Journal of Mathematical Imaging and Vision manuscript No.
(will be inserted by the editor)

Feature extraction by using dual generalized discriminative common vectors

Katerine Diaz-Chito · Jesús Martínez del Rincón · Marçal Rusiñol · Aura Hernández-Sabaté

Received: date / Accepted: date

Abstract In this paper, a dual online subspace-based learning method called Dual Generalized Discriminative Common Vectors (Dual-GDCV) is presented. The method extends Incremental GDCV by exploiting simultaneously both the concepts of incremental and decremental learning for supervised feature extraction and classification. Our methodology is able to update the feature representation space without recalculating the full projection or accessing the previously processed training data. It allows both adding information and removing unnecessary data from a knowledge base in an efficient way, while retaining the previously acquired knowledge. The proposed method has been theoretically proved and empirically validated in six standard face recognition and classification datasets, under two scenarios: 1) removing and adding samples of existent classes, and 2) removing and adding new classes to a classification problem. Results show a considerable computational gain without compromising the accuracy of the model in comparison with both batch methodologies and other state-of-art adaptive methods.

Keywords Online feature extraction · Generalized Discriminative Common Vectors · Dual learning · Incremental learning · Decremental learning.

Katerine Diaz-Chito, Marçal Rusiñol, Aura Hernández-Sabaté
Centre de Visió per Computador, Universitat Autònoma de Barcelona, Spain
E-mail: {kdiaz, marcal, aura}@cvc.uab.es

Jesús Martínez del Rincón
Centre for Secure Information Technologies, Queen's University Belfast, UK
E-mail: j.martinez-del-rincon@qub.ac.uk

1 Introduction

Subspace-based learning is a well-known branch of pattern recognition with multiple applications, such as automatic feature extraction, feature selection or dimensionality reduction. Algorithms such as Principal Component Analysis (PCA) or Linear Discriminant Analysis (LDA) are widely used in a multitude of learning systems and frameworks. Among their main advantages, they reduce the computation time and the memory footprint, they remove irrelevant and redundant information in order to improve the performance of a subsequent machine learning model, they are able to produce discriminative models and spaces with very limited data, and they facilitate the visualization of the data by reducing the data space to very low dimensions such as 2D or 3D. Due to their success, traditional subspace-based algorithms have evolved into multiple directions to cover a wider spectrum of applications, including supervised and unsupervised problems, and linear and non-linear spaces.

Adaptive methods are particularly interesting. While conventional algorithms require to work in batch mode, where all training samples have to be considered at once, adaptive versions allow evolving an initial model when more data become available. Furthermore, adaptive learning does not require full access to the initial training data, which may be lost or under restricted access. Not having to retrain from scratch leads to convenient trade-offs between computational and performance. It is common to find applications where a complete set of training samples is usually not known in advance but provided little by little, and therefore adaptive learning is best suited for the task. Examples are found in object tracking,[28] image classification,[5] stream processing,[31] and face recognition.[25]

The most common type of adaptive learning is incremental learning, where new samples or data are added to the knowledge base of the model. Thus, multiple incremental subspace-based learning techniques have been proposed, such as those based on PCA [6,10,16,22,27,31,33] , LDA[13,24,30] [2,17,18,25,32,34] or Discriminative Common Vector (DCV)[4,7] [5,8,19] methods.

However, these methods only address updating the system by adding new information, but they do not consider the double nature of adaptive learning, where previous information may need to be removed, if outliers or wrongly-labeled samples are detected a posteriori, or replaced, if the distribution of the data changes or drifts. A similar limitation occurs for the ubiquitous learning approaches. While their performance for feature extraction and classification is unquestionable regarding other traditional approaches, they do not easily allow for information removal without a hugely costly retraining. The deep learning paradigm can cover the incremental paradigm by using transfer learning and fine-tuning process if pretrained networks, but it does not allow for adaptive learning where both incremental and decremental is considered, as well as that these are not executed in a reasonable computational complexity since this is one of the major bottlenecks in deep learning.[1] It is in this aspect where dual learning has scope to be better integrated into neural networks and/or other methods to complement deep learning approaches.

Our method combines the advantages of update a model from an incremental and decremental learning point of view, with moderate computational complexity. In this sense, we define dual learning as an online process that allows adding and removing samples, classes or any initial information from a previously trained model. We postulate that removing or replacing information embedded in the model can be as important as adding information for automatic feature extraction. Several application fields will clearly benefit from the ability to decrement a learned model. For instance, biometric systems used to manage and identify a large population of users in big organizations may require updating the model when a user leaves the organization as well as replacing all obsolete samples after a major change in the user. This may be a laborious and long process, even impossible depending on the scale of the user database and antiquity of the model, which may result on delays and limited access to the other users, as well as privacy issues. Similarly, being able to remove a single instance from a complex model encompassing thousands of samples and classes, when this outlier has been introduced by mistake, is a desirable feature that will reduce the computational cost and the requirement of multiplicity of backed-up models. The use of dual subspace-based methods as part of hierarchical classification architecture, such as decision trees or cascade of classifiers, will also benefit from a decremental methods where more specific or compact models can be derived from a global.

Less research has been conducted on methodologies that allow both incremental and decremental learning. Two approaches have been proposed on dual versions of PCA.[11,12] The dual approach of[12] is based on an eigendecomposition (EVD) updating and downdating algorithm, referred to as EVD Dualdating (EVDD). The Merging and Splitting EigenSpaces (MSES) method, presented in,[11] where the subspace is extended by combining it with a new learned subspace, or reduced by dividing it. Both algorithms permit simultaneous arbitrary addition and deletion operations, by transforming the EVD of the covariance matrix into a Singular Value Decomposition (SVD) updating problem. However, since these approaches extend the PCA method, they are both unsupervised techniques and they may be ill-suited when applied to supervised classification problems.

To the best of our knowledge, only[23] have suggested a supervised dual learning framework, the LDA merging and LDA splitting (LDA-MS) which updates the scatter matrices in both ways. However, by relying on LDA, the methodology is susceptible to the Small Sample Size (SSS) problem and it cannot be applied when the dimension of the sample space is larger than the number of samples in the training set, since the within-class scatter matrix will be singular. Furthermore, incremental and decremental operations must be performed sequentially rather than in a single step, leading to higher computational costs.

In this paper, we propose a dual incremental and decremental subspace-based learning method called Dual Generalized Discriminative Common Vectors method (Dual-GDCV) suited for supervised feature extraction and classification. The method makes use of both concepts of incremental and decremental learning in order to allow simultaneously adding and/or removing samples, classes or any new information from a previously trained model or space. Our methodology is able to update the model without recalculating the full projection or accessing the previously processed training data, while preserving the previously acquired knowledge. In this paper, we aim to generate a model as accurate as the one calculated by batch processing, providing a tool for dual learning. The potential downfall of model updating that leads to model degeneration by providing bad or non-longer relevant samples is left to the user's discretion or as future work.

This paper builds on our previous work in,[5] where the Incremental-only GDCV method (IGDCV) was presented. The newly proposed method extends IGDCV by allowing simultaneously both adding and/or removing samples and classes. Its mathematical derivation and the computational complexity are described in detail in the current document. In addition, a more thorough evaluation is performed. Thus, 10-fold cross-validation is constantly applied in all experiments to mitigate the effect of random partitions, as opposed to.[5] Furthermore, 4 more datasets are used in the validation, including one with more than 200 classes to better show the effect on adding/removing classes. Moreover, a broader and more up-to-date comparison with the state of the art is provided under the very same data and experimental setup, which includes more than 11 methods comprising incremental-only, decremental-only and dual techniques.

The remainder of the paper is structured as follows. Section 2 briefly introduces the IGDCV method as background information. Section 3 presents the novel Dual-GDCV, the main contribution of this paper. Section 4 describes the empirical validation and presents the results and the analysis of the proposed approach as well as its comparison against the state of the art. Finally, Section 5 summarizes the main conclusions and results.

2 Background

Let the training set X be composed of samples belonging to c classes, where every class j has m_j samples. The total number of samples in the training set is $m = \sum_{j=1}^{c} m_j$. Let x_j^i be a d-dimensional column vector which denotes the ith sample from the jth class.

In order to obtain the optimal projection W of the samples X to the new subspace, the bases of such subspace U should be first calculated. These bases are obtained by solving the eigenproblem of the within-scatter matrix,

$$S_w^X = \sum_{j=1}^{c} \sum_{i=1}^{m_j} (x_j^i - \bar{x}_j)(x_j^i - \bar{x}_j)^T = X_c {X_c}^T,$$

where $\bar{x}_j$ is the average of the samples in the jth class, and the centered data matrix, X_c consists of column vectors $(x_j^i - \bar{x}_j)$ for all $j = 1 \dots c$ and $i = 1 \dots m_j$.

The eigendecomposition or eigen-value/vector decomposition (EVD) of S_w^X can be written in general as

$$EVD(S_w^X) : X_c {X_c}^T = U \Lambda U^T = [U_r \;\; U_o] \begin{bmatrix} \Lambda_r & \\ & 0 \end{bmatrix} \begin{bmatrix} {U_r}^T \\ {U_o}^T \end{bmatrix},$$

where $U = [u_1 \dots u_d]$ is a column matrix formed by the eigenvectors associated to the eigenvalues, $\lambda_1 \geq \dots \geq \lambda_d$, contained in the diagonal matrix Λ. r is the range of matrix S_w^X, where $\lambda_i = 0$ for all $i > r$.

