Chunk Incremental IDR/QR LDA Learning

Yiming Peng Shaoning Pang Gang Chen Abdolhossein Sarrafzadeh Tao Ban Daisuke Inoue

***Abstract*—Training data in real world is often presented in random chunks. Yet existing sequential Incremental IDR/QR LDA (s-QR/IncLDA) can only process data one sample after another. This paper proposes a constructive chunk Incremental IDR/QR LDA (c-QR/IncLDA) for multiple data samples incremental learning. Given a chunk of s samples for incremental learning, the proposed c-QR/IncLDA increments current discriminant model $\boldsymbol{\Omega}$, by implementing computation on the compressed the residue matrix $\boldsymbol{\Delta} \in \mathcal{R}^{d\times\eta}$, instead of the entire incoming data chunk $\widetilde{\boldsymbol{X}} \in \mathcal{R}^{d\times s}$, where $\eta \leq s$ holds. Meanwhile, we derive a more accurate reduced within-class scatter matrix $\boldsymbol{W}$ to minimize the discriminative information loss at every incremental learning cycle. It is noted that the computational complexity of c-QR/IncLDA can be more expensive than s-QR/IncLDA for single sample processing. However, for multiple samples processing, the computational efficiency of c-QR/IncLDA deterministically surpasses s-QR/IncLDA when the chunk size is large, i.e., $s \gg \eta$ holds. Moreover, experiments evaluation shows that the proposed c-QR/IncLDA can achieve an accuracy level that is competitive to batch QR/LDA and is consistently higher than s-QR/IncLDA.**

I. Introduction

Incremental learning (IL) addresses the real-world situation that data is always given in either sequential (i.e., one sample after another) or chunk manner (i.e., a group of samples is presented at a time) [1], [2]. Given an existing discriminant model $\boldsymbol{\Omega}$ (also called discriminant eigenspace model), the incremental learning of $\boldsymbol{\Omega}$ is an updating process as follow,

$$\boldsymbol{\Omega}' = \begin{cases} \mathcal{F}_{sq}(\boldsymbol{\Omega}, \widetilde{\boldsymbol{x}}) \\ \mathcal{F}_{ck}(\boldsymbol{\Omega}, \widetilde{\boldsymbol{X}}), \quad \widetilde{\boldsymbol{X}} = \{\widetilde{\boldsymbol{x}}_i\}_{i=1}^{s} \end{cases},$$

where $\widetilde{\boldsymbol{x}}$ denotes a newly presented data sample, and $\{\widetilde{\boldsymbol{x}}_i\}_{i=1}^{s}$ represents a chunk of s data samples. $\mathcal{F}_{sq}$ and $\mathcal{F}_{ck}$ perform respectively the sequential and chunk incremental learning to calculate the updated model $\boldsymbol{\Omega}'$ over any newly presented data, in expectation of $\boldsymbol{\Omega}' = \boldsymbol{\Omega}([\boldsymbol{X}\ \widetilde{\boldsymbol{x}}])$ or $\boldsymbol{\Omega}' = \boldsymbol{\Omega}([\boldsymbol{X}\ \widetilde{\boldsymbol{X}}])$. Normally $\mathcal{F}_{ck}$ is computationally less efficient than $\mathcal{F}_{sq}$, when the chunk size $s = 1$ [2], [3]. This paper studies incremental linear discriminant analysis (IncLDA) with a focus on how a $\mathcal{F}_{ck}$ is able to match and surpass the $\mathcal{F}_{sq}$ on the learning efficiency without sacrificing accuracy.

Linear Discriminant Analysis (LDA) seeks a linear projection of data that best discriminates two or more classes by the Fisher's criterion or its equivalents [4], [5], [6]. In principle, the computing of batch LDAs in different disciplines can be formulated as a n-tuple model. The classical Fisher LDA [5] constructs the discriminant eigenspace on the within-class scatter matrix $\boldsymbol{S}_w$ and between-class scatter matrices $\boldsymbol{S}_b$. Thus a Fisher LDA can be represented as a 3-tuple model $\boldsymbol{\Omega}(\boldsymbol{X}) = \{\boldsymbol{S_w}, \boldsymbol{S_b}\}$ [2]. The Least Square LDA [7], [8] transforms the LDA optimization into a multivariate linear regression, thus runs as a 3-tuple model $\boldsymbol{\Omega}(\boldsymbol{X}) = \{\boldsymbol{C}, \boldsymbol{Y}\}$ in which $\boldsymbol{C}$ and $\boldsymbol{Y}$ are the input centroid matrix and the output indicator matrix, respectively. The QR/LDA performs LDA through a QR decomposition on several dimension-reduced scatter matrices. The discriminant model can be written as $\boldsymbol{\Omega}(\boldsymbol{X}) = \{\boldsymbol{M}, \boldsymbol{W}, \boldsymbol{B}\}$ [9], where $\boldsymbol{M}$ is the centroid matrix, and $\boldsymbol{W}$ and $\boldsymbol{B}$ represent the reduced within-class and between-class scatter matrix respectively. In this sense, we understand and conduct IncLDA as an γ-tuple model updating process.

For IL evaluation, Polikar et. al[10] proposed four general criteria as,

1) It should be able to learn new information from new data.
2) It should not require access to and retain in memory the original data.
3) It should preserve previously acquired information.
4) It should be able to accommodate new classes that may be introduced with new data.

In the case of IncLDA learning, we discuss additionally the following two criteria: 5) It should be able to maintain the same capability as batch LDA on solving Small Sample Size (SSS) problem [5]. 6) It should be able to process multiple instances at one time. As discussed above, IncLDAs involve rounds of updating for renewing each component of LDA. The update rounds grow up, with the increase of newly presented samples. The loss of discriminant information accumulates over each update step. Evidently, the raise of updates gives rise to inefficiencies. Consequently, multiple samples processing is desirable for IncLDA, as it reduces update rounds at each IncLDA cycle.

In the above six criteria, this paper proposes a chunk IDR/QR IncLDA (c-QR/IncLDA) that implements a 3-tuple updating model $\mathcal{F}_{ck}$ as,

$$\begin{aligned} \boldsymbol{\Omega}' &= \mathcal{F}_{ck}(\{\boldsymbol{M}, \boldsymbol{W}, \boldsymbol{B}\}, \widetilde{\boldsymbol{X}}),\ \widetilde{\boldsymbol{X}} = \{\widetilde{\boldsymbol{x}}_i\}_{i=1}^{s} \\ &= \mathcal{F}_{ck}(\{\boldsymbol{M}, \boldsymbol{W}, \boldsymbol{B}\}, \widetilde{\boldsymbol{Y}}),\ \widetilde{\boldsymbol{Y}} = \{\widetilde{\boldsymbol{y}}_i\}_{i=1}^{\eta} \\ s.t. \quad & \eta < s \end{aligned} \tag{1}$$

where the IncLDA is conducted on the compressed data matrix $\widetilde{\boldsymbol{Y}} \in \mathbb{R}^{d\times\eta}$, instead of the original data matrix $\widetilde{\boldsymbol{X}} \in \mathbb{R}^{d\times s}$ newly presented. Hereafter, a symbol topped with "~" represents the same definition as without, but corresponds

Yiming Peng, Shaoning Pang, Gang Chen and Abdolhossein Sarrafzadeh are with the Department of Computing, Unitec Institute of Technology, Private Bag 92025, Auckland 1142, New Zealand (email: {ypeng, ppang, achen2, hsarrafzadeh}@unitec.ac.nz).

Tao Ban and Daisuke Inoue are with National Institute of Information and Communications Technology, 4-2-1 Nukui-Kitamachi, Koganei, Tokyo 184-8795, Japan (email: {bantao, dai}@nict.go.jp)

to newly added data samples. For example, n represents the total number of training samples so far, and $\widetilde{n}$ specifies the number of newly added samples.

II. Related Work

Conceptually, incremental learning (IL) needs to process each data sample to update every component of $\boldsymbol{\Omega}$ separately, especially sequential IL. Sequential IL updates learning model for merely one sample at one time. Adding one sample triggers a great number of update calculations. The inaccuracies and inefficiencies might build up quickly for IL as the number of data samples increases [11], [12]. Chunk IL is more likely to improve the accuracy and efficiency, because it processes data in chunk manner. In other words, multiple samples are processed at one time, which reduces substantially the number of updates for one incremental learning cycle.

