Non-Parametric Sparse Matrix Decomposition for Cross-View Dimensionality Reduction

Huawen Liu, Lin Liu, Thuc Duy Le *Member, IEEE*, Ivan Lee *Senior Member, IEEE*, Shiliang Sun *Member, IEEE* and Jiuyong Li *Member, IEEE*

Abstract—Cross-view data are collected from two different views or sources about the same subjects. As the information from these views often consolidate and/or complement each other, cross-view data analysis can gain more insights for decision making. A main challenge of cross-view data analysis is how to effectively explore the inherently correlated and high-dimensional data. Dimension reduction offers an effective solution for this problem. However, how to choose right models and parameters involved for dimension reduction is still an open problem. In this paper we propose an effective sparse learning algorithm for cross-view dimensionality reduction. A distinguished character of our model selection is that it is non-parametric and automatic. Specifically, we represent the correlation of cross-view data using a covariance matrix. Then we decompose the matrix into a sequence of low-rank ones by solving an optimization problem in an alternating least squares manner. More importantly, a new and non-parametric sparsity-inducing function is developed to derive a parsimonious model. Extensive experiments are conducted on real-world data sets to evaluate the effectiveness of the proposed algorithm. The results show that our method is competitive with the state-of-the-art sparse learning algorithms.

Keywords—Cross-view data, Matrix decomposition, Dimension reduction, Sparse learning, Sparsity-inducing function.

1 Introduction

Cross-view data, sampling from the same subjects with two different views, are ubiquitous in reality. For example, the data from one view is a collection of images while the data from another view is a set of captions of the images. Another example would be a data set with video (view 1) and audio signals (view 2) of a multimedia segment. The words "allô?" in French and "¡hola!" in Spanish have the same meaning, i.e., the greeting word "Hello".

Exploring cross-view data may benefit decision making, because such correlated or complementary data may provide more insights. From the learning perspective, building models from cross-view data jointly can lead to better generalization performance [1], [2]. Therefore, cross-view data analysis has attracted increasing attentions from diverse domains, such as image processing [3], object tracking [4], multimedia retrieval [5], bioinformatics, social network and recommended systems [6].

A main challenge for cross-view data analysis is the high-dimensional nature of the data, which may cause many problems, such as collinear and over-fitting, and pose great challenges to traditional learning algorithms [7]. Dimension reduction in this case is indispensable. With this technique, the original data can be re-organized within low-dimensional spaces where the content of the data is captured [8]. Typical examples of dimension reduction methods include principle component analysis (PCA) and canonical correlation analysis (CCA). However, the reduction results are hard to be interpreted, since the deduced latent variables in the low-dimensional spaces are weighted combinations of the original variables.

Sparse learning provides an effective solution to the problem by deriving parsimonious models, which are more comprehensible and interpretable [9]. Since sparse learning has solid mathematical foundations and good properties, it has been extensively studied and widely used in many fields like image and signal processing [10], optimization, machine learning, computer vision and bioinformatics [11]. Representative sparse learning methods include sparse PCA [12], sparse CCA [13]and penalized matrix decomposition (PMD) [14].

However, two major problems must be taken into account when sparse learning methods are used. The first one is which sparsity-inducing norms (or functions) should be adopted in learning models. Sparsity-inducing norms, such as Lasso (Least absolute shrinkage and selection operator) [15] and Elastic net [16], play an essential role in sparse learning. They aim at making the models sparse with some constraint conditions [11]. However, how to determine the right sparsity-inducing norms for a specific application is an open issue.

The second problem with sparse learning is how to set right parameters for a given model. There is no widely acceptable solution at the theoretical and practical aspects, because the parameters are data-driven and

H. Liu is with the Dept. of Computer Science, Zhejiang Normal University, 688 Yingbin Road, Jinhua 321004, P. R. China (e-mail: hwliu@zjnu.edu.cn).
L. Liu, T. D. Le, I. Lee and J. Li are with the School of Information Technology and Mathematical Sciences, University of South Australia, Mawson Lakes, SA 5095, Australia.
S. Sun is with the Dept. of Computer Science and Technology, East China Normal University, 500 Dongchuan Road, Shanghai 200241, P. R. China.

heavily depend on the specific contexts and applications. Parameter tuning is very difficult and time-consuming. Thus they are determined by a priori or empirical knowledge. However, the models with improper parameter values may lead to poor performance, or even contradictory results under the same situations. This hinders the applications of sparse learning in practice.

In this paper, we tackle the above problems by developing a novel, yet effective sparse learning method for cross-view dimensionality reduction. A distinct character of our method comparing with the existing solutions is that it is an automatic and non-parametric one which is less human-intensive. Specifically, we represent the inherent correlations of cross-view data using a covariance matrix. Then the matrix is decomposed into a sequence of low-rank ones in an alternating least squares way. Furthermore, to achieve a parsimonious model, we introduce a new and effective sparsity-inducing function, without involving any regularization parameters. To the best of our knowledge, little work has been done on the topic of non-parametric and automatic model selection so far.

The main contributions of this paper are highlighted as follows:

1) We propose a novel and automatic model selection framework for cross-view dimensionality reduction, which can handle the high-dimensional problems effectively.
2) More importantly, an effective sparsity-inducing function, which can automatically yield a parsimonious model according to the data at hand, is introduced.
3) The optimization function is non-parametric and can be solved by using the technique of alternating least squares regression automatically.

The rest of the paper is organized as follows. Section 2 briefly reviews the state-of-the-art sparse learning methods. Section 3 presents the notation used in the paper and the problem formulation. In Section 4, we describe the proposed non-parametric sparse learning method in detail. Experiments with real-world data sets are provided in Section 5, followed by the concluding remarks of the paper in Section 6.

2 Related Work

Many works related to dimension reduction have been done during the past years. Here we only review the work for cross-view data. More details can be found from good survey papers (see, e.g., [6]) and references therein.

Generally, two strategies are available for investigating the correlations of high-dimensional cross-view data. The first kind exploits matrix analysis techniques to explore the correlations of data. For example, after obtaining the correlation (or covariance) matrices of cross-view data, we can use principle component analysis (PCA) to identify the inherent relationships. Since each principal component of PCA is a linear combination of the original variables, the coefficients are typically nonzero, making the interpretation impossible. To yield sparse results, Zou et al. [12] introduced the Elastic net penalty into PCA and developed the sparse PCA (SPCA) method. Further, Croux et al. [17] exploited projection pursuit to get sparse principle components with many zero loadings. It should be pointed out that SPCA is good at identifying sparse principle components with high data variation, not for extracting the correlation.

Matrix factorization aims at decomposing a matrix into a series of low-rank matrices to find meaningful patterns. According to the additive property, the original matrix can be approximately reconstructed from the low-rank ones with a low loss [18]. Singular value decomposition (SVD) is a typical decomposition technique to analyze two-way dependencies. Like PCA, the interpretability of the derived models by SVD is relatively poor. To tackle this problem, Lee et al. [19] imposed the adaptive Lasso penalty on the left and right singular vectors to achieve sparse models. Recently, Hong and Lian [20] considered both smoothness and sparse properties of SVD by using the quadratic and Lasso penalties,while Yang et al. [21] extracted sparse and orthogonal singular vectors.

Non-negative matrix factorization (NMF), which takes the non-negative constraint into account, is a special decomposition technique. Besides, some additional constraints, like Lasso and the geometric penalty, are also involved in NMF [22].Tang et al. [23] exploited a sparse NMF to extract localities in videos and then developed a novel sparse ensemble learning scheme for concept detection. Liu et al. [24] extended NMF by incorporating the label information as additional constraints for semi-supervised image classification. Tan et al. [25] adopted supervised tensor factorization to explore structured visual features from mutually correlated multimedia data collections for classification. Similarly, Eweiwi et al. [26] took use of NMF to extract action-specific points or spatial regions of interest in still images from videos to identify human actions.

The second kind of correlation analysis is multivariate data analysis. CCA is a representative example, which mainly seeks a pair of linear transformations each for one set of variables such that the correlation of the projected variables is maximized [27]. CCA is a popular technique of multiview or multimodal feature learning from multimedia data [28]. For instance, Izadinia et al. [29] utilized CCA to identify the moving objects which are most correlated to the audio signal in audio-visual dynamics of videos, while Sargin et al. [30] resorted to CCA to implement multimodal fusion and synchronization method of the speech and lip features for open-set speaker identification. Since the classical CCA may have poor performance, several variants of CCA have been developed. As a typical example, Parkhomenko et al. [13] applied the Lasso penalty to enforce the loadings to be sparse. Yuan et al. [31] alleviated the deviations

of eigenvalues and singular values of CCA by virtue of fractional order and then developed a dimension reduction method for multi-view data. LapMCC [32] extends CCA with local structure information of nearest neighbor graphs to represent nonlinear correlation of multiview data in face and object image recognition.

Partial Least Squares (PLS), originally developed for econometrics and chemometrics, is another powerful and versatile technique to ease the multi-collinear problem stemmed from the "large p, small n" data [33]. PLS, which explores the correlations of sets of variables via regression, has also been widely applied in the field of multimedia. For example, Bakry and Elgammal [34] utilized kernel PLS to explore intrinsic nonlinear relations between speech contents and speakers. Wang et al. [35] constructed a no-reference assessment model for video quality, where PLS was used to get the relation of visual contents and network conditions. Zhong et al. [36] fulfilled the purpose of object tracking and segmentation in one stage by using PLS with structured labeling information. In [37], the correlation of object appearance and class labels from foreground and background is captured by PLS for object tracking. However, conventional PLS may encounter the over-fitting problem, and thus several sparse versions of PLS have been developed for the purpose of variable selection and dimension reduction. Sparse PLS (SPLS) [38] is a typical example, whose constraint condition is the ℓ_1 penalty. Li et al. [39] built a human detection model on the basis of channel features, which extracted by using sparse PLS with discriminative analysis.

The common character of the dimension reduction methods mentioned above is that they exploit different sparsity-inducing functions to make the derived models parsimonious [7]. However, selecting a right sparsity-inducing function for models becomes a difficult issue in reality, because it is varied from the specifical problems. Additionally, the more the regularization parameters involved, the higher complexity of the models and the higher probability of encountering the over-fitting problem. These motivate the work of this paper on an non-parametric sparse learning method for cross-view dimension reduction, where the approximate optimization values of parameters are determined adaptively.

