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

Yi-Ren Yeh, Yu-Chiang Frank Wang

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

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 In other words, our LSLDA is able to observe and model the data distribution changes, while the dependency on outdated data will be suppressed.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 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
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) [11] 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 received 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 $\beta$ 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 $W$ to the standard LDA solves the
been shown that needs a class indicator matrix presents a data instance (with the global mean removed). Besides, one entry satisfies the following equation:

\[ Y = [y_1, y_2, \ldots, y_n] \in \mathbb{R}^{n \times k}, \]

where each entry satisfies the following equation:

\[ Y(i,c) = \begin{cases} \sqrt{n}/n & \text{if } x_i \in c, \\ -\sqrt{n}/n & \text{otherwise.} \end{cases} \]  

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

\[ W_{\text{MLR}} = (A^\top A)^{-1} A^\top Y, \]

which is proved in [9] that \( W_{\text{MLR}} \) satisfies the following equation:

\[ W_{\text{MLR}} = [W_{\text{D}} D.0]Q^\top. \]

In (4), \( D \) is a diagonal matrix while \( Q \) is an orthogonal one. It has been shown that \( D \) would be an identity matrix under a mild condition which typically occurs when \( n < p \) [9]. The solution \( W_{\text{MLR}} \) in (4) indicates that the LSLDA model is derived from a multiple linear regression problem, and we will simply use \( W \) 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 \( Y \) (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

\[ Y_s(i,c) = \begin{cases} \frac{1}{\sqrt{n_c}} & \text{if } x_i \in c, \\ 0 & \text{otherwise.} \end{cases} \]

Using this form, the solution \( W_i \) to their LSLDA satisfies the following equation:

\[ W_i = \frac{1}{\sqrt{n}} W, \]

where \( W \) and \( W_i \) are the solutions of LSLDA using the coding schemes \( Y \) and \( Y_s \) 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 the 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 will 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.,

\[ Y_s(i,c) = \begin{cases} 1 & \text{if } x_i \in c, \\ 0 & \text{otherwise.} \end{cases} \]

In the following proposition, we will show that the use of our indicator matrix is equivalent to the use of previous ones (i.e., \( Y \) and \( Y_s \)). It is worth mentioning that there are only two different values 1 or 0 in each column of our indicator matrix \( Y_s \), 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 \( y_c \) is the \( c \)-th column of an indicator matrix (e.g., \( Y_1 \)). We have \( y_{c+} \in [s,t] \), i.e., the \( i \)-th entry in \( y_{c+} \) will be \( s \) if its corresponding instance is from class \( c \), or it equals \( t \) otherwise. Similar remarks apply to \( y_{c-} \), which is the \( c \)-th column of another indicator matrix whose \( y_{c-} \in [s',t'] \). Let \( A \) be the centered data matrix, and \( w_c \) and \( w_c^* \) be the least squares solutions with coding schemes \( y_{c+} \) and \( y_{c-} \), respectively. The two LDA solution models \( w_c \) and \( w_c^* \) will satisfy the following equation:

\[ w_c = \frac{t-s}{t-s'} w_c^*. \]

**Proof.** First, we have

\[ y_i = \frac{t-s}{t-s'} y_i^* - \frac{e(s(t-s) - s(t'-s'))}{t-s}. \]

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

\[ w_c = (A^\top A)^{-1} A^\top y_i,
= (A^\top A)^{-1} A^\top \left( \frac{t-s}{t-s'} y_i^* - \frac{e(s(t-s) - s(t'-s'))}{t-s} \right),
= (A^\top A)^{-1} A^\top \frac{t-s}{t-s'} y_i^* - \frac{e(s(t-s) - s(t'-s'))}{t-s} A^\top,
= \frac{t-s}{t-s'} w_c^* - \frac{e(s(t-s) - s(t'-s'))}{t-s} A^\top. \]

