it has been widely adopted to other models [37, 42]. However, none of these methods are suitable for matrix-shape datasets. Thus, the SMM cannot use the above-mentioned methods to speed up the training process. Fortunately, by imposing low-rank constraints on the regression matrix, the SMM can be considered a problem for minimizing the nuclear norm. While a great deal of work [15,21] has been done to develop effectual nuclear norm minimization solvers, the vast majority of them are still unable to solve large-scale problems. Although Yao et al. [36] proposed an effective algorithm to solve large-scale matrix completion problems, SMM cannot be directly equivalent to the problem. Recently, inspired by the idea that the nuclear norm is equivalent to the l_1-norm on singular values, Hsieh et al. [13] represented an active subspace selection method, which identifies a small active subspace and efficiently minimizes the reduced-sized nuclear norm minimization problem.

Motivated by the above studies, we first introduce the squared hinge loss-based support matrix machine (L_2-SMM), which is smooth and allows for better optimization. Subsequently, we develop a subspace elimination strategy for accelerating L_2-SMM (SES-L_2-SMM) to reduce the computational cost. The SES-L_2-SMM is presented by analyzing the regression matrix composed of the sum of rank one matrices. L_2-SMM implements low rank constraint of matrix by adopting nuclear norm regularization. The low rank of the matrix can be represented by the sparsity of the subspace. Therefore, we can identify the zero weight and eliminate the corresponding subspace. Thus we are able to obtain a lower-dimensional matrix variable, and accordingly, the scale of the L_2-SMM is reduced which is less expensive to optimize. Moreover, the SES is independent from the solver because it is implemented before solving the optimization problem. Therefore, the different efficient solvers can be

combined. We construct an alternating direction method of multipliers (ADMM) as our efficient solver.

The main contributions of our approach are summarized as follows:

(i) We propose a SES-L_2-SMM via sparse representation of the matrix to efficiently deal with the problem of large-scale matrix-variables.

(ii) To the best of our knowledge, this is the first attempt to apply the subspace elimination strategy to L_2-SMM.

(iii) The SES is independent from the solver as it is applied before solving L_2-SMM model. Hence, other efficient solvers can be combined. In this work, we construct the alternating direction method of multipliers (ADMM) as the efficient solver.

This paper is constructed as follows: Section 2 introduces the basic idea of subspace elimination strategy and reviews the basic concepts of SMM and L_2-SMM. In Section 3, details of SES for L_2-SMM are introduced. The calculation of reducing L_2-SMM is given in Section 4. Section 5 carries out numerical experiments on three real-world datasets to validate the efficacy of the presented algorithm. The last section draws some conclusions.

Notations: In this paper, scalar, vector, and matrix are represented by lowercase letter (e.g. w), lowercase bold letter (e.g. $\mathbf{w}$), and uppercase bold letter (e.g. $\mathbf{W}$), respectively. For a matrix $\mathbf{W} = [\mathbf{W}_{ij}] \in \mathbb{R}^{p\times q}$, its Frobenius norm is $\|\mathbf{W}\|_F = \sqrt{\mathrm{tr}(\mathbf{W}^T\mathbf{W})} = \sqrt{\sum_{i,j}\mathbf{W}_{ij}^2}$, nuclear norm is $\|\mathbf{W}\|_* = \sum_i \sigma_i(\mathbf{W})$, where $\sigma_i(\mathbf{W})$ is the singular value. $\|\cdot\|$ denotes the 2-norm of a vector or matrix. For two matrices $\mathbf{X}$, $\mathbf{W}$, the matrix trace operation is $\mathrm{tr}(\mathbf{W}^T\mathbf{X}) = \sum_i^p \sum_{j=1}^q \mathbf{W}_{ij}\mathbf{X}_{ij}$. For a smooth function g, ∇g denotes its gradient. The training data $\mathcal{T} = \{\mathbf{X}_i, y_i\}_{i=1}^n$, where $\mathbf{X}_i \in \mathbb{R}^{p\times q}$ and $y_i \in \{-1, 1\}$, contains n samples and each sample with $p \times q$ features. $\mathbf{X}_i$ is the i-th sample and y_i is the corresponding class label.

2 Preliminaries

In this section, we provide some basic knowledge of this paper, including the SMM model, its important properties and the central idea of subspace selection.

2.1 Support matrix machine with hinge loss

The structure information embedded in matrix data will inevitably be ignored by vectorizing the matrix-form data. Visually, we take the following formula into consideration:

$$\min_{\mathbf{W},b} \quad \frac{1}{2}\|\mathbf{W}\|_F^2 + C\sum_{i=1}^n \{1 - y_i[\mathrm{tr}(\mathbf{W}^T\mathbf{X}_i) + b]\}_+, \tag{2.1}$$

here, $\mathbf{W} \in \mathbb{R}^{p\times q}$ is the matrix of regression coefficients. Nonetheless, from the viewpoint of computation, when $\mathbf{w} = vec(\mathbf{W})$ and $\mathbf{x}_i = vec(\mathbf{X}_i)$, since $\|\mathbf{W}\|_F^2 = \mathrm{tr}(\mathbf{W}^T\mathbf{W}) = \mathbf{w}^T\mathbf{w} = \|\mathbf{w}\|^2$ and $\mathrm{tr}(\mathbf{W}^T\mathbf{X}_i) = \mathbf{w}^T\mathbf{x}_i$, Eq. (2.1) is essentially equivalent to the model of classical SVM. The method of directly executing classification algorithm with (2.1) cannot efficiently acquire the underlying structure of matrices efficiently.

To eliminate the adverse effects of vectorization, Luo *et al.* [18] developed an innovative algorithm called support matrix machine (SMM), which consists of the hinge loss and the *spectral elastic net* penalty are shown below:

$$\min_{\mathbf{W},b} \quad \frac{1}{2}\|\mathbf{W}\|_F^2 + \tau\|\mathbf{W}\|_* + C\sum_{i=1}^n \{1 - y_i[\mathrm{tr}(\mathbf{W}^T\mathbf{X}_i) + b]\}_+, \tag{2.2}$$

here, $\mathrm{tr}(\cdot)$ is the matrix trace operation, b is the bias term, and $\mathbf{W} \in \mathbb{R}^{p\times q}$ is the regression matrix. $\frac{1}{2}\|\mathbf{W}\|_F^2 + \tau\|\mathbf{W}\|_*$ is called *spectral elastic net* because $\|\mathbf{W}\|_F^2 = \sum_i \sigma_i(\mathbf{W})^2$ and $\|\mathbf{W}\|_* = \sum_i \sigma_i(\mathbf{W})$. According to [18], the spectral elastic net is analogous to the elastic net of [43]. τ and C are positive scalars chosen beforehand. Because of the properties of grouping effect of the spectral elastic net and the nature of low rank of the nuclear norm, SMM can powerfully catch the internal structural information within matrix-form data.

2.2 Subspace elimination strategy

The subspace elimination strategy (SES) is based on the idea of sparse representation of the matrix. Its main idea is as follows:

Generally, matrix $\mathbf{W} \in \mathbb{R}^{p\times q}$ can be presented as the sum of rank one matrices

$$\mathbf{W} = \sum_{i=1}^{p}\sum_{j=1}^{q} \mathbf{\Theta}_{ij}\mathbf{u}_i\mathbf{v}_j^T, \tag{2.3}$$

where $\mathbf{\Theta} \in \mathbb{R}^{p\times q}$, $\{\mathbf{u}_i \in \mathbb{R}^p\}_{i=1}^p$ and $\{\mathbf{v}_j \in \mathbb{R}^q\}_{j=1}^q$ are orthogonal bases in $\mathbb{R}^{p\times p}$ and $\mathbb{R}^{q\times q}$, respectively. The reason we impose the nuclear norm on regression matrix is that the regression matrix is assumed low-rank representable. On the other hand, the low rank of the matrix can be represented by the sparsity of the subspace. Thus the matrix $\mathbf{\Theta}$ is sparse. In other words, it will have numerous elements that are equal to zero. Consequently, the purpose of SES is to distinguish the set $\{\mathbf{u}_i\mathbf{v}_j^T|\mathbf{\Theta}_{ij} = 0\}$, namely inactive subspaces, before solving the final solution. Then we can only tackle the dimension reduction problem composed of the remaining rank one subspaces, which correspond to $\{\mathbf{u}_i\mathbf{v}_j^T|\mathbf{\Theta}_{ij} \neq 0\}$, namely active subspaces. Later, the scaled problem can be efficiently resolved.

3 Subspace Elimination Strategy for L_2-SMM

In this section, we first develop the L_2-SMM, then construct the SES procedure for L_2-SMM, and later obtain the reduction problem.

3.1 Support matrix machine with squared hinge loss function

Similar to SVM with the squared hinge loss function, we change the hinge loss to squared hinge loss (termed as L_2-SMM) to make it smooth for efficient optimization. Furthermore, the L_2-SMM is employed for subsequent search. Then, the formula of L_2-SMM is as follows:

$$\min_{\mathbf{W},b} \qquad \frac{1}{2}\|\mathbf{W}\|_F^2 + \tau\|\mathbf{W}\|_* + C\sum_{i=1}^{n}\max\{0, 1 - y_i[\mathrm{tr}(\mathbf{W}^T\mathbf{X}_i) + b]\}^2. \tag{3.1}$$

To avoid puzzlement, we follow the symbols in problem (2.2).