3 Problem Statement

Notation: Hereafter we assume that bold-faced lower case letters, e.g., $\mathbf{u}$ and $\mathbf{v}$, represent (column) vectors, and u_i indicates the i-th element of $\mathbf{u}$. Assume that $\mathbf{u}$ has p elements. Let $\mathcal{I}_p \subseteq \{1, .., p\}$ be an index set. $\mathbf{u}_{\mathcal{I}_p}$ is a sub-vector of $\mathbf{u}$, where each element of $\mathbf{u}_{\mathcal{I}_p}$ is also an element of $\mathbf{u}$. We take "$\sqsubseteq$" as a sparse structure relationship. For example, $\tilde{\mathbf{u}} \sqsubseteq \mathbf{u}$ denotes that $\tilde{\mathbf{u}}$ is a parsimonious vector of $\mathbf{u}$, where each element $\tilde{u}_i$ of $\tilde{\mathbf{u}}$ equals to zero or u_i. Besides, the transpose and the Frobenius norm of $\mathbf{u}$ are denoted as $\mathbf{u}^T$ and $\|\mathbf{u}\|_2$ respectively, where $\|\mathbf{u}\|_2^2 = \sum_i u_i^2$. For clarity, we always use the letters $\mathbf{x}$ and $\mathbf{y}$ to represent the two random variables (feature vectors) in the two views.

The problem: Let $\mathbf{x}_i \in \mathbb{R}^p$ and $\mathbf{y}_i \in \mathbb{R}^q$ be a sample of $\mathbf{x}$ and $\mathbf{y}$, respectively. $\mathbf{X}$=$[\mathbf{x}_1 ... \mathbf{x}_n]^T \in \mathbb{R}^{n \times p}$ and $\mathbf{Y}$=$[\mathbf{y}_1 ... \mathbf{y}_n]^T \in \mathbb{R}^{n \times q}$ consisting of n samples represent data collected from two different views characterized by p and q features (variables) respectively. As the dimensions p and q become large, the efficiency of building models from $\mathbf{X}$ and $\mathbf{Y}$ is very low.Besides, other problems like over-fitting and collinear also arise, resulting in the models with poor performance. Dimension reduction is an effective solution to alleviate the high-dimensional problems. It re-organizes original data within low-dimensional spaces to reveal significant latent information to build models.

Let $\tilde{\mathbf{X}}$ and $\tilde{\mathbf{Y}}$ be the projections of $\mathbf{X}$ and $\mathbf{Y}$ in the latent variable spaces $\mathbf{U}$=$\{\mathbf{u}_1, .., \mathbf{u}_{k_u}\}$ and $\mathbf{V}$=$\{\mathbf{v}_1, .., \mathbf{v}_{k_v}\}$, respectively. Usually, $\mathbf{U}$ and $\mathbf{V}$ have the same number of components when the correlation of $\mathbf{X}$ and $\mathbf{Y}$ is considered, that is, $k_u = k_v$. The purpose of dimension reduction is to obtain $\mathbf{U}$ and $\mathbf{V}$ such that the information loss through the projection is minimal, i.e.,

$$\arg\min \sum_{\mathbf{u}_i, \mathbf{v}_i} \ell(\mathbf{X}, \mathbf{Y} \| \tilde{\mathbf{X}}, \tilde{\mathbf{Y}}), \tag{1}$$

where ℓ is a loss function. Since $\tilde{\mathbf{X}}$=$\mathbf{X}\mathbf{U}$ and $\tilde{\mathbf{Y}}$=$\mathbf{Y}\mathbf{V}$ are the weighted combinations of $\mathbf{X}$ and $\mathbf{Y}$ respectively, they are hard to be interpreted or understood.

Making $\tilde{\mathbf{X}}$ and $\tilde{\mathbf{Y}}$ sparse seems to be an appealing solution, because parsimonious results are more preferable and acceptable than complex ones. Sparse learning offers such solutions of sparsity by imposing constraints on $\mathbf{u}_i$ and $\mathbf{v}_i$ in Eq.(1) as follows:

$$\arg\min \sum_{\theta, \mathbf{u}_i, \mathbf{v}_i} \ell(\mathbf{X}, \mathbf{Y} \| \tilde{\mathbf{X}}, \tilde{\mathbf{Y}}) + R(\mathbf{u}_i, \mathbf{v}_i, \theta), \tag{2}$$

where $R(\mathbf{u}_i, \mathbf{v}_i, \theta)$ is a sparsity-inducing function. It aims at enforcing $\mathbf{u}_i$ and $\mathbf{v}_i$ to be sparse by shrinking some components of $\mathbf{u}_i$ and $\mathbf{v}_i$ to be zero, under the constraint conditions θ, which is a set of regularization parameters.

For Eq.(2), it has different representations, resulting in generating different models. For example, the objective optimization function of the sparse CCA (SCCA) developed by Parkhomenko et al. [13] is

$$f(\mathbf{u}, \mathbf{v}) = \arg\min_{\mathbf{u}, \mathbf{v}} \|\mathbf{X}\mathbf{u} - \mathbf{Y}\mathbf{v}\|_2^2 + \lambda_u \sum |u_i| + \lambda_v \sum |v_i|, \tag{3}$$

where $\|\mathbf{X}\mathbf{u}\|_2^2 = 1$ and $\|\mathbf{Y}\mathbf{v}\|_2^2 = 1$. The sparse PLS (SPLS) [38] has the same sparsity-inducing function $R(\mathbf{u}_i, \mathbf{v}_i, \theta)$ to SCCA, but with different $\ell(\mathbf{X}, \mathbf{Y} \| \tilde{\mathbf{X}}, \tilde{\mathbf{Y}})$:

$$f(\mathbf{u}, \mathbf{v}) = \arg\min_{\mathbf{u}, \mathbf{v}} \|\mathbf{X}^T\mathbf{Y} - \mathbf{u}\mathbf{v}^T\|_2^2 + \lambda_u \sum |u_i| + \lambda_v \sum |v_i|, \tag{4}$$

where $\mathbf{u}^T\mathbf{u} = 1$ and $\mathbf{v}^T\mathbf{v} = 1$. Contrastively, the sparse SVD (SSVD) [19] takes use of the following function as its optimization problem:

$$f(\mathbf{u}, \mathbf{v}) = \arg\min_{\mathbf{u}, \mathbf{v}} \|\mathbf{X} - \delta\mathbf{u}\mathbf{v}\|_2^2 + \lambda_u \sum |\delta u_i| + \lambda_v \sum |\delta v_i|. \tag{5}$$

How to choose the right loss and sparsity-inducing functions for given data is still an open issue. Moreover, the regularization parameters involved within the functions make the problem more complicated. With more parameters, the models have higher complexity. Besides, the models tend to over-fit data, notwithstanding they have better performance during the training stage.

Our goal: In this paper we propose an automatic model selection method for cross-view dimension reduction. Specifically, our goal is threefold: fusing **X** and **Y** together so that we can analyze them jointly; seeking the latent variable spaces of **X** and **Y** such that the information loss is minimal; and obtaining a parsimonious learning model by using a new sparsity-inducing function without involving the regularization parameters.

4 Non-parametric Sparse Learning

Our learning method consists of three major stages: correlation representation, dimension reduction and sparsity making without regularization parameters. Specifically, we make use of the covariance matrix of **X** and **Y** to represent the correlation between them. Then we perform dimension reduction on the matrix with the technique of alternating least squares regression to obtain the latent variable spaces **U** and **V**. Finally, an effective and non-parametric sparsity-inducing function is introduced to enforce **U** and **V** to be sparse.

4.1 Matrix decomposition

The cross-view data **X** and **Y**, collected from different views, are semantically correlated as they describe the same subjects. This property may bring more insights and help with building models. We consider the covariance matrix Σ_{XY} of **X** and **Y** to express their correlations, because it can effectively measure the degree to which two variables vary together.

An entry, σ_{ij}=$E((\mathbf{X}_{.i} - \mu_{\mathbf{X}_{.i}})^T(\mathbf{Y}_{.j} - \mu_{\mathbf{Y}_{.j}}))$, of Σ_{XY} is the covariance of $\mathbf{X}_{.i}$ and $\mathbf{Y}_{.j}$, where $\mathbf{X}_{.i}$ (or $\mathbf{Y}_{.j}$) is the i-th (or j-th) column vector of **X** (or **Y**). $\mu_{\mathbf{X}_{.i}}$ and $\mu_{\mathbf{Y}_{.j}}$ are the expected values of $\mathbf{X}_{.i}$ and $\mathbf{Y}_{.j}$ respectively. If $\mathbf{X}_{.i}$ and $\mathbf{Y}_{.j}$ vary together in the same direction, their covariance is positive; otherwise the covariance is negative. Contrastively, their covariance is zero as there is no linear dependency between them. For the sake of simplicity, we assume that the columns of **X** and **Y** have been standardized to have zero mean and unit deviation.

Matrix decomposition is an effective technique to analyze and visualize high-dimensional data matrices. Since it can reveal the inherent characteristic and underlying structure of a matrix, helping users to interpret the meaning readily, it has now become a popular tool in data analysis.

Generally, the covariance matrix $\Sigma_{XY} \in \mathbb{R}^{p\times q}$ of **X** and **Y** can be decomposed to the following form:

$$\Sigma_{XY} = \mathbf{U}\mathbf{D}\mathbf{V}^T = \sum_{i=1}^{r} \delta_i \mathbf{u}_i \mathbf{v}_i^T \tag{6}$$

where r is the rank of Σ_{XY}, $\mathbf{D}$=$diag(\delta_1, ..., \delta_r)$ is a diagonal matrix such that $\delta_1 \geq \delta_2 \geq ... \geq \delta_r$. $\mathbf{u}_i$ and $\mathbf{v}_i$ are the left and right vectors corresponding to δ_i respectively. This definition means that the information embodied within Σ_{XY} can be summarized into δ_i, $\mathbf{u}_i$ and $\mathbf{v}_i$.

