Pattern Recognition 46 (2013) 1267–1276

Contents lists available at SciVerse ScienceDirect

Pattern Recognition

journal homepage: www.elsevier.com/locate/pr

A rank-one update method for least squares linear discriminant analysis with concept drift

Yi-Ren Yeh [a], Yu-Chiang Frank Wang [b,*]

[a] Intel-NTU Connected Context Computing Center, National Taiwan University, Taipei, Taiwan
[b] Research Center for Information Technology Innovation, Academia Sinica, Taipei, Taiwan

ARTICLE INFO

Article history:
Received 11 November 2011
Received in revised form
11 August 2012
Accepted 8 November 2012
Available online 16 November 2012

Keywords:
Linear discriminant analysis
Least squares solution
Rank-one update
Concept drift

ABSTRACT

Linear discriminant analysis (LDA) is a popular supervised dimension reduction algorithm, which projects the data into an effective low-dimensional linear subspace while the separation between the projected data from different classes is improved. While this subspace is typically determined by solving a generalized eigenvalue decomposition problem, its high computation costs prohibit the use of LDA especially when the scale and the dimensionality of the data are large. Based on the recent success of least squares LDA (LSLDA), we propose a novel rank-one update method with a simplified class indicator matrix. Using the proposed algorithm, we are able to derive the LSLDA model efficiently. Moreover, our LSLDA model can be extended to address the learning task of concept drift, in which the recently received data exhibit with gradual or abrupt changes in distribution. In other words, our LSLDA is able to observe and model the data distribution changes, while the dependency on outdated data will be suppressed. This proposed LSLDA will benefit applications of streaming data classification or mining, and it can recognize data with newly added class labels during the learning process. Experimental results on both synthetic and real datasets (with and without concept drift) confirm the effectiveness of our propose LSLDA.

1. Introduction

Linear discriminant analysis (LDA) [1] is a widely used supervised dimension reduction technique. Utilizing eigen-analysis, LDA projects the data into a low-dimensional linear subspace, and this subspace provides an improved visualization and separation between data with different class labels. For many real-world applications such as image and text categorization, it is very difficult and computationally expensive to perform eigen-analysis, since the scale and the dimensionality of the data are typically very large. Moreover, for incremental learning tasks, it is not desirable to re-train the LDA whenever a new data instance is received.

Besides the aforementioned problems, how to design an effective LDA model, which can adapt the recently received data with different distributions, also draws intensive attention from researchers. Such tasks are practical for applications like spam email or credit card fraud detection. In these problems, data distributions might change over time in an unforeseen way, and thus one needs to address the learning task of *concept drift*. Generally, the difference between the existing LDA model and the updated one (with new data instances received) should not be significant, when only a small amount of new data are received. This implies that the update process of the LDA model should not take much effort, and the developed updating algorithms will be benefit large-scale incremental learning problems [2–7].

Based on the recent success of least squares LDA (LSLDA) [8,9], we propose a rank-one update method for LSLDA. We present an improved recursive least squares technique [10] to update the LDA model when receiving a new data instance (with an existing or new class label). We note that, previous LSLDA algorithms need to keep the data covariance matrix during the update process, and it becomes very challenging to handle high-dimensional data. In this paper, our LSLDA is designed to handle high dimensional and concept-drifting data. We utilize the recursive least squares algorithm and present a rank-one update technique for LSLDA with a simplified class indicator matrix. This indicator matrix allows us to efficiently solve multiple linear regression problems, and we prove the equivalent relationship between different predetermined indicator matrices proposed by prior researchers and ours. Since our LSLDA does not need to store the entire covariance matrix (neither its inverse version) during the update process, our proposed method is computationally more efficient than prior LDA models. It is worth noting that, our LSLDA is able to handle streaming data with changes in distribution (i.e. concept drift), and data with newly added class labels. Both cases are common and important in many streaming data mining or

* Corresponding author. Tel.: +886 2 27872368.
E-mail addresses: yryeh@citi.sinica.edu.tw (Y.-R. Yeh), ycwang@citi.sinica.edu.tw (Y.-C. Wang).

0031-3203/$ - see front matter © 2012 Elsevier Ltd. All rights reserved.
http://dx.doi.org/10.1016/j.patcog.2012.11.008

information filtering applications. While some of prior LDA models can be extended to recognize recently received data with newly added class labels [11], they are not able to address the problem of concept drift.

It is worth noting that the influence of outliers is also a critical problem for real-world classification problems. Even few deviated data might lead to biased classification models and thus degrade the performance. Many methods have been proposed to perform outlier detection (e.g. [12,13]), which can be applied to identify and remove the deviated data in advance. It is worth noting that, our proposed LSLDA can be considered as an incremental learning approach, which updates the learned LDA model whenever a new instance is received while suppressing the information of outdated data (for the purposes of concept drift). If outlier data is presented during the incremental learning process, our rank-one updating technique will prevent the derived LSLDA model from fitting such few and deviated data (otherwise they will not be considered as outliers). Moreover, due to the introduced capability of dealing with concept drift, the presence of outlier data will be suppressed after the entire updating process is complete. Since the main purposes of this paper is to provide an efficient updating algorithm for LSLDA with concept drift, we do not explicitly discuss and conduct the experiments on datasets with outliers.

The remaining of this paper is organized as follows. Section 2 discusses prior works on LDA. We briefly review the least squares LDA, and prove the validity of our proposed simplified class indicator matrix for LSLDA in Section 3. We present the proposed rank-one update method for LSLDA with concept drift in Section 4. Experimental results and comparisons of different LDA methods are presented in Section 5. Finally, Section 6 concludes this paper.

2. Related work

Dimension reduction has been one of the main research topics in the areas of machine learning, pattern recognition, and statistics. It can be performed in a *supervised* or an *unsupervised* fashion, depending on the availability of class labels and the problems of interest. Supervised dimension reduction methods typically project the labeled data into a low-dimensional space, and this subspace provides improved discrimination between the projected data with different class labels.

Among supervised dimension reduction techniques, linear discriminant analysis (LDA) [1] is one of the most popular algorithms due to its simplicity and effectiveness. The idea of LDA is to seek the optimal low-dimensional space which minimizes the within-class variations while the between-class distances are simultaneously maximized. The standard LDA produces a linear transformation, which is determined by solving a generalized eigenvalue decomposition problem. Due to the need to calculate the inverse of the data covariance matrices, various methods have been proposed to reduce the computational complexity or to alleviate the singularity problem [14–16]. Several of them apply matrix factorization techniques such as QR factorization, singular value decomposition (SVD) or generalized SVD (GSVD) in their formulations. In addition, a least squares formulation for LDA was proposed in [9,17], which also converts the eigen-analysis problem into a multiple linear regression formulation. This least squares LDA model, or LSLDA in short, utilizes a predetermined class indicator matrix and solves the multiple linear regression task (see Section 3.2 for details). Since there is no need to perform eigen-analysis, LSLDA can be applied to those problems which require additional constraints (e.g. sparsity) on the LDA solution [9]. In [17], Cai et al. proposed an efficient algorithm, named SRDA, for performing discriminant analysis in a least squares formulation. The main purpose of this work is to reduce the computation complexity and memory requirement when computing the solution of LDA. It is worth noting that, SRDA is performed in batch mode and preferable for sparse data matrices (e.g. text classification data).

Due to the rise of streaming data mining and information filtering applications (e.g. spam detection, market analysis, etc.), incremental learning of streaming or time-varying data becomes a major research topic in machine learning and data mining communities. Several incremental LDA algorithms have been proposed [18–22,11], and they consider all data instances (newly receive or outdated ones) equally important. For example, in [18], an IDR/QR algorithm based on an approximated formulation of LDA with QR factorization (discussed in [16]) is proposed. Zhao et al. [21] pointed out that such an approximation of LDA subspace might not generalize well, and they presented a GSVD-ILDA algorithm to adopt a GSVD formulation as [15] did to update the LDA subspace. Although improved LDA results using GSVD-ILDA were reported, their approach still needs to perform QR factorization and SVD during their update process. Besides these incremental update methods for conventional LDA, an incremental update algorithm for LSLDA, LS-ILDA, is recently proposed in [11]. This LS-ILDA algorithm only invokes matrix manipulations without solving data inverse problems, and thus is computationally more efficient than methods performing matrix factorization. However, to the best of our knowledge, no prior LDA methods (incremental versions or not) have addressed the problem of concept drift. When newly received data have distribution changes over time in an unforeseen way, it is not clear how to apply (or modify) the aforementioned LDA approaches for learning the concept-drifting data.