2.1 Generalized Discriminative Common Vectors

GDCV method [5] constitutes a way to overcome the SSS singularity problem in LDA. The singularity is avoided by extending the null space of S_w^X to include not only null directions or basis vectors, i.e. $\lambda_i = 0$, but also with a set of almost null directions, $\lambda_i \approx 0$. This extension of the null space also implies the corresponding restriction of the range space. The projection basis U_α of the new restricted range space will be the basis of the learned subspace.

The scattering added to the null space is measured by the trace $tr(\cdot)$ as $tr(U_\alpha^T S_w^X U_\alpha)$. This quantity is at most $tr(S_w^X)$ when no directions are removed, $U_\alpha = U_r$, and decreases as more and more important directions disappear from U_r. Consequently, the scattering preserved after a projection, U_α, is written as follows

$$\alpha = 1 - \frac{tr(U_\alpha^T S_w^X U_\alpha)}{tr(S_w^X)} \qquad (1)$$

The parameter α takes values within the interval $[0, 1]$. When $\alpha = 0$, then $U_\alpha = U_r$. For individual values of $0 < \alpha < 1$, different projections are obtained with dissimilar levels of preserved variability. Figure 1 presents the main subspaces involved in the GDCV method.

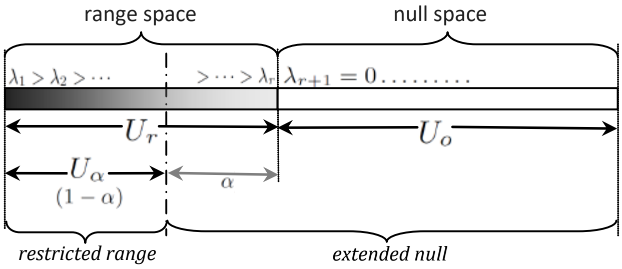


Fig. 1: Main subspaces involved in the GDCV method. U_r and U_o span the range and null space of S_w^X linked to the eigenvalues $\lambda_1 > \dots > \lambda_r$ and $\lambda_i = 0, \quad i \geq (r+1)$, respectively. U_α spans the restricted range of S_w^X according to α.

The generalized common vector are defined as: $x_{gcv}^j = \bar{x}_j - U_\alpha U_\alpha^T \bar{x}_j$, and the projection matrix is given by $W = orth(X_c^{com}) \in \mathbb{R}^{d \times (c-1)}$, where $X^{com} = [x_{gcv}^1 \dots x_{gcv}^c]$ and X_c^{com} be its centered version with regard to the mean $\bar{x}_{com} = (1/c) \sum_{j=1}^{c} x_{gcv}^j$ Finally, the generalized discriminative common vectors are defined by $W^T \bar{x}_j$.

2.2 Incremental Generalized Discriminative Common Vectors

The Incremental Generalized Discriminant Common Vector (IGDCV) method[5] allows the update of a given model by considering the addition of new examples even from unseen classes. IGDCV is based on the update of the projection matrices corresponding to the extended null space of the within-class scatter matrix, by relaxing the null space condition, such that the new projection matrix $\widetilde{W}$ and $\widetilde{U_\alpha}$ are obtained without fully recalculating the eigenproblem.

Let $I \in \mathbb{R}^{d \times m_I}$ the new training samples to be added, and $\widetilde{X} = [X \;\; I] \in \mathbb{R}^{d \times (m+m_I)}$ the resulting training set. $m_I =$

$\sum_{j=1}^{c_I} m_{I_j}$, where m_{I_j} is the number of samples in the j-class. $\overline{u}_{X_j}$ and $\overline{u}_{I_j}$ are the average of each class into the X and I, respectively. By using the previous notation, the updated average $\overline{u}_{\widetilde{X}_j}$ of each class is defined by

$$\overline{u}_{\widetilde{X}_j} = \frac{m_{X_j}\overline{u}_{X_j} + m_{I_j}\overline{u}_{I_j}}{(m_{X_j} + m_{I_j})}.$$

The within-class scatter matrix of the new training set $S_w^{\widetilde{X}}$ is calculated as:

$$S_w^{\widetilde{X}} = S_w^X + S_w^I + A_X A_X^T + A_I A_I^T$$

where $S_w^I = I_c I_c^T$, and I_c is the centered data matrix consists of column vectors $(x_j^i - \overline{u}_{I_j})$ for $i = 1 \ldots m_{I_j}$, for all $j = 1 \ldots c$.

A_X and A_I are matrices containing the differences on the average of each class between the updated training set and the initial one, respectively. Specifically, these matrices are given by $A_X = [a_{X_1} \ldots a_{X_c}]$, $a_{X_j} = \sqrt{m_{X_j}}(\overline{u}_{X_j} - \overline{u}_{\widetilde{X}_j})$ and $A_I = [a_{I_1} \ldots a_{I_c}]$, $a_{I_j} = \sqrt{m_{I_j}}(\overline{u}_{I_j} - \overline{u}_{\widetilde{X}_j})$.

The eigendecomposition of matrix $S_w^{\widetilde{X}}$ is approximated as:

$$\begin{aligned} EVD(S_w^{\widetilde{X}}) : S_w^{\widetilde{X}} &= \widetilde{U_r}\,\widetilde{\Lambda_r}\widetilde{U_r}^T \\ &\approx U_\alpha \Lambda_\alpha U_\alpha^T + I_c I_c^T + A_X A_X^T + A_I A_I^T . \end{aligned}$$

3 Dual formulation of extended null space methods

3.1 Problem Setting

Our aim is to investigate the role of dual learning on an initially calculated projection W and the corresponding subspace basis U_α so that the new projection $\widetilde{W}$ and subspace $\widetilde{U_\alpha}$ are obtained without fully recalculating the eigenproblem. In order to keep a consistent notation throughout the document, for any variable X, its updated version after adding and removing samples is denoted by $\widetilde{X}$. For example, the data matrix A is changed to $\widetilde{A}$ after updating.

Given an initial model $X = [C \;\; D] \in \mathbb{R}^{d \times m}$, and new training samples to be added, $m_I = \sum_{j=1}^{c_I} m_{I_j}$, $I \in \mathbb{R}^{d \times m_I}$, as well as obsolete training ones to be removed, $m_D = \sum_{j=1}^{c_D} m_{D_j}$, $D \in \mathbb{R}^{d \times m_D}$, the resulting training set should be effectively composed of $\widetilde{X} = [C \;\; I] \in \mathbb{R}^{d \times (m - m_D + m_I)}$, these sets are represented in Figure 2. m_{I_j} and m_{D_j} are the number of samples in the j-class, and $\overline{u}_{X_j}$, $\overline{u}_{I_j}$, and $\overline{u}_{D_j}$ are the average of each class into the X, I, and D training sets, respectively. By using the previous notation, the updated average $\overline{u}_{\widetilde{X}_j}$ of each class is defined by

$$\overline{u}_{\widetilde{X}_j} = \frac{m_{X_j}\overline{u}_{X_j} + m_{I_j}\overline{u}_{I_j} - m_{D_j}\overline{u}_{D_j}}{(m_{X_j} + m_{I_j} - m_{D_j})}. \tag{2}$$

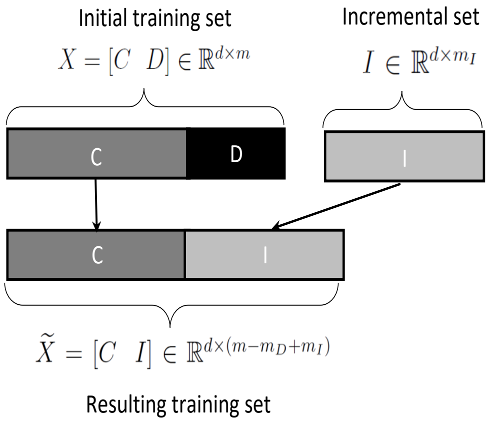

Initial training set
$X = [C \;\; D] \in \mathbb{R}^{d \times m}$
Incremental set
$I \in \mathbb{R}^{d \times m_I}$
C
D
I
C
I
$\widetilde{X} = [C \;\; I] \in \mathbb{R}^{d \times (m - m_D + m_I)}$
Resulting training set

Fig. 2: Main sets involved in the Dual-GDCV method. X is the initial training set. C is the results from removing D of X. I is the incremental training set.

Similarly to X_c in the previous section, the centered data matrices I_c and D_c of the incremental and removed training sets, consist of column vectors $(x_j^i - \overline{u}_{I_j})$ for $i = 1 \ldots m_{I_j}$ and $(x_j^i - \overline{u}_{D_j})$ for $i = 1 \ldots m_{D_j}$, for all $j = 1 \ldots c$, respectively. Their computation allows us to obtain their within-class scatter matrices as $S_w^I = I_c I_c^T$ and $S_w^D = D_c D_c^T$, for the incremental and removed training set respectively.