In real applications only the top k ($k < r$) values and vectors are taken into account. The rest elements corresponding to small values are often regarded as noise and have less useful information. As a result, Σ_{XY} can be approximately expressed as the sum of k rank-one matrices:

$$\Sigma_{XY} \approx \Sigma_{XY}^k = \sum_{i=1}^{k} \delta_i \mathbf{u}_i \mathbf{v}_i^T. \tag{7}$$

The top k left and right vectors can be extracted in a sequencial manner. Since the first pair of vectors **u** and **v** (i.e. $\mathbf{u}_1$ and $\mathbf{v}_1$ in Eq.(7), but for simplicity, the subscripts will be omitted in the rest of the paper), as well as the largest value δ of **D** (i.e. δ_1, and again for simplicity, the subscript is omitted hereafter), cover predominant information of Σ_{XY}, $\delta\mathbf{u}\mathbf{v}^T$ is the best rank-one matrix approximation of Σ_{XY}. If the loss function ℓ in Eq.(1) is the squared Frobenius norm, **u** and **v** can be extracted by solving the following optimization problem:

$$\begin{aligned} \arg\min_{\mathbf{u},\mathbf{v}} \quad & \frac{1}{2}\|\Sigma_{XY} - \delta\mathbf{u}\mathbf{v}^T\|_2^2 \\ s.t. \quad & \mathbf{u}^T\mathbf{u} = 1, \mathbf{v}^T\mathbf{v} = 1 \end{aligned} \tag{8}$$

To solve this optimization problem, our method adopts the Lemma 1.

Lemma 1: Assume that δ is the largest value of **D** extracted from Σ_{XY} by matrix decomposition. The solution of (8) is equivalent to the solution of following optimization problem:

$$\begin{aligned} \arg\max_{\mathbf{u},\mathbf{v}} \quad & \mathbf{u}^T\Sigma_{XY}\mathbf{v} \\ s.t. \quad & \mathbf{u}^T\mathbf{u} = 1, \mathbf{v}^T\mathbf{v} = 1 \end{aligned} \tag{9}$$

Note that the left and right singular vectors **u** and **v** of (9) are the first eigenvectors of $\Sigma_{XY}\Sigma_{XY}^T$ and $\Sigma_{XY}^T\Sigma_{XY}$ respectively. Thus, an intuitive solution of getting **u** and **v** is to obtain the first eigenvectors of $\Sigma_{XY}\Sigma_{XY}^T$ and $\Sigma_{XY}^T\Sigma_{XY}$ through SVD. Although this method sounds simple and good, its efficiency is questionable, especially when the size of Σ_{XY} is very large.

A cunning solution of getting **u** and **v** is the power iteration method. Indeed, the objective function of (9) is bilinear with respect to **u** and **v**, that is, if **v** (or **u**) is fixed the function is linear in **u** (or **v**). According to this property, **u** and **v** can be obtained via the power iteration method in an alternating way. Specifically, given an initial value $\mathbf{v}^{(0)}$ of **v**, we estimate the values $\mathbf{u}^{(i)}$ and $\mathbf{v}^{(i)}$ of **u** and **v** at the i-th ($i \geq 1$) iteration as follows:

$$\mathbf{u}^{(i)} = \frac{\Sigma_{XY}\mathbf{v}^{(i-1)}}{\|\Sigma_{XY}\mathbf{v}^{(i-1)}\|_2}, \quad \mathbf{v}^{(i)} = \frac{\Sigma_{XY}^T\mathbf{u}^{(i)}}{\|\Sigma_{XY}^T\mathbf{u}^{(i)}\|_2}. \tag{10}$$

Consequently, the final values $\mathbf{u}^{(t)}$ and $\mathbf{v}^{(t)}$ can be obtained as the desired results after t iterations.

The rest singular vectors $\mathbf{u}_i$ and $\mathbf{v}_i$ $(i = 2, .., k)$ can be extracted by applying the same procedure in a sequential way. For example, once the first singular vectors $\mathbf{u}$ and $\mathbf{v}$ and δ are obtained, they will be removed from Σ_{XY}. Thereafter, the next pair of singular vectors can be obtained by using the same extraction procedure on the residual matrix. The process is repeated until there is no information encoded in the residual matrix or no enough vectors are derived.

4.2 Non-parametric sparsity-inducing function

Interpreting and visualizing the derived results are very important in real-world scenarios, because they can help users to understand the models and make right decisions. However, as the dimensionality of data is high, interpretation and visualization become difficult and even impossible in some cases, because the derived results are the linear combinations of the original spaces. They may have good statistical properties, but have no physical meanings.

Sparse learning offers an effective strategy to improve the interpretability and help with visualization. It makes the derived results with parsimonious structures by enforcing their small-valued coefficients to zero under some constraints. Sparse learning is a hot topic in machine learning, and many sparsity-inducing functions have been witnessed and extensively studied. Typical examples include Lasso, Elastic net, group Lasso, fused Lasso and SCAD (smoothly clipped absolute deviation) [11].

Assume that $\tilde{\mathbf{u}}$ is a parsimonious counterpart of $\mathbf{u}$. They have element-wise one-to-one relationships. Let u_i be the i-th coefficient of $\mathbf{u}$. For the penalty function of Lasso, if the absolute value of u_i is large, depending on the sign of u_i, $\tilde{u}_i$ equals to $u_i - \lambda$ or $u_i + \lambda$ (λ is a regularization parameter) for the soft-thresholding function, or keeps unchanged for the hard-thresholding function. On the other hand, if the absolute value of u_i is small enough, both penalty functions enforce the corresponding value $\tilde{u}_i$ to zero.

Note that the degree of sparsity and the performance of models heavily rely on the function $R(\mathbf{u}, \mathbf{v}, \theta)$ and the regularization parameters θ. These parameters are data-driven and should be carefully tuned because different values may lead to different results. Unfortunately, assigning the parameters with appropriate values is a difficult task. There is no effective way to determine the optimal values for the parameters. A frequently adopted strategy is to take prior knowledge into consideration and empirically determine the optimal values with methods like cross-validation, sub-sampling, the mean squared prediction error (MSPE), Akaike information criterion (AIC) or Bayesian information criterion (BIC). This, however, will lead to high complexity and computational cost, and even over-fitting situation.

To cope with these problems, we introduce a new and non-parametric sparsity-inducing function. Our sparsity-inducing solution aims at seeking the parsimonious vectors $\tilde{\mathbf{u}} \sqsubseteq \mathbf{u}$ and $\tilde{\mathbf{v}} \sqsubseteq \mathbf{v}$ such that the value of $\tilde{\mathbf{u}}^T \Sigma_{XY} \tilde{\mathbf{v}}$ is maximal. Formally, given $\mathbf{u}$ and $\mathbf{v}$, we aim at solving the following optimization problem:

$$\begin{aligned} \arg\max \quad & \tilde{\mathbf{u}}^T \Sigma_{XY} \tilde{\mathbf{v}} \\ s.t. \quad & \tilde{\mathbf{u}} \sqsubseteq \mathbf{u}, \tilde{\mathbf{v}} \sqsubseteq \mathbf{v}, \tilde{\mathbf{u}}^T \tilde{\mathbf{u}} = 1, \tilde{\mathbf{v}}^T \tilde{\mathbf{v}} = 1 \end{aligned} \tag{11}$$

If the parsimonious vectors are the same as the original ones, i.e., $\tilde{\mathbf{u}} = \mathbf{u}$ and $\tilde{\mathbf{v}} = \mathbf{v}$, the problem of Eq.(11) is equivalent to Eq.(9). This implies that Eq.(11) is consistent with Eq.(9) and can be considered as a sub-optimization problem of Eq.(9) to some extent.

We resort to a heuristic strategy to approximately solve the optimization problem of Eq.(11) by exploiting inherent properties of the vectors. Indeed, solving Eq.(11) by checking each pair of $\tilde{\mathbf{u}}$ and $\tilde{\mathbf{v}}$ with a brute-force search approach is impractical, because there are 2^{p+q} pairs of $\tilde{\mathbf{u}}$ and $\tilde{\mathbf{v}}$, where p and q are the numbers of elements contained within $\mathbf{u}$ and $\mathbf{v}$ respectively, .

Generally, there are two kinds of internal relationships among $\mathbf{u}$, $\mathbf{v}$ and their counterparts (i.e., $\tilde{\mathbf{u}}$ and $\tilde{\mathbf{v}}$). The first one is the relationship between $\mathbf{u}$ and $\mathbf{v}$. As we know, Σ_{XY} generalizes the variances among $\mathbf{X}$ and $\mathbf{Y}$, where each element σ_{ij} is the covariance between the column vectors $\mathbf{X}_{.i}$ and $\mathbf{Y}_{.j}$. Thus, the first pair of vectors $\mathbf{u}$ and $\mathbf{v}$ represent the direction in which $\mathbf{X}$ and $\mathbf{Y}$ have the maximal variances. From the perspective of covariance, it can be safely interpreted that $\mathbf{u}$ is relevant to $\mathbf{v}$ with respect to Σ_{XY}.

When obtaining $\tilde{\mathbf{u}}$ and $\tilde{\mathbf{v}}$, the type of correlation should also be considered, otherwise the consistent property would not hold any more. Since $\mathbf{u}$ and $\mathbf{v}$ are the optimization solution of Eq.(9), they are positively correlated with each other with respect to Σ_{XY}. Therefore, $\tilde{u}_i$ and $\tilde{v}_i$ should also be positively correlated with respect to Σ_{XY}.

The second kind of relationships is the one-to-one mapping between $\mathbf{u}$ (or $\mathbf{v}$) and its corresponding parsimonious vector $\tilde{\mathbf{u}}$ (or $\tilde{\mathbf{v}}$). As mentioned above, for each coefficient (i.e., element) u_i of $\mathbf{u}$, if its absolute value is large, it will be preserved or changed slightly in $\tilde{\mathbf{u}}$ after the sparsity-inducing operation. On the contrary, if the absolute value of u_i is small enough, $\tilde{u}_i$ will be set to zero. This fact implies that the coefficients with large absolute values have strong and powerful effects, while the small ones are less useful.

These two properties give us the clues to solve the optimization problem of Eq.(11). The first clue is to sort the coefficients of $\mathbf{u}$ and $\mathbf{v}$. The purpose of sorting is to keep the coefficients in the same covariance direction. The second clue is to place more emphasis on the coefficients with large absolute values and less emphasis on the rests, when performing the sparsity-inducing operation.