It is worth noting that chunk IL recently has gained increasing attention [11], [13], [2], [14], [12], [15]. For incremental principle component analysis (IncPCA), Hall et.al [11] developed chunk IncPCA by merging original and newly created eigenspaces; Ozawa et. al [12] developed another version of chunk IncPCA by implementing both eigenspace rotation and augmentation. For incremental support vector machine (IncSVM), Karasuyama et. al [16] extended a sequential method [3] to chunk IncSVM algorithm by updating multiple Lagrange multipliers and keeping Karush-Kuhn-Tucker (KKT) balance.

The concept of chunk IncLDA is first discussed in Pang's algorithm (Fisher/IncLDA)[2], in which both sequential and chunk version of Fisher/IncLDA are developed and compared in terms of their computation efficiency, memory usage, and eigenspace discriminability. The comparison results indicate that the computational efficiency of c-Fisher/IncLDA outperforms sequential s-Fisher/IncLDA, and improves continuously with the increase of chunk size. However, this method cannot address the SSS problem [17]. To mitigate the problem, Zhao et.al proposed an alternative chunk IncLDA, called GSVD/IncLDA [18]. GSVD/IncLDA conducts a none one-pass incremental learning [19], [2], [20], as a result, all newly added data samples are always retained in memory at each incremental learning cycle. Therefore, it is not memory efficient for incremental learning over large size datasets. Considering the drawbacks of GSVD/IncLDA, Liu et. al [8] developed another very different IncLDA, called LS/IncLDA. Since the multivariate linear regression (MLR) is used for model construction instead of eigen-decomposition, it enjoys a better computational efficiency. However, the LS/IncLDA performs merely a sequential data processing, for which inefficiency builds up, since update calculation is performed for every data sample.

The s-QR/IncLDA, proposed by Ye et. al. [9], updates the batch QR/LDA by an efficient QR-updating approach. In practice, this method is applicable to any online feature extraction, because for a d-dimension k-class data space,

1) it conducts incremental dimensionality reduction in $d-k$ scale with the least loss of discriminative information.
2) it updates a QR-decomposition of a $d \times k$ matrix and solve an eigen-decomposition of a $k \times k$ matrix. Thus, it is computational efficient.
3) it is one-pass incremental learning, where IL is executed for every new data sample presented.
4) the SSS problem is well-handled in the algorithm by applying regularization on the reduced within-class scatter matrix $\boldsymbol{W}$.

An important drawback of s-QR/IncLDA is, it supports in principle just sequential mode IL. Also, we notice that approximation is involved for the updating of reduced within-class scatter matrix $\boldsymbol{W}$ and QR-decomposition. This results in the ineffectiveness of s-QR/IncLDA for data with large number of classes. Motivated by the chunk data processing of QR/IncLDA, we propose a new constructive incremental LDA learning method by learning QR/IncLDA from compressed data streams. The idea is, whatever big chunk of data presented for IL, we compress the data into a smaller size dataset, and update the QR/IncLDA by learning from the compressed data. The proposed c-QR/IncLDA inherits not only all remarkable characteristics of s-QR/IncLDA, but also gives in nature better performance than s-QR/IncLDA. In particular, it is capable of processing multiple samples at a time, so that its computational efficiency grows with the chunk size of data. Specifically, the bigger size the dataset provided at one time, the faster speed the proposed c-QR/IncLDA gives. Moreover, the eigenspace discriminablity is strengthened owing to the derivation of a deterministically more reliable meanwhile more accurate update of the reduced within-class scatter matrix $\boldsymbol{W}$.

III. Sequential IDR/QR IncLDA

Given $\boldsymbol{\Omega} = \{\boldsymbol{M}, \boldsymbol{W}, \boldsymbol{B}\}$ as a QR/LDA discriminant model on $\boldsymbol{X}$. Let $\widetilde{\boldsymbol{x}}$ be a new data sample presented; let $\widetilde{\boldsymbol{x}}$ belong to the ith class. Without loss of generality, let us assume that we have data from the first to the kth class, just before $\widetilde{\boldsymbol{x}}$ is presented. The sequential learning of IDR/QR IncLDA (s-QR/IncLDA) [9] can be described as

$$\boldsymbol{\Omega}' = \mathcal{F}_{sq}(\boldsymbol{\Omega}(\boldsymbol{X}), \widetilde{\boldsymbol{x}}) = \{\boldsymbol{M}', \boldsymbol{W}', \boldsymbol{B}'\}.$$

where we represent a primed variable as its updated version after data insertion. For example, n_i' denotes the updated number of elements in the ith class, and k' as the updated class number.

1) If $\widetilde{\boldsymbol{x}}$ belongs to an existing class (i.e., $i \leq k$), then $k' = k$, $n_i' = n_i + 1$ and $n' = n + 1$. The updating of the centroid matrix is calculated as

$$\boldsymbol{M}' = \boldsymbol{M} + \frac{\widetilde{\boldsymbol{x}} - \boldsymbol{m}_i}{n_i^{'}} \boldsymbol{g}^T, \tag{2}$$

where $\boldsymbol{m}_i$ is the ith class centroid, and $\boldsymbol{g} = (0, \ldots, 1, \ldots, 0)$ with 1 appearing at the ith position. By implementing QR-updating in two stages: 1) Rank-one QR-updating; and 2) QR-updating when inserting a new row, we can compute the updated QR-decomposition $\boldsymbol{Q}' \in \mathcal{R}^{d \times k}$ and $\boldsymbol{R}' \in \mathcal{R}^{k \times k}$ such that $\boldsymbol{Q}'\boldsymbol{R}' = \boldsymbol{M}'$.

The updating of the reduced within-class scatter matrix $\boldsymbol{W}$ is,

$$\boldsymbol{W}' = \boldsymbol{W} + (\boldsymbol{\alpha} - \boldsymbol{\beta})(\boldsymbol{\alpha} - \boldsymbol{\beta})^T + n_i \boldsymbol{\beta}\boldsymbol{\beta}^T, \quad (3)$$

where $\boldsymbol{\alpha} = \boldsymbol{Q}^{'T}(\widetilde{\boldsymbol{x}} - \boldsymbol{m}_i)$ and $\boldsymbol{\beta} = \boldsymbol{Q}^{'T}(\boldsymbol{m}'_i - \boldsymbol{m}_i)$.
The updating of the reduced between-class scatter matrix $\boldsymbol{B}$ is,

$$\boldsymbol{B}' = (\boldsymbol{R}'\boldsymbol{D} - (\frac{1}{n'}\boldsymbol{R}' \cdot r) \cdot h^T)(\boldsymbol{R}'\boldsymbol{D} - (\frac{1}{n'}\boldsymbol{R}' \cdot r) \cdot h^T)^T, \quad (4)$$

where $\boldsymbol{D} = diag(\sqrt{n'_1}, \dots, \sqrt{n'_k})$, $h = [\sqrt{n'_1}, \dots, \sqrt{n'_k}]^T$, and $r = (n'_1, \dots, n'_k)$.

2) If $\widetilde{\boldsymbol{x}}$ belongs to a new class (i.e., $i > k$), then $k' = k+1$, $n' = n + 1$, and for $i = 1, \dots, k$, $n'_i = n_i$, $n'_{k+1} = 1$. Correspondingly, the updated centroid matrix is,

$$\boldsymbol{M}' = [\boldsymbol{m}_1, \boldsymbol{m}_2, \dots, \boldsymbol{m}_k, \widetilde{\boldsymbol{x}}] = [\boldsymbol{M}, \widetilde{\boldsymbol{x}}]. \quad (5)$$

The updated reduced within-class scatter matrix is

$$\boldsymbol{W}' = \begin{pmatrix} \boldsymbol{W} & 0 \\ 0 & 0 \end{pmatrix}. \quad (6)$$

Similar to (4), the updating of reduced between-class scatter matrix is,

$$\boldsymbol{B}' = (\boldsymbol{R}'\boldsymbol{D} - (\frac{1}{n'}\boldsymbol{R}' \cdot r) \cdot h^T)(\boldsymbol{R}'\boldsymbol{D} - (\frac{1}{n'}\boldsymbol{R}' \cdot r) \cdot h^T)^T, \quad (7)$$

where $\boldsymbol{D} = diag(\sqrt{n'_1}, \dots, \sqrt{n'_{k+1}})$, $h = [\sqrt{n'_1}, \dots, \sqrt{n'_{k+1}}]^T$, and $r = (n'_1, \dots, n'_{k+1})$.