As stated before, in order to calculate $\widetilde{U_\alpha}$, the $EVD(S_w^{\widetilde{X}})$ problem should be solved. Assuming a common adaptive scenario, where the size of the update set is small in comparison with the initial training set, i.e. $m_I, m_D < m$, it is extremely inefficient from a computational complexity perspective to recalculate $EVD(S_w^{\widetilde{X}})$ from scratch every time knowing that the solution of $EVD(S_w^X)$ is already known and their computational costs are comparable. Furthermore, to make this update possible, the full set X should be accessible at any time, which leads to large memory and spatial complexity. The challenge then is to obtain the subspace, $\widetilde{U_\alpha}$, associated to $\widetilde{X}$ without explicitly having $\widetilde{X}$, $S_w^{\widetilde{X}}$, X or S_w^X.

3.2 Dual-Generalized Discriminative Common Vectors

By assuming the decomposition of the within-class scatter matrix as the sum of its components,[5, 11] the within-class scatter matrix of the new training set $S_w^{\widetilde{X}}$ is calculated as[1]:

$$S_w^{\widetilde{X}} = S_w^X + S_w^I + A_X A_X^T + A_I A_I^T - S_w^D - A_D A_D^T \tag{3}$$

where A_X, A_I, and A_D are matrices containing the differences on the average of each class between the updated training

[1] For more details see the Appendix

set and the initial, incremental, and removed training set, respectively. Specifically, these matrices are given by:

$$\begin{aligned} A_X &= [a_{X_1}\dots a_{X_c}] \quad & a_{X_j} &= \sqrt{m_{X_j}}(\bar{u}_{X_j} - \bar{u}_{\widetilde{X}_j}) \\ A_I &= [a_{I_1}\dots a_{I_c}] \quad & a_{I_j} &= \sqrt{m_{I_j}}(\bar{u}_{I_j} - \bar{u}_{\widetilde{X}_j}) \\ A_D &= [a_{D_1}\dots a_{D_c}] \quad & a_{D_j} &= \sqrt{m_{D_j}}(\bar{u}_{D_j} - \bar{u}_{\widetilde{X}_j}). \end{aligned}$$

The eigendecomposition of matrix $S_w^{\widetilde{X}}$ is approximated as:

$$\begin{aligned} EVD(S_w^{\widetilde{X}}) : S_w^{\widetilde{X}} &= \widetilde{U_r}\,\widetilde{\Lambda_r}\widetilde{U_r}^T \\ &\approx U_\alpha \Lambda_\alpha U_\alpha^T + I_c I_c^T + A_X A_X^T + A_I A_I^T \\ &\quad - D_c D_c^T - A_D A_D^T, \qquad (4) \end{aligned}$$

so that the basis that generates the range space of the new training set, $S_w^{\widetilde{X}}$, is obtained by:

$$\widetilde{U_r} \approx [U_\alpha \;\; V]R, \qquad (5)$$

Like in IGDCV,[5] R is a rotation matrix that controls the dimensionality of the range of $S_w^{\widetilde{X}}$. V contains the new orthogonal directions of both the added and removed centered data sets and the average differences in A. The orthogonal directions in V are obtained by projecting the corresponding vectors using $Y = [I_c \;\; A_X \;\; A_I \;\; D_c \;\; A_D]$:

$$V = orth(Y - U_\alpha U_\alpha^T Y). \qquad (6)$$

This construction constitutes a dual characterization of the range of $S_w^{\widetilde{X}}$. As both $\widetilde{U_r}$ and $[U_\alpha \;\; V]$ generate the same subspace, they are in general related by a rotation R.

By substituting Eq. 5 in the decomposition of Eq. 4 and projecting these matrices onto the range of $S_w^{\widetilde{X}}$ as $[U_\alpha \;\; V]^T(\cdot)[U_\alpha \;\; V]$, we obtain that:

$$\begin{aligned} M = &\begin{bmatrix} \Lambda_\alpha & 0 \\ 0 & 0 \end{bmatrix} + [U_\alpha \; V]^T I_c I_c^T [U_\alpha \; V] \\ &+ [U_\alpha \;\; V]^T A_X A_X^T [U_\alpha \; V] + [U_\alpha \; V]^T A_I A_I^T [U_\alpha \; V] \\ &- [U_\alpha \;\; V]^T D_c D_c^T [U_\alpha \;\; V] - [U_\alpha \;\; V]^T A_D A_D^T [U_\alpha \;\; V]. \qquad (7) \end{aligned}$$

Note that M is a square matrix of size at most $(m - m_D + m_I)$, and that both the rotation R and the new eigenvalues $\widetilde{\Lambda}$ are obtained by eigendecomposing M.

From the eigendecomposition of M, we extract the first eigenvectors, R_β, as the column vectors in R corresponding to the largest eigenvalues, $\widetilde{\Lambda}$, such that $tr(\Lambda_\beta) = \beta \cdot tr(\widetilde{\Lambda})$. Consequently, final accurate approximations for the *dual extended null* space projection with parameter α are written as:

$$\begin{aligned} \widetilde{U_\alpha} &\approx [U_\alpha \;\; V]R_\beta \\ \widetilde{\Lambda_\alpha} &\approx \Lambda_\beta. \end{aligned}$$

Note that the factor β is defined with regard to M, while α refers to $S_w^{\widetilde{X}}$. By considering the proposed approximation, the directions that are removed (depending on the α value), are compensated by adding directions from the remaining data (according to β).

Let $t_\alpha = tr(M) = tr(\widetilde{\Lambda})$, $t_X = tr(S_w^X) = tr(\Lambda)$, $t_D = tr(S_w^D)$, $t_I = tr(S_w^I)$, and $t_{\widetilde{X}} = t_C + t_I = t_X - t_D + t_I$. The appropriate value for β is obtained by $tr(\Lambda_\beta) = \beta \cdot t_\alpha$, and this scatter should ideally be equal to $\alpha \cdot t_{\widetilde{X}}$, such that:

$$\begin{aligned} \beta \cdot t_\alpha &= \alpha \cdot [t_C + t_I] \\ &= \alpha \cdot t_X - \alpha \cdot t_D + \alpha \cdot t_I. \end{aligned}$$

The final expression for β, written in terms of the ratio between the traces of the diagonal matrices is:

$$\beta = \frac{tr(\Lambda_\alpha)}{tr(\widetilde{\Lambda})} - \alpha\frac{tr(\Lambda_D)}{tr(\widetilde{\Lambda})} + \alpha\frac{tr(\Lambda_I)}{tr(\widetilde{\Lambda})}, \qquad (8)$$

using the facts that $tr(\Lambda_\alpha) = \alpha \cdot t_X$, $S_w^D = U_D \Lambda_D U_D^T$ and $S_w^I = U_I \Lambda_I U_I^T$.

The Dual-GDCV approach is presented in Algorithm 1 along with the asymptotic cost corresponding to each of its steps.

If some of the data vectors in I correspond to new classes which are not present in X, the expressions of the Dual-GDCV algorithm are valid by increasing the value of c and setting $m_j, m_{D_j} = 0$ in X and D for all new classes. If either m_j or m_{D_j}, m_{I_j} are zero for any class j, the corresponding average is undefined and the corresponding columns in A_D, A_I, a_{D_j}, a_{I_j}, should be set to zero. If all data vectors in I correspond to new classes, then the whole matrix A_I is the zero matrix and is removed from all expressions.

3.3 Computational and space complexity

The asymptotic cost of the Dual-GDCV is dominated by $O(d(m_D + m_I + c)^2 + (m_D + m_I + c + r)^3 + dr(m_D + m_I + c))$. In the case of the batch algorithm the complexity is dominated by $O(dm^2 + m^3 + dmr)$, when $d \gg m$, and $O(d^2m + d^3)$, when $d \leq m$. We can seen that the Dual-GDCV approach is more efficient than the batch algorithm in both cases since $(m_D + m_I + c) \ll m$, and $(m_D + m_I + c + r) \leq min(d, m)$. Table 1 shows this comparison between the Dual-GDCV approach and the batch method.

Dual-GDCV	**Batch GDCV**	
$d \gg (m_D, m_I)$	$d \gg m$	$d < m$
$d(m_D + m_I + c)^2$	dm^2	d^2m
$(m_D + m_I + c + r)^3$	m^3	d^3
$dr(m_D + m_I + c)$	drm	

Table 1: Comparison between the main computational complexities of the Dual-GDCV and the batch GDCV.