Keeping these two ideas in mind, we propose a new sparsity-inducing function as formalized in the following lemma:

Lemma 2: Assume that $\mathbf{x} \in \mathbb{R}^n$ and $\mathbf{y} \in \mathbb{R}^n$ are two sorted vectors with an increasing order, and $\mathcal{I}_i \subseteq$

$\{1,...,i\}(i \leq n)$ is an index set such that $\mathbf{x}_{\mathcal{I}_i}^T\mathbf{y}_{\mathcal{I}_i}$ is maximal. There exists a parsimonious vector $\tilde{\mathbf{x}}$ of $\mathbf{x}$, such that $\tilde{\mathbf{x}}^T\mathbf{y}$= $\mathbf{x}_{\mathcal{I}_n}^T\mathbf{y}_{\mathcal{I}_n}$, where

$$\tilde{x}_i = \begin{cases} 0, & \mathbf{x}_{\{1..i\}}^T\mathbf{y}_{\{1..i\}} \leq \mathbf{x}_{\mathcal{I}_{i-1}}^T\mathbf{y}_{\mathcal{I}_{i-1}}; \\ x_i, & Otherwise. \end{cases} \quad (12)$$

proof. Let $\mathcal{S}_i$ be the maximal sub-sum of product of $\mathbf{x}$ and $\mathbf{y}$ from the first to the i-th element, i.e., $\mathcal{S}_i = \max_{S\subseteq\{1,...,i\}}(\mathbf{x}_S^T\mathbf{y}_S)$. Let $\mathcal{I}_i \subseteq \{1,...,i\}$ be the index set of $\mathbf{x}$ (or $\mathbf{y}$) leading to $\mathcal{S}_i$. Thus, we have $\mathcal{S}_i = \mathbf{x}_{\mathcal{I}_i}^T\mathbf{y}_{\mathcal{I}_i}$.

For the sparse vector $\tilde{\mathbf{x}} \sqsubseteq \mathbf{x}$, we set the first element of $\tilde{\mathbf{x}}$ as the first one of $\mathbf{x}$, i.e., $\tilde{x}_1 = x_1$, and the rests zero. In this case, we have $\mathcal{S}_1 = \tilde{x}_1 y_1$ and $\mathcal{I}_1 = \{1\}$.

Assume that for the first $i-1$ elements, we have $\mathcal{I}_{i-1}\subseteq\{1..i-1\}$ and $\mathcal{S}_{i-1} = \tilde{\mathbf{x}}^T\mathbf{y} = \mathbf{x}_{\mathcal{I}_{i-1}}^T\mathbf{y}_{\mathcal{I}_{i-1}} \geq \mathbf{x}_{\{1..i-1\}}^T\mathbf{y}_{\{1..i-1\}}$. This means that the first $i-1$ elements of the parsimonious vector $\tilde{\mathbf{x}}$ is identified. Now let's turn our attention to the i-th element of $\tilde{\mathbf{x}}$. In terms of the definition, we know that

$$\begin{aligned} \mathcal{S}_i &= \max(\mathcal{S}_{i-1}, \mathbf{x}_{\{1..i\}}^T\mathbf{y}_{\{1..i\}}) \\ &= \max(\mathcal{S}_{i-1}, \mathbf{x}_{\{1..i-1\}}^T\mathbf{y}_{\{1..i-1\}} + x_i y_i) \end{aligned}$$

On one hand, if $\mathcal{S}_{i-1} < \mathbf{x}_{\{1..i\}}^T\mathbf{y}_{\{1..i\}}$, we should keep the element x_i down, i.e., $\tilde{x}_i = x_i$ and $\mathcal{I}_i = \mathcal{I}_{i-1}\cup\{i\}$, because the i-th element can increase the sum of the previous ones. On the other hand, $\mathcal{S}_{i-1} \geq \mathbf{x}_{\{1..i\}}^T\mathbf{y}_{\{1..i\}}$ means that $\mathbf{x}_i$ can not bring any change to $\mathcal{S}_{i-1}$, because the maximal sum of product of $\mathbf{x}_{\{1..i\}}$ and $\mathbf{y}_{\{1..i\}}$ is still equal to $\mathcal{S}_{i-1}$ identified previously. In this case, the fact of $\tilde{x}_i = 0$ will not change the fact that the sum of product of $\mathbf{x}_{\mathcal{I}_{i-1}}$ and $\mathbf{y}_{\mathcal{I}_{i-1}}$ is maximal.

When $i = n$, the parsimonious vector $\tilde{\mathbf{x}}$ of $\mathbf{x}$ is generated, and ultimately we have $\mathcal{I}_n \subseteq \{1,...,n\}$ and $\tilde{\mathbf{x}}^T\mathbf{y} = \max_{\mathcal{I}_n}\mathbf{x}_{\mathcal{I}_n}^T\mathbf{y}_{\mathcal{I}_n}$. □

Lemma 2 provides a good instruction to make the vectors sparse. We can approximately solve the optimization problem of Eq.(11) according to Eq.(12) in an alternating way. Specifically, let $g(\mathbf{u},\mathbf{v}) = \mathbf{u}^T\Sigma_{XY}\mathbf{v}$, s.t. $\mathbf{u}^T\mathbf{u} = 1, \mathbf{v}^T\mathbf{v} = 1$. The parsimonious vector $\tilde{\mathbf{u}} \sqsubseteq \mathbf{u}$ can be obtained as $\mathbf{x} = \mathbf{u}$ and $\mathbf{y} = \Sigma_{XY}\mathbf{v}$ in Eq.(12) when $\mathbf{v}$ is fixed. Similarly, the parsimonious vector $\tilde{\mathbf{v}} \sqsubseteq \mathbf{v}$ of $\mathbf{v}$ can be obtained as $\mathbf{x} = \mathbf{v}$ and $\mathbf{y} = \Sigma_{XY}^T\mathbf{u}$ (or $\mathbf{y} = \Sigma_{XY}^T\tilde{\mathbf{u}}$) in Eq.(12) when $\mathbf{u}$ (or $\tilde{\mathbf{u}}$) is fixed.

4.3 The non-parametric sparse learning method

The proposed Non-parametric Sparse Matrix Decomposition (NSMD) method is summarized in Alg. 1. It mainly consists of four steps, i.e., getting $\mathbf{u}$ and $\mathbf{v}$, making them sparse, normalizing $\mathbf{u}$ and $\mathbf{v}$, and updating the residual matrix.

The first two steps correspond to the optimization problems discussed previously (see, Eq.(9) and Eq.(11)). Before making $\mathbf{u}$ and $\mathbf{v}$ sparse, they should be sorted in decreasing or ascending order to keep the same direction of covariance. After the sparsity-inducing stage, $\mathbf{u}$ and $\mathbf{v}$ are not of unit norms. Thus, they are normalized in the third step. The purpose of the last step is to obtain the residual matrix of Σ_{XY} by subtracting the effects of the preceding vectors found, so the subsequent vectors can be extracted sequentially.

Alg. 1 NSMD: Non-parametric Sparse Matrix Decomposition

Input: The cross-view data $\mathbf{X} \in \mathbb{R}^{n\times p}$ and $\mathbf{Y} \in \mathbb{R}^{n\times q}$;
Output: Sparse coefficient matrices $\mathbf{U} \in \mathbb{R}^{p\times k}$ and $\mathbf{V} \in \mathbb{R}^{q\times k}$;
1: Obtain the covariance matrix Σ_{XY} of $\mathbf{X}$ and $\mathbf{Y}$;
2: **For** i=1,...,k **do**
3: Initialize $\mathbf{u}$ and $\mathbf{v}$ with unit norm;
4: Get initial values of $\mathbf{u}$ and $\mathbf{v}$ in terms of Eq.(**??**);
5: Make $\mathbf{u}$ and $\mathbf{v}$ sparse according to Eq.(12), i.e., $\mathbf{u}\leftarrow g(\mathbf{u},\Sigma_{XY}\mathbf{v})$; $\mathbf{v}\leftarrow g(\mathbf{v},\Sigma_{XY}^T\mathbf{u})$;
6: Normalize $\mathbf{u}$ and $\mathbf{v}$;
7: Combine $\mathbf{u}$ and $\mathbf{v}$ into $\mathbf{U}$ and $\mathbf{V}$ respectively ;
8: $\delta \leftarrow tr(\Sigma_{XY}\mathbf{v}\mathbf{u}^T)$;
9: Update the residual matrix: $\Sigma_{XY} \leftarrow \Sigma_{XY} - \delta\mathbf{u}\mathbf{v}^T$;
10: **End**
11: Return $\mathbf{U}$ and $\mathbf{V}$ as the final results.

In Alg. 1, the number of iterations, k, is the desirable number of the singular vectors. Its value can be determined adaptively by using prior knowledge. For example, it can be set to r, the rank of Σ_{XY}, or a pre-specified constant for simplicity. Besides, the cross validation can also be used to set an optimal value to k for a specific application. For simplicity, in our subsequent experiments it was simply assigned to the minimal dimensionality of the training data set.

The proposed learning algorithm has relatively low computational costs. For our method, the most time-consuming steps are getting the singular vectors $\mathbf{u}$ and $\mathbf{v}$ (i.e., Line 4), and marking them sparse (i.e., Line 5). Let $\mathbf{X} \in \mathbb{R}^{n\times p}$, $\mathbf{Y} \in \mathbb{R}^{n\times q}$ and $p > q$. The time complexity of the former is $O(tnp)$, where t is the number of convergence iterations. t is normally set as a constant (e.g., 100). Our experiments show that the convergence iteration was often terminated within 30. The time complexities of the latter (i.e., sorting and sparse) stages are $O(p\log p)$ and $O(p)$. Consequently, the total computational cost of NSMD is $O(ktnp + kp\log p)$.

5 Experiments and Results

5.1 Experimental settings

To demonstrate the effectiveness of the proposed method, we made a comparison between NSMD and the following four kinds of popular sparse learning algorithms with the ℓ_1 and $\ell_{1,2}$-norm penalties. They are SCCA [13], SSVD [19], SPLS [38] and PMD [14]. As discussed above, they stand for four different kinds of learning techniques. Since their objective functions only involve the ℓ_1-norm penalty, we also took the $\ell_{1,2}$-norm penalty into consideration to make a more comprehensive comparison.

Apart from the sparse algorithms, two traditional learning methods, CCA and ALPCCA [40], were also

used to compare with NSMD in the experiments. ALPCCA is a supervised variant of CCA for classification problems. It exploits the local structure information of data within clusters to enhance the discriminative capability of classifiers.