In this paper, we propose a rank-one update method for incremental LSLDA with a simplified class indicator matrix. Our LDA formulation focuses on reducing the computational cost during the updating process, and we add the additional ability of dealing with concept-drifting data. As discussed and summarized in [23–25], there are three different types of approaches for handling the task of concept drift: instance selection, ensemble learning, and instance weighting. The idea of instance selection is to select data instances which are relevant to the current concept of interest. Such a concept can be detected by exploring the relationship between the newly arrived data and instances which were previously received [26,27]. For ensemble learning methods, the adaptivity of the classifier is achieved by advancing fusion rules [28,29]. Therefore, how to properly determine the weights for fusing different models becomes a critical issue. Typically, one considers the weight of each model as a function of the associated performance, or cross-validation can be utilized to determine the weights. Different from the above two methods, instance weighting achieves the model adaptivity by assigning different weights for each data instance. Unlike ensemble learning approaches determine the weight for each learning model, one typically uses the age or competence of an instance as its weight, and these weighted instances are applied to learn a single model or an ensemble for handling concept drift [30]. Our proposed work can be categorized as an instance weighting approach. The proposed forgetting factor β suppresses the influence of outdated data, so that the adaptivity of our LSLDA model can be achieved. Our experiments will validate the use of our proposed method for data with and without concept drift. Comparing with conventional LDA or prior LSLDA models, we will show that our LSLDA model produces better performances when the task of concept drift is of major concern.

3. Least squares linear discriminant analysis

3.1. Least squares formulations for LDA

Given n data instances from k different classes in a p-dimensional space, the solution $\mathbf{W}$ to the standard LDA solves the

following generalized eigenvalue decomposition problem:

$$\mathbf{\Sigma}_b\mathbf{W}^{LDA}=\mathbf{\Lambda}\mathbf{\Sigma}_t\mathbf{W}^{LDA}, \tag{1}$$

where

$$\mathbf{\Sigma}_t=\frac{1}{n}\sum_{i=1}^{n}(\mathbf{x}_i-\boldsymbol{\mu})(\mathbf{x}_i-\boldsymbol{\mu})^\top$$

and

$$\mathbf{\Sigma}_b=\frac{1}{n}\sum_{c=1}^{k}n_c(\boldsymbol{\mu}_c-\boldsymbol{\mu})(\boldsymbol{\mu}_c-\boldsymbol{\mu})^\top$$

are the total and between-class scatter covariance matrices, respectively. $\mathbf{\Lambda}$ is a diagonal matrix with the corresponding generalized eigenvalues. Note that $\boldsymbol{\mu}$ is the global mean, $\boldsymbol{\mu}_c$ is the mean of class c, and n_c is the number of instances in class c.

To avoid solving the above eigenvalue decomposition problem, Ye [9] proposed a least squares solution to LDA, which converts the conventional LDA formulation into a multiple linear regression problem. To solve the LSLDA model, a centered data matrix is constructed as $\mathbf{A}=[\mathbf{x}_1^\top;\mathbf{x}_2^\top;\ldots;\mathbf{x}_n^\top]\in\mathbb{R}^{n\times p}$, where each row $\mathbf{x}_i$ represents a data instance (with the global mean removed). Besides, one needs a class indicator matrix $\mathbf{Y}=[\mathbf{y}_1^\top;\mathbf{y}_2^\top;\ldots;\mathbf{y}_n^\top]\in\mathbb{R}^{n\times k}$, where each entry satisfies the following equation:

$$\mathbf{Y}(i,c)=\begin{cases}\sqrt{\frac{n}{n_c}}\sqrt{\frac{n_c}{n}} & \text{if } \mathbf{x}_i\in c,\\ -\sqrt{\frac{n_c}{n}} & \text{otherwise.}\end{cases} \tag{2}$$

According to [9], the solution to the above LSLDA is

$$\mathbf{W}^{MLR}=(\mathbf{A}^\top\mathbf{A})^{-1}\mathbf{A}^\top\mathbf{Y}, \tag{3}$$

which is proved in [9] that $\mathbf{W}^{MLR}$ satisfies the following equation:

$$\mathbf{W}^{MLR}=[\mathbf{W}^{LDA}\mathbf{D},0]\mathbf{Q}^\top. \tag{4}$$

In (4), $\mathbf{D}$ is a diagonal matrix while $\mathbf{Q}$ is an orthogonal one. It has been shown that $\mathbf{D}$ would be an identity matrix under a mild condition which typically occurs when $n\ll p$ [9]. The solution $\mathbf{W}^{MLR}$ in (4) indicates that the LSLDA model is derived from a multiple linear regression problem, and we will simply use $\mathbf{W}$ in the remaining of this paper for the sake of simplicity.

3.2. The proposed simplified coding scheme for LSLDA

As discussed above, the LSLDA proposed in [9] requires a predetermined indicator matrix $\mathbf{Y}$ (determined by (2)), the coding scheme of [9] needs to know the data size n and the number of instances for each class n_c in advance. However, these numbers are not known a priori, especially for incremental or streaming cases. Recently, Liu et al. [11] redefined the indicator matrix and proposed a least squares incremental LDA. The indicator matrix determined in [11] is simply determined as

$$\mathbf{Y}_1(i,c)=\begin{cases}\frac{1}{\sqrt{n_c}} & \text{if } \mathbf{x}_i\in c,\\ 0 & \text{otherwise.}\end{cases} \tag{5}$$

Using this form, the solution $\mathbf{W}_1$ to their LSLDA satisfies the following equation:

$$\mathbf{W}_1=\frac{1}{\sqrt{n}}\mathbf{W}, \tag{6}$$

where $\mathbf{W}$ and $\mathbf{W}_1$ are the solutions of LSLDA using the coding schemes $\mathbf{Y}$ and $\mathbf{Y}_1$ respectively. In other words, using the simplified coding scheme of [11], the LSLDA model can be derived by (3) with a scaling factor $1/\sqrt{n}$. However, this coding scheme still requires the number of instances for each class n_c a priori; if one needs to apply this coding scheme for incremental learning or streaming data problems, the indicator matrix needs to be updated whenever a data instance is received. Detailed derivations and discussions of least squares and incremental least squares LDA models can be found in [9,11]. In this paper, we focus on a rank-one update approach for LSLDA, which would benefit high dimensional classification problems (with or without concept drift). To alleviate computational complexity for our rank-one update method, and to avoid to recalculate the class indicator matrix as [9,11] did, we advance a simplified coding scheme to define a novel class indicator matrix, i.e.

$$\mathbf{Y}_2(i,c)=\begin{cases}1 & \text{if } \mathbf{x}_i\in c,\\ 0 & \text{otherwise.}\end{cases} \tag{7}$$

In the following proposition, we will show that the use of our indicator matrix is equivalent to the use of previous ones (i.e. $\mathbf{Y}$ and $\mathbf{Y}_1$). It is worth mentioning that there are only two different values 1 or 0 in each column of our indicator matrix $\mathbf{Y}_2$, and thus it is very easy to update our LSLDA solution than prior LSLDA methods. We now show our LDA solution is effectively the same as those derived by [9,11], except for a constant term which can be easily calculated.

Proposition 1. *Suppose that $\mathbf{y}_c$ is the cth column of an indicator matrix (e.g. $\mathbf{Y}_1$). We have $y_{ic}\in\{s,t\}$, i.e. the ith entry in $\mathbf{y}_c$ will be s if its corresponding instance is from class c, or it equals t otherwise. Similar remarks apply to $\mathbf{y}'_c$, which is the cth column of another indicator matrix whose $y'_{ic}\in\{s',t'\}$. Let $\mathbf{A}$ be the centered data matrix, and $\mathbf{w}_c$ and $\mathbf{w}'_c$ be the least squares solutions with coding schemes $\mathbf{y}_c$ and $\mathbf{y}'_c$, respectively. The two LDA solution models $\mathbf{w}_c$ and $\mathbf{w}'_c$ will satisfy the following equation*:

$$\mathbf{w}_c=\frac{t-s}{t'-s'}\mathbf{w}'_c. \tag{8}$$

Proof. First, we have

$$\mathbf{y}_c=\frac{t-s}{t'-s'}\mathbf{y}'_c-\mathbf{e}\frac{s'(t-s)-s(t'-s')}{t'-s'}, \tag{9}$$

where $\mathbf{e}=[1,1,\ldots,1]\in\mathbb{R}^{p\times 1}$. Thus

$$\begin{aligned}\mathbf{w}_c&=(\mathbf{A}^\top\mathbf{A})^{-1}\mathbf{A}^\top\mathbf{y}_c\\ &=(\mathbf{A}^\top\mathbf{A})^{-1}\mathbf{A}^\top\frac{t-s}{t'-s'}\mathbf{y}'_c\\ &\quad-(\mathbf{A}^\top\mathbf{A})^{-1}\mathbf{A}^\top\mathbf{e}\frac{s'(t-s)-s(t'-s')}{t'-s'}\\ &=(\mathbf{A}^\top\mathbf{A})^{-1}\mathbf{A}^\top\frac{t-s}{t'-s'}\mathbf{y}'_c\\ &=\frac{t-s}{t'-s'}\mathbf{w}'_c. \qquad \square\end{aligned} \tag{10}$$