It is worth noting that when data is presented in chunk manner, s-QR/IncLDA performs learning inefficiently for a single sample at a time. Just for one sample, a large number of updating iterations are carried out in s-QR/IncLDA, however this sample might be discriminatively redundant (i.e., gives no contribution to current discriminant model). Furthermore, the class separability of s-QR/IncLDA compromises for datasets with large number of classes. This is because that serious discriminative information lost occurs for $\boldsymbol{W}$ updating: 1) when the newly presented sample belongs to an existing class, s-QR/IncLDA uses the assumption of $Q' \approx Q$ in (3); and 2) when the newly presented sample comes from a new class, uses another assumption of $\boldsymbol{W'} \approx \boldsymbol{W}$ in (6). However when the class number is large, both assumptions lose generality.

IV. Proposed Chunk IDR/QR IncLDA

Given a batch QR/LDA eigenspace model $\boldsymbol{\Omega} = \{\boldsymbol{M}, \boldsymbol{W}, \boldsymbol{B}\}$, and a chunk of instances $\widetilde{\boldsymbol{X}}$ in $\widetilde{k}$ classes,which can involve both existing and entirely new categories. Straightforwardly, the set of total class labels is updated as $\boldsymbol{C'} = \boldsymbol{C} \cup (\widetilde{\boldsymbol{C}} - \boldsymbol{C} \cap \widetilde{\boldsymbol{C}})$ and $k' = |\boldsymbol{C'}|$. The number of total training samples is renewed as $n' = n + \widetilde{n}$. Then, the problem of the proposed c-QR/IncLDA can be described as

$$\boldsymbol{\Omega}' = \mathcal{F}_c(\boldsymbol{\Omega}, \widetilde{\boldsymbol{X}}) = \{\boldsymbol{M'}, \boldsymbol{W'}, \boldsymbol{B'}\}.$$

$\mathcal{F}_c$ updates $\boldsymbol{\Omega}$ on $\widetilde{\boldsymbol{X}}$ in three steps: (1) Updating centroid matrix $\boldsymbol{M}$ and its QR-decomposition; (2) Updating the reduced within-class scatter matrix $\boldsymbol{W}$; and (3) Updating the reduced between-class scatter matrix $\boldsymbol{B}$. Note that, differing from s-QR/IncLDA, c-QR/IncLDA can process newly added data from existing and new classes simultaneously.

A. Updating centroid matrix $\boldsymbol{M}$ and its QR-decomposition

To derive $\boldsymbol{M'} \in \mathcal{R}^{d \times k'}$ from $\boldsymbol{M} \in \mathcal{R}^{d \times k}$, the dimensionality of $\boldsymbol{M}$ is augmented from k to k' in order to accommodate new classes in the centroid matrix. In doing so, we insert zero vectors $\mathbf{0}$ at ith column of $\boldsymbol{M}$ for $i = k+1, \dots, k'$ and jth column of $\widetilde{\boldsymbol{M}}$ for $j = \widetilde{k} + 1, \dots, k'$. Thus, $\boldsymbol{M} \in \mathcal{R}^{d \times k}$ is augmented to $\hat{\boldsymbol{M}} \in \mathcal{R}^{d \times k'}$ upon the presence of $\widetilde{\boldsymbol{X}}$. Here, '^' refers to augmented matrices, and will be applied throughout the paper.

Applying matrix augmentation to $\boldsymbol{M} = \boldsymbol{QR}$, we have $\hat{\boldsymbol{M}} = \hat{\boldsymbol{Q}}\hat{\boldsymbol{R}}$. Straightforwardly, $\hat{\boldsymbol{Q}}\hat{\boldsymbol{R}}$ can be obtained by recomputing a QR-decomposition on $\hat{\boldsymbol{M}}$, or by a QR-updating on $\boldsymbol{QR}$. However, both approaches are computational expensive due to the high complexity of matrix decomposition or multiplication. For efficient $\hat{\boldsymbol{Q}}\hat{\boldsymbol{R}}$ augmentation, we propose a rule of matrix augmentation for QR-decomposition in Proposition 1, which involves only manipulations of matrix partition or combination.

Proposition 1: Given $\boldsymbol{A} = (\boldsymbol{A}_1, \boldsymbol{A}_2) = \boldsymbol{QR} \in \mathcal{R}^{d \times k}$ with $\boldsymbol{A}_1 \in \mathcal{R}^{d \times (i-1)}$, and $\boldsymbol{A}_2 \in \mathcal{R}^{d \times (k-i)}$. If a zero vector $\mathbf{0}$ is inserted as $\hat{\boldsymbol{A}} = (\boldsymbol{A}_1, \mathbf{0}, \boldsymbol{A}_2)$, then the QR-decomposition $\hat{\boldsymbol{Q}}\hat{\boldsymbol{R}}$ of $\hat{\boldsymbol{A}}$ can be computed as,

$$\hat{\boldsymbol{Q}}\hat{\boldsymbol{R}} = (\boldsymbol{Q}_1, \mathbf{0}, -\boldsymbol{Q}_2) \begin{pmatrix} \boldsymbol{R}_1 & \mathbf{0} & \boldsymbol{R}_2 \\ \mathbf{0}^T & 1 & \mathbf{0}^T \\ \boldsymbol{R}_3 & \mathbf{0} & -\boldsymbol{R}_4, \end{pmatrix}$$

where $\boldsymbol{Q}_1 \in \mathcal{R}^{d \times (i-1)}$, $\boldsymbol{Q}_2 \in \mathcal{R}^{d \times (k-i)}$, $\boldsymbol{R}_1 \in \mathcal{R}^{(i-1) \times (i-1)}$, $\boldsymbol{R}_2 \in \mathcal{R}^{(i-1) \times (k-i)}$, $\boldsymbol{R}_3 \in \mathcal{R}^{(k-i) \times (i-1)}$, and $\boldsymbol{R}_4 \in \mathcal{R}^{(k-i) \times (k-i)}$. Note that, $\boldsymbol{Q} = (\boldsymbol{Q}_1, \boldsymbol{Q}_2)$, $\boldsymbol{R} = \begin{pmatrix} \boldsymbol{R}_1 & \boldsymbol{R}_2 \\ \boldsymbol{R}_3 & \boldsymbol{R}_4 \end{pmatrix}$, and $\boldsymbol{R}_3$ is a zero matrix. The detailed proof is given in the appendix.

According to the definition of the centroid matrix, we reformulate $\boldsymbol{M'} = [\frac{\boldsymbol{m}_1 n_1 + \widetilde{\boldsymbol{m}}_1 \widetilde{n}_1}{n'_1}, \dots, \frac{\boldsymbol{m}_{k'} n_{k'} + \widetilde{\boldsymbol{m}}_{k'} \widetilde{n}_{k'}}{n'_{k'}}]$ as,

$$\boldsymbol{M'} = \hat{\boldsymbol{M}} + \boldsymbol{\Delta}. \quad (8)$$

Therefore $\boldsymbol{\Delta} = [\frac{\widetilde{n}_1}{n'_1}(\widetilde{\boldsymbol{m}}_1 - \boldsymbol{m}_1), \dots, \frac{\widetilde{n}_{k'}}{n'_{k'}}(\widetilde{\boldsymbol{m}}_{k'} - \boldsymbol{m}_{k'})] \in \mathcal{R}^{d \times k'}$. Note that $\boldsymbol{\Delta}$ here identifies the residue information of newly presented data.