We applied these learning algorithms to classification problems and investigated their classification performance. Since the nearest neighbor (1-NN) classifier is one of the most widely used learning methods in practice, we chose it as our base classifier for the sake of simplicity. All the experiments were conducted in the R environment (version 3.0.1) running on a PC with dual core 3.4GHz CPU, 4GB RAM and 64-bit Operation System.

5.2 Results and discussions

5.2.1 Handwritten numeral recognition

Handwritten digit recognition is a classical and popular research topic in pattern recognition. The handwritten digit data used in our experiments were downloaded from the UCI machine learning repository[1]. There are six data sets representing different features of the handwritten digits from '0' to '9' that were extracted from a collection of Dutch utility maps. A summary of the data sets is provided in the top part of Table 1. Each of them contains 2000 samples (i.e. patterns of '0'-'9'). The samples are divided evenly into 10 groups, representing the patterns of the digits from '0' to '9' respectively. That is, the first 200 samples corresponds to '0', followed by sets of 200 samples for each of '1'-'9'.

TABLE 1
Summary of the experimental data sets

Name	#Samp.	#Var.	#Lab.	Description
Handwritten digit				
fac	2000	216	10	Fourier coefficients
fou	2000	76	10	profile correlations
kar	2000	64	10	Karhunen-Love coefficients
mor	2000	6	10	morphological features
pix	2000	240	10	pixel averages
zer	2000	47	10	Zernike moments
Yale face				
Ori	165	4096	15	The original data (64×64)
LBP	165	944	15	LBP format
Wav	165	1024	15	Wav format
Course				
View1	1051	87	2	Course information
View2	1051	2332	2	Web text

In the experiments, each time we took two data sets as $\mathbf{X}$ and $\mathbf{Y}$ to test the performance of the sparse learning algorithms. Thus, in total there are fifteen different pairs of data sets obtained from the six data sets. With each pair of data sets, we performed the sparse learning algorithms and extracted k pairs of $\mathbf{u}$ and $\mathbf{v}$ from the training data. As described in the previous section, k was assigned to the minimal value of the sizes of the training data.

Firstly, NSMD is used to compare with the sparse learning algorithms with the ℓ_1-norm penalty, where λ_u (λ_v) was equal to 0.4, 0.05, 0.05 and 0.1 for SCCA, SSVD, SPLS and PMD respectively. The reason of setting such values for λ_u and λ_v was that the sparse learning algorithms achieved their best performance (see Fig. 1) with these values. The experimental results are given in Table 2, where the values in bold-face represent the best results in the same row.

From Table 2, we know that NSMD outperformed the other parametric sparse learning algorithms in many cases, notwithstanding it is non-parametric. For instance, NSMD slightly under-performed SSVD and SPLS on only five combinations of data. On the rest data sets, NSMD had the best performance than the others. This indicates that NSMD has competitive performance in comparing with the parametric learning methods. It should be pointed out that NSMD is non-parametric, while others need to carefully choose right values for the regularization parameters to achieve better performance.

As discussed above, for the parametric sparse learning methods, their performance heavily depends on the values of regularization parameters within the sparsity-inducing functions. To testify their impacts on the classification performance, we applied the sparse learning methods on the digit data sets with different values of λ_u (λ_v) varying from 0.05 to 0.5, with an interval of 0.05. The results are presented in Fig. 1, where the performance is the mean accuracy over the fifteen combinations of data.

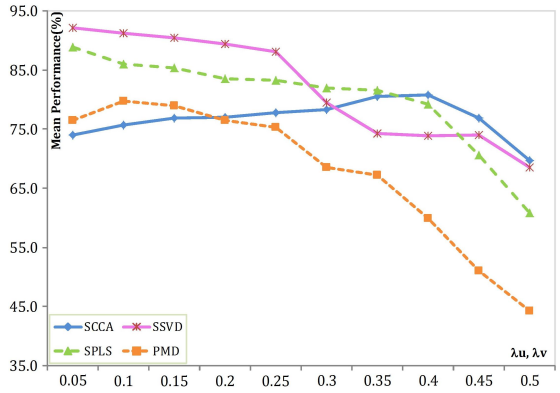

Fig. 1. Mean accuracy (%) of the parametric sparse learning algorithms with the ℓ_1-norm penalty, where $\lambda_u(\lambda_v)$ varied from 0.05 to 0.5.

According to Fig. 1, the performance of the parametric sparse learning methods greatly varied with the regularization parameters λ_u (λ_v), and different algorithms had better performance at different values of the parameters. For example, SCCA, SSVD, SPLS and PMD achieved their better mean performance when λ_u (λ_v) was 0.4, 0.05, 0.05 and 0.1 respectively. Moreover, the performance of SSVD, SPLS and PMD decreased when λ_u (λ_v)

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

TABLE 2
Classification accuracy (%) of the sparse learning algorithms with the ℓ_1-norm penalty, where λ_u (λ_v)= 0.4, 0.05, 0.05 and 0.1 for SCCA, SSVD, SPLS and PMD respectively.

	SCCA	SSVD	SPLS	PMD	NSMD
fac-fou	95.15±0.16	97.45±0.14	96.45±0.19	96.20±0.23	**97.55±0.11**
fac-mor	87.45±0.22	90.25±0.21	93.35±0.25	89.20±0.28	**94.15±0.18**
fac-kar	84.90±0.21	96.75±0.17	96.25±0.21	82.10±0.24	**97.15±0.09**
fac-pix	92.95±0.11	**97.75±0.15**	95.85±0.11	91.00±0.18	97.40±0.11
fac-zer	92.10±0.18	**96.55±0.23**	95.85±0.26	87.60±0.21	96.21±0.15
fou-kar	82.45±0.13	97.24±0.12	94.80±0.19	79.00±0.11	**97.25±0.07**
fou-mor	64.30±0.09	79.90±0.11	**80.95±0.15**	73.05±0.18	78.60±0.10
fou-pix	68.95±0.23	97.75±0.21	96.70±0.25	70.95±0.26	**98.10±0.19**
fou-zer	70.80±0.13	82.80±0.10	82.00±0.16	68.50±0.13	**83.95±0.12**
kar-mor	85.50±0.08	87.05±0.06	90.25±0.12	86.30±0.15	**91.15±0.08**
kar-pix	92.80±0.17	97.35±0.16	96.95±0.21	86.80±0.26	**97.40±0.13**
kar-zer	80.75±0.12	**95.35±0.11**	93.85±0.17	83.85±0.13	94.70±0.14
mor-pix	86.25±0.10	**96.00±0.11**	68.60±0.14	62.50±0.13	95.45±0.10
mor-zer	56.25±0.11	73.00±0.11	54.55±0.13	55.75±0.15	**75.75±0.11**
pix-zer	70.65±0.19	96.50±0.15	96.15±0.23	82.80±0.21	**96.90±0.15**
Mean	80.75	92.11	88.84	79.71	92.78

increased, especially when the parameters were larger than 0.4. This indicates that it is impossible to determine the optimal values of the regularization parameters for all applications.

Besides the ℓ_1-norm penalty, we also adopted the $\ell_{1,2}$-norm penalty as the sparsity-inducing function for the parametric sparse learning methods. Specifically, for each parametric learning algorithm, the latent variables $\mathbf{u}$ and $\mathbf{v}$ were constrained by the $\ell_{1,2}$-norm penalty. As we know, the $\ell_{1,2}$-norm penalty has two regularization parameters, λ and α, to control the degree of sparsity for each latent variable [16]. Thus for each of the parametric sparse methods, there are four regularization parameters, λ_u, λ_v, α_u and α_v, to be tuned. How to get optimal values for them is impossible in practice. For simplicity, in our experiments we observed the changes of performance with λ_u and λ_v when α_u and α_v were fixed.

Table 3 lists the comparison results of the sparse learning algorithms with the $\ell_{1,2}$-norm penalty, where $\alpha_u(\alpha_v)$=0.9 and $\lambda_u(\lambda_v)$=0.2, 0.15, 0.05 and 0.1 for SCCA, SSVD, SPLS and PMD respectively (also refer to Fig. 2 which showing that the parametric algorithms achieved better performance with these parameter values). Additionally, in Table 3 the values in bold-face represent the best results in the same row.

One may observe that the case of the $\ell_{1,2}$-norm penalty was similar to that of the ℓ_1-norm penalty, that is, NSMD also had the best overall performance in comparing with the parametric sparse methods with the $\ell_{1,2}$-norm penalty. For example, NSMD had the best performance on all data sets, except *fou-mor* where NSMD was worse than SPLS, but still better than others.

Another interesting finding is that SSVD had good performance with the ℓ_1-norm penalty, but poor performance with the $\ell_{1,2}$-norm penalty. This implies that for the parametric sparse learning methods, selecting a right sparsity-inducing function is very important, and which sparsity-inducing function should be adopted is determined by the specific problems and data at hand.

Similar to the situations of the ℓ_1-norm penalty, we also compared the performance changes of the parametric sparse learning algorithms with the $\ell_{1,2}$-norm penalty under different parameter values. Fig. 2 presents the mean performance over the fifteen combinations of data sets, where λ_u (λ_v) varied from 0.05 to 0.5. The dotted lines (i.e., the methods marked with '*') and solid lines (i.e., the methods without '*') denote $\alpha_u(\alpha_v)$ was of 0.1 and 0.9 respectively. From Fig. 2, one may observe that the performance of the parametric sparse algorithms was affected by the regularization parameters greatly. When $\alpha_u(\alpha_v)$ were fixed, the performance was varied along with λ_u (λ_v) greatly as those with the ℓ_1-norm penalty (see Fig. 1).

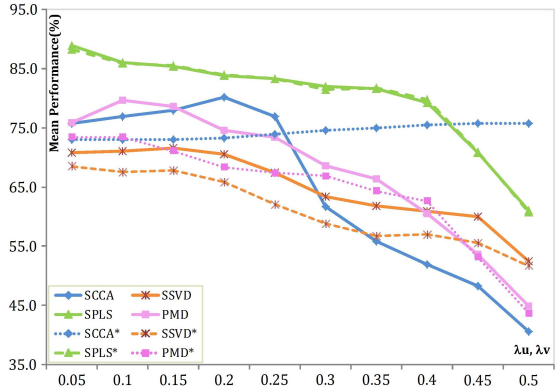

Fig. 2. Mean accuracy (%) of the parametric sparse learning algorithms with the $\ell_{1,2}$-norm penalty, where λ_u (λ_v) varied from 0.05 to 0.5, and dotted and solid lines denote $\alpha_u(\alpha_v)$=0.1 and 0.9 respectively.