We note that $\mathbf{A}^\top\mathbf{e}$ in the above equation is equal to $\mathbf{0}$, since A is centered (i.e. global mean removed). Based on Proposition 1, we have the following relationship between the three different indicator matrices discussed above

$$\mathbf{w}_{2c}=\frac{1}{\sqrt{n_c}}\mathbf{w}_{1c}=\frac{1}{\sqrt{n_c n}}\mathbf{w}_c \quad \text{for } c=1,\ldots,k, \tag{11}$$

where $\mathbf{w}_c$, $\mathbf{w}_{1c}$, and $\mathbf{w}_{2c}$ are the least squares solutions using $\mathbf{Y}$, $\mathbf{Y}_1$, and our simplified indicator matrix $\mathbf{Y}_2$, respectively. This verifies that the use of our indicator matrix results in valid LDA subspace (and an equivalent solution model). Our LDA solutions can be normalized by a factor $\sqrt{n_c}$ or $\sqrt{n_c n}$ if necessary (e.g. if n_c is very different between classes). This simplified indicator matrix $\mathbf{Y}_2$ will be applied in our proposed rank-one update approach for LSLDA, as we detail in the following section.

4. LSLDA with concept drift

4.1. Our rank-one update method for LSLDA

As previously discussed, the least squares LDA solution $\mathbf{W}$ can be computed as [9]

$$\mathbf{W}=(\mathbf{A}^\top\mathbf{A})^{-1}\mathbf{A}^\top\mathbf{Y}. \tag{12}$$

As the proof provided in Section 3.2, we can simply use a binary class indicator matrix $\mathbf{Y}$ in (12) to derive the LSLDA solution $\mathbf{W}$. When a new data instance is received (e.g. incremental or streaming data problems), the calculation of $(\mathbf{A}^\top\mathbf{A})^{-1}$ is the key to derive the updated LSLDA model. Suppose $\mathbf{A}_i$ is the current data matrix and $\mathbf{A}_{i+1}$ is the data matrix with a new instance $\mathbf{x}_{i+1}$. The new solution to this LSLDA can be calculated as

$$\mathbf{W}_{i+1}=(\mathbf{A}_{i+1}^\top\mathbf{A}_{i+1})^{-1}\mathbf{A}_{i+1}^\top\mathbf{Y}_{i+1}, \tag{13}$$

which does not involve with the solution from the previous iteration

$$\mathbf{W}_i=(\mathbf{A}_i^\top\mathbf{A}_i)^{-1}\mathbf{A}_i^\top\mathbf{Y}_i. \tag{14}$$

To avoid computing $(\mathbf{A}_{i+1}^\top\mathbf{A}_{i+1})^{-1}$ whenever a new data point is received, the Woodbury matrix identity [31] can be applied to utilize the information from the current $(\mathbf{A}_i^\top\mathbf{A}_i)^{-1}$. This popular identity technique provides a rank-k correction to the inverse of the original matrix $\mathbf{M}$, i.e.

$$(\mathbf{M}+\mathbf{UCV})^{-1}=\mathbf{M}^{-1}-\mathbf{M}^{-1}\mathbf{U}(\mathbf{C}^{-1}+\mathbf{VM}^{-1}\mathbf{U})^{-1}\mathbf{VM}^{-1}, \tag{15}$$

where $\mathbf{M}\in\mathbb{R}^{n\times n}$, $\mathbf{U}\in\mathbb{R}^{n\times k}$, $\mathbf{C}\in\mathbb{R}^{k\times k}$, and $\mathbf{V}\in\mathbb{R}^{k\times n}$.

We apply the above property and compute the solution $\mathbf{W}_{i+1}$ with $\mathbf{W}_i$. To be more precise, let $\mathbf{M}_i=\mathbf{A}_i^\top\mathbf{A}_i$ and $\mathbf{N}_i=\mathbf{A}_i^\top\mathbf{Y}_i$, we have

$$\begin{aligned}\mathbf{W}_{i+1}&=\mathbf{M}_{i+1}^{-1}\mathbf{N}_{i+1}\\&=\mathbf{M}_{i+1}^{-1}(\mathbf{N}_i+\mathbf{x}_{i+1}\mathbf{y}_{i+1}^\top)=\mathbf{M}_{i+1}^{-1}(\mathbf{M}_i\mathbf{W}_i+\mathbf{x}_{i+1}\mathbf{y}_{i+1}^\top)\\&=\mathbf{M}_{i+1}^{-1}((\mathbf{M}_{i+1}-\mathbf{x}_{i+1}\mathbf{x}_{i+1}^\top)\mathbf{W}_i+\mathbf{x}_{i+1}\mathbf{y}_{i+1}^\top)\\&=\mathbf{W}_i+\mathbf{M}_{i+1}^{-1}\mathbf{x}_{i+1}(\mathbf{y}_{i+1}^\top-\mathbf{x}_{i+1}^\top\mathbf{W}_i)\\&=\mathbf{W}_i+\frac{\mathbf{M}_i^{-1}\mathbf{x}_{i+1}}{1+\mathbf{x}_{i+1}^\top\mathbf{M}_i^{-1}\mathbf{x}_{i+1}}(\mathbf{y}_{i+1}^\top-\mathbf{x}_{i+1}^\top\mathbf{W}_i),\end{aligned} \tag{16}$$

where $\mathbf{M}_{i+1}^{-1}\mathbf{x}_{i+1}=\mathbf{M}_i^{-1}\mathbf{x}_{i+1}/(1+\mathbf{x}_{i+1}^\top\mathbf{M}_i^{-1}\mathbf{x}_{i+1})$ is derived by Woodbury matrix identity [10]. It is worth noting that this update process needs to keep and update the inverse matrix $\mathbf{M}_i^{-1}\in\mathbb{R}^{p\times p}$ in each iteration. For large-scale problems with low-dimensional data ($n\gg p$), this update will work efficiently. However, this technique prohibits rank-one updates when the dimensionality p of data is large.

In order to alleviate the problem when dealing with high-dimensional data, we choose to update the vector $\mathbf{M}_i^{-1}\mathbf{x}_{i+1}$ directly. Our recursive rank-one update process can be achieved by utilizing the Woodbury matrix identity recursively in the following way:

$$\begin{aligned}\mathbf{M}_i^{-1}\mathbf{x}_{i+1}&=\mathbf{M}_{i-1}^{-1}\mathbf{x}_{i+1}-\frac{\mathbf{M}_{i-1}^{-1}\mathbf{x}_i\mathbf{x}_i^\top\mathbf{M}_{i-1}^{-1}}{1+\mathbf{x}_i^\top\mathbf{M}_{i-1}^{-1}\mathbf{x}_i}\mathbf{x}_{i+1}\\&=\mathbf{M}_{i-2}^{-1}\mathbf{x}_{i+1}-\frac{\mathbf{M}_{i-2}^{-1}\mathbf{x}_{i-1}\mathbf{x}_{i-1}^\top\mathbf{M}_{i-2}^{-1}}{1+\mathbf{x}_{i-1}^\top\mathbf{M}_{i-2}^{-1}\mathbf{x}_{i-1}}\mathbf{x}_{i+1}-\frac{\mathbf{M}_{i-1}^{-1}\mathbf{x}_i\mathbf{x}_i^\top\mathbf{M}_{i-1}^{-1}}{1+\mathbf{x}_i^\top\mathbf{M}_{i-1}^{-1}\mathbf{x}_i}\mathbf{x}_{i+1}\\&\vdots\\&=\mathbf{M}_0^{-1}\mathbf{x}_{i+1}-\sum_{k=0}^{i-1}\frac{\mathbf{M}_k^{-1}\mathbf{x}_{k+1}\mathbf{x}_{k+1}^\top\mathbf{M}_k^{-1}}{1+\mathbf{x}_{k+1}^\top\mathbf{M}_k^{-1}\mathbf{x}_{k+1}}\mathbf{x}_{i+1}\\&=\mathbf{x}_{i+1}-\sum_{k=0}^{i-1}\frac{\mathbf{t}_k\mathbf{t}_k^\top}{s_k}\mathbf{x}_{i+1}=\mathbf{t}_i,\end{aligned} \tag{17}$$

where

$$\mathbf{t}_i=\mathbf{M}_i^{-1}\mathbf{x}_{i+1}, s_i=1+\mathbf{x}_{i+1}^\top\mathbf{M}_i^{-1}\mathbf{x}_{i+1} \quad\text{and}\quad \mathbf{M}_0^{-1}=\mathbf{I}.$$