To calculate $\boldsymbol{Q'R'}$, we apply the thin Singular Value Decomposition on $\boldsymbol{\Delta}$, namely $\boldsymbol{\Delta} = \boldsymbol{U \Sigma V^T}$ and obtain a summation of rank one matrices products [21] as,

$$\boldsymbol{\Delta} = \sum_{j=1}^{\eta} \boldsymbol{\Sigma}(\boldsymbol{j}, \boldsymbol{j}) \boldsymbol{U}(\boldsymbol{1} : \boldsymbol{d}, \boldsymbol{j}) \boldsymbol{V^T}(\boldsymbol{j}, \boldsymbol{1} : \boldsymbol{k'}) = \sum_{j=1}^{\eta} \boldsymbol{u_j} \boldsymbol{v_j}^T \quad (9)$$

where $\eta = \text{rank}(\boldsymbol{\Delta})$, $\boldsymbol{u_j} = \boldsymbol{\Sigma}(\boldsymbol{j}, \boldsymbol{j}) \boldsymbol{U}(\boldsymbol{1} : \boldsymbol{d}, \boldsymbol{j})$ and $\boldsymbol{v_j} = \boldsymbol{V}(\boldsymbol{j}, \boldsymbol{1} : \boldsymbol{k'})$.

Substituting (9) into (8), we can calculate $\boldsymbol{Q}'\boldsymbol{R}'$ by an iterative QR-updating procedure as,

$$\begin{aligned}\boldsymbol{Q}'\boldsymbol{R}' &= \hat{\boldsymbol{Q}}^{(\eta+1)}\hat{\boldsymbol{R}}^{(\eta+1)}\\ &= \hat{\boldsymbol{Q}}^{(\eta)}\hat{\boldsymbol{R}}^{(\eta)} + \boldsymbol{u}_\eta\boldsymbol{v}_\eta^T\\ &\dots\\ &= (\hat{\boldsymbol{Q}}^{(j)}\hat{\boldsymbol{R}}^{(j)} + \boldsymbol{u}_j\boldsymbol{v}_j^T) + \textstyle\sum_{j=j+1}^{\eta}\boldsymbol{u}_j\boldsymbol{v}_j^T\\ &\dots\\ &= (\hat{\boldsymbol{Q}}^{(2)}\hat{\boldsymbol{R}}^{(2)} + \boldsymbol{u}_2\boldsymbol{v}_2^T) + \textstyle\sum_{j=3}^{\eta}\boldsymbol{u}_j\boldsymbol{v}_j^T\\ &= (\hat{\boldsymbol{Q}}^{(1)}\hat{\boldsymbol{R}}^{(1)} + \boldsymbol{u}_1\boldsymbol{v}_1^T) + \textstyle\sum_{j=2}^{\eta}\boldsymbol{u}_j\boldsymbol{v}_j^T\\ &= \hat{\boldsymbol{Q}}\hat{\boldsymbol{R}} + \textstyle\sum_{j=1}^{\eta}\boldsymbol{u}^j\boldsymbol{v}_j^T.\end{aligned} \tag{10}$$

Here, we define the recursive function $\mathcal{P}$ such that $\hat{\boldsymbol{Q}}^{(j+1)}\hat{\boldsymbol{R}}^{(j+1)} = \mathcal{P}(\hat{\boldsymbol{Q}}^{(j)}\hat{\boldsymbol{R}}^{(j)})$. Thus, for each iteration of (10),

$$\begin{aligned}\mathcal{P}(\hat{\boldsymbol{Q}}^{(j)}\hat{\boldsymbol{R}}^{(j)}) &= \hat{\boldsymbol{Q}}^{(j)}\hat{\boldsymbol{R}}^{(j)} + \boldsymbol{u}_j\boldsymbol{v}_j^T\\ &= \hat{\boldsymbol{Q}}^{(j)}(\hat{\boldsymbol{R}}^{(j)} + \hat{\boldsymbol{Q}}^{(j)T}\boldsymbol{u}_j\boldsymbol{v}_j^T)\\ &\quad + (\boldsymbol{I} - \hat{\boldsymbol{Q}}^{(j)}\hat{\boldsymbol{Q}}^{(j)T})\boldsymbol{u}_j\boldsymbol{v}_j^T\\ &= \hat{\boldsymbol{Q}}^{(j)}(\hat{\boldsymbol{R}}^{(j)} + \boldsymbol{w}^{(j)}\boldsymbol{v}_j^T) + \boldsymbol{f}^{(j)}\boldsymbol{v}_j^T,\end{aligned} \tag{11}$$

where $j = [1, \dots, \eta]$, and $\boldsymbol{w}^{(j)} = \hat{\boldsymbol{Q}}^{(j)T}\boldsymbol{u}_j$ and $\boldsymbol{f}^{(j)} = (\boldsymbol{I} - \hat{\boldsymbol{Q}}^{(j)}\hat{\boldsymbol{Q}}^{(j)T})\boldsymbol{u}_j$. $\boldsymbol{Q}'\boldsymbol{R}'$ can be calculated by applying the two-stage QR-updating method on (11) as in [9].

B. Updating the Reduced Within-Class Scatter Matrix $\boldsymbol{W}$

For updating the reduced within-class scatter matrix $\boldsymbol{W}$, similar as the updating of $\boldsymbol{M}$, the k-dimension $\boldsymbol{W}$ is augmented to k'-dimension $\hat{\boldsymbol{W}}$. Again to avoid matrix multiplication, we simplify the augmentation of $\boldsymbol{W}$ following the rule of Proposition 2,

Proposition 2: Given $\boldsymbol{W} = \boldsymbol{Q}^T\boldsymbol{H}_w\boldsymbol{H}_w^T\boldsymbol{Q} \in \mathcal{R}^{k\times k}$, if a zero vector $\boldsymbol{0}$ is inserted as the i-th column into $\boldsymbol{Q} \in \mathcal{R}^{d\times k}$, such that $\hat{\boldsymbol{Q}} = (\boldsymbol{Q}_1, \boldsymbol{0}, -\boldsymbol{Q}_2) \in \mathcal{R}^{d\times(k+1)}$ where $\boldsymbol{Q}_1 \in \mathcal{R}^{d\times(i-1)}$ and $\boldsymbol{Q}_2 \in \mathcal{R}^{d\times(k-i)}$, then we have

$$\begin{aligned}\hat{\boldsymbol{W}} &= \hat{\boldsymbol{Q}}^T\boldsymbol{H}_w\boldsymbol{H}_w^T\hat{\boldsymbol{Q}}\\ &= \begin{pmatrix}\boldsymbol{W}_1 & \boldsymbol{0} & -\boldsymbol{W}_2\\ \boldsymbol{0}^T & 0 & \boldsymbol{0}^T\\ -\boldsymbol{W}_3 & \boldsymbol{0} & \boldsymbol{W}_4,\end{pmatrix},\end{aligned}$$

where $\boldsymbol{W}_1 \in \mathcal{R}^{(i-1)\times(i-1)}$, $\boldsymbol{W}_2 \in \mathcal{R}^{(i-1)\times(k-i)}$, $\boldsymbol{W}_3 \in \mathcal{R}^{(k-i)\times(i-1)}$, $\boldsymbol{W}_4 \in \mathcal{R}^{(k-i)\times(k-i)}$, and $\boldsymbol{W} = \begin{pmatrix}\boldsymbol{W}_1 & \boldsymbol{W}_2\\ \boldsymbol{W}_3 & \boldsymbol{W}_4,\end{pmatrix}$. The detailed proof is given in the appendix.

Based on $\hat{\boldsymbol{W}}$, the reduced within-class scatter matrix $\boldsymbol{W'}$ can be determined with the help of Proposition 3,

Proposition 3: Given an augmented reduced within-class scatter matrix $\hat{\boldsymbol{W}}$ and a set of new training samples $\widetilde{\boldsymbol{X}}$, the reduced within-class scatter matrix can be updated as

$$\boldsymbol{W'} = \bar{\boldsymbol{W}} + \widetilde{\boldsymbol{W}} + \boldsymbol{E} \tag{12}$$

where $\bar{\boldsymbol{W}} = \boldsymbol{Q'}^T\boldsymbol{H_w}\boldsymbol{H_w}^T\boldsymbol{Q'}$, $\widetilde{\boldsymbol{W}} = \boldsymbol{Q'}^T\widetilde{\boldsymbol{H}}_w\widetilde{\boldsymbol{H}}_w^T\boldsymbol{Q'}$, $\boldsymbol{E} = \sum_{i=1}^{k'}(\widetilde{n}_i(\boldsymbol{\alpha}' - \boldsymbol{\beta}')(\boldsymbol{\alpha}' - \boldsymbol{\beta}')^T + n_i\boldsymbol{\beta}'\boldsymbol{\beta}'^T))$, $\boldsymbol{\alpha}' = \boldsymbol{Q'}^T(\widetilde{\boldsymbol{m}}_{\boldsymbol{i}} - \boldsymbol{m}_{\boldsymbol{i}}$, and $\boldsymbol{\beta}' = \boldsymbol{Q'}^T(\boldsymbol{m}'_{\boldsymbol{i}} - \boldsymbol{m}_{\boldsymbol{i}})$. The detailed proof is given in the appendix.