To further demonstrate the advantage of NSMD, we also compared it with two non-sparse learning methods, CCA and ALPCCA. The comparison results are illustrated in Fig. 3. As shown in Fig. 3, NSMD outperformed

TABLE 3
Classification accuracy (%) of the sparse learning algorithms with the $\ell_{1,2}$-norm penalty, where $\alpha_u(\alpha_v)$=0.9 and λ_u (λ_v)=0.2, 0.15, 0.05 and 0.1 for SCCA, SSVD, SPLS and PMD respectively

	SCCA	SSVD	SPLS	PMD	NSMD
fac-fou	95.25±0.18	73.25±0.14	96.80±0.12	96.25±0.21	**97.55±0.11**
fac-mor	91.20±0.13	52.25±0.12	93.35±0.16	89.60±0.19	**94.15±0.09**
fac-kar	92.45±0.21	86.80±0.18	96.40±0.25	82.20±0.26	**97.15±0.17**
fac-pix	95.50±0.23	84.15±0.17	96.10±0.21	91.00±0.25	**97.40±0.15**
fac-zer	89.70±0.15	76.75±0.12	95.85±0.16	86.45±0.13	**96.25±0.10**
fou-kar	79.65±0.17	89.35±0.13	95.20±0.19	80.45±0.15	**97.25±0.13**
fou-mor	58.40±0.12	47.10±0.09	**81.00±0.10**	73.30±0.14	78.60±0.14
fou-pix	69.65±0.22	95.75±0.21	96.60±0.18	70.80±0.19	**98.10±0.18**
fou-zer	68.95±0.16	58.05±0.13	81.55±0.19	68.45±0.21	**83.95±0.12**
kar-mor	86.80±0.11	73.70±0.10	90.20±0.13	85.40±0.12	**91.15±0.10**
kar-pix	91.15±0.18	83.10±0.15	97.00±0.19	85.30±0.17	**97.40±0.16**
kar-zer	79.20±0.11	54.85±0.11	93.65±0.12	83.15±0.15	**94.70±0.11**
mor-pix	78.65±0.12	89.65±0.12	68.60±0.15	62.35±0.16	**95.45±0.09**
mor-zer	55.65±0.10	38.20±0.11	54.55±0.13	56.35±0.12	**75.75±0.10**
pix-zer	69.85±0.22	70.15±0.17	95.95±0.21	83.10±0.23	**96.90±0.14**
Mean	80.14	71.54	88.85	79.61	92.78

CCA and ALPCCA across all data sets, although NSMD was slightly better than CCA and ALPCCA on the *fou-mor*, *kar-zer* and *mor-zer* data sets.

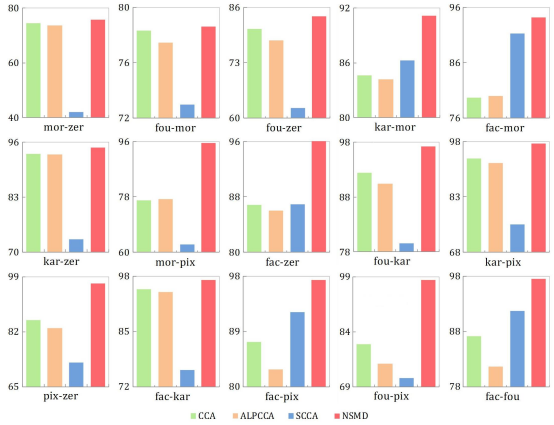

Fig. 3. Classification accuracy (%) of NSMD, CCA and ALPCCA.

Generally speaking, additional data often provide helpful information and potentially improve classification performance. To demonstrate this assertion, we performed NSMD and PMD on the individual (i.e., **X**) and the combined (i.e., **X** and **Y**) data sets respectively. Fig. 4 shows the performance of NSMD and PMD, where the classifiers performed on the individual data **X** are marked with '*'.

It is noticeable that the classification performance of NSMD and PMD on **X** and **Y** was significantly better than **X**. This implies that additional information encoded within **Y** was helpful in classification. It is also interesting to find out that the performance of NSMD on **X** was also comparable to PMD on the combination of **X** and **Y** (see Fig. 4). Similar situations can be found for other parametric sparse learning methods, such as SCCA, SPLS and SSVD. Due to the limitation of space, the results are not shown here.

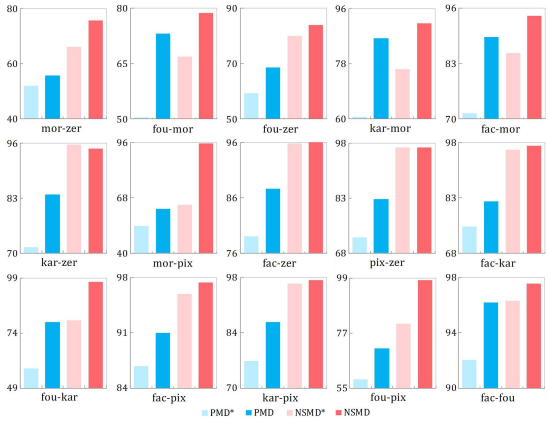

Fig. 4. Classification accuracy (%) of NSMD and PMD with the ℓ_1-norm penalty (λ_u=λ_v=0.1) over the individual (marked with '*') and the combined data sets.

5.2.2 *Yale face recognition*

The data collection of Yale faces, which was downloaded online[2], has been widely used in computer vision research. It involves 165 grayscale images in GIF format of 15 persons, and for each person, there are 11 images with different facial expression or ambient (background) illumination including center-light, left-light, right-ligh, w/glasses, w/no glasses, normal, happy, sad, sleepy, surprised, and wink. The original images are in the size of 64×64 (see Table 1). Apart from the original face images, we also considered two other types of images by using the LBP and wavelet methods on the original ones [40], called *LBP* and *Wav* data respectively in this paper. Thus, there are three pairs of data sets totally.

Table 4 reports the results of NSMD and the parametric sparse methods with the ℓ_1 and $\ell_{1,2}$-norm penalties on the Yale face data, where *Ori* stands for the original data. As shown in Table 4, NSMD has compet-

2. http://cvc.yale.edu/projects/yalefaces/yalefaces.html

TABLE 4
Classification accuracy (%) of the sparse learning algorithms with different λ_u (λ_v) on the Yale face data.

	SCCA	SSVD	SPLS	PMD	NSMD
The ℓ_1-norm penalty					
Data	λ_u (λ_v)				
	0.05	0.05	0.4	0.35	-
Ori-Wav	64.12±0.11	**69.15±0.12**	66.51±0.13	61.14±0.12	65.99±0.11
Ori-LBP	54.60±0.15	**70.95±0.13**	70.11±0.16	58.09±0.15	68.97±0.14
Wav-LBP	65.40±0.18	71.75±0.16	**72.54±0.12**	64.74±0.17	69.56±0.13
The $\ell_{1,2}$-norm penalty with $\alpha_u(\alpha_v)$=0.9					
Data	λ_u (λ_v)				
	0.05	0.15	0.4	0.4	-
Ori-Wav	67.12±0.10	57.50±0.11	**67.17±0.13**	62.35±0.12	65.99±0.11
Ori-LBP	36.25±0.12	62.39±0.13	68.90±0.10	56.91±0.12	**68.97±0.10**
Wav-LBP	64.74±0.15	59.34±0.16	**72.20±0.13**	64.71±0.14	69.56±0.12

itive performance in comparison with the parametric methods. Particularly, the performance of NSMD was better than the corresponding PMD, which is a matrix decomposition method too. Additionally, NSMD outperformed SCCA roundly. Although SSVD with the ℓ_1-norm penalty achieved better performance than NSMD, it had relatively worse performance when the sparsity-inducing function was the $\ell_{1,2}$-norm penalty.

SPLS achieved relatively good performance on the Yale face data. As we know, the dimensionality of the Yale face data is high, while the number of samples is relatively small. This implies that PLS is good at dealing with the so called problem of "large p, small n". The technique of matrix decomposition may not be about to tackle such highly collinear data very well.

Fig 5 shows the mean performance of the parametric sparse algorithms with the ℓ_1-norm penalty over the three combinations of data with different parameter values. From Fig 5, one may observe that unlike the handwritten digit data, the mean performance of SSVD, SPLS and PMD on the Yale face data was changed slightly along with λ_u (λ_v). The reason is that the Yale data set contains less samples, but its dimensionality is relatively high. On the other hand, for a sparse learning algorithm the optimal values of the regularization parameters vary from the data.

Similarly, we also tested the performance of the parametric sparse learning algorithms with the $\ell_{1,2}$-norm penalty with different regularization parameters on the Yale face data. Fig. 6 records the mean performance of the parametric sparse algorithms over the three combinations of data sets, where λ_u (λ_v) varied from 0.05 to 0.5 when α_u (α_v) was fixed at 0.1 and 0.9 respectively. In Fig. 6, the dotted lines (i.e., the methods marked with '*') and the solid lines (i.e., the methods without '*') denote $\alpha_u(\alpha_v)$ was of 0.1 and 0.9 respectively. Observing from Fig. 6, we can make the same conclusions like the case of the Handwritten digit data, that is, the performance of the parametric sparse algorithms was affected by the regularization parameters. For example, the performance of SCCA was changed greatly if $\lambda_u(\lambda_v)$ was greater than 0.1.

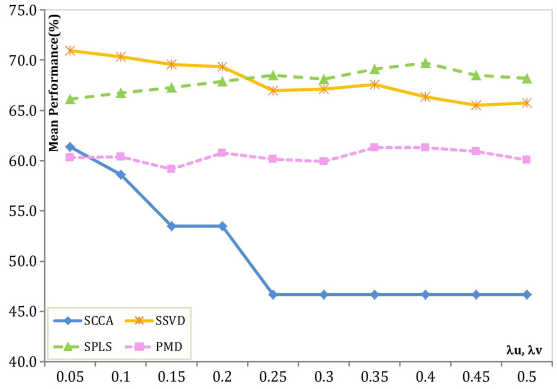

Fig. 5. Mean accuracy (%) of the parametric sparse learning algorithms with the ℓ_1-norm penalty, where $\lambda_u(\lambda_v)$ varied from 0.05 to 0.5.