We note that \( A^\top e \) in the above equation is equal to \( 0 \), 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

\[ w_{c+} = \frac{1}{\sqrt{n_c}} w_{c+} \]

where \( w_{c+} \), \( w_{c-} \), and \( w_{c+} \) are the least squares solutions using \( Y \), \( Y_s \), and our simplified indicator matrix \( Y_s \), 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 \( Y_s \) 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 $W$ can be computed as [9]

$$W = (A^T A)^{-1} A^T Y.$$  
(12)

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

$$W_{i+1} = (A_i^T A_i)^{-1} A_i^T Y_i,$$  
(13)

which does not involve with the solution from the previous iteration

$$W_i = (A_i^T A_i)^{-1} A_i^T Y_i.$$  
(14)

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

$$(M + UCV)^{-1} = M^{-1} - M^{-1} UC^{-1} VM^{-1} U^T C^{-1} VM^{-1},$$  
(15)

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

We apply the above property and compute the solution $W_{i+1}$ with $W_i$. To be more precise, let $M_i = A_i^T A_i$ and $N_i = A_i^T Y_i$, we have

$$W_{i+1} = M_i^{-1} N_i + x_{i+1} y_{i+1}^T = M_i^{-1} (M_i W_i + x_{i+1} y_{i+1}^T)$$

$$= M_i^{-1} (W_i + M_i^{-1} x_{i+1} y_{i+1}^T W_i)$$

$$= W_i + M_i^{-1} x_{i+1} (y_{i+1}^T M_i^{-1} x_{i+1})^{-1} y_{i+1}^T W_i,$$  
(16)

where $M_i^{-1} x_{i+1} = M_i^{-1} x_{i+1} / (1 + x_{i+1}^T M_i^{-1} x_{i+1})$ is derived by the Woodbury matrix identity [10]. It is worth noting that this update process needs to keep and update the inverse matrix $M_i^{-1}$ in $\mathbb{R}^{p \times p}$ in each iteration. For large-scale problems with low-dimensional data ($n > p$), 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 $M_i^{-1} x_{i+1}$. Our recursive rank-one update process can be achieved by utilizing the Woodbury matrix identity recursively in the following way:

$$M_i^{-1} x_{i+1} = - M_i^{-1} x_{i+1} y_{i+1}^T M_i^{-1} x_{i+1} / (1 + x_{i+1}^T M_i^{-1} x_{i+1})$$

$$= M_i^{-1} x_{i+1} - M_i^{-1} x_{i+1} y_{i+1}^T M_i^{-1} x_{i+1} / (1 + x_{i+1}^T M_i^{-1} x_{i+1})$$

...  

$$= M_i^{-1} x_{i+1} - \sum_{k=0}^{i-1} \frac{M_k^{-1} x_{i+1} y_{i+1}^T M_k^{-1} x_{i+1}}{1 + x_{i+1}^T M_k^{-1} x_{i+1}}$$

$$= x_{i+1} - \sum_{k=0}^{i-1} \frac{t_k x_{i+1}}{s_k} = t_i,$$  
(17)

where $t_i = M_i^{-1} x_{i+1}, s_i = 1 + x_{i+1}^T M_i^{-1} x_{i+1}$ and $M_0^{-1} = I$.

The update of $W$ is thus reformulated as follows:

$$W_{i+1} = W_i + \frac{t_i (y_{i+1}^T x_{i+1} W_i)}{s_i}.$$  
(18)

From (17) and (18), we see that $t_i$ can be calculated by $T = [t_0, t_1, \ldots, t_i] \in \mathbb{R}^{i+1}$ from previous iterations, and thus we only need to keep $i$ of these $p$-dimensional $t_i$ vectors (i.e. $T$) when computing $W_{i+1}$. 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 ($i > n$).