Algorithm 1 *Dual-GDCV Algorithm*

```
Parameter: $\alpha$, $0 \leq \alpha < 1$
Input: $D \in \mathbb{R}^{d \times m_D}$, $I \in \mathbb{R}^{d \times m_I}$
From previous iteration: $U_\alpha \in \mathbb{R}^{d \times r}$, $\Lambda_\alpha \in \mathbb{R}^{r \times r}$, $\overline{X} \in \mathbb{R}^{d \times c}$
Output: $\widetilde{U_\alpha} \in \mathbb{R}^{d \times \widetilde{r}}$, $\widetilde{\Lambda_\alpha} \in \mathbb{R}^{\widetilde{r} \times \widetilde{r}}$, $\overline{X} \in \mathbb{R}^{d \times c}$
Method:
 1. Compute:
        $D$ and $I$
        regarding its average to obtain $D_c$ and $I_c$                  // $O(dm_D + dm_I)$
 2. Compute:
        $A_X$, $A_I$, $A_D$                                                 // $O(dc)$
 3. Compute:
        $EVD(D_c D_c^T) : U_D \Lambda_D U_D^T$                              // $O(d^2 m_D + d^3)$
        if $d > m_D$ use:
            $EVD(D_c^T D_c) : U_D \Lambda_D U_D^T$                          // $O(d{m_D}^2 + {m_D}^3)$
 4. Compute:
        $EVD(I_c I_c^T) : U_I \Lambda_I U_I^T$                              // $O(d^2 m_I + d^3)$
        if $d > m_I$ use:
            $EVD(I_c^T I_c) : U_I \Lambda_I U_I^T$                          // $O(d{m_I}^2 + {m_I}^3)$
 5. Compute:
        $V = orth(Y - U_\alpha U_\alpha^T Y)$, and                        // $O(d(m_D + m_I + c)r + d(m_D + m_I + c)^2)$
        $Y = [U_I \;\; A_X \;\; A_I \;\; U_D \;\; A_D]$
 6. Build $M$ using Eq. 7                                               // $O(d(m_D + m_I + c)(r + m_D + m_I + c))$
 7. Eigendecompose $M$ in $R\widetilde{\Lambda}R^T$                       // $O((r + m_D + m_I + c)^3)$
    to obtain the eigenvalues $\widetilde{\Lambda_\alpha} = \Lambda_\beta$ within $\widetilde{\Lambda}$ according to $\beta$ Eq. 8
 8. Compute the generalized common vector as:
        $x_{gcv}^j = \overline{u}_{\widetilde{X}_j} - \widetilde{U_\alpha}\widetilde{U_\alpha}^T \overline{u}_{\widetilde{X}_j}$          // $O(d\widetilde{r}c)$
 9. Compute the generalized common vector as:
        $x_{gcv}^j = \overline{u}_{\widetilde{X}_j} - \widetilde{U_\alpha}\widetilde{U_\alpha}^T \overline{u}_{\widetilde{X}_j}$
10. Define:
        $X^{com} = [x_{gcv}^1 \ldots x_{gcv}^c]$, and
        $X_c^{com}$ its centered version with regard to the mean $\overline{x}_{com} = (1/c)\sum_{j=1}^{c} x_{gcv}^j$.
11. Compute the projection matrix such that:
        $\widetilde{W} = orth(X_c^{com}) \in \mathbb{R}^{d \times (c-1)}$
12. Obtain the generalized discriminative common vectors as:
        $\widetilde{W}^T \overline{u}_{\widetilde{X}_j}$
13. To test a new sample, $x_{test}$, project it as $\widetilde{W}^T x_{test}$.
        The label is allocated from the minimum distance between the projected sample and the discriminative common vectors.
```

As expected, the closer the value between the number of samples added and/or deleted and the size of the initial training set, the smaller the computational gain by using a dual approach, since previous disparities are not fulfilled. Thus, if most samples/classes of the initial training set are to be deleted, it is simpler to train the system from scratch. However, this scenario will only be possible for very small and simple problems and toy examples, not in real life problems and big sets. Regarding space complexity, the batch method exhibits a $O(min(d,m)^2)$ while the dual algorithm has a $O((m_D + m_I + c + r)^2)$.

4 Experiments and Results

4.1 Experimental setup

To observe the advantages of the Dual-GDCV approach to add and delete samples into the initial training data of a classification problem, we select six datasets of images as test bench. As classifier, a simple 1-Nearest Neighbors classifier is employed, using the Euclidean distance between the trained discriminative common vectors and the test samples projected into the discriminant subspace. The simplicity of the classifier is justified for our aim to demonstrate the accuracy and approximation of our method to obtain a projection into another space where the relevant information is easily separable into the different classes. Table 2 shows the main characteristics of the datasets Coil-20,[21] CMU-PIE,[29] AR,[20] BANCA,[14] Altkom[14] and FERET,[26] where the size of image has been normalized to 40×40.

	Coil-20	CMU-PIE	AR	BANCA	Altkom	FERET
c	20	68	50	52	80	200
m_j	72	56	14	10	15	4

Table 2: Datasets used in validation along with their corresponding details. c is the number of classes. m_j is the number of samples per class.

The proposed Dual-GDCV algorithm is empirical validated in both cases, when $d \leq M$ and $d > M$, both in terms of the accuracy and computational time and compared against both the batch GDCV algorithm, and state-of-art dual[12,23] and incremental approaches.[5,14,25,32] In this validation, two main scenarios are considered using raw pixel images as input. In the first one, the number of classes in the training set is fixed over the updates, with samples of each class being added and/or removed. In the second scenario, the number of classes is allowed to change over the updates. In all experiments, an initial model is obtained using the corresponding batch algorithm. For each scenario, the proposed algorithm is evaluated in dual-learning, decremental-only and incremental-only experiments. Afterwards, DGDCV is applied using input features derived from a Convolutional Neural Network (CNN), in order to evaluate the potential of our approach to allow dual learning in a Deep Learning framework. Finally, a large scale experiment is presented to show the limitations of our methodology.

Cross validation is applied as evaluation protocol to avoid bias to a particular training/testing split. Each experiment is run 10 times with different random training/testing sample choices. Graphs show the average result over the iterations as well as dispersion bars. All algorithms have been run on a computer with a Intel(R) Core(TM) i7-4790 CPU @ 3.60GHz, 3601 Mhz, and 32-GB RAM.

4.2 First scenario: Constant number of classes

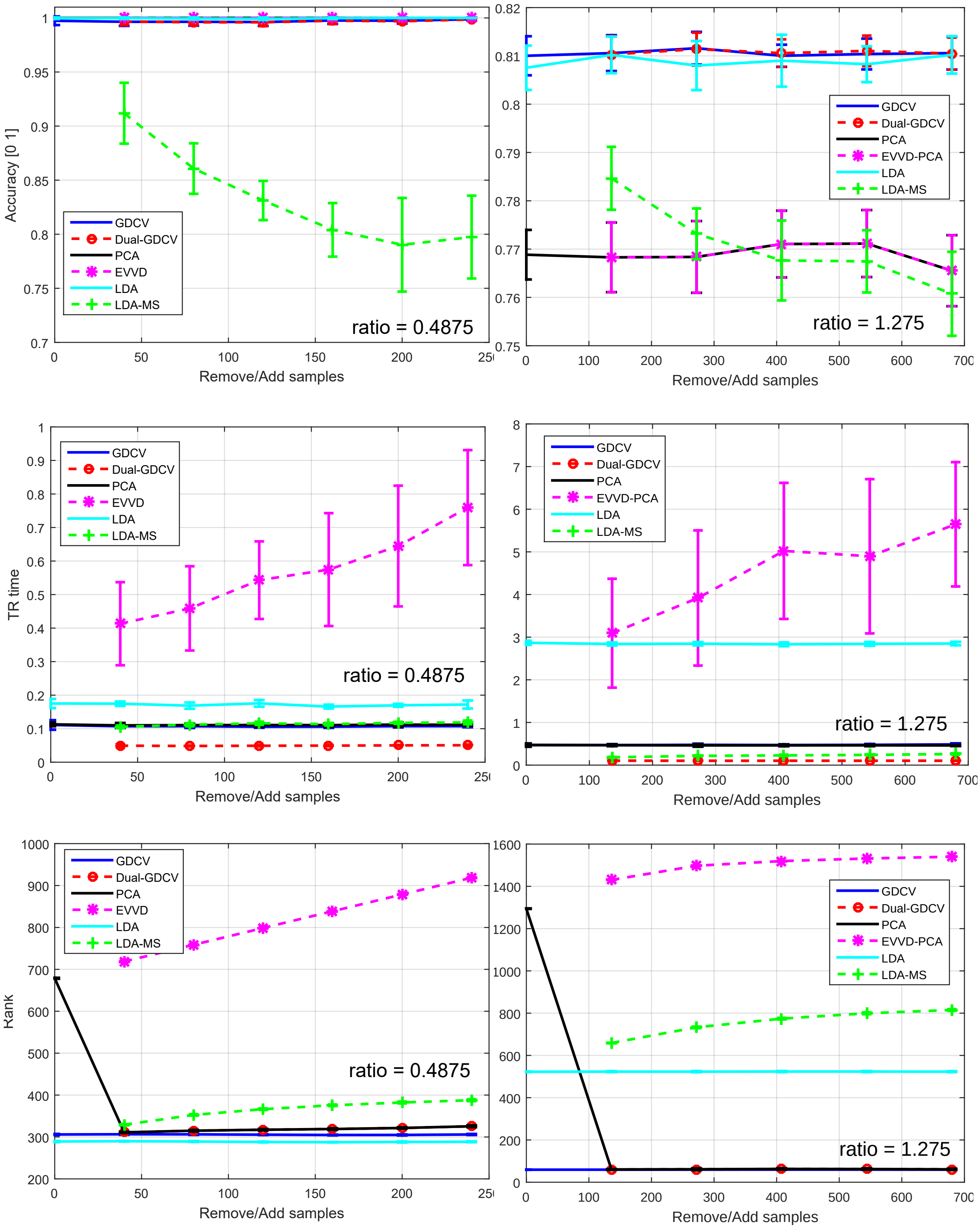
In this first scenario, new samples are added and old samples are deleted to and from the initial training set. Since the number of classes is not allowed to change during the updates, samples for every class should be in the initial set.