The update of $\mathbf{W}$ is thus reformulated as follows:

$$\mathbf{W}_{i+1}=\mathbf{W}_i+\frac{\mathbf{t}_i}{s_i}(\mathbf{y}_{i+1}^\top-\mathbf{x}_{i+1}^\top\mathbf{W}_i). \tag{18}$$

From (17) and (18), we see that $\mathbf{t}_i$ can be calculated by $\mathbf{T}=[\mathbf{t}_0^\top,\mathbf{t}_1^\top,\ldots,\mathbf{t}_{i-1}^\top]\in\mathbb{R}^{i\times p}$ from previous iterations, and thus we only need to keep i of these p-dimensional $\mathbf{t}_i$ vectors (i.e. $\mathbf{T}$) when computing $\mathbf{W}_{i+1}$. This avoids the limitation in prior LSLDA methods which require to store a $p\times p$ data covariance matrix while the data dimensionality is very large (i.e. $p\gg n$).

4.2. LSLDA with concept drift via rank-one updates

To address the problem of concept drift, the dependency of the LSLDA model on the outdated data should be decreased, and one should determine such dependency by the type of concept drift (i.e. gradual or abrupt). To add this ability to our rank-one update formulation for LSLDA, we introduce a forgetting factor $\beta<1$ into (13). As a result, we have

$$\mathbf{A}_{i+1}^\top\mathbf{A}_{i+1}=\tilde{\mathbf{M}}_{i+1}=\beta\tilde{\mathbf{M}}_i+\mathbf{x}_{i+1}\mathbf{x}_{i+1}^\top \tag{19}$$

and

$$\mathbf{A}_{i+1}^\top\mathbf{Y}_{i+1}=\tilde{\mathbf{N}}_{i+1}=\beta\tilde{\mathbf{N}}_i+\mathbf{x}_{i+1}\mathbf{y}_{i+1}. \tag{20}$$

We note that $\tilde{\mathbf{M}}$ and $\tilde{\mathbf{N}}$ indicate the data matrices with concept drift. From the above equations, it can be seen that we suppress the influence of outdated data by a factor of β when calculating the latest outer product matrix. The value of this forgetting factor (<1) can be adjusted, depending on the type of concept drift. Using this forgetting factor β, we now have a new formulation of $\mathbf{W}$, i.e.

$$\begin{aligned}\tilde{\mathbf{W}}_{i+1}&=\tilde{\mathbf{M}}_{i+1}^{-1}\tilde{\mathbf{N}}_{i+1}\\&=\tilde{\mathbf{W}}_i+\frac{\tilde{\mathbf{M}}_i^{-1}\mathbf{x}_{i+1}}{\beta+\mathbf{x}_{i+1}^\top\tilde{\mathbf{M}}_i^{-1}\mathbf{x}_{i+1}}(\mathbf{y}_{i+1}^\top-\mathbf{x}_{i+1}^\top\tilde{\mathbf{W}}_i).\end{aligned} \tag{21}$$

In the above equation, we can further simplify the numerator of the second term by specifying

$$\begin{aligned}\tilde{\mathbf{M}}_i^{-1}\mathbf{x}_{i+1}&=\beta^{-i}\mathbf{x}_{i+1}-\beta^{-i}\frac{\tilde{\mathbf{t}}_0\tilde{\mathbf{t}}_0^\top}{\tilde{s}_0}\mathbf{x}_{i+1}-\beta^{-i}\frac{\tilde{\mathbf{t}}_1\tilde{\mathbf{t}}_1^\top}{\tilde{s}_1}\mathbf{x}_{i+1}\\&\quad-\cdots-\beta^{-i}\frac{\tilde{\mathbf{t}}_{i-1}\tilde{\mathbf{t}}_{i-1}^\top}{\tilde{s}_{i-1}}\mathbf{x}_{i+1}\\&=\beta^{-i}\left(\mathbf{x}_{i+1}-\frac{\tilde{\mathbf{t}}_0\tilde{\mathbf{t}}_0^\top}{\tilde{s}_0}\mathbf{x}_{i+1}-\cdots-\frac{\tilde{\mathbf{t}}_{i-1}\tilde{\mathbf{t}}_{i-1}^\top}{\tilde{s}_{i-1}}\mathbf{x}_{i+1}\right)=\beta^{-i}\tilde{\mathbf{t}}_i,\end{aligned} \tag{22}$$

where

$$\tilde{s}_i=\beta^{i+1}+\beta^i\mathbf{x}_{i+1}^\top\tilde{\mathbf{M}}_i^{-1}\mathbf{x}_{i+1}=\beta^{i+1}+\mathbf{x}_{i+1}^\top\tilde{\mathbf{t}}_i.$$

Recall that $\tilde{\mathbf{W}}$, $\tilde{\mathbf{M}}$, and $\tilde{\mathbf{t}}$ denote the solutions with concept drift, and we have $\tilde{\mathbf{M}}_0=\mathbf{I}$ and $\tilde{\mathbf{W}}_0=\mathbf{0}$ for initialization. Therefore, the final solution of our LSLDA via rank-one update can be expressed as follows:

$$\begin{aligned}\tilde{\mathbf{W}}_{i+1}&=\tilde{\mathbf{W}}_i+\frac{\beta^{-i}\tilde{\mathbf{t}}_i}{\beta+\mathbf{x}_{i+1}^\top\beta^{-i}\tilde{\mathbf{t}}_i}(\mathbf{y}_{i+1}-\mathbf{x}_{i+1}^\top\tilde{\mathbf{W}}_i)\\&=\tilde{\mathbf{W}}_i+\frac{\tilde{\mathbf{t}}_i}{\tilde{s}_i}(\mathbf{y}_{i+1}^\top-\mathbf{x}_{i+1}^\top\tilde{\mathbf{W}}_i).\end{aligned} \tag{23}$$

The pseudo-code of our proposed LSLDA with concept drift is described in Algorithm 1.

Algorithm 1. Rank-one update LSLDA with concept drift.

Require: The new instance $\mathbf{x}_{i+1}$ with the corresponding class label y_{i+1}, the current mean $\boldsymbol{\mu}_i$,

Table 1
Description of classification datasets.

Dataset	Classes	Training/testing	Attributes
pendigits	10	7494/3498	16
letter	26	16 000/4000	16
reuters-top2	2	4522/1805	11 941
medline	5	1250/1250	22 095

$\tilde{\mathbf{T}}=[\tilde{\mathbf{t}}_0\cdots\tilde{\mathbf{t}}_{i-2},\tilde{\mathbf{t}}_{i-1}]$, $\tilde{\mathbf{s}}=[\tilde{s}_0\cdots\tilde{s}_{i-2},\tilde{s}_{i-1}]$, $\tilde{\mathbf{W}}_i$, β
Ensure: The resulting LDA solution $\tilde{\mathbf{W}}_{i+1}\in\mathbb{R}^{p\times k}$.
$\boldsymbol{\mu}_{i+1}\leftarrow\frac{i}{i+1}\boldsymbol{\mu}_i+\frac{1}{i+1}\mathbf{x}_{i+1}$
$\mathbf{x}_{i+1}\leftarrow\mathbf{x}_{i+1}-\boldsymbol{\mu}_{i+1}$
$\mathbf{y}_{i+1}\leftarrow\mathbf{0}\in\mathbb{R}^{1\times k}$
if $y_{i+1}\in c$ **then**
  $\mathbf{y}_{i+1}(c)=1$
end if
$\tilde{\mathbf{t}}_i\leftarrow\mathbf{x}_{i+1}-\frac{\tilde{\mathbf{t}}_0\tilde{\mathbf{t}}_0^\top}{\tilde{s}_0}\mathbf{x}_{i+1}-\cdots-\frac{\tilde{\mathbf{t}}_{i-1}\tilde{\mathbf{t}}_{i-1}^\top}{\tilde{s}_{i-1}}\mathbf{x}_{i+1}$
$\tilde{s}_i\leftarrow\beta^{i+1}+\mathbf{x}_{i+1}^\top\tilde{\mathbf{t}}_i$
$\tilde{\mathbf{W}}_{i+1}\leftarrow\tilde{\mathbf{W}}_i+\frac{\tilde{\mathbf{t}}_i}{\tilde{s}_1}(\mathbf{y}_{i+1}^\top-\mathbf{x}_{i+1}^\top\tilde{\mathbf{W}}_i)$