3.2 SES-L_2-SMM

We need to search a set $\{\mathbf{u}_i\mathbf{v}_i|\mathbf{\Theta}_{ij} = 0\}$ according to the basic idea of SES. Assume the optimal solution to problem (3.1) is $\mathbf{W}^* = \sum_{i=1}^p\sum_{j=1}^q \mathbf{\Theta}_{ij}^*\mathbf{u}_i\mathbf{v}_j^T$, if there exists a particular subspace $\mathbf{u}_i\mathbf{v}_j$, the value of $\mathbf{\Theta}_{ij}^*$ will be 0 if and only if

$$\mathbf{u}_i^T\mathbf{W}^*\mathbf{v}_j = 0, \tag{3.2}$$

since $\{\mathbf{u}_i\mathbf{v}_j, i = 1, 2, \cdots, p, \quad j = 1, 2, \cdots, q\}$ are orthogonal to each other. Then we have

$$\mathbf{u}_i^T \mathbf{U}\mathbf{V}^T \mathbf{v}_j = 0. \tag{3.3}$$

The problem (3.1) can be represented as a function with respect to $\mathbf{\Theta}_{ij}$ by the form (2.3). We first give the symbol shorthand for facilitating notation.

$$\begin{aligned} G(\mathbf{W}, b) &= h(\mathbf{W}, b) + \tau\|\mathbf{W}\|_*, \\ h(\mathbf{W}, b) &= \frac{1}{2}\|\mathbf{W}\|_F^2 + C\sum_{i=1}^{n} \max\{0, 1 - y_i[\mathrm{tr}(\mathbf{W}^T\mathbf{X}_i) + b]\}^2. \end{aligned}$$

Then, two Lemmas for the computation of the sub-differential $\partial_{\mathbf{\Theta}_{ij}} G(\mathbf{W}, b)$ are provided below.

Lemma 3.1. *[34] For a real matrix $\mathbf{W} \in \mathbb{R}^{p\times q}$, if the singular value decomposition (SVD) of $\mathbf{W}$ is $\mathbf{U}\mathbf{\Sigma}\mathbf{V}$, the sub-differential of $\|\mathbf{W}\|_*$ is defined as*

$$\partial\|\mathbf{W}\|_* = \left\{\mathbf{U}\mathbf{V}^{\mathbf{T}} + \mathbf{P} : \mathbf{P} \in \mathbb{R}^{p\times q}, \mathbf{U}^{\mathbf{T}}\mathbf{P} = \mathbf{0}, \mathbf{P}\mathbf{V} = \mathbf{0}, \|\mathbf{P}\| \le 1\right\}. \tag{3.4}$$

Lemma 3.2. *(Cauchy Schwarz). For $x, y \in \mathbb{R}^n$*

$$|\langle x, y\rangle| \le \|x\|\|y\|. \tag{3.5}$$

Subsequently, we calculate the sub-differential and estimate its range. Here, we only consider $\mathbf{\Theta}_{ij} = 0$. The sub-differential with respect to $\mathbf{\Theta}_{ij}$ can be represented as

$$\begin{aligned} \partial_{\mathbf{\Theta}_{ij}} G(\mathbf{W}, b) &= \mathbf{u}_i^T \nabla_{\mathbf{W}} h(\mathbf{W}, b)\mathbf{v}_j + \tau\mathbf{u}_i^T(\mathbf{U}\mathbf{V}^T + \mathbf{P})\mathbf{v}_j \\ &= \mathbf{u}_i^T \nabla_{\mathbf{W}} h(\mathbf{W}, b)\mathbf{v}_j + \tau\mathbf{u}_i^T\mathbf{P}\mathbf{v}_j, \end{aligned} \tag{3.6}$$

where the second equation holds because of the formula (3.3).

Meanwhile, we have the following inequations from Lemmas 3.1 and 3.2

$$\begin{aligned} \tau|\mathbf{u}_i^T\mathbf{P}\mathbf{v}_j| &= \tau|\langle \mathbf{u}_i, \mathbf{P}\mathbf{v}_j\rangle| \\ &\le \tau\|\mathbf{u}_i\| \cdot \|\mathbf{P}\mathbf{v}_j\| \\ &\le \tau\|\mathbf{u}_i\| \cdot \|\mathbf{P}\| \\ &= \tau\|\mathbf{P}\| \\ &\le \tau, \end{aligned} \tag{3.7}$$

Later, we can get

$$\partial_{\mathbf{\Theta}_{ij}} G(\mathbf{W}, b) \in [\mathbf{u}_i^T \nabla_{\mathbf{W}} h(\mathbf{W}, b)\mathbf{v}_j - \tau, \mathbf{u}_i^T \nabla_{\mathbf{W}} h(\mathbf{W}, b)\mathbf{v}_j + \tau].$$

Finally, a Theorem is given to define the inactive subspace and active subspace as follows:

Theorem 3.3. *For problem (3.1), we define the inactive subspace set $\mathcal{L}$ as*

$$\mathcal{L} = \{\mathbf{u}\mathbf{v}^T | \mathbf{u}^T\mathbf{W}\mathbf{v} = 0 \ \wedge \ |\mathbf{u}^T \nabla_{\mathbf{W}} h(\mathbf{W}, b)\mathbf{v}| \le \tau\},$$

and the active subspace set $\mathcal{E}$ as

$$\mathcal{E} = \{\mathbf{u}\mathbf{v}^T | \mathbf{u}^T\mathbf{W}\mathbf{v} \neq 0 \ \vee \ |\mathbf{u}^T \nabla_{\mathbf{W}} h(\mathbf{W}, b)\mathbf{v}| > \tau\},$$

where $\wedge$ denotes logical operation "and" and $\vee$ represents logical operation "or".

Proof. Let $0 \in \partial_{\mathbf{\Theta}_{ij}} G(\mathbf{W}, b)$, we could easily obtain that

$$|\mathbf{u}_i^T \nabla_{\mathbf{W}} h(\mathbf{W}, b) \mathbf{v}_j| \leq \tau.$$

The above conclusion can be established only when the formula (3.2) is satisfied. Thus the inactive subspace is defined as:

$$\mathcal{L} = \{\mathbf{u}\mathbf{v}^T | \mathbf{u}^T \mathbf{W} \mathbf{v} = 0 \ \wedge \ |\mathbf{u}^T \nabla_{\mathbf{W}} h(\mathbf{W}, b) \mathbf{v}| \leq \tau\}.$$

Therefore, all the rank one subspace in $\mathcal{L}$ have zero weight in the current solution $\mathbf{W}$. Afterward, the active subspace is as follows:

$$\mathcal{E} = \{\mathbf{u}\mathbf{v}^T | \mathbf{u}^T \mathbf{W} \mathbf{v} \neq 0 \ \vee \ |\mathbf{u}^T \nabla_{\mathbf{W}} h(\mathbf{W}, b) \mathbf{v}| > \tau\},$$

which is the complementary set of $\mathcal{L}$.

This completes the proof. □

If we can find the elements that belong to the above set, we can distinguish inactive subspaces and active subspaces. To do this, we first give a Lemma, and then provide a Theorem.

Lemma 3.4. *Let $\mathbf{U}\mathbf{\Sigma}\mathbf{V}^T$ represent the SVD of matrix $\mathbf{W}$, then, the singular value thresholding (SVT) operator [4] is as follows*

$$\mathcal{D}_\tau(\mathbf{W}) = \mathbf{U}\mathbf{\Sigma}_\tau\mathbf{V}^T, \tag{3.8}$$

where $(\mathbf{\Sigma}_\tau)_{ii} = max(0, \mathbf{\Sigma}_{ii} - \tau)$.

For simplicity, let $\mathbf{Z} = \mathbf{W} - \nabla_{\mathbf{W}} h(\mathbf{W}, b)$, then we give the following Theorem.

Theorem 3.5. *Assume $\mathbf{W} = \mathbf{U}\mathbf{\Sigma}\mathbf{V}^T$ is the reduced SVD of $\mathbf{W}$ ($\mathbf{U} \in \mathbb{R}^{p\times k}, \mathbf{V} \in \mathbb{R}^{q\times k}$ and $\mathbf{\Sigma}$ has positive diagonal values, and $\mathcal{D}_\tau(\mathbf{Z}) = \mathbf{U}_g\mathbf{\Sigma}_g\mathbf{V}_g$ (also a reduced SVD). Let $\mathbf{U}_s$ be an orthonormal basis of span$(\mathbf{U}, \mathbf{U}_g)$, $\mathbf{V}_s$ be an orthonormal basis of span $(\mathbf{V}, \mathbf{V}_g)$, and $\mathbf{U}_s^\perp, \mathbf{V}_s^\perp$ are orthonormal complements to $\mathbf{U}_s, \mathbf{V}_s$, then*

$$\{\mathbf{u}\mathbf{v}^T | \mathbf{u} \in \mathbf{U}_s^\perp \ \vee \ \mathbf{v} \in \mathbf{V}_s^\perp\} \subset \mathcal{L}. \tag{3.9}$$

Proof. We first prove $\mathbf{u}\mathbf{v}^T \in \mathcal{L}$ for all $\mathbf{u}$ if $\mathbf{v} \in \mathbf{V}_s^\perp$.

If $\mathbf{v} \in \mathbf{V}_s^\perp$, notice that $\mathbf{V}_s$ is an orthonormal basis of $span\{\mathbf{V}, \mathbf{V}_g\}$, the following equalities will be obtained:

$$\begin{aligned}
&\mathbf{V}^T\mathbf{v} = 0 \Rightarrow \mathbf{W}\mathbf{v} = 0, \\
&\mathbf{V}_g^T\mathbf{v} = 0 \Rightarrow \ \text{for } \forall\ \mathbf{u},\ |\mathbf{u}^T\mathbf{Z}\mathbf{v}| < \tau \Rightarrow |\mathbf{u}^T \nabla_{\mathbf{W}} h(\mathbf{W}, b)\mathbf{v}| \leq \tau.
\end{aligned}$$

Combine above two formulas, we have $\mathbf{u}\mathbf{v}^T \in \mathcal{L}$ for all $\mathbf{u}$ if $\mathbf{v} \in \mathbf{V}_s^\perp$. For $\mathbf{u} \in \mathbf{U}_s^\perp$, the proof is omitted since it is similar. Therefore, (3.9) is demonstrated to be true. □

Thus, $\mathbf{U}_s$ and $\mathbf{V}_s$ are the active subspace. Subsequently, we discuss how to calculate the values of $\mathbf{U}_s$ and $\mathbf{V}_s$. Note that to solve the issue, the values of $\mathbf{U}$ ($\mathbf{V}$) and $\mathbf{U}_g$ ($\mathbf{V}_g$) need be computed.