Consider $\bar{\boldsymbol{W}} = \boldsymbol{Q'}^T\boldsymbol{H_w}\boldsymbol{H_w}^T\boldsymbol{Q'}$ and $\boldsymbol{Q}^{(j+1)} = \mathcal{P}(\boldsymbol{Q}^{(j)})$, there should exist an iterative function $\mathcal{P}_W$ for $\boldsymbol{W}$

$$\boldsymbol{W}^{(j+1)} = \mathcal{P}_W(\boldsymbol{W}^{(j)}), j = 1, \dots, \eta \tag{13}$$

such that $\bar{\boldsymbol{W}} = \boldsymbol{W}^{(\eta+1)}$ when $j = \eta$

To simplify deduction, we define two sets of the Givens rotations [21] $\boldsymbol{G}^{(j)}$ and $\boldsymbol{J}^{(j)}$, which represent $\boldsymbol{G}_{1,2}^{(j)}, \dots, \boldsymbol{G}_{k'-1,k'}^{(j)}$ and $\boldsymbol{J}_{k'-1,k'}^{(j)}, \dots, \boldsymbol{J}_{1,2}^{(j)}$ respectively. Then following (13), we have

$$\begin{aligned}\boldsymbol{W}^{(j+1)} &= \hat{\boldsymbol{Q}}^{(j+1)T}\boldsymbol{H}_w\boldsymbol{H}_w^T\hat{\boldsymbol{Q}}^{(j+1)}\\ &= [\mathcal{P}(\hat{\boldsymbol{Q}}^{(j)T})]\boldsymbol{H}_w\boldsymbol{H}_w^T[\mathcal{P}(\hat{\boldsymbol{Q}}^{(j)})]\\ &\approx \boldsymbol{G}^{(j)T}\cdot\boldsymbol{J}^{(j)T}\cdot\hat{\boldsymbol{Q}}^{(j)T}\boldsymbol{H}_w\boldsymbol{H}_w^T\hat{\boldsymbol{Q}}^{(j)}\cdot\boldsymbol{J}^{(j)}\cdot\boldsymbol{G}^{(j)}\\ &= \boldsymbol{G}^{(j)T}\cdot\boldsymbol{J}^{(j)T}\cdot\hat{\boldsymbol{W}}^{(j)}\cdot\boldsymbol{J}^{(j)}\cdot\boldsymbol{G}^{(j)},\end{aligned} \tag{14}$$

where "$\cdot$" is used to represent the applications of the Givens rotations to $\hat{\boldsymbol{W}}^{(j)}$. Note that approximation in (14) is made by assuming that $\hat{\boldsymbol{W}}^{(j)}$ is not varied in the second QR-updating stage.

C. Updating the Reduced Between-Class Scatter Matrix $\boldsymbol{B}$

We use the same process for updating $\boldsymbol{B}$ as in [9], which can be written as

$$\boldsymbol{B'} = \left(\ \boldsymbol{R'D} - (\tfrac{1}{n'}\boldsymbol{R'r})\boldsymbol{h}^T\ \right)\left(\ \boldsymbol{R'D} - (\tfrac{1}{n'}\boldsymbol{R'r})\boldsymbol{h}^T\ \right)^T, \tag{15}$$

where $\boldsymbol{D} = \mathrm{diag}(\sqrt{n'_1}, \dots, \sqrt{n'_{k'}})$, $\boldsymbol{r} = (n'_1, \dots, n'_{k'})^T$, and $\boldsymbol{h} = [\sqrt{n'_1}, \dots, \sqrt{n'_{k'}}]^T$

V. Time Complexity Analysis

This section provides the time complexity analysis of the proposed c-QR/IncLDA with a comparison to the s-QR/IncLDA. The term "flam", which denotes an addition and a multiplication, is used for presenting operation counts [22]. Consider a single round of updating the 3-tuple model $\boldsymbol{\Omega} = \{\boldsymbol{M}, \boldsymbol{W}, \boldsymbol{B}, \}$ by both s-QR/IncLDA and c-QR/IncLDA. We measure the time complexity of updating $\boldsymbol{M}$, $\boldsymbol{W}$, $\boldsymbol{B}$ separately for every single new sample presented for IL. Our aim is to find the maximum time cost regardless of whatever class the incoming sample belongs to.

For s-QR/IncLDA, the maximum time cost occurs when the new sample belongs to an existing class, since it involves more complicated calculations on $\boldsymbol{W}$ and $\boldsymbol{QR}$ than the case when new sample comes from a new class. In this case, the updating of $\boldsymbol{M}$ requires dk flam, namely $O(dk)$. According to [23], total operation counts for the two-stage QR-updating is $9dk + \frac{9}{2}k^2$ flam, so its time complexity should be $O(dk + k^2)$. With regard to the scatter matrices, it requires $dk + 2k^2 + k$ flam for updating $\boldsymbol{W}$ and $\frac{3}{2}k^3 + 2k^2$ flam for $\boldsymbol{B}$. Hence, the time complexity for updating $\boldsymbol{W}$ and $\boldsymbol{B}$ is determined as $O(dk + k^2)$ and $O(k^3 + k^2)$, respectively.

In the proposed c-QR/IncLDA, an matrix augmentation operation is employed before all updates. We have $\boldsymbol{M} \in \mathcal{R}^{d\times k}$, $\widetilde{\boldsymbol{M}} \in \mathcal{R}^{d\times \widetilde{k}}$, $\boldsymbol{Q} \in \mathcal{R}^{d\times k}$, $\boldsymbol{R} \in \mathcal{R}^{k\times k}$ and $\boldsymbol{W} \in$

Step	s-QR/IncLDA	c-QR/IncLDA
Matrices Augmentation	N/A	$O(dk' + k'^2)$
Update $\boldsymbol{M}$	$O(dk)$	$O(dk')$
Update $\boldsymbol{Q}$ and $\boldsymbol{R}$	$O(dk + k^2)$	$O(dk'^2 + k'^3 + dk' + k'^2)$
Update $\boldsymbol{W}$	$O(dk + k^2)$	$O(k'^3 + dk' + k'^2)$
Update $\boldsymbol{B}$	$O(k^3 + k^2)$	$O(k'^3 + k'^2)$
General	$O(k^3 + dk)$	$O(dk'^2 + k'^3 + dk')$

TABLE I: Comparison of Time Complexity for incremental learning on a single sample, s-QR/IncLDA vs c-QR/IncLDA

$\mathcal{R}^{k\times k}$ augmented from k or $\widetilde{k}$ to k', so the total time complexity of augmentation is $O(dk' + k'^2)$. For updating $\hat{\boldsymbol{M}}$, it requires dk' flam and thus the time complexity is $O(dk')$. For updating the $\hat{\boldsymbol{Q}}\hat{\boldsymbol{R}}$, the SVD on $\boldsymbol{\Delta}$ requires $4dk'^2 + 8k'^3$ flam according to [22], and the subsequent two-stage QR-updating on $\hat{\boldsymbol{Q}}\hat{\boldsymbol{R}}$ is a η-iteration process, which requires $9\eta dk' + \frac{9}{2}\eta k'^2$ flam. Adding up the above two steps, the total time complexity for updating $\hat{\boldsymbol{Q}}\hat{\boldsymbol{R}}$ is therefore $O(dk'^2 + k'^3 + dk' + k'^2)$. As discussed in Section IV, the update of the reduced within-class matrix $\hat{\boldsymbol{W}}$ follows Proposition 3, in which the operation counts for computing $\bar{\boldsymbol{W}}$ by (13) and (14) are $6\eta k'^2 + k'^3 + dk' + 6\eta k' + k'^2$ flam, and the operation counts for computing $\widetilde{\boldsymbol{W}}$ and $\boldsymbol{E}$ are $d'k' + k'^2$ and k'^3 flam. Hereby for updating $\hat{\boldsymbol{W}}$, the total operation counts is $2k'^3 + 2dk' + 6\eta k'^2 + 2k'^2 + 6\eta k'$ flam, namely $O(k'^3 + dk' + k'^2)$. For updating the reduced between-class scatter matrix $\boldsymbol{B}$, the time complexity is similarly to s-QR/IncLDA as $O(k'^3 + k'^2)$ based on (15). In general, for incremental learning with respect to a single sample, the total complexity of c-QR/IncLDA is $O(dk'^2 + k'^3 + dk')$, which is greater than the complexity of s-QR/IncLDA, i.e., $O(k^3 + dk)$, since $d \gg k$ and $k' \geq k$.