5. Experimental results

5.1. Classification tasks without concept drift

5.1.1. Multi-class classification

We first compare our LSLDA with standard LDA, LSLDA, and LS-ILDA on four different datasets. Note that we chose to compare our proposed algorithm with LSLDA and LS-ILDA, while the LS-ILDA has been shown to outperform several state-of-the-art incremental LDA approaches (e.g. [21]) in [11]. The detailed information of these datasets is described in Table 1, including the sizes of training and test sets and the number of feature attributes for each. The first two datasets in Table 1, `pendigits`, and `letter`, are available at UCI Machine Learning Repository [32] and the UCI Statlog[1] collection. The other two datasets, `medline` and `reuters-top2`,[2] are for text categorization. For the `reuters` dataset, we only include the two categories with the largest numbers of instances for our experiments. One important characteristic of the datasets for text categorization is that the number of features is typically much larger than that of the data instances (i.e. $p\gg n$), which makes the calculation of the inverse of scatter matrices very computationally expensive.

For each LDA model considered in our experiments, we first determine its optimal linear subspace using training set data, and we project the test data accordingly. For simplicity, a nearest neighbor classifier is applied for classification in all experiments. Table 2 lists the classification performances of different LDA models. It is clear that our LSLDA method performed as well as conventional LDA, LSLDA and LS-ILDA did, and this verifies the effectiveness of our proposed LSLDA approach for different types of classification problems. Although only the recognition rate with $\beta=1$ is presented in Table 2, our empirical results did not observe a significant variation in terms of recognition rate when other $\beta<1$ values were used. This is because none of the datasets considered here have concept-drifting data, and thus our method produced comparable results with $\beta\leq 1$. However, we expect (and will observe) the advantage of our proposed LSLDA method for problems with concept drift, as we discuss in the next subsection.

Table 2
Recognition performance of LDA, LSLDA, LS-ILDA and our proposed method on different datasets. Note that $\beta=1$ is used on our formulation, and it shows that we achieved comparable results with state-of-the-art LDA approaches when no concept drifting data is presented.

Dataset	LDA	LSLDA [9]	LS-ILDA [11]	Ours
pendigits	0.9440	0.9434	0.9483	0.9451
letter	0.9150	0.9320	0.9556	0.9554
reuters-top2	0.9490	0.9629	0.9607	0.9651
medline	0.8816	0.8768	0.8648	0.8828

It is worth noting that, since the dimensionality p of the data in `medline` and `reuters` datasets is large, it is difficult to determine the LDA solution using standard LDA or LSLDA methods. In this paper, we apply linear kernel techniques to compute their solutions, which have been shown to produce comparable performances as the standard LDA method does [33].

Table 3 compares computation complexities and memory requirements of different LDA models. Since the standard LDA and LSLDA methods need to store either the data matrix or the scatter covariance matrix, their memory requirement is $O(\max(n,p)\times p)$. The time complexity of these two methods is $O(p^3)$ due to their need to compute the inverse of the scatter matrices. As discussed in Section 4.1, when the dimensionality is much greater than the dataset size (i.e. $p\gg n$), our proposed LSLDA only needs to store the matrix $\mathbf{T}\in\mathbb{R}^{n\times p}$, and only $O(n\times p)$ is required to produce the LDA model (see (17)). On the other hand, if $n\gg p$, it is easier and more efficient for us to keep the matrix $\mathbf{M}^{-1}\in\mathbb{R}^{p\times p}$, and thus the computational complexity is $O(p\times p)$ (see (16)). For completeness, we also compare these requirements to those reported by LS-ILDA in [11]. However, it is worth repeating that LS-ILDA is *not* designed to handle concept-drifting data. When dealing with general classification problems without concept drift, we do not expect significant recognition differences by LS-ILDA. Therefore, we did not report their performances in this experiment.

5.1.2. Classification with newly added classes

One of the advantages of our rank-one update LSLDA is the ability to recognize data with new class labels in online or streaming data applications. In such cases, we can easily extend our class indicator matrix by adding an additional column for the new class label of interest. Thus, our new LDA solution can be computed as

$$\tilde{\mathbf{W}}_{i+1}=\tilde{\mathbf{W}}_i+\frac{\tilde{\mathbf{t}}_i}{\beta^{i+1}\mathbf{x}_{i+1}^\top\tilde{\mathbf{t}}}(\tilde{\mathbf{y}}_{i+1}-\mathbf{x}_{i+1}^\top\tilde{\mathbf{W}}_i),\tag{24}$$

where $\tilde{\mathbf{W}}_{i+1}\in\mathbb{R}^{p\times(k+1)}$ and $\tilde{\mathbf{W}}_i=[\tilde{\mathbf{W}}_i,\mathbf{0}]\in\mathbb{R}^{p\times(k+1)}$, and $\tilde{\mathbf{y}}_{i+1}\in\mathbb{R}^{1\times(k+1)}$.

To evaluate the classification performance for this case, we consider the `medline` dataset and use the data from the first three classes (i.e. classes 1–3) to design the initial LDA model. Samples from classes 4 to 5 are considered as newly added data to be recognized, but they are not used in the beginning of this learning process. In other words, we extract 50% of the training data from classes 1 to 3, and use them to design the associated LDA model beforehand. When the experiment starts, we sequentially add the remaining of the training data from *all* classes 1 to 5 in a streaming fashion, and we evaluate the classification performance on the test data from all classes. Fig. 1 shows the classification accuracy of this learning task. The horizontal axis

[1] http://www.is.umk.pl/projects/datasets-stat.html.
[2] The `reuters` and `medline` dataset is available at http://www.cc.gatech.edu/~hpark/data.html.

Table 3
Comparisons of LDA, LSLDA, and our proposed LSLDA in terms of computational complexity and memory requirements. Note that n and p are the numbers of instances and dimensions, respectively.

	LDA	LSLDA [9]	LS-ILDA [11]	Ours
Computation complexity	$O(p^3)$	$O(p^3)$	$O(\min(n,p)\times p)$	$O(\min(n,p)\times p)$
Memory requirement	$O(\max(n,p)\times p)$	$O(\max(n,p)\times p)$	$O(\min(n,p)\times p)$	$O(\min(n,p)\times p)$

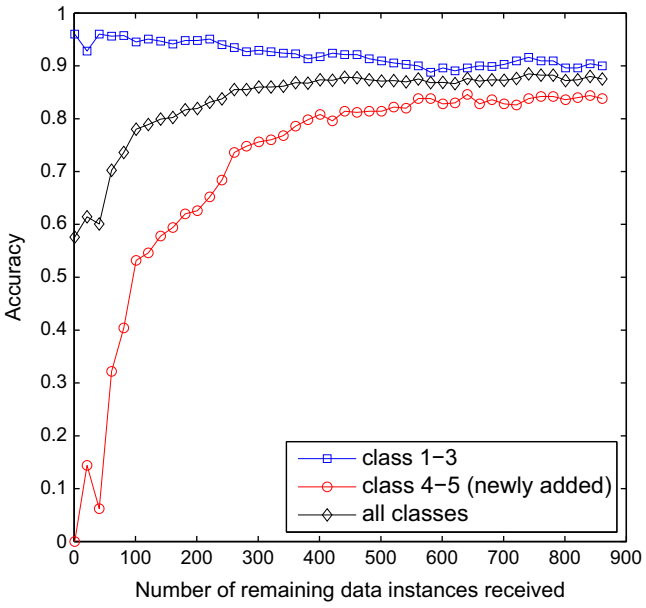

Fig. 1. Classification results of newly added classes. (For interpretation of the references to color in this figure caption, the reader is referred to the web version of this article.)

in Fig. 1 indicates the number of the training data received, and the vertical axis shows the classification rate for different cases. Three accuracy curves can be observed in Fig. 1, i.e. classes 1–3 (blue), classes 4–5 (red), and the average one (black). We see that the information of newly added data is gradually adopted in the designed LDA model when the number of receiving data increases. These results verify the feasibility of our LDA method for learning tasks with newly added data of interest, which is practical in many real-life online classification applications.