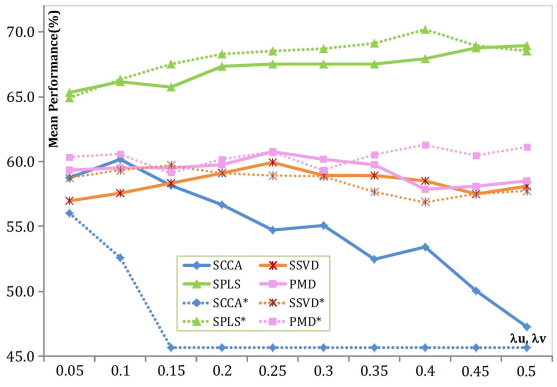

Fig. 6. Mean accuracy (%) of the parametric sparse learning algorithms with the $\ell_{1,2}$-norm penalty, where λ_u (λ_v) varied from 0.05 to 0.5, and the dotted and solid lines denote $\alpha_u(\alpha_v)$=0.1 and 0.9 respectively.

5.2.3 *Course categorization*

This data collection was extracted from 1051 web pages at four U.S. universities' web sites [41]. Each page was preprocessed by removing stop words, numbers and then stemming words. The obtained words in each page were divided into two groups, representing two different

views. The first one refers to those words linking to course web pages, whereas the second group contains the rest words. In addition, the words that occurred in five or fewer documents were ignored. As a result, the first and second views were represented as 87- and 2332-dimensional data.

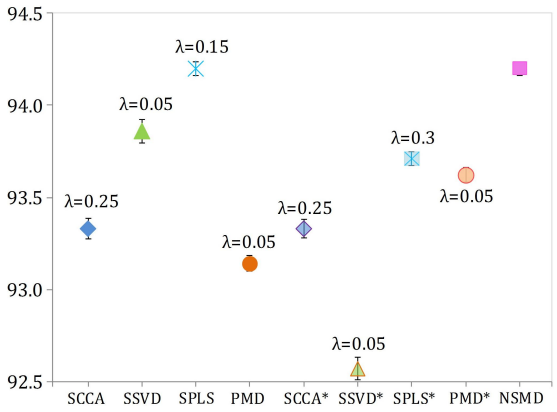

Fig. 7. Classification accuracy (%) of the sparse learning methods on the course data, where the methods marked without or with '*' represent the ℓ_1 or $\ell_{1,2}$-norm penalty respectively.

Fig. 7 shows the classification performance of the comparing methods on the course data collection, where the methods marked without or with '*' denote the ℓ_1 and $\ell_{1,2}$-norm penalty respectively. The value of λ above each indicator means that the corresponding method achieved the best performance in this case. According to the results, one can note that our non-parametric learning method is superior to the parametric ones on this data. Among the parametric methods, SPLS with the ℓ_1-norm penalty has relatively good performance with $\lambda_u(\lambda_v)$=0.15. However, getting such optimal value for $\lambda_u(\lambda_v)$ is a time-consuming thing. Meanwhile SPLS with the $\ell_{1,2}$-norm penalty is not good enough.

Fig. 8 gives the performance changes of the parametric sparse learning methods along with the values of $\lambda_u(\lambda_v)$, where the dotted and solid lines denote the ℓ_1 or $\ell_{1,2}$-norm penalty respectively. As illustrated in Fig. 8, the performance of the matrix factorization methods, i.e., SSVD and PMD, varied greatly when $\lambda_u(\lambda_v)$ changed, while the performance of SPLS was relatively stable. As discussed above, the underlying reason is that the dimensionality of the course data is larger than the quantity of samples. An interesting fact is that the sparse learning algorithms, except SCCA, with different penalty constraints have similar performance.

Fig. 9 presents the performance changes of the parametric sparse learning methods with the $\ell_{1,2}$-norm penalty, where λ_u (λ_v) varied from 0.05 to 0.5, and the dotted and solid lines denote $\alpha_u(\alpha_v)$=0.1 and 0.9 respectively. Like Fig. 8, similar conclusions can be made. For example, SPLS had relatively stable performance, while PMD and SSVD changed greatly. It is noticeable

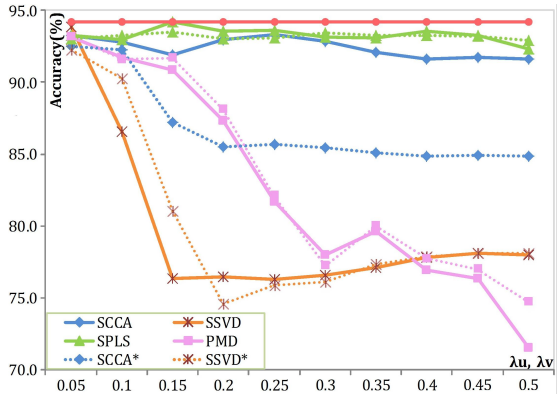

Fig. 8. Classification accuracy (%) of the sparse learning methods along with $\lambda_u(\lambda_v)$, where the dotted and solid lines denote the ℓ_1 or $\ell_{1,2}$-norm penalty respectively.

that the performance of the parametric learning algorithms, except SCCA, varied not much as the parameter α changed. Fig. 8 and Fig. 9 tell us a fact that the proposed non-parametric learning method, NSMD, still had better performance, without such cumbersome issue of assigning the appropriate values to the regularization parameters.

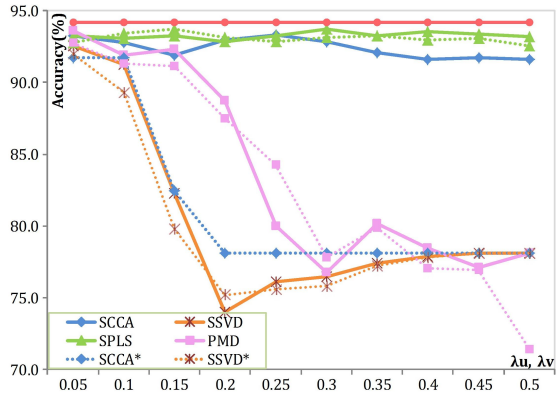

Fig. 9. Classification accuracy (%) of the sparse learning methods with the $\ell_{1,2}$-norm penalty, where λ_u (λ_v) varied from 0.05 to 0.5, and the dotted and solid lines denote $\alpha_u(\alpha_v)$=0.1 and 0.9 respectively.

5.2.4 *Sparsity degree*

To demonstrate the sparse capability of the non-parametric sparse model, we carried out additional experiments and then obtained the sparsity degree, i.e., the ratio of the coefficients with zero values to all coefficients, on the experimental data. The experimental results are listed in Table 5, where the higher the sparsity degree, the more the coefficients with zero values. From the experimental results, we can observe that the proposed model has good sparsity capabilities on most of the data. For the *mor* data, the coefficients with zero values achieved by NSMD are less. The reason is that there are only six variables within this data.

TABLE 5
Mean sparsity degree (%) of NSMD on the data.

Data	**u**	**v**	Data	**u**	**v**
fac-fou	51.03	44.29	fac-mor	32.01	15.56
fac-kar	46.30	40.99	fac-pix	56.79	54.83
fac-zer	51.83	34.03	fou-kar	45.60	42.21
fou-mor	54.58	8.61	fou-pix	47.28	46.42
fou-zer	48.19	39.19	kar-mor	47.58	12.78
kar-pix	47.57	46.36	kar-zer	46.97	33.61
mor-pix	6.11	23.65	mor-zer	0.00	7.30
pix-zer	55.88	40.00	Ori-Wav	89.97	89.16
Ori-LBP	54.09	57.36	LBP-Wav	60.00	53.79
Course	73.47	61.60			

6 Conclusions

In this paper, we have proposed a novel dimension reduction method for cross-view data analysis. Comparing with the traditional learning methods which require careful selection of appropriate sparsity-inducing functions and tuning of the regularization parameters, the distinct characteristic of our method is that it is automatic and non-parametric one, and does not require human-involvement. The experiments conducted on synthetic and real-world data have shown that the performance of the parametric sparse learning algorithms change with the values of the regularization parameters, while our automatic learning method does not involve any regularization parameters, but has stable performance.

Although the proposed method did not achieve good performance as SPLS did on the Yale face data, it still outperformed PMD significantly. Both of them belong to the techniques of matrix decomposition. Indeed, PLS is good at handling the high-dimensional data with less samples (i.e., the "large p, small n" problem). In future, we will apply the non-parametric sparsity-inducing function to PLS to cope with the "large p, small n" problem effectively.

In real-world applications, data are often collected from multiple domains or views, resulting in a more complex and common situation. It has been demonstrated that tensor is an effective technique for the multiple-view data collections in the literature. In our future work, we will extend the idea of our model to the tensor technique, so that it can handle the high-dimensional and massive multiple-view data.

Acknowledgment

The authors would like to thank the anonymous referees and the associate editor for their valuable comments and suggestions, which have improved the paper vastly. This work was partially supported by the National NSF of China grant 61572443 and 61673179, the ARC Discovery grant DP130104090, and the Shanghai Key Laboratory of Intelligent Information Processing grant IIPL-2016-001.

References

[1] F. Wu, Z. Yu, Y. Yang, S. Tang, Y. Zhang, and Y. Zhuang, "Sparse multi-modal hashing," *IEEE Transactions on Multimedia*, vol. 16, no. 2, pp. 427–439, 2014.

[2] X. Wu, X. Zhu, G.-Q. Wu, and W. Ding, "Data mining with big data," *IEEE Transactions on Knowledge and Data Engineering*, vol. 26, no. 1, pp. 97–107, Jan. 2014.

[3] M. Katsurai, T. Ogawa, and M. Haseyama, "A cross-modal approach for extracting semantic relationships between concepts using tagged images," *IEEE Transactions on Multimedia*, vol. 16, no. 4, pp. 1059–1074, 2014.

[4] S. Zhang, X. Yu, Y. Sui, S. Zhao, and L. Zhang, "Object tracking with multi-view support vector machines," *IEEE Transactions on Multimedia*, vol. 17, no. 3, pp. 265–278, 2015.

[5] C. Kang, S. Xiang, S. Liao, C. Xu, and C. Pan, "Learning consistent feature representation for cross-modal multimedia retrieval," *IEEE Transactions on Multimedia*, vol. 17, no. 3, pp. 370–381, 2015.