In the procedure of SES, the matrix $\mathbf{W}$ maintains a low-rank decomposition, namely $\mathbf{W} = \mathbf{U}\mathbf{\Sigma}\mathbf{V}^T$. Hence we do not need to calculate the values of $\mathbf{U}$ and $\mathbf{V}$.

To compute $\mathbf{U}_g$ and $\mathbf{V}_g$, we have to reckon the SVD of $\mathbf{Z}$ and then compute $\mathcal{D}_\tau(\mathbf{Z})$. The process of SVD is often time-consuming, particularly for large-scale matrix data. Based on this fact, we mainly manage to calculate effectively SVD of matrix $\mathbf{Z}$. Following the literature [12], we first utilize an auxiliary variable computed from the following Proposition and then give the singular vectors we need. Later, we could obtain the $\mathbf{U}_g$ and $\mathbf{V}_g$.

Proposition 3.6. *Let $\tilde{m}$ be the number of singular values in $\mathbf{Z} \in \mathbb{R}^{p\times q}$ larger than τ, and $\mathbf{Q} \in \mathbb{R}^{p\times m}$, where $m \geq \tilde{m}$, be orthogonal and contains the subspaces spanned by the top $\tilde{m}$ left singular vectors of $\mathbf{Z}$. Then, $\mathcal{D}_\tau(\mathbf{Z}) = \mathbf{Q}\mathcal{D}_\tau(\mathbf{Q}^T\mathbf{Z})$.*

Thus, if the span of $\mathbf{Z}$'s top m left singular vectors are given, we only need to perform SVD on a smaller $\mathbf{Q}^T\mathbf{Z} \in \mathbb{R}^{m\times q}$ (instead of $\mathbf{Z} \in \mathbb{R}^{p\times q}$). The matrix $\mathbf{Q}$ can be found by using the power method [12] detailed in Algorithm 1.

Algorithm 1 Power method

Input: $\mathbf{Z} \in \mathbb{R}^{p\times q}$, $\mathbf{R} \in \mathbb{R}^{q\times m}$, and the number of iterations *Iter*;
Output: $\mathbf{Q}$;
1: initialize $\mathbf{Q}_0 = QR(\mathbf{ZR})$; /* $QR(\cdot)$ is QR factorization */
2: **for all** $i = 1, 2, \cdots, Iter$ **do**
3: $\quad \mathbf{Q}_i = QR(\mathbf{Z}(\mathbf{Z}^T\mathbf{Q}_{i-1}))$;
4: **end for**
5: **return** $\mathbf{Q}_{Iter}$.

Based on the results above, we calculate the SVD of $\mathbf{Q}^T\mathbf{Z}$ and $\mathbf{Q}\mathcal{D}_\tau(\mathbf{Q}^T\mathbf{Z})$ to get $\mathbf{U}_g$ and $\mathbf{V}_g$. The procedure is shown in Algorithm 2.

Algorithm 2 Calculating $\mathcal{D}_\tau(\mathbf{Z})$

Input: $\mathbf{Z} \in \mathbb{R}^{p\times q}$, $\mathbf{R} \in \mathbb{R}^{q\times m}$, *Iter*, and $\tau \geq 0$;
Output: $\mathbf{U}_g$, $\mathbf{V}_g$;
1: $\mathbf{Q} = Powermethod(\mathbf{Z}, \mathbf{R}, Iter)$; $\leftarrow$ Algorithm 1
2: $[\tilde{\mathbf{U}}, \mathbf{\Sigma}, \mathbf{V}] = \text{SVD}(\mathbf{Q}^T\mathbf{Z})$; /* Calculate the SVD of a smaller matrix */
3: $\mathbf{U} = \mathbf{Q}\tilde{\mathbf{U}}$; /* Form the Orthogonal matrix */
4: $\mathbf{U}_g = \{\mathbf{u}_{ii} | \mathbf{\Sigma}_{ii} > \tau\}$;
5: $\mathbf{V}_g = \{\mathbf{v}_{ii} | \mathbf{\Sigma}_{ii} > \tau\}$;
6: **return** $\mathbf{U}_g$, $\mathbf{V}_g$.

Since $\mathbf{U}_s$ ($\mathbf{V}_s$) is an orthogonal basis of $span(\mathbf{U}, \mathbf{U}_g)$ ($span(\mathbf{V}, \mathbf{V}_g)$), we utilize the QR factorization to get that in our method.

3.3 The reduced problem of L_2-SMM

Given $\mathbf{U}_s$, $\mathbf{V}_s$, we can construct the column basis $\tilde{\mathbf{U}} = [\mathbf{U}_s, \mathbf{U}_s^\perp]$ and row basis $\tilde{\mathbf{V}} = [\mathbf{V}_s, \mathbf{V}_s^\perp]$. Subsequently, regression matrix $\mathbf{W}$ can be re-parameterize as $\mathbf{W} = \tilde{\mathbf{U}}\tilde{\mathbf{\Omega}}\tilde{\mathbf{V}}^T$, here $\tilde{\mathbf{\Omega}} \in \mathbb{R}^{p\times q}$. Therefore, the optimization problem (3.1) can be rewritten as

$$\underset{\tilde{\mathbf{\Omega}}\in\mathbb{R}^{p\times q}, b}{\arg\min}\, G(\tilde{\mathbf{U}}\tilde{\mathbf{\Omega}}\tilde{\mathbf{V}}^T, b) = h(\tilde{\mathbf{U}}\tilde{\mathbf{\Omega}}\tilde{\mathbf{V}}^T, b) + \tau\|\tilde{\mathbf{U}}\tilde{\mathbf{\Omega}}\tilde{\mathbf{V}}^T\|_*. \tag{3.10}$$

Suppose both $\mathbf{U}_s$ and $\mathbf{V}_s$ have k columns. Through the preceding analysis, only the $k \times k$ upper-left corner sub-matrix of $\tilde{\mathbf{\Omega}}$ is the active subspace set, while the other is the inactive subspace set. By employing the SES, the problem (3.10) downsizes to the below equivalence problem:

$$\underset{\mathbf{\Omega}\in\mathbb{R}^{k\times k}, b}{\arg\min}\, G(\mathbf{U}_s\mathbf{\Omega}\mathbf{V}_s^T, b) = h(\mathbf{U}_s\mathbf{\Omega}\mathbf{V}_s^T, b) + \tau\|\mathbf{U}_s\mathbf{\Omega}\mathbf{V}_s^T\|_*, \tag{3.11}$$

here, $\mathbf{\Omega} = \tilde{\mathbf{\Omega}}_{1:k,1:k} \in \mathbb{R}^{k\times k}$. Additionally, $\mathbf{U}_s, \mathbf{V}_s$ are orthogonal bases, which indicates that $\|\mathbf{U}_s\mathbf{\Omega}\mathbf{V}_s^T\|_* = \|\mathbf{\Omega}\|_*$. Afterward, the reduced problem (3.11) is written as follows:

$$\arg\min_{\mathbf{\Omega}\in\mathbb{R}^{k\times k},b} \tilde{G}(\mathbf{\Omega}, b) = \tilde{h}(\mathbf{\Omega}, b) + \tau\|\mathbf{\Omega}\|_*, \tag{3.12}$$

where $\tilde{h}(\mathbf{\Omega}, b) = h(\mathbf{U}_s\mathbf{\Omega}\mathbf{V}_s^T, b)$. In problem (3.12), we just need to solve a reduced optimization problem with a $k \times k$ matrix variables instead of $p \times q$ as in problem (3.1), which can potentially improve computation speed. However, in problem (3.12), we do not know the values of $\mathbf{U}_s$ and $\mathbf{V}_s$. Subsequently, we will discuss how to obtain $\mathbf{U}_s$ and $\mathbf{V}_s$, and how to solve problem (3.12).