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

$$A_i^{-1} A_{i+1} = M_i^{-1} N_i + \beta x_{i+1} y_{i+1}^T$$

and

$$A_i^{-1} y_{i+1} = N_i + \beta x_{i+1} y_{i+1}^T.$$  
(19)

We note that $M_i$ and $N_i$ 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 $\beta$ when calculating the latest outer product matrix. The value of this forgetting factor ($\beta < 1$) can be adjusted, depending on the type of concept drift. Using this forgetting factor $\beta$, we now have a new formulation of $W$, i.e.

$$W_{i+1} = M_i^{-1} N_i + \beta x_{i+1} y_{i+1}^T - (y_{i+1}^T x_{i+1} W_i).$$  
(20)

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

$$M_i^{-1} x_{i+1} = \beta^{-1} x_{i+1} - \beta^{-1} \frac{t_{i-1} x_{i-1}}{s_{i-1}} - \beta^{-1} \frac{t_{i-2} x_{i-2}}{s_{i-2}} - \beta^{-1} \frac{t_{i-3} x_{i-3}}{s_{i-3}} - \beta^{-1} \frac{t_{i-4} x_{i-4}}{s_{i-4}} - \beta^{-1} \frac{t_{i-5} x_{i-5}}{s_{i-5}} - \beta^{-1} \frac{t_{i-6} x_{i-6}}{s_{i-6}} - \beta^{-1} \frac{t_{i-7} x_{i-7}}{s_{i-7}} - \beta^{-1} \frac{t_{i-8} x_{i-8}}{s_{i-8}} - \beta^{-1} \frac{t_{i-9} x_{i-9}}{s_{i-9}} - \beta^{-1} \frac{t_{i-10} x_{i-10}}{s_{i-10}}$$

$$= \beta^{-1} x_{i+1} - \beta^{-1} \frac{t_{i-1} x_{i-1}}{s_{i-1}} - \beta^{-1} \frac{t_{i-2} x_{i-2}}{s_{i-2}} - \beta^{-1} \frac{t_{i-3} x_{i-3}}{s_{i-3}} - \beta^{-1} \frac{t_{i-4} x_{i-4}}{s_{i-4}} - \beta^{-1} \frac{t_{i-5} x_{i-5}}{s_{i-5}} - \beta^{-1} \frac{t_{i-6} x_{i-6}}{s_{i-6}} - \beta^{-1} \frac{t_{i-7} x_{i-7}}{s_{i-7}} - \beta^{-1} \frac{t_{i-8} x_{i-8}}{s_{i-8}} - \beta^{-1} \frac{t_{i-9} x_{i-9}}{s_{i-9}} - \beta^{-1} \frac{t_{i-10} x_{i-10}}{s_{i-10}}$$

(21)

$$= \beta^{-1} x_{i+1} - \beta^{-1} t_i x_i$$

(22)

where

$$s_i = \beta^{i+1} + \beta^i x_{i+1} M_i^{-1} x_{i+1} = \beta^{i+1} + x_{i+1}^T t_i.$$ Recalling that $W_i, M_i$ and $t_i$ denote the solutions with concept drift, and we have $M_0 = I$ and $W_0 = 0$ for initialization. Therefore, the final solution of our LSLDA via rank-one update can be expressed as follows:

$$W_{i+1} = W_i + \frac{t_i (y_{i+1}^T x_{i+1} W_i)}{s_i}$$

$$= W_i + \frac{t_i (y_{i+1}^T x_{i+1} W_i)}{s_i}.$$  
(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 $x_{i+1}$ with the corresponding class label $y_{i+1}$, the current mean $\mu_i$. 

The update of $W$ is thus reformulated as follows:
Table 1
Description of classification datasets.

<table>
<thead>
<tr>
<th>Dataset</th>
<th>Classes</th>
<th>Training/testing Attributes</th>
</tr>
</thead>
<tbody>
<tr>
<td>pendigits</td>
<td>10</td>
<td>7494/3498</td>
</tr>
<tr>
<td>letter</td>
<td>26</td>
<td>16 000/4000</td>
</tr>
<tr>
<td>reuters-top2</td>
<td>2</td>
<td>4522/1805</td>
</tr>
<tr>
<td>medline</td>
<td>5</td>
<td>1250/1250</td>
</tr>
</tbody>
</table>

Table 2
Recognition performance of LDA, LSLDA, LS-ILDA and our proposed method on different datasets. Note that β = 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 present.