The time complexity discussed above is summarized in Table I. As seen in Table I, for incremental learning with respect to a single sample, the total complexity of c-QR/IncLDA is $O(dk'^2 + k'^3 + dk')$, which is greater than the complexity of s-QR/IncLDA, i.e., $O(k^3 + dk)$, since $d \gg k$ and $k' \geq k$.

However for incremental learning on a chunk of s samples, s-QR/IncLDA can only process the data iteratively in s cycles. Thus, it gives $O(s(k^3 + dk))$. In contrast, the proposed c-QR/IncLDA processes the whole chunk of s samples at one time. According to Proposition 3, the actual time cost is determined by η, the rank of $\boldsymbol{\Delta}$. Hence, the time complexity for c-QR/IncLDA is $O(\eta(dk'^2 + k'^3 + dk'))$.

Matching the running efficiency of the c-QR/IncLDA with s-QR/IncLDA, $s(k^3 + dk) = \eta(dk'^2 + k'^3 + dk')$, since $k \leq k'$, we have the minimal chunk size s as,

$$s = \frac{\eta(dk'^2 + k'^3 + dk')}{k^3 + dk} \geq \frac{\eta k'^2 d}{k^3 + dk} + \eta. \qquad (16)$$

Note that s is greater than η. Proof: consider $k \ll d$, without loss of generality, we enlarge the denominator of (16) by replacing the k^3 item with dk^2, thus we have

$$s \geq \frac{\eta k'^2 d}{dk^2 + dk} + \eta = \frac{\eta k'^2}{k(k+1)} + \eta > \eta. \qquad (17)$$

Thus for incremental learning of one chunk of s samples, c-QR/IncLDA can compress information of the data into η equivalent samples. This reduces $s - \eta$ cycles learning as compared to the s-QR/IncLDA. As such learning reduction $s - \eta$ increases, the computational efficiency of c-QR/IncLDA (in terms of the total time cost to complete the learning of a whole dataset) is expected to match and surpass s-QR/IncLDA, despite the higher complexity of c-QR/IncLDA for incremental learning of single sample.

VI. Results and Discussion

We examined the accuracy and efficiency of the proposed c-QR/IncLDA with a comparison to s-QR/IncLDA and batch QR/LDA, based on equality to the ground truth eigenspace, class separability of the embedding eigenspace, and execution time. All the experiments were implemented on a Intel 2.26GHZ Core I5 PC with 4GB Ram.

A. Data Description

We selected five benchmarked facial recognition datasets and one combined face dataset for the following performance and efficiency evaluation experiments.

1) The AT&T (ORL) face dataset * consists of 400 face images from 40 persons (10 images per person). Each image has the size of 92×112 pixels. The images were taken at the same background but at different time, with faces varied at either facial expressions (e.g., open/closed eyes and smiling/non-smiling) or facial details (e.g., glasses/no-glasses).
2) The Altkom dataset † [24], [25] is composed of 1200 face images from 80 persons (15 images per person) with different poses. The size of image is normalized to 46×56.
3) The MPEG-7 dataset has 1355 face images from 271 persons (5 images per person) [26], [27], [13]. Each image has the size of 23×28 pixels. The dataset is collected from AR(Purdue), AT&T, Yale, UMIST, University of Berne, and some of them are obtained from MPEG-7 news Videos.
4) The XM2VTS dataset ‡ [24], [25] contains 2950 face images from 5 recordings of 295 persons. Each recording has two head shots taken at two different time (i.e., 10 images per person). Each image has the same size of 46×56 pixels.
5) The extended version 1 MPEG dataset § [24], [25] is a combination of several public face sets (e.g., AR and ORL). It collects 3175 images from 635 persons (5 images per person) with the exhibition of illumination and view variations. All images have the same size of 46×56.
6) The Combined face dataset includes total 5050 images from 1010 persons (5 images per person). Among all

*http://www.cl.cam.ac.uk/research/dtg/attarchive/facedatabase.html.

†http://www.iis.ee.ic.ac.uk/∼tkkim/code.htm.

‡http://www.iis.ee.ic.ac.uk/∼tkkim/code.htm.

§http://www.iis.ee.ic.ac.uk/∼tkkim/code.htm.

the images, 400 images are selected from the Altkom dataset (80 persons), 1475 images from XM2VTS dataset (295 persons), and 3175 images from the version 1 MPEG dataset (635 persons). The size of image is kept the same as 46×56.

B. Experiment Setup

For incremental learning over each dataset, we first construct an initial discriminant eigenspace using 10% of the face images, in which at least two classes of data are guaranteed to be included according to the definition of classic LDA [5]. The remaining 90% training instances are divided equally and arbitrarily into nine chunks/groups and presented to the learner sequentially. As a reference, we also apply batch QR/LDA to perform the same incremental learning task. As the incremental learning proceeds, the learner is provided with chunks of new data in a sequential manner until all data from the dataset are consumed.

C. Similarity to Ground Truth Eigenspace

To examine the similarity between the discriminant eigenspace from the c-QR/IncLDA and that from the ground truth QR/LDA, we set deliberatively the chunk size s to 1, and use the c-QR/IncLDA for sequential incremental learning. We examine c-QR/IncLDA, s-QR/IncLDA and the batch LDA on the UCI Iris dataset [28] which has three classes and 150 samples in a 4-dimension space. We calculate the inner products between the discriminant eigenvectors from the c-QR/IncLDA and the counterpart of the batch QR/LDA at every learning stage, and record the converging procedure of the c-QR/IncLDA and s-QR/IncLDA, respectively.

Fig 1 reveals the difference between two IncLDAs on their converging procedure to batch QR/LDA. As seen, both IncLDAs converge to the Batch QR/LDA as the incremental learning proceeds. This shows that the c-QR/IncLDA is equivalent to the ground truth on Iris dataset. The proposed c-QR/IncLDA has not only fewer but also far smaller fluctuations than the s-QR/IncLDA on all three axes. Specifically on the $d1$ axis, the c-QR/IncLDA is found converging to the ground truth over 100 stages earlier than the s-QR/IncLDA; and on the other two axises performing similarly to the s-QR/IncLDA. This suggests that with reference to the ground truth, the proposed c-QR/IncLDA has deterministically less discriminative information lost for incremental QR/LDA learning, thus is expected to have better stage learning performance (in terms of stage class separability evaluation) than the s-QR/IncLDA, even if it is just being used in sequential learning mode.

D. Class Separability

To evaluate the class separability of the proposed c-QR/IncLDA, we project the data presented so far to the current LDA eigenspace and then classify the data using a k-Nearest Neighbor (kNN) classifier. The dimensions of the embedding spaces of both s-QR/IncLDA and c-QR/IncLDA are set to the number of classes, and the classification accuracy is measured by leave-one-out (LOO) cross-validation.