[6] S. Sun, "A survey of multi-view machine learning," *Neural Comput & Applic*, vol. 23, no. 7-8, pp. 2031–2038, Dec. 2013.

[7] H. Liu, Z. Ma, S. Zhang, and X. Wu, "Penalized partial least square discriminant analysis with ℓ_1-norm for multi-label data," *Pattern Recognition*, vol. 48, no. 5, pp. 1724–1733, 2015.

[8] X. Yang, T. Zhang, and C. Xu, "Cross-domain feature learning in multimedia," *IEEE Transactions on Multimedia*, vol. 17, no. 1, pp. 64–78, 2015.

[9] C. Shi, Q. Ruan, G. An, and R. Zhao, "Hessian semi-supervised sparse feature selection based on $L_{2,1/2}$ -matrix norm," *IEEE Transactions on Multimedia*, vol. 17, no. 1, pp. 16–28, 2015.

[10] C. Luo, B. Ni, S. Yan, and M. Wang, "Image classification by selective regularized subspace learning," *IEEE Transactions on Multimedia*, vol. 18, no. 1, pp. 40–50, 2016.

[11] F. Bach, R. Jenatton, J. Mairal, and G. Obozinski, "Optimization with sparsity-inducing penalties," *Found. Trends Mach. Learn.*, vol. 4, no. 1, pp. 1–106, Jan. 2012.

[12] H. Zou, T. Hastie, and R. Tibshirani, "Sparse principal component analysis," *Journal of Computational and Graphical Statistics*, vol. 15, no. 2, pp. 262–286, 2004.

[13] E. Parkhomenko, D. Tritchler, and J. Beyene, "Sparse canonical correlation analysis with application to genomic data integration," *Statistical Applications in Genetics and Molecular Biology*, vol. 8, no. 1, p. Article 1, 2009.

[14] D. M. Witten, R. Tibshirani, and T. Hastie, "A penalized matrix decomposition, with applications to sparse principal components and canonical correlation analysis," *Biostatistics*, vol. 10, no. 3, pp. 515–534, 2009.

[15] R. Tibshirani, "Regression shrinkage and selection via the lasso," *Journal of the Royal Statistical Society, Series B*, vol. 58, no. 1, pp. 267–288, 1996.

[16] R. Mazumder, J. H. Friedman, and T. Hastie, "Sparsenet: Coordinate descent with nonconvex penalties," *Journal of the American Statistical Association*, vol. 106, no. 495, pp. 1125–1138, 2011.

[17] C. Croux, P. Filzmoser, and H. Fritz, "Robust sparse principal component analysis," *Technometrics*, vol. 55, no. 2, pp. 202–214, 2013.

[18] Y.-X. Wang and Y.-J. Zhang, "Nonnegative matrix factorization: A comprehensive review," *IEEE Transactions on Knowledge and Data Engineering*, vol. 25, no. 6, pp. 1336–1353, 2013.

[19] M. Lee, H. Shen, J. Huang, and J. S. Marron, "Biclustering via sparse singular value decomposition," *Biometrics*, vol. 66, pp. 1087–1095, 2010.

[20] Z. Hong and H. Lian, "Sparse-smooth regularized singular value decomposition," *Journal of Multivariate Analysis*, vol. 117, pp. 163–174, May 2013.

[21] D. Yang, Z. Ma, and A. Buja, "A sparse singular value decomposition method for high-dimensional data," *Journal of Computational and Graphical Statistics*, vol. 23, no. 4, pp. 923–942, 2014.

[22] D. Cai, X. He, J. Han, and T. S. Huang, "Graph regularized nonnegative matrix factorization for data representation," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 33, no. 8, pp. 1548–1560, Aug. 2011.

[23] S. Tang, Y.-T. Zheng, Y. Wang, and T.-S. Chua, "Sparse ensemble learning for concept detection," *IEEE Transactions on Multimedia*, vol. 14, no. 1, pp. 43–54, 2012.

[24] H. Liu, Z. Wu, X. Li, D. Cai, and T. S. Huang, "Constrained nonnegative matrix factorization for image representation," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 34, no. 7, pp. 1299–1311, 2012.

[25] X. Tan, F. Wu, X. Li, S. Tang, W. Lu, and Y. Zhuang, "Structured visual feature learning for classification via supervised probabilistic tensor factorization," *IEEE Transactions on Multimedia*, vol. 17, no. 5, pp. 660–673, 2015.

[26] A. Eweiwi, M. Cheema, and C. Bauckhage, "Action recognition in still images by learning spatial interest regions from videos," *Pattern Recognition Letters*, vol. 51, pp. 8–15, 2015.

[27] D. R. Hardoon and J. Shawe-Taylor, "Sparse canonical correlation analysis," *Machine Learning*, vol. 83, no. 3, pp. 331–353, 2011.

[28] A. K. Katsaggelos, S. Bahaadini, and R. Molina, "Audiovisual fusion: Challenges and new approaches," *IEEE Transactions on Multimedia*, vol. 103, no. 9, pp. 1635–1653, 2015.

[29] H. Izadinia, I. Saleemi, and M. Shah, "Multimodal analysis for identification and segmentation of moving-sounding objects," *IEEE Transactions on Multimedia*, vol. 15, no. 2, pp. 378–390, 2013.

[30] M. Sargin, Y. Yemez, E. Erzin, and A. Tekalp, "Audiovisual synchronization and fusion using canonical correlation analysis," *IEEE Transactions on Multimedia*, vol. 9, no. 7, pp. 1396–1403, 2007.

[31] Y.-H. Yuan, Q.-S. Sun, and H.-W. Ge, "Fractional-order embedding canonical correlation analysis and its applications to multi-view dimensionality reduction and recognition," *Pattern Recognition*, vol. 47, no. 3, pp. 1411–1424, 2014.

[32] Y.-H. Yuan, Y. Li, X.-B. Shen, Q.-S. Sun, and J.-L. Yang, "Laplacian multiset canonical correlations for multiview feature extraction and image recognition," *Multimedia Tools and Applications*, vol. 10.1007/s11042-015-3070-y, pp. 1–25, 2015.

[33] A.-L. Boulesteix and K. Strimmer, "Partial least squares: a versatile tool for the analysis of high-dimensional genomic data," *Brief Bioinform.*, vol. 8, no. 1, pp. 32–44, Jan. 2007.

[34] A. Bakry and A. Elgammal, "Mkpls: Manifold kernel partial least squares for lipreading and speaker identification," in *IEEE Conf. on Computer Vision and Pattern Recognition (CVPR 2013)*. Portland, OR, USA: IEEE, 2013, pp. 684–691.

[35] Z. Wang, W. Wang, Z. Wan, Y. Xia, and W. Lin, "No-reference hybrid video quality assessment based on partial least squares regression," *Multimedia Tools and Applications*, vol. 74, no. 23, pp. 10 277–10 290, 2015.

[36] B. Zhong, X. Yuan, R. Ji, Y. Yan, Z. Cui, X. Hong, Y. Chen, T. Wang, D. Chen, and J. Yu, "Structured partial least squares for simultaneous object tracking and segmentation," *Neurocomputing*, vol. 133, pp. 317–327, 2014.

[37] Q. Wang, F. Chen, W. Xu, and M.-H. Yang, "Object tracking via partial least squares analysis," *IEEE Transactions on Image Processing*, vol. 21, no. 10, pp. 4454–4465, 2012.

[38] K.-A. Lê Cao, D. Rossouw, C. Robert-Granié, and P. Besse, "A sparse pls for variable selection when integrating omics data," *Stat. Appl. Genet. Mol. Biol.*, vol. 7, no. 1, pp. Article 35: 1–37, Nov. 2008.

[39] Q. Li, Y. Yan, and H. Wang, "Discriminative weighted sparse partial least squares for human detection," *IEEE Transactions on Intelligent Transportation Systems*, vol. PP, no. 99, pp. 1–10, 2015.

[40] F. Wang and D. Zhang, "A new locality-preserving canonical correlation analysis algorithm for multi-view dimensionality reduction," *Neural Process Lett*, vol. 37, no. 2, pp. 135–146, Apr. 2013.

[41] S. Sun and D. R. Hardoon, "Active learning with extremely sparse labeled examples," *Neurocomputing*, vol. 73, pp. 2980–2988, 2010.

Huawen Liu is an associate professor at the Department of Computer Science, Zhejiang Normal University, P.R. China. He received his master and PhD degree in computer science from Jilin University, P.R. China in 2007 and 2010 respectively. His interests include data mining, feature selection and sparse learning for machine learning.

Lin Liu is a senior lecturer at the School of Information Technology and Mathematical Sciences, University of South Australia (UniSA). She received her bachelor and master degrees in Electronic Engineering from Xidian University, China in 1991 and 1994 respectively, and her PhD degree in computer systems engineering from UniSA in 2006. Dr Liu's research interests include data mining and bioinformatics, as well as Petri nets and their applications to protocol verification and network security analysis.

Thuc Duy Le is a research Fellow at the University of South Australia (UniSA). He received his BSc (2002) and MSc (2006) in pure Mathematics from the University of Pedagogy, Ho Chi Minh City, Vietnam, and BSc (2010) in Computer Science from UniSA. He received his PhD degree in Computer Science (Bioinformatics) in 2014 from UniSA. His research interests are Bioinformatics, data mining, and machine learning.

Ivan Lee received his BE, MER, MCom, and PhD degrees from the University of Sydney, Australia. He worked at Cisco Systems, Remotek Corporation, and Ryerson University. He is currently a Senior Lecturer at the School of Information Technology and Mathematical Sciences, University of South Australia. His research interests are in multimedia systems, computer vision, biomedical engineering and data analytics.

Shiliang Sun received his BSc in Automatic Control from Beijing University of Aeronautics and Astronautics (BUAA), and MSc and PhD degrees from Tsinghua University. In 2007, he joined the Department of Computer Science and Technology, East China Normal University (ECNU). He serves on editorial boards of multiple international academic journals, e.g., associate editorship for Neurocomputing, IEEE Transactions on Intelligent Transportation Systems and Information Fusion.

Jiuyong Li received his PhD degree in computer science from Griffith University in Australia. He is currently a professor at the University of South Australia. His research interests are in data mining, privacy preservation and biomedical informatics. He has published more than 90 papers and his research has been supported by multiple Discovery Grants of Australian Research Council.