5.2. Classification of concept drifting data

5.2.1. Synthetic 2D dataset with concept drift

We first consider a 2D synthetic dataset with a total of 2000 instances for binary classification (see Fig. 2). The first 1000 instances are sampled from the following multivariate normal distributions (500 instances for each class):

$$\mathbf{x}_+ \sim N(\mu_+,\sigma), \tag{25}$$

$$\mathbf{x}_- \sim N(\mu_-,\sigma), \tag{26}$$

where

$$\begin{bmatrix}\mu_+\\ \mu_-\end{bmatrix}=\begin{bmatrix}1 & 1\\ -1 & -1\end{bmatrix} \quad \text{and} \quad \sigma=\begin{bmatrix}0.8 & 0\\ 0 & 0.8\end{bmatrix}.$$

To simulate the concept-drifting scheme, the remaining 1000 instances are sampled from a different distribution setting

$$\mathbf{x}'_+ \sim N(\mu'_+,\sigma), \tag{27}$$

$$\mathbf{x}'_- \sim N(\mu'_-,\sigma), \tag{28}$$

where

$$\begin{bmatrix}\mu'_+\\ \mu'_-\end{bmatrix}=\begin{bmatrix}1 & -1\\ -1 & 1\end{bmatrix}.$$

When the forgetting factor $\beta=1$ is used in our LSLDA (i.e. equivalent to LS-ILDA [11]), all 2000 data instances are considered equally important, and thus we will produce an LDA solution without concept drift. Fig. 3a shows the result of the LS-ILDA solution after receiving the first 1000 instances. It can be seen that this projection provides excellent separation between the projected data, and thus is a good estimate of LDA solution. After all 2000 instances are received, the resulting projection direction (black line) is shown in Fig. 3b. We see that the first 1000 data points are now considered as outdated data, and thus the associated projection is shown in gray for better visualization. The final LS-ILDA solution (i.e. the black line without concept drift $\beta=1$) is equivalent to that of the conventional LDA, since both consider the entire dataset for calculating the LDA solution.

Next, we consider the case of concept drift with $\beta=0.99$ in our LSLDA, and show the results in Fig. 3c and d. In Fig. 3c, our LSLDA again estimates the projection well for the first 1000 instances, and the solution is very similar to that in Fig. 3a. When receiving the next 1000 instances as shown in Fig. 3d, our LSLDA results in a projection (black line) which provides excellent discrimination between the recently received 1000 instances. Therefore, these results verify the use of our forgetting factor for concept drifting and data stream tasks, where the influence of outdated data should be suppressed when updating the LDA solution.

5.2.2. Recognition of UCI data with concept drift

We now perform recognition tasks with concept drift using the `pendigits` dataset. The scenario of concept drift is created by changing the classes of interest during the learning process. More specifically, we load the training data from digits 1 and 3 in a streaming fashion to design our initial LSLDA model to recognize the test data from these two classes. Once all training samples from these two classes are used, we change the classes of interest to digits 2 and 4 and use their training data instead (also in a streaming way), and the associated test inputs will be from these two new classes as well. The purpose of this is to see whether how quickly our LDA model learns from these data with different distributions. To evaluate the performance, we calculate the recognition rate using all test data points from the corresponding classes (but not in a steaming fashion) whenever a training sample is received during the above learning process.

Fig. 4 shows recognition results using LS-ILDA and our proposed LSLDA with $\beta=0.99$, 0.95, and 0.9. When LS-ILDA is used, the derived LDA model considers all the previously received data equally important. As can be seen from Fig. 4, the concept drift occurs when the 1500th instance is received. Although LS-ILDA incrementally updates its solution, it is not designed to handle concept drifting data and thus requires the longest time to recover the drop of recognition rate. On the other hand, our LSLDA exhibits its ability to learn concept-drifting data and quickly update the LDA model right after the concept drift occurs. From Fig. 4, it can also be observed that our LSLDA solution with a

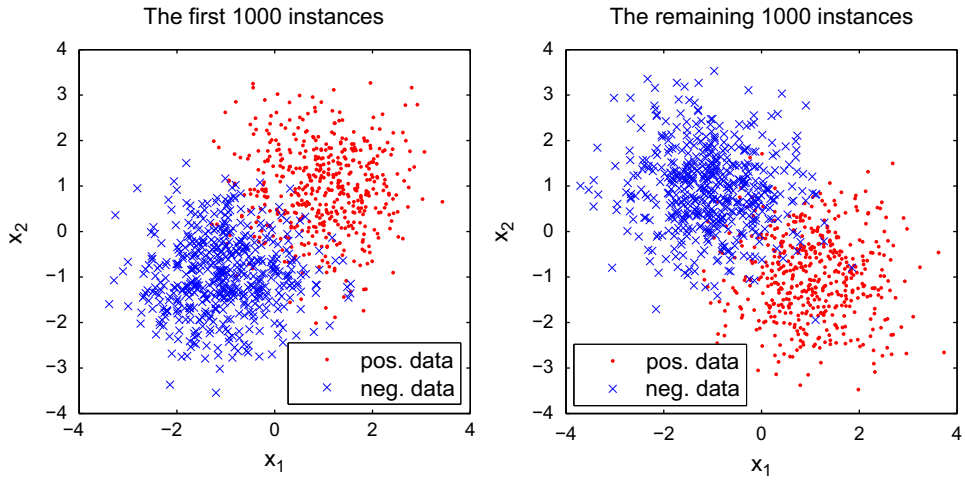

Fig. 2. Our 2D synthetic data: the first 1000 instances (left) and the next 1000 instances (right). The instances from class 1 are shown in red, while those from class −1 are shown in blue. (For interpretation of the references to color in this figure caption, the reader is referred to the web version of this article.)

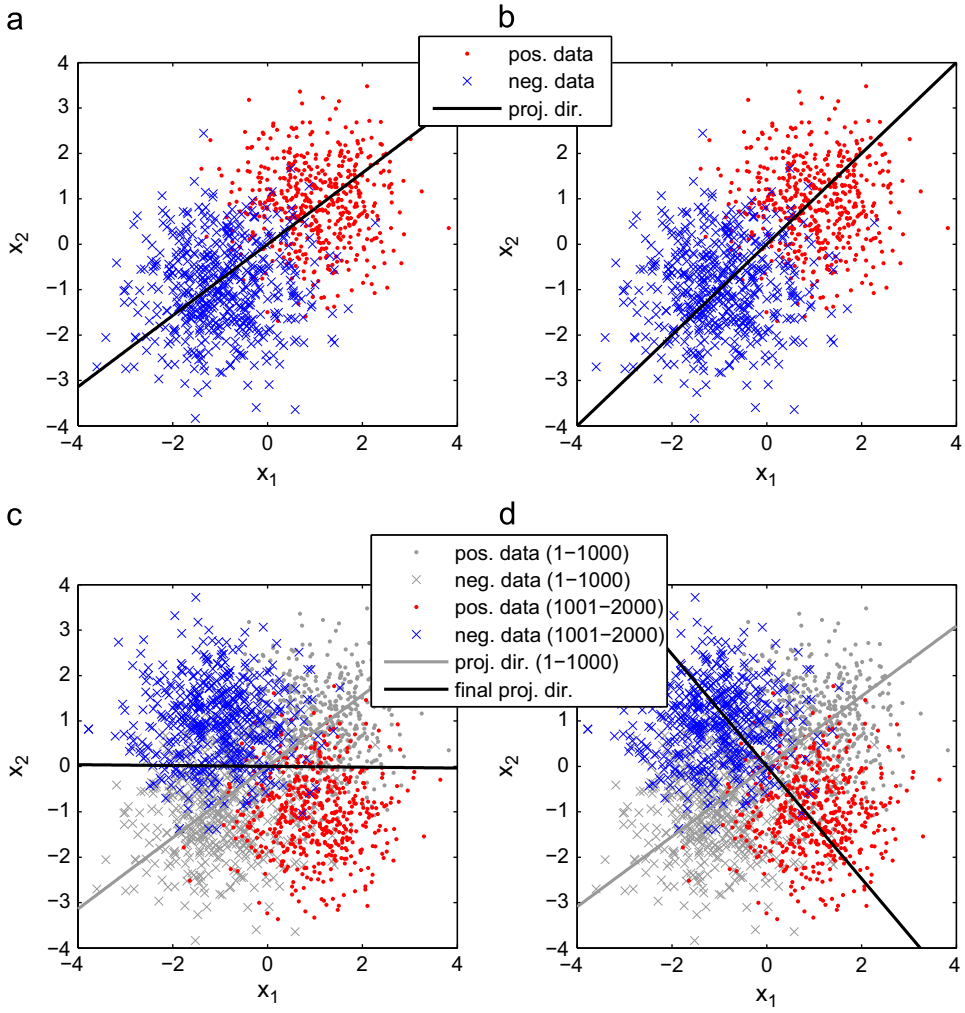

Fig. 3. LDA models produced by LS-ILDA [11] ((a) and (c)) and our proposed LSLDA ((b) and (d)) before and after concept drift occurs. Note that $\beta = 0.99$ is used in our LSLDA to handle concept-drifting data.

smaller β value (e.g. $\beta = 0.9$) is less stable than those derived by larger β. This is because, when a smaller β is used to produce our LSLDA model, the mostly recent received data instances will be utilized in updating the associated LSLDA model.