We validate our approach using Coil-20[21] and CMU-PIE,[29] since they contain a big enough number of samples per class to perform a healthy number of update iterations. Note that we are evaluating the Dual-GDCV into both cases, $d \leq M$ and $d > M$, since the ratio (M/d) in Coil-20 and CMU-PIE is 0.4875 and 1.275, respectively. For each dataset, three subsets are generated: the initial training set containing 52% of the samples, a incremental set with 18% and the test set with the remaining 30%. In those experiments where decremental learning is applied, 18% of the samples are deleted, such that only the (34%) of the initial training set is preserved. Subspace-based learning parameters were empirically optimized to give the best possible accuracy in the batch version of each method under comparison. In our GDCV framework, resulting scattering parameter is set to $\alpha = 0.99$ in Coil-20 and $\alpha = 0.96$ in CMU-PIE. For PCA and LDA, the energy parameter ranged from 0.99 to 1.

Dual learning experiment. In this experiment samples from each class are added and deleted in each updating step. Specifically, 2 samples per class are added and another 2 samples are deleted simultaneously. Our approach is compared against the dual approaches in the literature EVVD-PCA[12] -unsupervised-, the LDA-MS[23] -supervised-, and their batch versions.

Figure 3 shows the accuracy, computational training time and the rank of $S_w^{\widetilde{X}}$. Table 3 shows the accuracy rate, the computational time and the rank, as well as their standard deviations, for the last iteration of each method. We can see that the Dual-GDCV shows the same discriminant properties than its batch version GDCV but exhibits a significant saving in its computational time. The computational time in Coil-20 reduced by 53%, and in CMU-PIE by 79%. In comparison to other dual methods, Dual-GDCV outperforms both supervised and unsupervised methods in both accuracy and cost overall. Dual-GDCV shows almost the same performance that PCA and EVVD-PCA[12] in Coil-20, where the difference is not significative, and superior performance in CMU-PIE. In both sets, Dual-GDCV has a significant computational advantage, specially when compared with the dual version EVVD-PCA, which has to extend the range during the updates to preserve the same discriminant properties than its batch version, leading to a higher computational cost than the batch PCA. Dual-GDCV also shows better performance and less computational time than LDA-MS[23] in both sets, whose accuracy degrades over time due to cumulative errors in the adaptive approximation. As additional advantage, while EVVD-PCA and Dual-GDCV are fully dual, that is, samples can be added and deleted in the same operation, LDA-MS needs to do two different merging and splitting operations for each dual step, leading to higher computational cost.

Incremental-only experiment. In this experiment samples from each class are added in each updating step. Our approach is compared against the dual approaches in the



(e) Coil-20

(f) CMU-PIE

Fig. 3: Comparison between the Dual-GDCV and the EVVD-PCA,[12] the LDA-MS,[23] and their batch versions for dual-training at sample level. Accuracy, CPU computational training time and the rank of $S_w^{\widetilde{X}}$ are in first, second and third row, respectively, and datasets Coil-20 (left) and CMU-PIE (right) in the columns.

Method	Coil-20					
	Acc	Std	TR time	Std	Rank	Std
PCA[12]	1.0000	0	0.1114	0.0026	325.6	2.1705
EVDD-PCA[12]	1.0000	0	0.7594	0.1715	918.7	1.2517
LDA[23]	**1.0000**	**0**	0.1721	0.0121	288.6	0.9661
LDA-MS[23]	0.7974	0.0383	0.1189	0.0056	387.8	2.2509
GDCV[5]	**0.9988**	**0.0013**	0.1074	0.0030	306.0	1.2472
Dual-GDCV	**0.9986**	**0.0012**	**0.0506**	**0.0021**	325.6	2.1705
	CMU-PIE					
PCA[12]	0.7655	0.0073	0.4619	0.0156	60.9	2.8460
EVDD-PCA[12]	0.7655	0.0073	5.6462	1.4572	1540.3	1.3374
LDA[23]	**0.8102**	**0.0038**	2.8502	0.0400	523.2	1.8737
LDA-MS[23]	0.7607	0.0086	0.2606	0.0149	815.0	3.8586
GDCV[5]	**0.8105**	**0.0033**	0.4853	0.0253	59.5	0.8498
Dual-GDCV	**0.8104**	**0.0033**	**0.1027**	**0.0030**	60.9	2.8460

Table 3: Performance at the last update step for dual-training at sample level.

literature EVVD-PCA[12] -unsupervised-, the LDA-MS[23] -supervised-, and other common incremental only state-of-art approaches, namely IncLDA,[14] GSVD-ILDA,[32] cQR-ILDA[25] and IGDCV,[5] as well as their batch versions.

Tables 4 show the accuracy, computational training time and the rank of $S_w^{\widetilde{X}}$ when only new samples are added into the initial training set to each method. Accuracy results for DGCV and IGCV in this incremental-only experiment are exactly the same since, in our formulation, the latter is a particular case of the former. Their computational costs are also equivalent, with small differences due to implementation that are not significative. Regarding its comparison to others incremental-only methods, only QR-LDA's performance is better that the Dual-GDCV, in both the accuracy rate and the CPU time and in both sets. IncLDA[14] shows a similar performance to our approach in CMU-PIE its cost multiplies by 4. These incremental-only subspace-based methods are, however, not able to remove samples from the training set.

Decremental-only experiment. In this experiment samples from each class are deleted in each updating step. Since no decremental-only learning methodology exists, our approach is compared against the dual approaches in the literature EVVD-PCA[12] - unsupervised -, the LDA-MS[23] - supervised -, and their batch versions.

Table 5 shows the accuracy, computational training time and the rank of $S_w^{\widetilde{X}}$ when only samples are deleted from the initial training set. Similarly to previous experiments, we can see that in both datasets the Dual-GDCV presents better performance both in accuracy rate and computational time. Regarding methods with comparable accuracy rate, Dual-GDCV exhibits computational gains of 67% over PCA, 94% over PCA-EVVD, 80% over LDA, and 65% over GDCV in Coil-20, and 98% over LDA, and 90% over GDCV in CMU-Pie.

4.3 Second scenario: Variable number of classes

In this second scenario, new classes are added to the initial training set by adding all samples belonging to classes that did not exist initially. Similarly, old classes are deleted from the initial training set by removing all samples belonging to a existing class at once.

We validated our approach using the remaining datasets AR,[20] BANCA,[14] Altkom[14] and FERET.[26] Scattering parameter is set to $\alpha = 0.98$ in AR, $\alpha = 0.80$ in BANCA and $\alpha = 0.99$ in Altkom and FERET. For each dataset, training and testing are generated randomly containing 70% and 30% of the samples of each class respectively. 30% of the classes (the last ones) are using as incremental set and another 30% of the classes (the first ones) are using as decremental set. At each iteration one class is added and/or deleted from the model. Our method is compared in this second scenario against the same baseline methods for each of the settings than the first scenario, but only including the supervised methods among them. Comparison against unsupervised methods is not performed due to the importance of adding classes in the reshaping of the subspace.

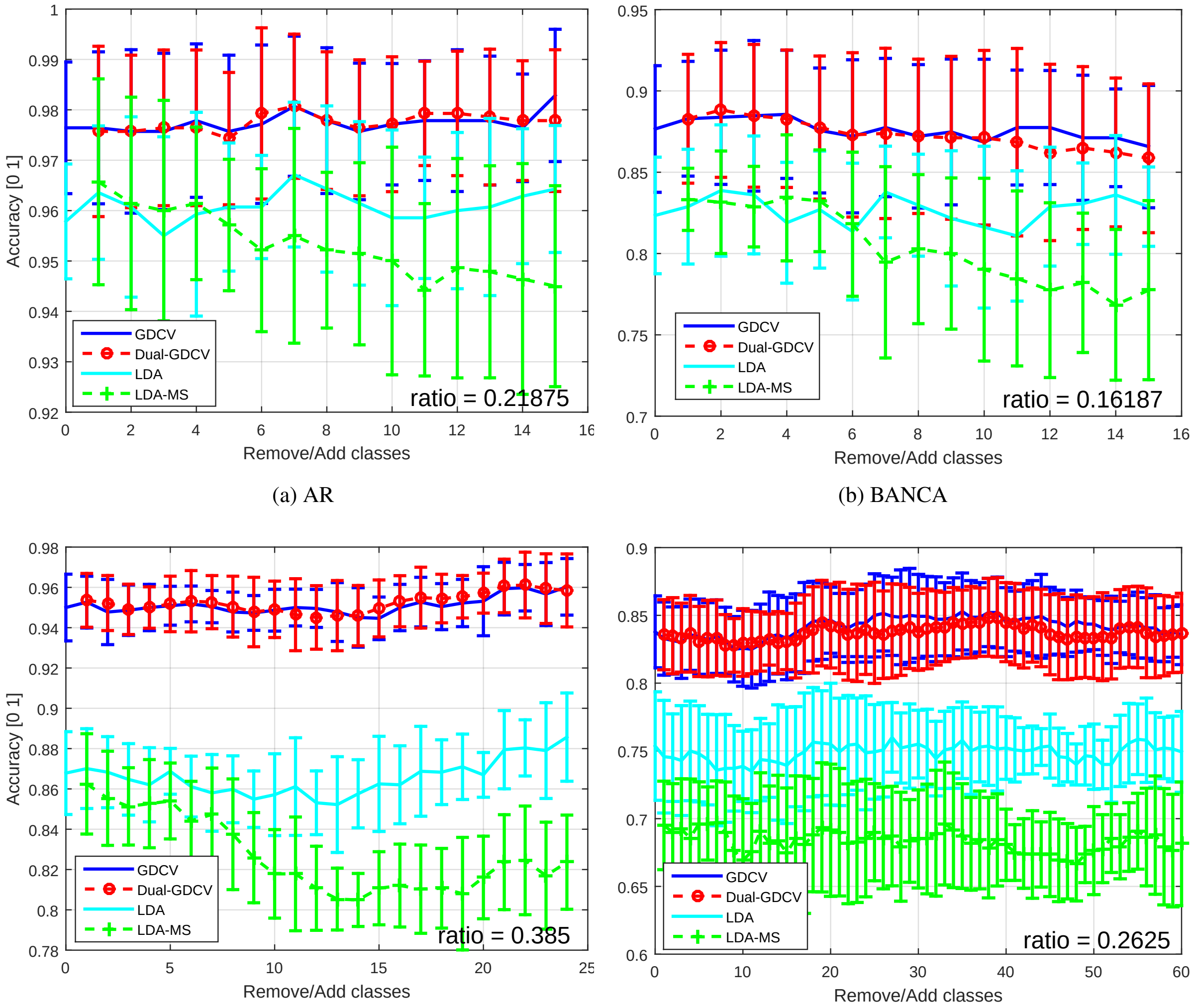
Dual learning experiment. In this experiment a class is added and another class is deleted in each updating step.