4 ADMM

In this section, we solve the problem (3.12). As $\|\mathbf{\Omega}\|_*$ is convex but non-smooth, it is difficult to directly solve the problem (3.12). Fortunately, the ADMM method is an efficient solution method for low rank constrain problem [9], therefore, the problem (3.12) is solved with ADMM. To adopt ADMM strategy to our problem, we need to make our objective function separable. Therefore, we introduce one auxiliary variable $\mathbf{L}$ to replace $\mathbf{\Omega}$ in the nuclear term of our objective function. Hence, the objective function of (3.12) becomes two separable objective functions with a linear constraint:

$$\begin{aligned} \min_{\mathbf{\Omega},b,\mathbf{L}} \quad & \tilde{h}(\mathbf{\Omega}, b) + \tau\|\mathbf{L}\|_* \\ \text{s.t.} \quad & \mathbf{\Omega} = \mathbf{L}. \end{aligned} \tag{4.1}$$

For problem (4.1), the augmented Lagrangian function is given by

$$\mathcal{L}(\mathbf{\Omega}, b, \mathbf{L}, \mathbf{\Gamma}) = \tilde{h}(\mathbf{\Omega}, b) + \tau\|\mathbf{L}\|_* + \mathrm{tr}(\mathbf{\Gamma}^T(\mathbf{L} - \mathbf{\Omega})) + \frac{\mu}{2}\|\mathbf{L} - \mathbf{\Omega}\|_F^2, \tag{4.2}$$

where $\mathbf{\Gamma} \in \mathbb{R}^{k\times k}$ is Lagrangian multiplier and $\mu > 0$ is the positive penalty parameter. ADMM algorithm iteratively estimates the optimal solutions by minimizing problem (4.2). In each iteration process, a variable is solved by fixing the remaining variables, and the updating order for all variables is reported below:

$$\begin{aligned} (\mathbf{\Omega}_l, b_l) &= \arg\min_{\mathbf{\Omega},b} \mathcal{L}(\mathbf{\Omega}, b, \mathbf{L}_{l-1}, \mathbf{\Gamma}_{l-1}), \\ \mathbf{L}_l &= \arg\min_{\mathbf{L}} \mathcal{L}(\mathbf{\Omega}_l, b_l, \mathbf{L}, \mathbf{\Gamma}_{l-1}), \\ \mathbf{\Gamma}_l &= \mathbf{\Gamma}_{l-1} + \mu(\mathbf{L}_l - \mathbf{\Omega}_l), \end{aligned} \tag{4.3}$$

where $l \in \mathbb{N}$ indicates the index of iteration. Then, the detailed solutions of above subproblems are described as follows.

(1) ($\mathbf{\Omega}$, b)-subproblem: To update $\mathbf{\Omega}$ and b, we first fix the other variables, and then consider the problem described below:

$$(\mathbf{\Omega}^*, b^*) = \arg\min_{\mathbf{\Omega},b} \frac{1}{2}\tilde{h}(\mathbf{\Omega}, b) - \mathrm{tr}(\mathbf{\Gamma}^T\mathbf{\Omega}) + \frac{\mu}{2}\|\mathbf{L} - \mathbf{\Omega}\|_F^2. \tag{4.4}$$

For $\|\mathbf{U}_s\mathbf{\Omega}\mathbf{V}_s^T\|_F^2 = \mathrm{tr}(\mathbf{V}_s\mathbf{\Omega}^T\mathbf{U}_s^T\mathbf{U}_s\mathbf{\Omega}\mathbf{V}_s) = \mathrm{tr}(\mathbf{\Omega}^T\mathbf{\Omega})$, the above problem is equivalent to solving the following problem:

$$\begin{aligned} \min_{\mathbf{\Omega},b,\boldsymbol{\xi}} \quad & \frac{1}{2}\,\mathrm{tr}(\mathbf{\Omega}^T\mathbf{\Omega}) - \mathrm{tr}(\mathbf{\Gamma}^T\mathbf{\Omega}) + \frac{\mu}{2}\|\mathbf{L}-\mathbf{\Omega}\|_F^2 + C\sum_{i=1}^{n}\xi_i^2 \\ \text{s.t.} \quad & y_i\left[\mathrm{tr}(\mathbf{V}_s\mathbf{\Omega}^T\mathbf{U}_s^T\mathbf{X}_i) + b\right] \geq 1-\xi_i, \quad i = 1, 2, \cdots, n, \end{aligned} \tag{4.5}$$

where $\boldsymbol{\xi}$ is slack vector and $C > 0$ is parameter chosen in advance.

To solve the optimization problem (4.5), we construct the following Lagrangian function

$$\begin{aligned} \mathcal{L}'(\mathbf{\Omega}, b, \boldsymbol{\xi}, \boldsymbol{\alpha}) = & \frac{1}{2}\,\mathrm{tr}\left(\mathbf{\Omega}^T\mathbf{\Omega}\right) - \mathrm{tr}\left(\mathbf{\Gamma}^T\mathbf{\Omega}\right) + \frac{\mu}{2}\|\mathbf{L}-\mathbf{\Omega}\|_F^2 \\ & + C\sum_{i=1}^{n}\xi_i^2 - \sum_{i=1}^{n}\alpha_i\left\{y_i\left[\mathrm{tr}\left(\mathbf{V}_s\mathbf{\Omega}^T\mathbf{U}_s^T\mathbf{X}_i\right) + b\right] - 1 + \xi_i\right\}, \end{aligned} \tag{4.6}$$

where $\boldsymbol{\alpha} \geq 0$ is the Lagrangian multiplier.

According to the KKT conditions, the following equations are calculated as follows:

$$\begin{aligned} & \frac{\partial\mathcal{L}'}{\partial\mathbf{\Omega}} = 0 \rightarrow \mathbf{\Omega} = \frac{1}{1+\mu}(\mathbf{\Gamma} + \mu\mathbf{L} + \sum_{i=1}^{n}\alpha_i y_i \mathbf{U}_s^T\mathbf{X}_i\mathbf{V}_s), \\ & \frac{\partial\mathcal{L}'}{\partial b} = 0 \rightarrow \sum_{i=1}^{n}\alpha_i y_i = 0, \\ & \frac{\partial\mathcal{L}'}{\partial\xi_i} = 0 \rightarrow \xi_i = \frac{1}{2C}\alpha_i. \end{aligned} \tag{4.7}$$

Substituting the above formulations into (4.6), we have

$$\begin{aligned} \mathcal{L}'(\mathbf{\Omega}, b, \boldsymbol{\xi}, \boldsymbol{\alpha}) = & \frac{1}{2}\,\mathrm{tr}(\mathbf{\Omega}^T\mathbf{\Omega}) - \mathrm{tr}(\mathbf{\Gamma}^T\mathbf{\Omega}) + \frac{\mu}{2}\|\mathbf{L}-\mathbf{\Omega}\|_F^2 - \frac{1}{4C}\boldsymbol{\alpha}^T\boldsymbol{\alpha} \\ & - \sum_{i=1}^{n}\alpha_i\{y_i[\mathrm{tr}(\mathbf{V}_s\mathbf{\Omega}^T\mathbf{U}_s^T\mathbf{X}_i)] - 1\} \\ = & \frac{1+\mu}{2}\,\mathrm{tr}(\mathbf{\Omega}^T\mathbf{\Omega}) - \mathrm{tr}\left[(\mathbf{\Gamma}^T + \mu\mathbf{L}^T + \sum_{i=1}^{n}\alpha_i y_i \mathbf{V}_s^T\mathbf{X}_i^T\mathbf{U}_s)\mathbf{\Omega}\right] \\ & - \frac{1}{4C}\boldsymbol{\alpha}^T\boldsymbol{\alpha} + \boldsymbol{\alpha} \\ = & -\frac{1+\mu}{2}\,\mathrm{tr}(\mathbf{\Omega}^T\mathbf{\Omega}) - \frac{1}{4C}\boldsymbol{\alpha}^T\boldsymbol{\alpha} + \boldsymbol{\alpha} \\ = & -\frac{1}{2(1+\mu)}\left[\sum_{i=1}^{n}\sum_{j=1}^{n}\alpha_i\alpha_j y_i y_j\,\mathrm{tr}(\mathbf{X}_i^T\mathbf{X}_j)\right] \\ & - \frac{1}{1+\mu}\sum_{i=1}^{n}\alpha_i y_i\,\mathrm{tr}[(\mathbf{\Gamma} + \mu\mathbf{L})^T\mathbf{U}_s^T\mathbf{X}_i\mathbf{V}_s] - \frac{1}{4C}\boldsymbol{\alpha}^T\boldsymbol{\alpha} + \boldsymbol{\alpha} + const \\ = & -\frac{1}{2}\boldsymbol{\alpha}^T(\mathbf{H} + \frac{1}{2C}\mathbf{I})\boldsymbol{\alpha} + \mathbf{f}^T\boldsymbol{\alpha} + const, \end{aligned} \tag{4.8}$$

where $\mathbf{H} = [\mathbf{H}_{ij}] \in \mathbb{R}^{n\times n}$, $\mathbf{f} \in \mathbb{R}^n$, and *const* is a constant, specifically,

$$\mathbf{H}_{ij} = \frac{y_i y_j \operatorname{tr}(\mathbf{X}_i^T \mathbf{X}_j)}{1+\mu}, \quad \mathbf{f}_i = 1 - \frac{y_i \operatorname{tr}[(\mathbf{\Gamma} + \mu\mathbf{L})^T \mathbf{U}_s^T \mathbf{X}_i \mathbf{V}_s]}{1+\mu},$$
$$const = -\frac{1}{2(1+\mu)} \operatorname{tr}[\mathbf{\Gamma}^T\mathbf{\Gamma} + 2\mu\mathbf{\Gamma}^T\mathbf{L} + \mu^2\mathbf{L}^T\mathbf{L}]. \tag{4.9}$$

Through the KKT conditions (4.7) and $\alpha_i \geq 0$, we derive the dual problem of (4.5) below:

$$\begin{aligned} \max_{\boldsymbol{\alpha}} \quad & -\frac{1}{2}\boldsymbol{\alpha}^T(\mathbf{H} + \frac{1}{2C}\mathbf{I})\boldsymbol{\alpha} + \mathbf{f}^T\boldsymbol{\alpha} \\ \text{s.t.} \quad & \sum_{i=1}^{n} \alpha_i y_i = 0, \\ & \boldsymbol{\alpha} \geq 0. \end{aligned} \tag{4.10}$$

By solving the problem (4.10) with "*quadprog*" toolbox of MATLAB, we could get the vector $\boldsymbol{\alpha}^*$, then we can obtain $\mathbf{\Omega}^*$ from the following formulation:

$$\mathbf{\Omega}^* = \frac{1}{1+\mu}(\mathbf{\Gamma} + \mu\mathbf{L} + \sum_{i=1}^{n} \alpha_i^* y_i \mathbf{U}_s^T \mathbf{X}_i \mathbf{V}_s). \tag{4.11}$$

As for the optimal solution of b, we consider the KKT conditions, which provide

$$\alpha_i\{y_i[\operatorname{tr}(\mathbf{\Omega}^T\mathbf{X}_i) + b] - 1 + \xi_i\} = 0, \tag{4.12}$$

for $\forall \alpha_i > 0$, we have

$$y_i[\operatorname{tr}(\mathbf{\Omega}^T\mathbf{X}_i) + b] - 1 + \xi_i = 0, \tag{4.13}$$

according to (4.7), the following formulation is derived

$$y_i[\operatorname{tr}(\mathbf{\Omega}^T\mathbf{X}_i) + b] = 1 - \frac{\alpha_i^*}{2C} \rightarrow b = y_i(1 - \frac{\alpha_i^*}{2C}) - \operatorname{tr}(\mathbf{\Omega}^T\mathbf{X}_i). \tag{4.14}$$

Practically, we compute the optimal b with the averaging solution

$$b^* \quad = \frac{1}{|\mathcal{E}^*|}\sum_{i\in\mathcal{E}^*}\{y_i(1 - \frac{\alpha_i^*}{2C}) - \operatorname{tr}(\mathbf{\Omega}^T\mathbf{X}_i)\}, \tag{4.15}$$

where $\mathcal{E}^* = \{i|\alpha_i^* > 0\}$.