As one of the major contributions of our work, we introduce a forgetting factor $\beta \leq 1$ in the proposed rank-one update formulation for LSLDA, so that our LSLDA is able to suppress the influence of outdate data accordingly. The value of β determines how fast

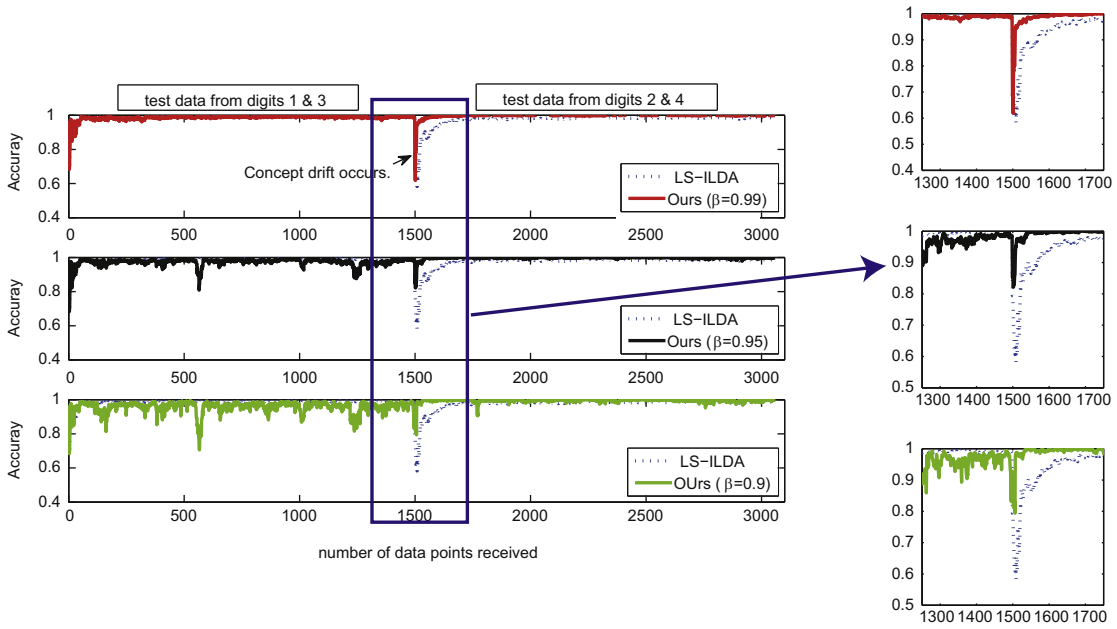

Fig. 4. Recognition performance of the `pendigits` dataset with concept drift using LS-ILDA [11] and our LSLDA with different β values.

the LSLDA model can be adapted to the recently received instances. If a gradual concept drift occurs, one can choose a larger β (i.e. closer to one) since data received earlier still contribute to the construction of the resulting LSLDA model. On the other hand, if an abrupt concept drift is of concern, one will choose a smaller β, which focuses more on the newly arrived data. However, a smaller β value would cause the stability of the derived LSLDA. This is because that the derived LSLDA will be highly dependent on the most recently received instances, which might be corrupted due to noise, etc. presented. Therefore, for practical problems, the *adaptivity* and the *stability* will be a tradeoff for the proposed LSLDA, which is controlled by β. It is worth noting that, whether the concept drift of concern is gradual or abrupt, the final LSLDA model (i.e. the model derived after *all* instances are received) will *not* be sensitive to the choice of β.

5.2.3. Recognition of real-world concept-drifting data

In the final part of our experiments, we evaluate the performance of our LSLDA on a real-world news dataset with concept drift. This news dataset has been studied in [34], which uses Usenet articles from 20 Newsgroups[3] collection; this dataset consists of 5995 instances, each with a total of 27 893 features. Table 4 shows the newsgroups of interest before and after the concept drift occurs. We see that the newsgroups of interest changes after the 3000th instance. As a result, we only report the recognition performance from the 3001th to 3300th instances, since it will be easier to observe the difference between LDA models with and without the ability to handle concept drift.

To conduct the experiment on this dataset, the initial LS-ILDA and our LSLDA are both trained on the first 3000 data instances. When the concept drift occurs (i.e. the 3001th data instance is received), the existing LDA model will be used to predict the label of that data point. Whether the prediction is correct or not, this received data instance and its ground truth label will be used to update the LS-ILDA and our LSLDA models accordingly.

[3] The 20 Newsgroups dataset is available at http://kdd.ics.uci.edu/databases/20newsgroups/20newsgroups.html.

Fig. 5 shows the resulting recognition performance of LS-ILDA and our LSLDA. We note that the vertical axis of Fig. 5 indicates the average recognition accuracy when the $(3000+i)$th data point is received. More precisely, the numerator of the average recognition accuracy is i, while the denominator is the number of correct prediction among the first i test inputs. From this figure, we see that our proposed method has an improved recognition rate compared with the LS-ILDA. This is due to the additional ability of our LSLDA to recognize concept-drifting data. It is also worth noting that, since the two types of concepts listed in Table 4 are not mutually exclusive of each other, we expect both LDA models will eventually achieve comparable recognition performance when more data points (3001th–4500th) are received. This is why the difference between the two curves in Fig. 5 becomes smaller toward the end (right-hand side) of the figure. From the last two parts of our experiments, we verify that our proposed LSLDA not only achieves satisfying recognition performance as prior LDA models do, our method also exhibits excellent ability in learning concept-drifting data, which cannot be easily achieved by prior LDA approaches.

6. Conclusion

We proposed a rank-one update approach for least squares LDA with concept drift, which efficiently updates the LDA subspace in an incremental fashion while significantly reducing the computation complexity and memory requirement. Using our simplified class indicator matrix for multiple linear regression, our approach updates and derives the LSLDA model efficiently, and we verified that the use of the proposed class indicator matrix is equivalent to more complex ones which were previously used in prior LSLDA models. The introduction of the forgetting factor to our LSLDA model makes our solutions adaptive to newly received data with distribution changes, and thus decreases the dependency of the resulting LDA model on outdated data. Our experimental results confirmed the effectiveness of our LSLDA on learning tasks with and without concept drift. We also compared computational complexities and memory requirements

Table 4
The news dataset and its newsgroups of interest over time.

Newsgroup	Instances	
	1–3000	3001–4500
Comp.pc.hardware	Yes	Yes
Comp.mac.hardware	No	No
Rec.autos	Yes	Yes
Rec.motorcycles	No	No
Rec.sport.baseball	Yes	
Rec.sport.hockey	No	
Sci.med		Yes
Sci.space		No

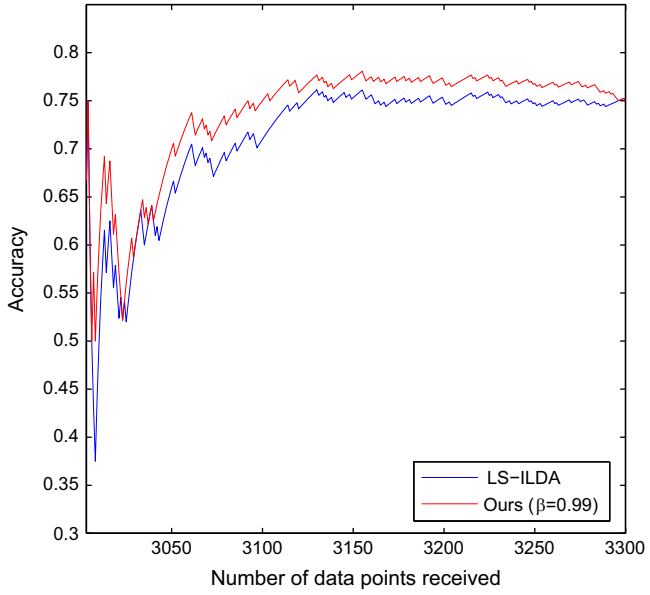

Fig. 5. Recognition of the news dataset when concept drift occurs.

of different LDA models, and we verified that ours is among the most efficient ones while achieving comparable recognition performance as standard LDA and LSLDA do.