Method	Coil-20					
	Acc	Std	TR time	Std	Rank	Std
PCA[12]	1.0000	0	0.1925	0.0151	373.8	1.6865
EVDD-PCA[12]	1.0000	0	0.6240	0.2369	918.9	1.1972
LDA[23]	1.0000	0	0.3426	0.0300	329.0	0
LDA-MS[23]	0.6710	0.0512	0.3426	0.0300	524.8	0.9189
IncLDA[14]	1.0000	0	**0.0403**	**0.0034**	294.9	1.5951
GSVD-LDA[32]	0.9905	0	0.7400	0.0566	336.0	0
GSVD-ILDA[32]	0.9905	0.0032	0.2335	0.0091	72.0	0.4714
QR-LDA[25]	1.0000	0	**0.0206**	**0.0012**	20.0	0
cQR-ILDA[25]	1.0000	0	0.1544	0.0075	20.0	0
GDCV[5]	1.0000	0	0.1820	0.0099	347.0	0
IGDCV[5]	0.9981	0.0015	0.1134	0.0081	373.8	1.6865
Dual-GDCV	0.9981	0.0015	**0.0433**	**0.0027**	373.8	1.6865
	CMU-PIE					
PCA[12]	0,8097	1,17e-16	0,4929	0,0107	63,3	2,2136
EVDD-PCA[12]	0,8097	1,17e-16	2,6278	0,0379	1600	0
LDA[23]	**0,8254**	**1,17e-16**	6,7477	0,0738	573	0
LDA-MS[23]	0,5097	0,0249	6,7477	0,0738	1186	2,2608
IncLDA[14]	0,8199	0,0019	0,1625	0,0087	506	2,2608
GSVD-LDA[32]	0,8199	2,34e-16	2,4807	0,0377	550	0
GSVD-ILDA[32]	0,8197	0,0022	0,1567	0,0047	144,4	2,1187
QR-LDA[25]	**0,8254**	**1,17e-16**	**0,0709**	**0,0025**	68	0
cQR-ILDA[25]	**0,8202**	**0,0020**	0,4494	0,0079	68	0
GDCV[5]	0,8125	0	0,6051	0,0154	60	0
IGDCV[5]	0,8115	0,0015	0,0753	0,0044	63,3	2,2136
Dual-GDCV	0,8115	0,0015	**0,0481**	**0,0014**	63,3	2,2136

Table 4: Performance at the last update step for incremental-training at sample level

Method	Coil-20					
	Acc	Std	TR time	Std	Rank	Std
PCA[12]	1.0000	0	0.1141	0.0055	309.2	1.1353
EVDD-PCA[12]	1.0000	0	0.6712	0.0205	831.0	0
LDA[23]	0.9998	0.0008	0.1876	0.0109	293.7	1.7029
LDA-MS[23]	0.8186	0.0258	0.0765	0.0039	334.3	1.0593
GDCV[5]	0.9998	0.0008	0.1083	0.0039	305.8	1.2293
Dual-GDCV	0.9995	0.0010	**0.0378**	**0.0023**	309.2	1.1353
	CMU-PIE					
PCA[12]	0,7666	0,0085	0,5186	0,02468	55,3	0,6749
EVDD-PCA[12]	0,7666	0,0085	3,9214	0,07640	1378	0
LDA[23]	**0,8098**	**0,0041**	3,0180	0,06854	522	1,9436
LDA-MS[23]	0,7942	0,0049	0,1143	0,01007	590	0
GDCV[5]	**0,8075**	**0,0048**	0,5352	0,02245	59,2	0,42163
Dual-GDCV	**0,8070**	**0,0039**	**0,0523**	**0,00127**	55,3	0,67494

Table 5: Performance at the last update step for decremental-training at sample level



(a) AR

(b) BANCA

(c) Altkom

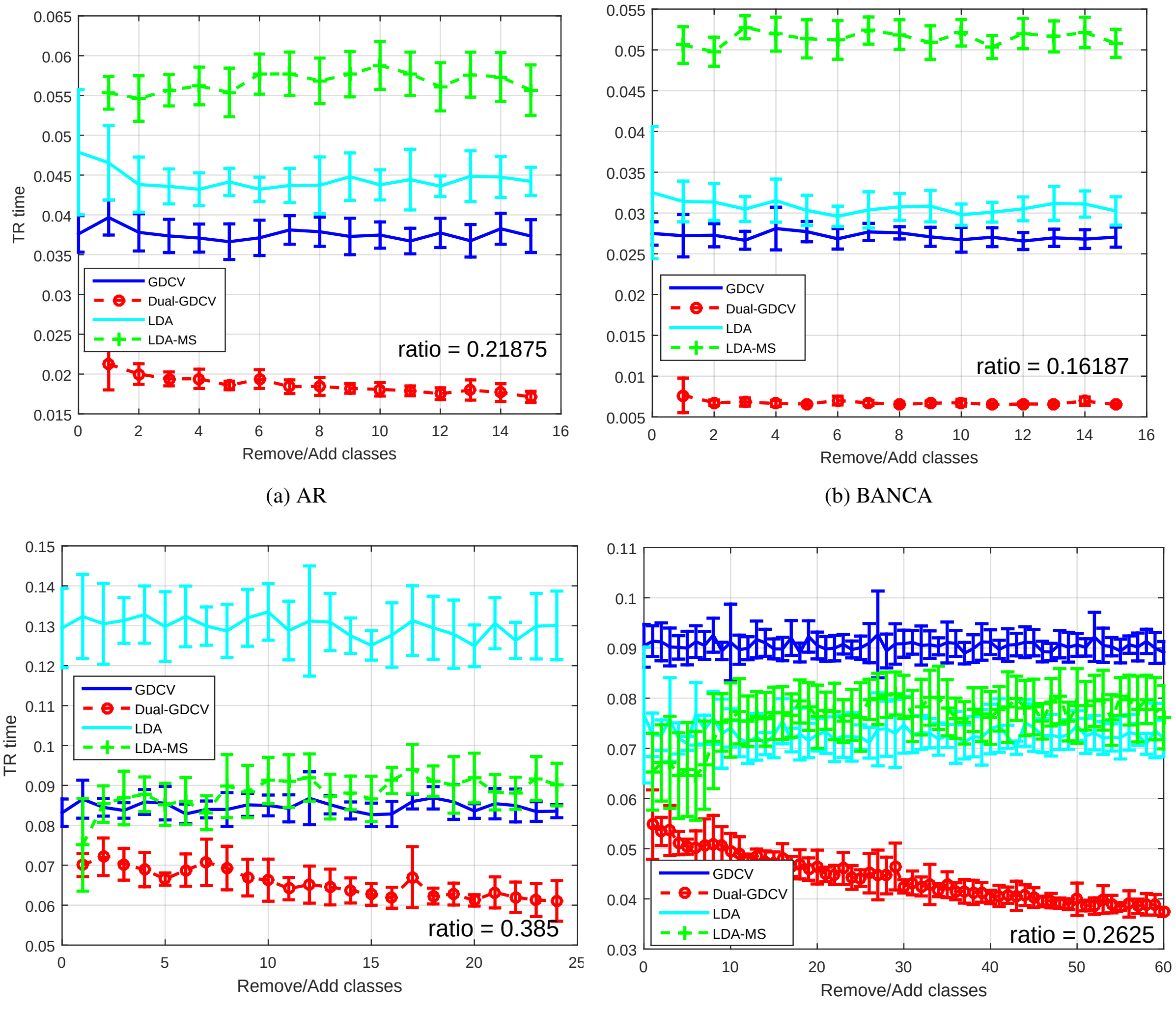
(d) FERET

Fig. 4: Accuracy comparison for dual training at class level on AR, BANCA, Altkom and FERET datasets

Figures 4, 5 and 6 show the accucary, computational training time and the rank of $S_w^{\widetilde{X}}$, respectively, when new classes are added and old classes are deleted into the initial training set. Table 6 shows the accuracy rate, computational time and the rank in the last iteration for each dataset and method. The advantages of Dual-GDCV regarding its batch version and other dual methods are more evident in this scenario, since more samples are added/deleted at each step. Dual-GDCV shows the same discriminant properties than its batch version but with a reduction in computational cost of 54% in AR, 76% in BANCA, 27% in Altkom and 58% in FERET. Moreover, it clearly outperforms the other dual-learning methodology LDA-MS in all datasets, both in time and accuracy.