(2) L-subproblem: In this step, we fix the other variables, and update $\mathbf{L}$ as follows:

$$\mathbf{L}^* = \arg\min_{\mathbf{L}} \tau\|\mathbf{L}\|_* + \operatorname{tr}(\mathbf{\Gamma}^T\mathbf{L}) + \frac{\mu}{2}\|\mathbf{L} - \mathbf{\Omega}\|_F^2. \tag{4.16}$$

Since $\|\mathbf{L}\|_*$ is non-smooth and non-differentiable, we derive the sub-gradient for $\mathbf{L}$ as

$$0 \in \tau\partial\|\boldsymbol{L}\|_* + \mathbf{\Gamma} + \mu(\mathbf{L} - \mathbf{\Omega}), \tag{4.17}$$

where $\partial\|\mathbf{\Omega}\|_*$ is denoted as the sub-gradient of the nuclear norm.

To solve (4.16), suppose the SVD of $\mu\mathbf{\Omega} - \mathbf{\Gamma}$ as

$$\mu\mathbf{\Omega} - \mathbf{\Gamma} = \mathbf{U}_1\mathbf{\Sigma}_1\mathbf{V}_1^T + \mathbf{U}_{-1}\mathbf{\Sigma}_{-1}\mathbf{V}_{-1}^T, \tag{4.18}$$

where $\mathbf{\Sigma}_1$ is a diagonal matrix whose diagonal values are all greater than τ. $\mathbf{U}_1$ and $\mathbf{V}_1$ are the corresponding singular matrices. By contrast, $\mathbf{\Sigma}_{-1}$ is a diagonal matrix whose diagonal values are all not greater than τ. $\mathbf{U}_{-1}$ and $\mathbf{V}_{-1}$ are the corresponding singular matrices.

Similar to [40], let $\mathbf{L} = \frac{1}{\mu}\mathbf{U}_1(\mathbf{\Sigma}_1 - \tau\mathbf{I})\mathbf{V}_1^T$, and thus we can obtain

$$\begin{aligned}\mu(\mathbf{\Omega} - \mathbf{L}) - \mathbf{\Gamma} =& \mu\mathbf{\Omega} - \mathbf{\Gamma} - \mu\mathbf{L} \\ =& \mathbf{U}_1\mathbf{\Sigma}_1\mathbf{V}_1^T + \mathbf{U}_{-1}\mathbf{\Sigma}_{-1}\mathbf{V}_{-1}^T - \mathbf{U}_1(\mathbf{\Sigma}_1 - \tau\mathbf{I})\mathbf{V}_1^T \\ =& \tau(\mathbf{U}_1\mathbf{V}_1 + \frac{1}{\tau}\mathbf{U}_{-1}\mathbf{\Sigma}_{-1}\mathbf{V}_{-1}^T).\end{aligned} \tag{4.19}$$

According to Lemma 3.1, let $\mathbf{P} = \frac{1}{\tau}\mathbf{U}_{-1}\mathbf{\Sigma}_{-1}\mathbf{V}_{-1}^T$. It is easy to verify that $\mathbf{U}_1\mathbf{P}^T = 0, \mathbf{P}\mathbf{V}_1 = 0$, and $\|\mathbf{P}\| \leq 1$ for the diagonal matrix $\mathbf{\Gamma}_1$ is bounded by τ. Therefore, we have $\mu(\mathbf{\Omega} - \mathbf{L}) - \mathbf{\Gamma} \in \tau\partial\|\mathbf{\Omega}\|_*$, and the updating formula for $\mathbf{L}$ is obtained as

$$\mathbf{L} = \frac{1}{\mu}\mathbf{U}_1(\mathbf{\Sigma}_1 - \tau\mathbf{I})\mathbf{V}_1^T = \frac{1}{\mu}\mathcal{D}_\tau(\mu\mathbf{\Omega} - \mathbf{\Gamma}). \tag{4.20}$$

(3) Update Multiplier: We update the multiplier by

$$\mathbf{\Gamma} = \mathbf{\Gamma} + \mu(\mathbf{L} - \mathbf{\Omega}). \tag{4.21}$$

Finally, to accelerate ADMM and guarantee convergence, we use a restart rule [11]. The details are represented in Algorithm 3, where the stopping criteria are the number of iterations l reaches the maximum number *maxIter*, which is set to 1000, or the algorithm satisfies the following convergence condition:

$$\frac{|obj_l - mean(obj_{recent})|}{|mean(obj_{recent})|} < 10^{-5}, \tag{4.22}$$

where obj denotes the value of the objective function, obj_l is the value of the l-th iteration, and obj_{recent} represents the value of the last 50 times (it implies that the method at least iterates 50 times).

Furthermore, our proposed subspace elimination strategy for accelerating L_2-SMM can be concluded in Algorithm 4. In our experiments, the maximum number $MaxIter$ of SES is set to 10 and convergence criterion is

$$errW = \sqrt{\frac{\|\mathbf{W}_m - \mathbf{W}_{m-1}\|_F^2}{numel(\mathbf{W})}} \leq 10^{-5}, \tag{4.23}$$

where $\mathbf{W}_m$ is the value of m-th iteration and $numel(\mathbf{W})$ denotes the number of elements in matrix $\mathbf{W}$.

5 Numerical Experiments

To illustrate the efficacy of our proposed acceleration scheme, experiments on multiple datasets are conducted in this section. All experiments are conducted in MATLAB R2019a on Windows 7 running on a PC with system configuration Intel(R) Core(TM) i5-4590 CPU(3.30 GHZ) with 8.00 GB of RAM. Moreover, to demonstrate the effectiveness of our SES-L_2-SMM, five other algorithms are compared in the experiments, i.e., Sparse SVM (SSVM) [41], bilinear SVM (BSVM) [16], Sparse SMM (SSMM) [39], SMM [18], and L_2-SMM. Among them, SSVM is the representative vector-based method and others are state-of-the-art matrix classifiers. The package is download from the website.

SSMM: https://github.com/zhengqq/SSMM. SMM: http://bcmi.sjtu.edu.cn/luoluo/code/smm.zip. SSVM and BSVM can be regard as the special cases of SSMM when parameter $\tau = 0$ and $\gamma = 0$.

Algorithm 3 ADMM algorithm for problem (3.12)

Input: $\mathbf{\Omega} \in \mathbb{R}^{k\times k}$, $\mathbf{U}_s \in \mathbb{R}^{p\times k}$, $\mathbf{V}_s \in \mathbb{R}^{q\times k}$, $\tau \geq 0$, $C \geq 0$, and $maxIter$;
Output: $\mathbf{\Omega}, b$;

Initialize: $\mathbf{L}_{-1} = \tilde{\mathbf{L}} = \mathbf{0}, \mu = 10$
repeat
  update $(\mathbf{\Omega}_l, b_l) = \arg\min_{\mathbf{\Omega},b} \mathcal{L}(\mathbf{\Omega}, b, \tilde{\mathbf{L}}_l, \tilde{\mathbf{\Gamma}}_l) \leftarrow$ Formula (4.11) and (4.15);
  update $\mathbf{L}_l = \arg\min_{\mathbf{L}} \mathcal{L}(\mathbf{\Omega}_l, b_l, \mathbf{L}, \tilde{\mathbf{\Gamma}}_l) \leftarrow$ Formula (4.20);
  update $\mathbf{\Gamma}_l = \tilde{\mathbf{\Gamma}}_l + \mu(\mathbf{L}_l - \mathbf{\Omega}_l) \leftarrow$ Formula (4.21);
    /* restart rule */
  $c_l = \mu^{-1}\|\mathbf{\Gamma}_l - \tilde{\mathbf{\Gamma}}_l\|_F^2 + \mu\|\mathbf{L}_l - \tilde{\mathbf{L}}_l\|_F^2$;
  if $c_l < \eta c_{l-1}$ **then**
    $\gamma_{l+1} = \frac{1+\sqrt{1+4\gamma_l^2}}{2}$;
    $\tilde{\mathbf{L}}_{l+1} = \mathbf{L}_l + \frac{\gamma_l - 1}{\gamma_{l+1}}(\mathbf{L}_l - \mathbf{L}_{l-1})$;
    $\tilde{\mathbf{\Gamma}}_{l+1} = \mathbf{\Gamma}_l + \frac{\gamma_l - 1}{\gamma_{l+1}}(\boldsymbol{L}_l - \boldsymbol{L}_{l-1})$;
  else
    $\gamma_{l+1} = 1$;
    $\tilde{\mathbf{L}}_{l+1} = \mathbf{L}_{l-1}$;
    $\tilde{\mathbf{\Gamma}}_{l+1} = \mathbf{\Gamma}_{l-1}$;
    $c_l = \eta^{-1} c_{l-1}$;
  end if
until $l \geq maxIter$ *or convergence*
return $\mathbf{\Omega}$, b.