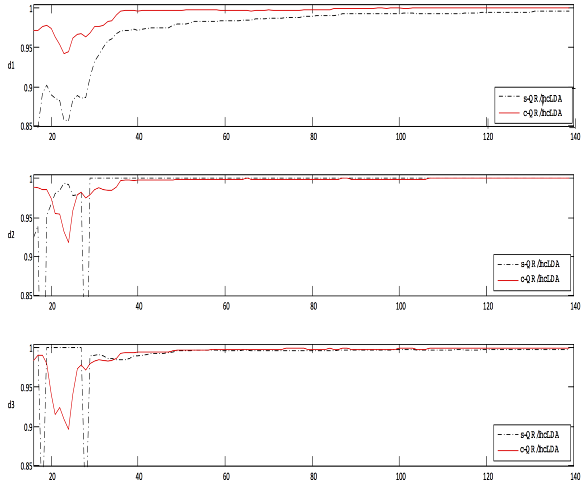


Fig. 1: inner products of eigenvectors, c-QR/IncLDA vs. s-QR/IncLDA on their converging procedure to batch QR/LDA

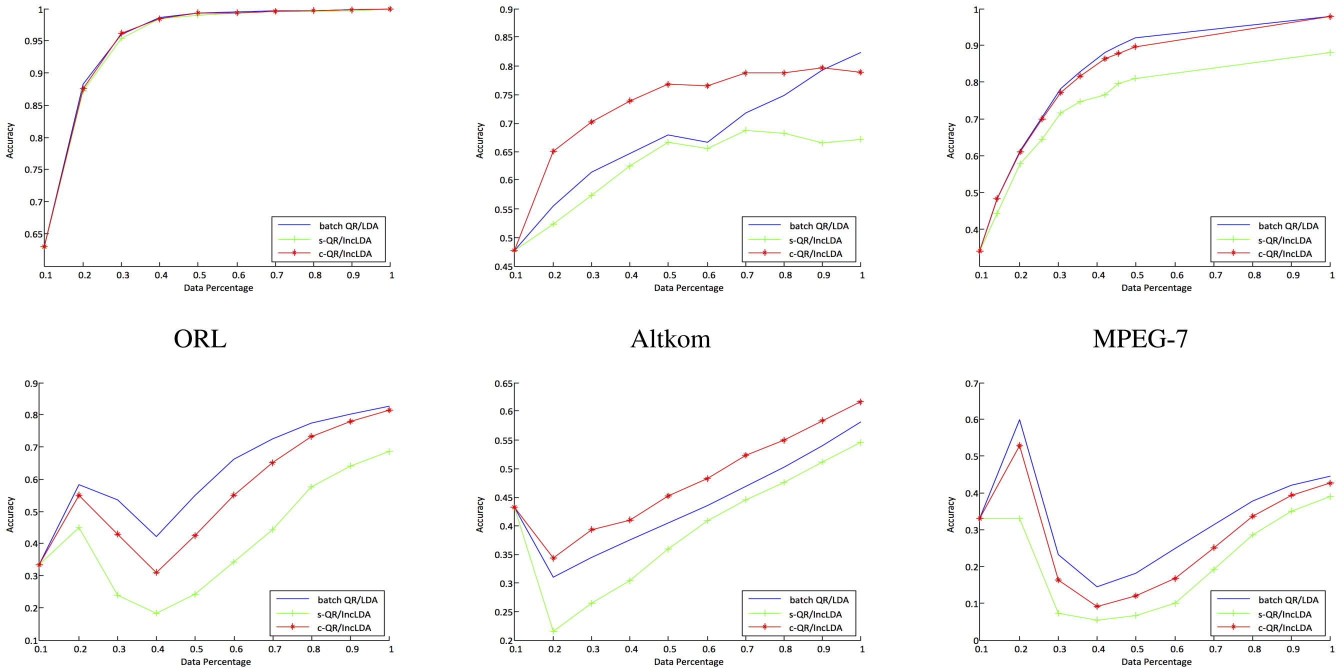
Fig.2 shows the experimental results, where the proposed c-QR/IncLDA is compared to s-QR/IncLDA and batch QR/LDA, at every incremental learning stage, which is identified as the percentage of data presented so far. As seen in the figure, although the c-QR/IncLDA resembles closely the trend of s-QR/IncLDA for all six datasets, the performance of the c-QR/IncLDA apparently approaches the ground truth batch QR/LDA and is better than the s-QR/IncLDA. Such performance difference arises at the beginning stage and amplifies quickly in a few cycles of learning. For dataset Altkom and XM2V., c-QR/IncLDA seems to even outperform the batch QR/LDA.

E. Computational Efficiency

In this experiment, *c-QR/IncLDA* is compared with s-QR/IncLDA on CPU time cost. To observe the relationship between the execution time and *chunk size* (size of the added subsets), we use different chunk sizes ranging from 5 to 1220 with a step of 5 in the experiment, and record the differences between c-QR/IncLDA and s-QR/IncLDA at every incremental learning stage.

Fig 3 (a) compares the time cost between c-QR/IncLDA and s-QR/IncLDA at each stage when chunk size is set to 100, 250, and 350, respectively. As seen, when the chunk size is 100, the time cost of stage learning of s-QR/IncLDA is smaller than that of c-QR/IncLDA, and the difference is as large as 20 seconds on average. However at 250, the efficiency of c-QR/IncLDA is competitive to s-QR/IncLDA, and its efficiency surpasses the s-QR/IncLDA, when the chunk size is greater 100.

We further measure the total time cost of c-QR/IncLDA to compute a discriminant model for all 1,355 samples of MPEG-7 dataset under different chunk sizes. To better clarify the comparison between c-QR/IncLDA and s-QR/IncLDA, we show the s-QR/IncLDA time minus the c-QR/IncLDA



ORL

Altkom

MPEG-7

MPEG

XM2V

Combined

Fig. 2: The comparison of class separability between the proposed c-QR/IncLDA and s-QR/IncLDA, with reference to the batch QR/LDA

time in Fig. 3 (b). As seen, the efficiency of c-QR/IncLDA improves quickly with the increase of the chunk size. When chunk size reaches 240, c-QR/IncLDA matches s-QR/IncLDA in efficiency. The superiority of c-QR/IncLDA in efficiency occurs and stands when the incremental sizes are greater than 290. The difference scales up to over 100 seconds, when the chunk size is increased to 1220.

VII. Conclusions and Future Work

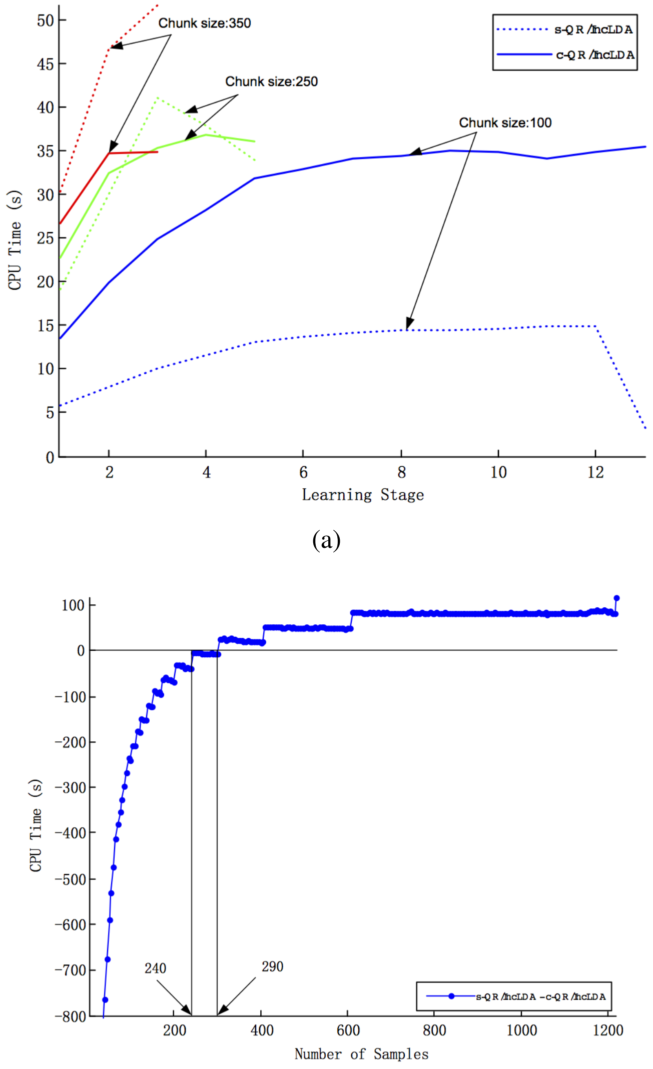
Incremental learning increments a system with the knowledge acquired from newly presented data. In literature, such knowledge increment is achieved mostly by incremental learning from raw data, such as GSVD/IncLDA [18], LS/IncLDA [8], s-QR/IncLDA [9], and Fisher/IncLDA [2] etc., or more efficiently by knowledge merging (i.e., append knowledge extracted from raw data to current system), such as PCA merging [11] and LDA merging [29]. Alternatively, this paper proposes a new solution to incrementing a QR/LDA system by learning efficiently over the compressed data of any newly presented raw data.