Incremental-only experiment. In this experiment a class is added in each updating step. Tables 7 and 8 show the accuracy, computational training time and the rank of $S_w^{\widetilde{X}}$ when only new classes are added. Dual-GDCV outperforms any other method, batch, incremental or dual in all datasets, both in time and accuracy. Similarly to the previous scenario, IGDCV provides the same accuracy results since it is a particular case of DGDCV.

Decremental-only experiment. In this experiment a class is deleted in each updating step. Table 9 shows the accuracy, computational training time and the rank of $S_w^{\widetilde{X}}$ when only old classes are deleted. Again, Dual-GDCV outperforms any other method, batch, incremental or dual in all datasets, both in time and accuracy.



(a) AR

(b) BANCA

(c) Altkom

(d) FERET

Fig. 5: CPU training time comparison for dual training at class level on AR, BANCA, Altkom and FERET datasets

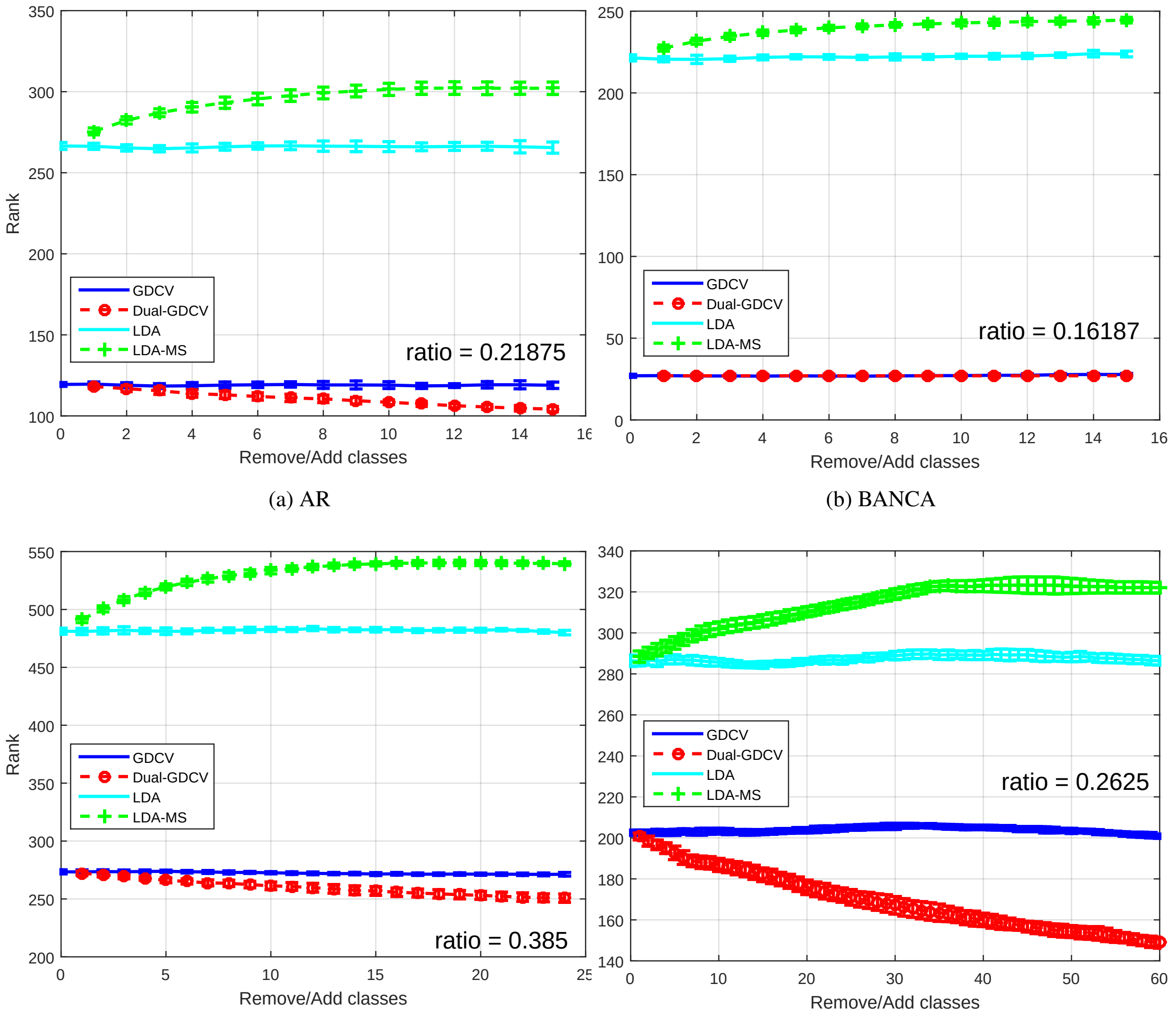
4.4 Application to state-of-art feature vectors

This section aims to explore the potential of our approach to be used in a Deep Learning framework and enable dual learning in such feature space. Thus, this experiment proposes the use of DGDCV where the input comes from the pretrained AlexNet convolutional neural network.[15] The Alexnet model has been trained on a subset of the ImageNet database[3] with more than a million images, so that a rich feature representation has been learned for a wide range of images. A feature space of 4096 dimensions is generated by cropping the last classfifcation layer of Alexnet. Evaluation of GDCV is done under a dual-learning setting for both new samples and new classes as in sections 4.2 and 4.3.

Figure 7 shows the accuracy and the computational training time on the dual-learning at sample level, and figures 8 and 9 presents the accuracy and computational time at class level. Results shows similar conclusions and trends to the raw pixel experiment, with our approach surpassing all other state of the art dual-methods, but with a small overall improve in performance.

4.5 On the ordering of updating steps

In this section, an experiment was designed to validate our approach for applications where temporal or order information is relevant to the updating steps. Object classification is used as target problem in the Coil-20[21] dataset but using



(a) AR

(b) BANCA

(c) Altkom

(d) FERET

Fig. 6: Rank comparison for dual training at class level on AR, BANCA, Altkom and FERET datasets

the camera angle as ordering and the potential drift on the object's distribution.

Coil-20[21] was captured on a motorized turntable, where the turntable was rotated from 0 to 360 degrees, at intervals of 5 degrees, to vary object pose with respect to a fixed camera. This resulted on 72 images per object. Two experimental settings are used: incremental and dual learning as depicted in Figure 10. For incremental learning, samples of all objects viewed between 165 and 180 degrees are used as training set, while images between 0 and 110 degrees are in the initial training. Images viewed from 110 to 165 degrees are used as ascending-ordered updating set. In the dual setup, the same process is followed but using the initial angels between 0 and 45 degrees as decremental set to be remove at the same time than an incremental set is added. As reference to evaluate the importance of this ordering, the same experiments are reverted (decreasing ordering), where the less relevant samples (0 to 45) are added and the more relevant ones are decremented (dual setting).

Figure 11 shows the accuracy on the incremental-learning and dual learning with ordered samples on the updating training set. For the incremental setup (Fig. 11a), when the incremental learning is provided with increasingly relevant samples, the classification performance improves significantly from 0.75 up to 1, suggesting that the distribution adapts to the end target. On the contrary, if the added samples are the less relevant ones, the performance, which starts in a higher baseline since the most relevant information is already in the initial training, barely improves from 0.95 up to 1. This

Method	AR					
	Acc	Std	TR time	Std	Rank	Std
LDA[23]	0.9643	0.0126	0.0442	0.0018	265.5	3.4400
LDA-MS[23]	0.9450	0.0199	0.0557	0.0032	302.1	3.8427
GDCV[5]	**0.9829**	**0.0131**	0.0373	0.0021	118.9	1.9692
Dual-GDCV	**0.9779**	**0.0141**	**0.0171**	**0.0007**	104.2	1.6193
	BANCA					
LDA[23]	0.8288	0.0244	0.0303	0.0017	223.8	1.7512
LDA-MS[23]	0.7775	0.0551	0.0508	0.0017	244.5	1.7159
GDCV[5]	**0.8658**	**0.0376**	0.0270	0.0012	27.8	0.6325
Dual-GDCV	**0.8586**	**0.0458**	**0.0065**	**0.0002**	27.0	0.9428
	Altkom					
LDA[23]	0.8857	0.0219	0.1301	0.0086	480.0	1.9437
LDA-MS[23]	0.8237	0.0234	0.0902	0.0053	539.6	1.6465
GDCV[5]	**0.9603**	**0.0140**	0.0835	0.0016	271.3	1.5670
Dual-GDCV	**0.9585**	**0.0181**	**0.0611**	**0.0051**	250.7	3.6225
	FERET					
LDA[23]	0.7493	0.0298	0.0720	0.0036	286.4	2.1187
LDA-MS[23]	0.6814	0.0456	0.0762	0.0063	322.0	2.5820
GDCV[5]	**0.8357**	**0.0221**	0.0891	0.0022	200.8	1.0328
Dual-GDCV	**0.8371**	**0.0291**	**0.0374**	**0.0009**	148.9	2.7264

Table 6: Performance at the last update step for dual-training at class level.