Algorithm 4 SES-L_2-SMM

Input: dataset $\{\mathbf{X}_i, y_i\}_{i=1}^n$, parameter $C \geq 0$, and $\tau \geq 0$;
Output: $\mathbf{W}, b$;

repeat
  $\mathbf{Z} = \mathbf{W} - \nabla_{\mathbf{W}} h(\mathbf{W}, b)$;
  $[\mathbf{U}_g, \mathbf{\Sigma}_g, \mathbf{V}_g] = \mathcal{D}_\tau(\mathbf{Z}) \leftarrow$ Algorithm 2
  $\mathbf{U}_s = QR([\mathbf{U}_g, \mathbf{U}])$; /* $\mathbf{U}_s \in \mathbb{R}^{p\times k_m}$ */
  $\mathbf{V}_s = QR([\mathbf{V}_g, \mathbf{U}])$; /* $\mathbf{V}_s \in \mathbb{R}^{q\times k_m}$ */
  Calculate $(\mathbf{\Omega}, b) \leftarrow$ Algorithm 3;
  $\mathbf{W} = \mathbf{U}_s \mathbf{\Omega} \mathbf{V}_s^T$;
  $m = m + 1$;
until $m \geq MaxIter$ *or convergence*
return $\mathbf{W}$, b.

5.1 Experiments settings

We first introduce the settings of our experiment. Parameters C and τ are tuned by altering the values of the parameters on a grid of 5×5, which gains values from $C = \{2^i | i = -4 : 2 : 4\}$ and $\tau = \{0.1, 0.5, 1, 5, 10\}$. For fair comparison, the free parameters of all competitive algorithms are carefully adjusted to acquire the best classification results. Additionally, five-fold cross-validation is employed in our experiments.

5.2 Real datasets

A. Fish dataset

Disease and decay of seafood can cause serious human health problems and economic losses. Therefore, the evaluation of the quality of seafood is highly crucial. However, since disease and decay of seafood show different symptoms in different species, it is first necessary to classify the species. In this section, we carry out extensive experiments on a large-scale fish dataset (Fish dataset) [30]. The dataset contains 9 different seafood types which are widely consumed. All the fish in the process of image acquisition are fresh, and they are placed at various displacements and angles, but the light conditions do not change distinctly. The background of images is blue and noisy to make the dataset useable for the study of practical problems. The size of each image is 2832×2128 or 1024×768. There are approximately 30-50 images each class.

In our experiments, we select alphabetically the first 5 classes, namely, black sea sprat (B), gilt head bream (G), horse mackerel (H), red mullet (R1), and red sea bream (R2). Some samples are shown in Fig. 1. We adopt a "1-versus-1" structure and form $C_5^2 = 10$ binary

Figure 1: Some samples from Fish dataset, from top to bottom belong to black sea sprat (B), gilt head bream (G), horse mackerel(H), red mullet (R1), and red sea bream (R2), respectively.

classification problems. For facilitation, we will use the abbreviations of these datasets, for instance, "BG" indicates that "Black sea sprat-versus-Gilt head bream". In addition, some image samples contain time stamp at the bottom right of the picture. To prevent this information from being learned as additional information, which would unfairly affect the results, we delete these images. We first convert the images into gray scale images and then use the pixel values as features. Finally, the detailed description of our used dataset is

displayed in Table 1.

B. MaskedFace-Net

COVID-19 causes an outbreak of severe pneumonia with worldwide human-to-human transmission and leads to significant morbidity and mortality. To a certain extent, wearing face masks might limit the spread of COVID-19. It makes sense to use a recognition system to check whether people are wearing masks in regulated areas. However, it is even more crucial to wear a mask correctly. Actually, plenty of people are not properly wearing their masks because of bad practices, bad behavior or the vulnerability of individuals. For the above reasons, [3] collected masked face detection dataset, that is, MaskedFace-Net data. Subsequently, we perform experiments on that dataset to detect faces having their masks correctly worn or incorrectly worn. MaskedFace-Net data is composed of two types, namely, the Correctly Masked Face Data set (CMFD) and the Incorrectly Masked Face Data set (IMFD). Besides, IMFD is subdivided into three subclasses, as depicted in Fig. 2. The

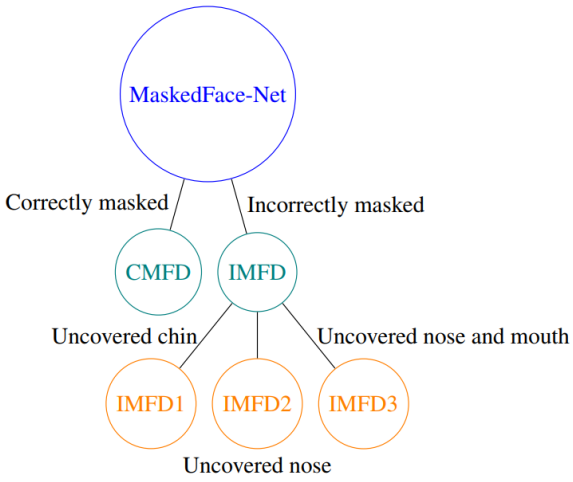

Figure 2: View of the MaskedFace-Net data tree.

dataset contains 67049 images with CMFD and 66734 images with IMFD at 1024×1024.

In our experiments, we use CMFD (C) and IMFD1 (I1), IMFD2 (I2), IMFD3 (I3) to form three binary classifications, respectively. Analogously, the abbreviation of the dataset is used for the name of the dataset, for example, "CI1" represents "CMFD-versus-IMFD1". Some instances are deployed in Fig. 3. We select part of the data for the experiment and the detailed information is shown in Table 1. Similar to Fish date set, we convert the images into gray scale images and extract the gray pixel values as the input matrix-form features.

C. GAMEEMO

Numerical experiments are further conducted on the application of EEG Emotion data classification. Emotion impacts individuals' daily life and plays a crucial role. Unfavorable

C is the first 100 images of "00000" folder; I2 is the top 100 images of IMFD2 in "00000" folder; I1 and I3 is selected from "00000" folder.

Figure 3: The picture on the left shows samples of correctly masked faces; The picture on the right represents samples of incorrectly masked faces; specifically, in the right picture, the first two rows are uncovered chin (IMFD1), the middle two rows are uncovered nose (IMFD2), and the last two rows are uncovered nose and mouth (IMFD3).

emotions can lead people to develop poor mental health conditions while positive emotions provide a better standard of living. GAMEEMO [1] includes games-based EEG signals. They are collected from 28 different subjects with wearable and portable EEG device called 14 channel Emotiv Epoc+. Subjects played emotionally 4 different computer games, including Train Sim World, Unravel, Slender-The Arrival, and Goat, to evoke the target emotion, that is, boring (B), calm (C), horror (H), and funny (F), respectively. EEG signals are collected from various EEG channels during the data obtaining process. There are 38252 samples in an EEG channel, total 14 EEG channels, therefore, our feature matrix is 14×38252. Similarly, we form $C_4^2 = 6$ binary classification datasets. "BC" consists of boring EEG emotion signal and calm EEG emotion signal. The detailed description is also shown in Table 1.

5.3 Experimental results

The performance comparisons are listed in Tables 2-4, where "Acc" is the best testing accuracy among all parameters; "Time" shows the average time of the cross validation, which contains the SES time and training time for each pair of parameters. "k" ($k = (k_1 + \cdots + k_m)/m$) is the mean value of *step 4* in Algorithm 4. "Speedup" is calculated from the ratio of solving time of L_2-SMM and that of SES-L_2-SMM.

Table 2 shows the results of Fish dataset. It is can be seen that SES-L_2-SMM has the superior prediction performance and the admirable computational speed. SES-L_2-SMM significantly reduces the solving time and its maximum speedup is up to 10.58 on the R1R2 of the Fish dataset. As Table 2 clearly demonstrates, matrix classifiers outperform the vector classifier on most datasets because they take advantage of the structural information

Table 1: The statistics of three datasets.

Data sets	Subsets	#Instances	#Features
Fish date set	BG	79	1024×768
	BH	80	1024×768
	BR1	83	1024×768
	BR2	81	1024×768
	GH	59	1024×768
	GR1	62	1024×768
	GR2	60	1024×768
	HR1	63	1024×768
	HR2	61	1024×768
	R1R2	64	1024×768
MaskedFace-Net	CI1	181	1024×1024
	CI2	200	1024×1024
	CI3	176	1024×1024
GAMEEMO	BC	56	14×38252
	BH	56	14×38252
	BF	56	14×38252
	CH	56	14×38252
	CF	56	14×38252
	HF	56	14×38252

the within feature matrix. Besides, it must also be mentioned that the accuracies of SMM, L_2-SMM, and our SES-L_2-SMM are very close. The main reason for this situation is that these algorithms are based on the same SMM model. The accuracy differences between these methods are tiny in this data set, but our method performs better than the vector classifier SSVM in both time and accuracy in most cases. These results highlight the effectiveness of matrix classifiers and illustrate the significance of our proposed SES for accelerating the L_2-SMM.

Table 2: Performance comparisons of six algorithms on Fish dataset.