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>pendigits</td>
<td>0.9440</td>
<td>0.9434</td>
<td>0.9483</td>
<td>0.9451</td>
</tr>
<tr>
<td>letter</td>
<td>0.9150</td>
<td>0.9320</td>
<td>0.9556</td>
<td>0.9554</td>
</tr>
<tr>
<td>reuters-top2</td>
<td>0.9490</td>
<td>0.9629</td>
<td>0.9651</td>
<td>0.9651</td>
</tr>
<tr>
<td>medline</td>
<td>0.8816</td>
<td>0.8768</td>
<td>0.8648</td>
<td>0.8828</td>
</tr>
</tbody>
</table>

\[ T = \{t_0, \cdots, t_{n-1}\}, \quad S = \{s_0, \cdots, s_{n-2}, s_{n-1}\}, \quad W_i, \quad \beta \]

**Ensure:** The resulting LDA solution \( W_{i+1} \in \mathbb{R}^{p \times k} \).

\[
\begin{align*}
\mu_i & = \frac{1}{n_i} \sum_{x \in \mathcal{C}_i} x_i + \beta \sum_{x \in \mathcal{S}} x_i \\
\Sigma_i^{-1} & = \frac{1}{n_i} \sum_{x \in \mathcal{C}_i} (x_i - \mu_i) (x_i - \mu_i)^T + \beta \sum_{x \in \mathcal{S}} (x_i - \mu_i) (x_i - \mu_i)^T \\
y_{i+1} & = 0 \in \mathbb{R}^{1 \times k} \\
\text{if } y_{i+1} & \text{ then} \\
W_{i+1} & = W_i + \gamma_i (y_{i+1}^T x_i^T)
\end{align*}
\]

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 standard LDA 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 features 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

(continued)
To simulate the concept-drifting scheme, the remaining 1000 instances are sampled from a different distribution setting $X’_+ \sim N(\mu’_+, \sigma)$,\\
$X’_- \sim N(\mu’_-, \sigma)$,

where $\begin{bmatrix} \mu’_+ \\ \mu’_- \end{bmatrix} = \begin{bmatrix} 1 & -1 \\ -1 & 1 \end{bmatrix}$ and $\sigma = \begin{bmatrix} 0.8 & 0 \\ 0 & 0.8 \end{bmatrix}$.

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 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

| 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. |
|---|---|---|---|---|
| Computation complexity | $O(p^2)$ | $O(p^2)$ | $O(\min(n,p) \times p)$ | $O(\min(n,p) \times p)$ |
| 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.)
smaller $\beta$ value (e.g. $\beta = 0.9$) is less stable than those derived by larger $\beta$. This is because, when a smaller $\beta$ 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 $\beta$ determines how fast
Table 4 shows the newsgroups of interest before and after the concept drift. This news dataset has been studied in [34], which uses Usenet articles from 20 Newsgroups collection; this dataset consists of 5995 instances, each with a total of 27,893 features.

Models with and without the ability to handle concept drift. Since it will be easier to observe the difference between LDA and our LSLDA, we only report the concept drift occurs. We see that the newsgroups of interest are 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 $\beta$. 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 $\beta$.

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 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 change 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.

![Fig. 4. Recognition performance of the pendigits dataset with concept drift using LS-ILDA [11] and our LSLDA with different $\beta$ values.](image)

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.

<table>
<thead>
<tr>
<th>Newsgroup</th>
<th>Instances</th>
</tr>
</thead>
<tbody>
<tr>
<td>Comp.pc.hardware</td>
<td>Yes</td>
</tr>
<tr>
<td>Comp.mac.hardware</td>
<td>No</td>
</tr>
<tr>
<td>Rec.autos</td>
<td>Yes</td>
</tr>
<tr>
<td>Rec.motorcycles</td>
<td>No</td>
</tr>
<tr>
<td>Rec.sport.baseball</td>
<td>Yes</td>
</tr>
<tr>
<td>Rec.sport.hockey</td>
<td>No</td>
</tr>
<tr>
<td>Sci.med</td>
<td>Yes</td>
</tr>
<tr>
<td>Sci.space</td>
<td>No</td>
</tr>
</tbody>
</table>

1–3000 | 3001–4500

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


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 (CTI) 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.