The proposed c-QR/IncLDA processes a whole chunk of data for every incremental learning cycle. The core techniques of c-QR/IncLDA are summarized as below

1) Two efficient matrix augmentation methods have been proposed. They allow us to easily accommodate information of new classes contained in chunk data through expanding the dimensions of matrices $\boldsymbol{Q}$ and $\boldsymbol{W}$.
2) A new method for incremental updating of $\boldsymbol{W}$, which is more accurate than the existing approach, has been successfully developed in this paper. This is achieved by relaxing a key assumption on $\boldsymbol{Q}$ and $\boldsymbol{W}$ that is vital to existing algorithms but may be significantly violated as the number of classes increases.

Unlike Pang's IncLDAs [2] where only the chunk data of one class can be acquired each time, the proposed c-QR/IncLDA is capable of absorbing newly presented data samples of arbitrary classes. Experimental results show that the performance (i.e., accuracy) is improved in our algorithm, especially when processing data with large number of classes. In addition, comparing with existing s-QR/IncLDA, the proposed algorithm also produces transformation matrices that are much similar to those obtained by Batch QR/LDA.

As we noted in the paper, the reduced between-class scatter matrix $\boldsymbol{B}$ in the existing s-QR/IncLDA and the proposed c-QR/IncLDA are not incrementally updated but completely recomputed. As a matter of fact, whenever new data are presented for the incremental learning procedure, there should exist an incremental learning rule such as $\boldsymbol{B'} = \boldsymbol{B} + \boldsymbol{\Sigma}$. One important future challenge would be to develop such an incremental mechanism for calculating $\boldsymbol{\Sigma}$ and subsequently updating $\boldsymbol{B}$. Once realized, we believe the efficiency of c-QR/IncLDA could be improved further. As analyzed in section V, the time complexity for recomputing $\boldsymbol{B}$ is $O(k^3)$. It is straightforward to see that if the reduced between-class scatter matrix can be updated through $\boldsymbol{B'} = \boldsymbol{B} + \boldsymbol{\Sigma}$, which only requires $O(k^2)$, significant reduction of computation cost can be achieved. Additionally, our time complexity analysis showed that when $s > \eta$, the computational efficiency of c-QR/IncLDA can match and even exceed that of s-QR/IncLDA. This has been mathematically proved. However, it remains as another future challenge to find a general formula for determining the exact match point and surpass point.



(a)

(b)

Fig. 3: Comparison on time cost with different incremental sizes, c-QR/IncLDA vs. s-QR/IncLDA.

References

[1] C. Giraud-Carrier, “A note on the utility of incremental learning,” *AI Communications*, vol. 13, no. 4, pp. 215–223, 2000.

[2] S. Pang, S. Ozawa, and N. Kasabov, “Incremental linear discriminant analysis for classification of data streams,” *IEEE Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics*, vol. 35, no. 5, pp. 905–914, 2005.

[3] G. Cauwenberghs and T. Poggio, “Incremental and decremental support vector machine learning,” in *NIPS*, pp. 409–415.

[4] R. A. Fisher, “The use of multiple measurements in taxonomic problems,” *Annals of Human Genetics*, vol. 7, no. 2, pp. 179–188, 1936.

[5] K. Fukunaga, *Introduction to statistical pattern recognition*. Academic Press, 1990.

[6] G. McLachlan, *Discriminant analysis and statistical pattern recognition*. Wiley-Interscience, 2004.

[7] J. Ye, “Least squares linear discriminant analysis,” in *Proceedings of the 24th International Conference on Machine Learning*. ACM, pp. 1087–1093.

[8] L. Liu, Y. Jiang, and Z. Zhou, “Least square incremental linear discriminant analysis,” in *Ninth IEEE International Conference on Data Mining*, pp. 298–306.

[9] J. Ye, Q. Li, H. Xiong, H. Park, R. Janardan, and V. Kumar, “Idr/qr: An incremental dimension reduction algorithm via qr decomposition,” *IEEE Transactions on Knowledge and Data Engineering*, vol. 17, no. 9, pp. 1208–1222, 2005.

[10] R. Polikar, L. Upda, S. S. Upda, and V. Honavar, “Learn++: an incremental learning algorithm for supervised neural networks,” *Systems, Man, and Cybernetics, Part C: Applications and Reviews, IEEE Transactions on*, vol. 31, no. 4, pp. 497–508, 2001.

[11] P. Hall, D. Marshall, and R. Martin, “Merging and splitting eigenspace models,” *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 22, no. 9, pp. 1042–1049, 2000.

[12] S. Ozawa, S. Pang, and N. Kasabov, “An incremental principal component analysis for chunk data,” in *2006 IEEE International Conference on Fuzzy Systems*, pp. 2278–2285.

[13] S. Pang, S. Ozawa, and N. Kasabov, *One-Pass Incremental Membership Authentication by Face Classification*, ser. Lecture Notes in Computer Science. Springer Berlin / Heidelberg, 2004, vol. 3072, pp. 1–44.

[14] ——, *Chunk Incremental LDA Computing on Data Streams*, ser. Lecture Notes in Computer Science. Springer Berlin / Heidelberg, 2005, vol. 3497, pp. 829–831.

[15] J.-G. Wang, E. Sung, and W.-Y. Yau, “Incremental two-dimensional linear discriminant analysis with applications to face recognition,” *Journal of Network and Computer Applications*, vol. 33, no. 3, pp. 314–322, 2010.

[16] M. Karasuyama and I. Takeuchi, “Multiple incremental decremental learning of support vector machines,” *Trans. Neur. Netw.*, vol. 21, no. 7, pp. 1048–1059, 2010.

[17] F. Song, H. Liu, D. Zhang, and J. Yang, “A highly scalable incremental facial feature extraction method,” *Neurocomputing*, vol. 71, no. 10-12, pp. 1883–1888, 2008.

[18] H. Zhao and P. C. Yuen, “Incremental linear discriminant analysis for face recognition,” *IEEE Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics*, vol. 38, no. 1, pp. 210–221, 2008.

[19] H. G. Loos, “Parity madeline: a neural net with complete boolean repertoire capable of one-pass learning,” in *Neural Networks, 1989. IJCNN., International Joint Conference on*, pp. 111–118.

[20] N. Kasabov, *Evolving Connectionist Systems: The Knowledge Engineering Approach (new edition)*. London: Springer Verlag, 2007, 10.1007/978-1-84628-347-5_2.

[21] G. H. Golub and C. F. Van Loan, *Matrix computations*, 3rd ed. Baltimore, M.D.: Johns Hopkins University Press, 1996.

[22] S. G.W., *Baisc Decompositions*. SIAM, 1998, vol. I.

[23] J. W. Daniel, W. B. Gragg, L. Kaufman, and G. W. Stewart, “Reorthogonalization and stable algorithms for updating the gram-schmidt qr factorization,” *Mathematics of Computation*, vol. 30, no. 136, pp. 772–795, 1976.

[24] T.-K. Kim, H. Kim, W. Hwang, and J. Kittler, “Component-based lda face description for image retrieval and mpeg-7 standardisation,” *Image Vision Comput.*, vol. 23, no. 7, pp. 631–642, 2005.

[25] T.-K. Kim, B. Stenger, J. Kittler, and R. Cipolla, “Incremental linear discriminant analysis using sufficient spanning sets and its applications,” *International Journal of Computer Vision*, vol. 91, no. 2, pp. 216–232, 2011.

[26] S. Pang, D. Kim, and S. Y. Bang, “Membership authentication in the dynamic group by face classification using svm ensemble,” *Pattern Recognition Letters*, vol. 24, no. 1-3, pp. 215–225, 2003.

[27] M.-S. Kim, D. Kim, and S.-Y. Lee, “Face recognition using the embedded hmm with second-order block-specific observations,” *Pattern Recognition*, vol. 36, no. 11, pp. 2723–2735, 2003.

[28] A. Frank and A. Asuncion, “UCI machine learning repository,” 2010. [Online]. Available: http://archive.ics.uci.edu/ml

[29] S. Pang, T. Ban, Y. Kadobayashi, and N. Kasabov, “Incremental and decremental lda learning with applications,” pp. 1–8, 18-23 July 2010 2010.