Data	SSVM		BSVM		SSMM		SMM		L_2-SMM		SES-L_2-SMM			
	Acc	Time	Acc	Time	Acc	Time	Acc	Time	Acc	Time	Acc	Time	k	Speedup
BG	78.31	378.21	91.64	225.44	89.82	311.07	98.06	473.72	98.33	449.6	98.33	80.06	16	5.62
BH	86.16	396.1	87.14	272.22	87.98	335	93.36	481.49	93.36	482.23	94.2	173.98	26	2.77
BR1	67.86	419.5	69.06	329.2	66.03	376.29	66.48	441.89	67.28	439.38	67.28	182.75	107	2.4
BR2	87.42	387.2	90.53	235.7	91.83	325.01	92.89	472.25	92.89	489.5	94.22	46.1	34	10.62
GH	80.71	362.89	90	230.57	89.76	322.16	87.86	440.77	85.71	438.92	85.71	61.86	43	7.1
GR1	81.15	363.87	93.93	213.42	90.63	312.17	86.83	444.92	86.83	443.86	86.83	76.09	21	5.83
GR2	76.9	354.05	85.95	235.57	87.62	319.14	80.95	463.91	80.95	459.94	80.95	45.8	29	10.04
HR1	81.67	389.63	88.41	252.39	87.46	380.08	88.81	450.47	88.81	444.14	88.81	116.1	31	3.83
HR2	90.71	364.92	91.11	254.59	91.11	356.26	90.63	437.87	90.63	437.29	90.63	48.44	31	9.03
R1R2	86.65	367.73	92.05	226.9	91.5	333.48	97.38	445.92	97.38	448.37	97.38	42.39	30	10.58

The results of MaskedFace-Net are given in Table 3. Significantly, since the number of variables in the feature matrix is much greater than that of the Fish dataset, the computation time is increased across all algorithms. Our proposed strategy still has a significant speedup effect.

From Tables 2-4, the value of $k \times k$ is much less than the value of $p \times q$. Even if we iterate over Algorithm 3 many times, our proposed SES-L_2-SMM is still faster than L_2-SMM. It demonstrates that reducing the number of matrix variables can accelerate the solving speed

of optimization problems to a certain extent.

Table 3: Performance comparisons of six algorithms on MaskedFace-Net.

Data	SSVM		BSVM		SSMM		SMM		L_2-SMM		SES-L_2-SMM			
	Acc	Time	Acc	Time	Acc	Time	Acc	Time	Acc	Time	Acc	Time	k	Speedup
CI1	84.95	1883.97	84.32	787.17	89.27	1296.42	89.08	1066.73	88.83	1044.75	88.83	488.17	29	2.14
CI2	87.11	1450.44	85.09	1177.25	87.08	1400.64	88.9	1035.44	88.96	1002.33	88.96	711.45	33	1.41
CI3	88.67	1323.23	81.38	1071.02	88.03	1249.26	88.97	1145.29	88.96	1165.2	89.36	435.57	25	2.68

Table 4 shows the performance comparisons on GAMEEMO. It is clear that the matrix classifiers perform significantly better than the vector classifier. One of the primary causes is that EEG signals are usually highly relevant. Matrix classifiers have better performance because they can capture inherent structure information. It demonstrates the strong efficiency of matrix classifiers in the task of EEG emotion signal classification. Obviously, our proposed algorithm is consistently superior to all competing classifiers on all binary matrix subsets of GAMEEMO for time comparison. That fully demonstrates that our algorithm is significantly effective.

Table 4: Performance comparisons of six algorithms on GAMEEMO.

Data	SSVM		BSVM		SSMM		SMM		L_2-SMM		SES-L_2-SMM			
	Acc	Time	Acc	Time	Acc	Time	Acc	Time	Acc	Time	Acc	Time	k	Speedup
BC	66.32	135.23	96.19	86.65	88.57	112.74	94.29	24.01	94.29	31.86	93.33	10.39	26	3.07
BH	67.36	133.71	81.9	94.7	96.19	111.21	91.6	24.98	88.83	32	90.65	10.87	26	2.94
BF	66.32	137.42	88.92	84.95	86.06	112.51	86.15	26.89	86.15	30.85	90.65	16.34	26	1.89
CH	66.32	136.66	90.48	95.41	90.48	110.84	88.92	30.8	88.92	30.93	92.55	23.05	26	1.34
CF	66.32	138.24	91.6	88.07	74.72	111.47	88.83	25.91	88.83	30.15	90.65	16.76	26	1.8
HF	59.65	143.59	79.13	107.78	64.59	116.13	87.19	27.63	87.19	30.31	90.82	17.37	26	1.74

5.4 Experimental result analysis

To visually explicate the speedup of SES-L_2-SMM, the bar charts of three datasets are depicted in Fig. 4 of two algorithms using the results in Tables 2-4. It should be noted that the time consumption of different datasets varies greatly, thus the relative time consumption is used as the ordinate in this paper. Relative time-consuming means that the time consumption of L_2-SMM is set to 1, and the computation time of SES-L_2-SMM is the relative time compared with L_2-SMM. We can observe that our method have significantly acceleration effect on all datasets. In addition, SES provides first-rate acceleration performance across the Fish dataset. The reason for this situation might be that the data sample itself has a lower rank due to the fact that the image of the dataset consists of only two parts: the blue background and the fish.

Furthermore, by taking the Fish dataset and MaskedFace-Net for example, we explore the influence of parameters C and τ on subspace elimination. The results are reported in Fig. 6, where "ratio" is calculated by the following formula:

$$ratio = (1 - \frac{k \times k}{p \times q}) * 100.$$

As is obvious from Fig. 6 where the speedup effect of SES-L_2-SMM is stable when the parameters C and τ are small. However, when the parameter C is large, the elimination

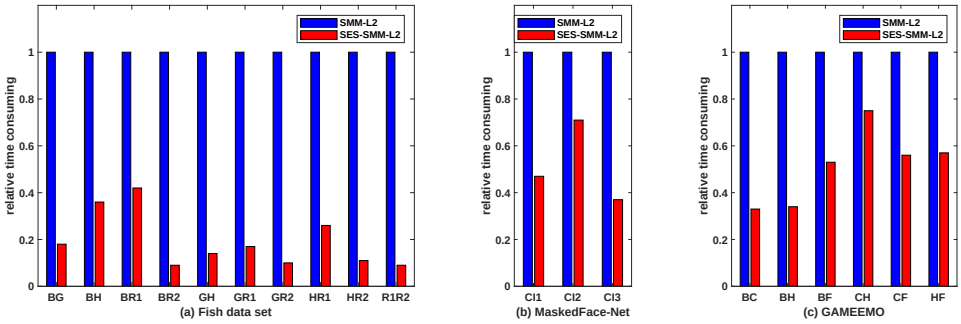

Figure 4: The graphical representation about relative time-consuming for L_2-SMM and SES-L_2-SMM on three datasets.

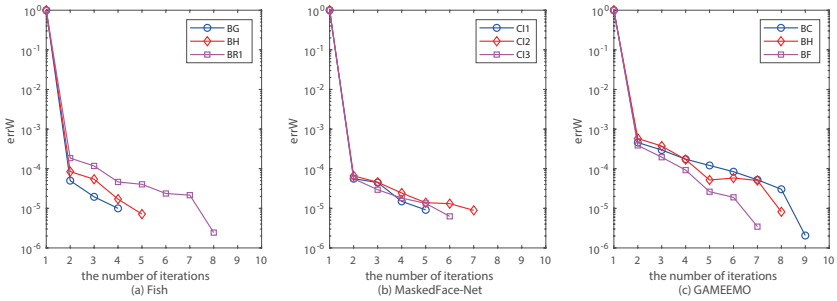

Figure 5: Convergence process for SES-L_2-SMM on the first three subsets of three datasets, respectively.

Fig. 5 shows the convergence process of SES-L_2-SMM on the first three subsets of three datasets, respectively. In the picture, "errW" denotes the value of equation (4.23) in each iteration and "the number of iterations" represents the value of m in Algorithm 4. It verifies that our algorithm converges fast in ten steps, which is consistent with the description in literature [13]. Similar phenomena also occur when using SES-L_2-SMM in other datasets. It shows the efficacy of our method.

ratio decreases with the increase of the parameter τ. Similarly, when the parameter τ is large, the elimination ratio grows as the increase of parameter C. In other words, when a parameter is small, the elimination ratio is higher.

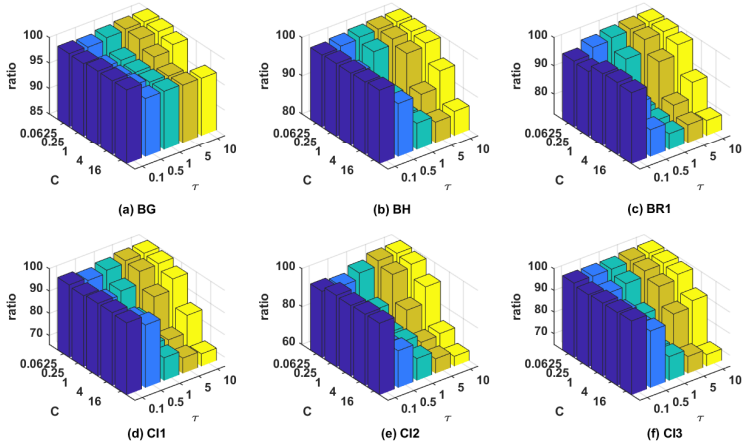

Figure 6: Ratio of subspace elimination across two datasets. The first row is a subset of the Fish dataset and the second row is the MaskedFace-Net.

Later, GAMEEMO is taken as an example to show that our subspace elimination strategy converges very quickly. We compute the similarity [13] between $(\mathbf{U}_s)_l$ ($\mathbf{U}_s$ at the l-th iteration) and $\mathbf{U}^*$, which is measured by the smallest singular value of $(\mathbf{U}^*)^T(\mathbf{U}_s)_l$. If $\mathbf{U}^* \subset span((\mathbf{U}_s)_l)$, this value will be 1. From Fig. 7, we can see the value converges to 1 in ten steps.