References

[1] T. Hastie, R. Tibshirani, J. Friedman, The Elements of Statistical Learning: Data Mining, Inference, and Prediction, Springer, 2009.
[2] C.-N. Hsu, H.-S. Huang, Y.-M. Chang, Y.-J. Lee, Periodic step-size adaptation in second-order gradient descent for single-pass on-line structured labeling, Machine Learning 77 (2) (2009) 195–224.
[3] L. Bottou, Y.L. Cun, Large scale online learning, in: Proceedings of Advances in Neural Information Processing Systems, 2004, pp. 217–224.
[4] Q. Dua, H. Ren, Real-time constrained linear discriminant analysis to target detection and classification in hyperspectral imagery, Pattern Recognition 36 (2003) 1–12.
[5] X. Wu, Yunde Jia, Wei Liang, Incremental discriminant-analysis of canonical correlations for action recognition, Pattern Recognition 43 (2010) 4190–4197.
[6] Y. Pang, H. Yan, Y. Yuan, K. Wang, Robust coHOG feature extraction in human-centered image/video management system, IEEE Transactions on Systems, Man, and Cybernetics, Part B 42 (2) (2012) 458–468.
[7] Y. Pang, Y. Yuan, K. Wang, Learning optimal spatial filters by discriminant analysis for brain–computer-interface, Neurocomputing 77 (1) (2012) 20–27.
[8] L. Sun, S. Ji, J. Ye, A least squares formulation for a class of generalized eigenvalue problems in machine learning, in: Proceedings of International Conference on Machine Learning, 2009, pp. 977–984.
[9] J. Ye, Least squares linear discriminant analysis, in: Proceedings of International Conference on Machine Learning, 2007.
[10] S. Haykin, Adaptive Filter Theory, 4th edition, Prentice Hall, 2002.
[11] L.-P. Liu, Y. Jiang, Z.-H. Zhou, Least squares incremental linear discriminant analysis, in: Proceedings of IEEE International Conference on Data Mining, 2009, pp. 298–306.
[12] I. Higuchi, S. Eguchi, Robust principal component analysis with adaptive selection for tuning parameters, Journal of Machine Learning Research 5 (2004) 453–471.
[13] S.-Y. Huang, Y.-R. Yeh, S. Eguchi, Robust kernel principal component analysis, Neural Computation 21 (11) (2009) 1–35.
[14] P. Howland, M. Jeon, H. Park, Structure preserving dimension reduction for clustered text data based on the generalized singular value decomposition, SIAM Journal on Matrix Analysis and Applications 25 (1) (2003) 165–179.
[15] J. Ye, R. Janardan, C.H. Park, H. Park, An optimization criterion for generalized discriminant analysis on undersampled problems, IEEE Transactions on Pattern Analysis and Machine Intelligence 26 (2004) 982–994.
[16] J. Ye, Q. Li, LDA/QR: an efficient and effective dimension reduction algorithm and its theoretical foundation, Pattern Recognition 37 (4) (2004) 851–854.
[17] D. Cai, X. He, J. Han, Srda: an efficient algorithm for large-scale discriminant analysis, IEEE Transactions on Knowledge and Data Engineering 20 (20) (2008) 1–12.
[18] J. Ye, Q. Li, H. Xiong, H. Park, R. Janardan, V. Kumar, IDR/QR: an incremental dimension reduction algorithm via qr decomposition, IEEE Transactions on Knowledge and Data Engineering 17 (9) (2005) 1208–1222.
[19] S. Pang, S. Ozawa, N. Kasabov, Incremental linear discriminant analysis for classification of data streams, IEEE Transactions on Systems, Man, and Cybernetics, Part B 35 (5) (2005) 905–914.
[20] T.-K. Kim, S.-F. Wong, B. Stenger, J. Kittler, R. Cipolla, Incremental linear discriminant analysis using sufficient spanning set approximations, in: Proceedings of IEEE Conference on Computer Vision and Pattern Recognition, 2007.
[21] H. Zhao, P.C. Yuen, Incremental linear discriminant analysis for face recognition, IEEE Transactions on Systems, Man, and Cybernetics, Part B 38 (1) (2008) 210–221.
[22] E.A.K. James, S. Annadurai, Implementation of incremental linear discriminant analysis using singular value decomposition for face recognition, in: Proceedings of International Conference on Advanced Computing, 2009, pp. 172–175.
[23] S. Delany, P. Cunningham, A. Tsymbal, A comparison of ensemble and case-base maintenance techniques for handling concept drift in spam filtering, in: Proceeding of American Association for Artificial Intelligence, 2006.
[24] L.L. Minku, A.P. White, X. Yao, The impact of diversity on online ensemble learning in the presence of concept drift, IEEE Transactions on Knowledge and Data Engineering 22 (5) (2010) 730–742.
[25] I. Zliobaite, Learning Under Concept Drift: An Overview, Technical Report, 2010.
[26] R. Klinkenberg, Learning drifting concepts: example selection vs. example weighting, Intelligent Data Analysis 8 (3) (2004) 281–300.
[27] G. Widmer, M. Kubat, Learning in the presence of concept drift and hidden contexts, Machine learning 23 (1) (1996) 69–101.
[28] H. Wang, W. Fan, P. Yu, J. Han, Mining concept-drifting data streams using ensemble classifiers, in: Proceedings of ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, 2003.
[29] P. Zhang, X. Zhu, Y. Shi, Categorizing and mining concept drifting data streams, in: Proceeding of ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, 2009.
[30] J. Oommen, L. Rueda, Stochastic learning-based weak estimation of multinomial random variables and its applications to pattern recognition in non-stationary environments, Pattern Recognition 39 (3) (2006) 328–341.
[31] N. Higham, Accuracy and Stability of Numerical Algorithms, SIAM, 2002.
[32] A. Asuncion, D. Newman, UCI repository of machine learning databases, 〈http://www.ics.uci.edu/∼mlearn/mlrepository.html〉, 2007.
[33] Y.-R. Yeh, S.-Y. Huang, Y.-J. Lee, Nonlinear dimension reduction with kernel sliced inverse regression, IEEE Transactions on Knowledge and Data Engineering 21 (11) (2009) 1590–1603.
[34] I. Katakis, G. Tsoumakas, E. Banos, N. Bassiliades, I. Vlahavas, An adaptive personalized news dissemination system, Journal of Intelligent Information Systems 32 (2) (2009) 191–212.

Yi-Ren Yeh received his M.S. and Ph.D. degrees from the Department of Computer Science and Information Engineering, National Taiwan University of Science and Technology, Taiwan, in 2006 and 2010, respectively. From August 2008 to May 2009, he was a visiting scholar of CyLab, Carnegie Mellon University, Pittsburgh, USA. He was a postdoctoral research fellow of the Research Center for Information Technology Innovation (CITI) at Academia Sinica, Taipei, Taiwan. He is currently a postdoctoral research fellow of Intel-NTU Connected Context Computing Center, National Taiwan University, Taipei, Taiwan. His research interests include machine learning, data mining, optimization, numerical methods, and pattern recognition.

Yu-Chiang Frank Wang received his B.S. degree in Electrical Engineering from National Taiwan University, Taipei, Taiwan, in 2001. From 2001 to 2002, he worked as a research assistant at the National Health Research Institutes, Taiwan. He received his M.S. and Ph.D. degrees in Electrical and Computer Engineering from Carnegie Mellon University, Pittsburgh, USA, in 2004 and 2009, respectively.

Dr. Wang joined the Research Center for Information Technology Innovation (CITI) of Academia Sinica, Taiwan, in 2009, where he holds the position as a tenure-track assistant research fellow. He leads the Multimedia and Machine Learning Lab at CITI, and works in the fields of signal and image processing, computer vision, and machine learning. From 2010 to 2011, he is a visiting scholar of the Department of Computer Science and Information Engineering at National Taiwan University Science and Technology, Taiwan.