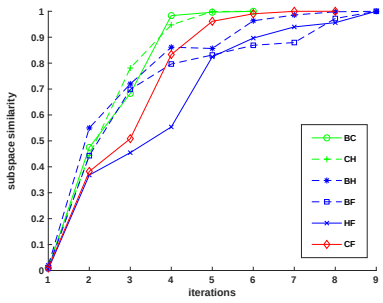

Figure 7: Subspace similarity between active subspace and the final solution on GAMEEMO.

6 Conclusion

In this paper, a novel subspace elimination strategy based on rank-one factorization for L_2-SMM is exploited to reduce computational cost and improve efficacy. Our method could eliminate most of the active subspaces before solving optimization problem, thus we only need to solve a smaller problem. Our approach is performed independently of the solver,

hence other effective solvers can be integrated into this method. We further provide ADMM-L_2-SMM as an efficacious solver in this paper. In our numerical experiments, we carry out the SES-L_2-SMM with other algorithms on multiple real-world datasets. The experimental results verify the validity of our algorithm. How to apply the subspace elimination strategy to other matrix classifiers is worthy of future research.

References

[1] T.B. Alakus, M. Gonen and I. Turkoglu, Database for an emotion recognition system based on EEG signals and various computer games-GAMEEMO, *Biomed. Signal Proces.* 60 (2020): 101951.

[2] A. Bonnefoy, V. Emiya, L. Ralaivola and R. Gribonval, Dynamic screening: accelerating first-order algorithms for the Lasso and group-Lasso, *IEEE Trans. Signal Process.* 63 (2015),5121–5132.

[3] A. Cabani, K. Hammoudi, H. Benhabiles and M. Melkemi, Masked Face-Net-A dataset of correctly/incorrectly masked face images in the context of COVID-19, *Smart Health.* 19 (2021): 100144.

[4] J. F. Cai, E. J. Candès and Z. Shen, A singular value thresholding algorithm for matrix completion, *SIAM J. Optim.* 20(4) (2010), 1956–1982.

[5] X Deng, P Jiang, X. Peng and C. Mi, An intelligent outlier detection method with one class support tucker machine and genetic algorithm toward big sensor data in internet of things, *IEEE Trans. Ind. Electron.* 66 (2019) 4672–4683.

[6] M. Dyrholm, C. Christoforou and L.C. Parra, Bilinear discriminant component analysis, *J. Mach. Learn. Res.* 8 (2007) 1097–1111.

[7] O. Fercoq, A. Gramfort, J. Salmon, Mind the duality gap: Safer rules for the lasso, in: *ICML*, 2015, pp. 333–342.

[8] X. Gao, L. Fan and H. Xu, A novel method for classification of matrix data using twin multiple rank SMMs, *Appl. Soft Comput.* 48 (2016) 546–562.

[9] E. Ghadimi, A. Teixeira, I. Shames and M. Johansson, Optimal parameter selection for the alternating direction method of multipliers (ADMM): quadratic problems, *IEEE Trans. Automat. Contr.* 60 (2014) 644–658.

[10] L.E. Ghaoui, V. Viallon and T. Rabbani, Safe feature elimination in sparse supervised learning, *Pac. J. Optim.* 8 (2012), 667–698.

[11] T. Goldstein, B. O'Donoghue, S. Setzer and R. Baraniuk, Fast alternating direction optimization methods, *SIAM J. Imaging Sci.* 7 (2014) 1588–1623.

[12] N. Halko and P.G. Martinsson and J. Tropp, Finding structure with randomness: Probabilistic algorithms for constructing approximate matrix decompositions, *SIAM Rev.* 53 (2011) 217–288.

[13] C.J. Hsieh and P.A. Olsen, Nuclear norm minimization via active subspace selection, in: *ICML*, 2014, pp. 575–583.

[14] S. Hua and Z. Sun, Support vector machine approach for protein subcellular localization prediction, *Bioinformatics.* 17 (2001) 721–728.

[15] S. Ji and J. Ye, An accelerated gradient method for trace norm minimization, in: *ICML*, 2009, pp. 457–464.

[16] T. Kobayashi and N. Otsu, Efficient optimization for low-rank integrated bilinear classifiers, in: *ECCV*, 2012, pp. 474–487.

[17] X. Lu, W. Liu, C. Zhou and M. Huang, Robust least-squares support vector machine with minimization of mean and variance of modeling error, *IEEE Trans. Neural Netw. Learn. Syst.* 29 (2018), 2909–2920.

[18] L. Luo, Y. Xie, Z. Zhang and W. J. Li, Support matrix machines, in: *ICML*, 2015, pp. 938–947.

[19] J. Mathew, C.K. Pang, M. Luo and W.H. Leong, Classification of imbalanced data by oversampling in kernel space of support vector machines, *IEEE Trans. Neural Netw. Learn. Syst.* 29 (2018) 4065–4076.

[20] L.M. Manevitz and M. Yousef, One-class SVMs for document classification, *J. Mach. Learn. Res.* 2 (2001) 139–154.

[21] R. Mazumder, T. Hastie and R. Tibshirani, Spectral regularization algorithms for learning large incomplete matrices, *J. Mach. Learn. Res.* 11 (2010) 2287–2322.

[22] E. Ndiaye, O. Fercoq, A. Gramfort and J. Salmon, GAP safe screening rules for sparse multi-task and multi-class models, in: *NIPS*, 2015, pp. 811–819.

[23] E. Ndiaye, O. Fercoq, A. Gramfort and J. Salmon, Gap safe screening rules for sparsity enforcing penalties, *J. Mach. Learn. Res.* 18 (2017) 4671–4703.

[24] K. Ogawa, Y. Suzuki and I. Takeuchi, Safe screening of non-support vectors in pathwise SVM computation. in: *ICML*, 2013, pp. 1382–1390.

[25] X. Pan and Y. Xu, A novel and safe two-stage screening method for the support vector machine, *IEEE Trans. Neural Netw. Lear.* 30 (2018) 2263–2274.

[26] Y. Pang, K. Zhang, Y. Yuan and K. Wang, Distributed object detection with linear SVMs, *IEEE Trans. Cybern.* 44 (2014) 2122–2133.

[27] H. Pirsiavash, D. Ramanan and C. C. Fowlkes, Bilinear classifiers for visual recognition, in: *NIPS*, 2009, pp. 1482–1490.

[28] M. Pontil and A. Verri, Support vector machines for 3D object recognition, *IEEE Trans. Pattern Anal. Mach. Intell.* 20 (1998) 637–646.

[29] A. Rakotomamonjy, G. Gasso and J. Salmon, Screening rules for Lasso with non-convex Sparse Regularizers. in: *ICML*, 2019, pp. 5341–5350.

[30] O. Ulucan, D. Karakaya and M. Turkan, A large-scale dataset for fish segmentation and classification, in: *Conf. Intell. Syst. Appl.*, 2020, pp. 1–5.

[31] V. Vapnik, *The Nature of Statistical Learning Theory*, Springer, New York, 1995.

[32] V. Vapnik, *Statistical Learning Theory*, John Wiley, New York, 1998.

[33] J. Wang, P. Wonka and J. Ye, Scaling SVM and least absolute deviations via exact data reduction, in: *ICML*, 2014, pp. 1912–1927.

[34] G. A. Watson, Characterization of the subdifferential of some matrix norms, *Linear Algebra Appl.* 170 (1991), 33–45.

[35] L. Wolf, H. Jhuang and T. Hazan, Modeling Appearances with Low-Rank SVM, in: *CVPR*, 2007, pp. 1–6.

[36] Q. Yao and J. T. Kwok, Accelerated and inexact soft-impute for large-scale Matrix and tensor completion, *IEEE T. Knowl. Data En.* 31 (2019) 1665–1679.

[37] M. Yuan and Y. Xu, Bound estimation-based safe acceleration for maximum margin of twin spheres machine with pinball loss, *Pattern Recogn.* 114 (2021): 107860.

[38] W. Zhang, T. Yoshida and X. Tang, Text classification based on multi-word with support vector machine, *Knowl.-Based Syst.* 21 (2008), 879–886.

[39] Q. Zheng, F. Zhu, J. Qin, B. Chen and P. Heng, Sparse Support Matrix Machine, *Pattern Recogn.* 76 (2018) 715–726.

[40] Q. Zheng, F. Zhu and P. A. Heng, Robust support matrix machine for single trial EEG classification, *IEEE Trans. Neural Syst. Rehabil. Eng.* 26 (2018) 551–562.

[41] J. Zhu, S. Rosset, T. Hastie and R. Tibshirani, L_1-Norm support vector machines, in: *NIPS*, volume 16, 2004, pp. 49–56.

[42] J. Zimmert, C. S. de Witt, G. Kerg and M. Kloft, Safe screening for support vector machines, in: *NIPS*, 2015, pp. 1–5.

[43] H. Zou and T. Hastie, Regularization and variable selection via the elastic net, *Journal of the Royal Statistical Society.* 67 (2005) 301–320.

Manuscript received 30 June 2021
revised 18 September 2021
accepted for publication 5 October 2021

Renxiu Feng
College of Information and Electrical Engineering
China Agricultural University, Beijing 100083, China
E-mail address: s20193081402@cau.edu.cn

Peiwei Zhong
College of Science, China Agricultural University, Beijing 100083, China
E-mail address: S20203101881@cau.edu.cn

Yitian Xu
College of Science, China Agricultural University, Beijing 100083, China
E-mail address: xytshuxue@126.com