Method	AR					
	Acc	Std	TR time	Std	Rank	Std
LDA[23]	0.9560	0.0133	0.0786	0.0022	337.7	1.6364
LDA-MS[23]	0.9565	0.0149	0.0686	0.0011	402.6	0.6992
GSVD-LDA[32]	0.9580	0.0130	0.1693	0.0041	457.0	0
GSVD-ILDA[32]	0.7280	0.0178	0.1887	0.0072	459.0	0
QR-LDA[25]	0.8820	0.0272	0.0231	0.0020	50.00	0
cQR-ILDA[25]	0.9080	0.0190	0.1834	0.0038	50.0	0
GDCV[5]	0.9770	0.0111	0.0592	0.0024	142.4	1.1738
IGDCV[5]	0.9790	0.0127	0.0173	0.0013	155.1	1.7288
Dual-GDCV	**0.9790**	**0.0127**	**0.0162**	**0.0012**	155.1	1.7288
	BANCA					
LDA[23]	0.8186	0.0432	0.0471	0.0009	279.1	0.9944
LDA-MS[23]	0.8019	0.0317	0.0082	0.0005	307.9	1.6633
GSVD-LDA[32]	0.8327	0.0292	0.1204	0.0011	341.0	0
GSVD-ILDA[32]	0.3006	0.0509	0.1274	0.0013	334.0	0
QR-LDA[25]	0.7301	0.0323	0.0220	0.0002	52.0	0
cQR-ILDA[25]	0.7006	0.0306	0.1887	0.0007	52.0	0
GDCV[5]	0.8724	0.0255	0.0413	0.0018	29.7	0.4830
IGDCV[5]	0.8750	0.0226	0.0098	0.0010	30.1	1.6633
Dual-GDCV	**0.8750**	0.0226	**0.0072**	**0.0009**	30.1	1.6633

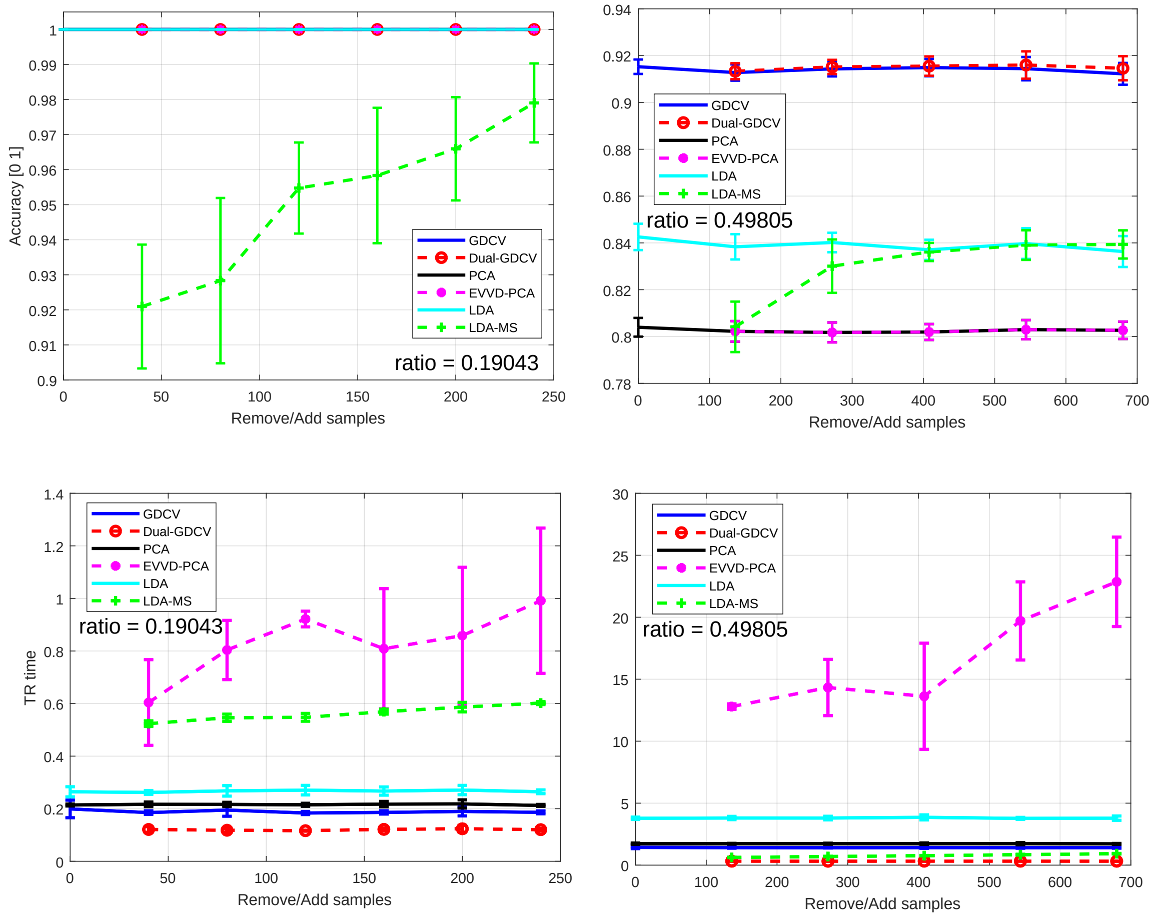
Table 7: Performance at the last update step for incremental-training at class level.

Method	Altkom					
	Acc	Std	TR time	Std	Rank	Std
LDA[23]	0.8409	0.0265	0.2922	0.0171	591.7	1.3375
LDA-MS[23]	0.8000	0.0113	0.1998	0.0101	721.8	1.3984
GSVD-LDA[32]	0.8153	0.0243	0.4980	0.0206	784.1	0.3162
GSVD-ILDA[32]	0.4309	0.0284	0.5244	0.0161	804.6	0.5164
QR-LDA[25]	0.6656	0.0570	0.0418	0.0026	80.0	0
cQR-ILDA[25]	0.7200	0.0269	0.2690	0.0022	80.0	0
GDCV[5]	**0.9281**	**0.0093**	0.1535	0.0145	322.2	1.0328
IGDCV[5]	**0.9266**	**0.0142**	**0.0441**	**0.0011**	357.8	1.8738
Dual-GDCV	**0.9266**	**0.0142**	**0.0470**	**0.0020**	357.8	1.8738
	FERET					
LDA[23]	0.6835	0.0299	0.1446	0.0080	356.5	1.4337
LDA-MS[23]	0.6240	0.0307	0.1175	0.0022	404.9	1.2867
GSVD-LDA[32]	0.6265	0.0336	0.3520	0.0118	549.2	0.4216
GSVD-ILDA[32]	0.4530	0.0242	0.2444	0.0091	510.9	0.3162
QR-LDA[25]	0.7500	0.0318	0.1004	0.0034	200.0	0
cQR-ILDA[25]	0.7140	0.0290	0.0803	0.0067	200.0	0
GDCV[5]	0.7705	0.0281	0.1506	0.0051	261.9	1.4491
IGDCV[5]	**0.7750**	**0.0253**	**0.0567**	**0.0024**	256.6	1.4298
Dual-GDCV	**0.7750**	**0.0253**	**0.0572**	**0.0027**	256.6	1.4298

Table 8: Performance at the last update step for incremental-training at class level.

Method	AR					
	Acc	Std	TR time	Std	Rank	Std
LDA[23]	0.9579	0.0119	0.0436	0.0024	266.3	1.4181
LDA-MS[23]	0.9629	0.0125	0.0425	0.0019	298.4	1.8974
GDCV[5]	**0.9771**	**0.0100**	0.0378	0.0011	119.2	1.1353
Dual-GDCV	**0.9764**	**0.0107**	**0.0137**	**0.0006**	115.2	2.0976
	BANCA					
LDA[23]	0.8234	0.0301	0.0298	0.0015	224.3	1.7029
LDA-MS[23]	0.7991	0.0368	0.0378	0.0014	241.9	1.7288
GDCV[5]	**0.8559**	**0.0321**	0.0275	0.0006	27.0	0.4714
Dual-GDCV	**0.8333**	**0.0385**	**0.0055**	**0.0001**	15.2	0.9189
	Altkom					
LDA[23]	0.8647	0.0197	0.1279	0.0117	481.5	1.4337
LDA-MS[23]	0.8268	0.0153	0.0841	0.0162	543.9	1.4491
GDCV[5]	**0.9562**	**0.0136**	0.0838	0.0047	269.9	0.9944
Dual-GDCV	**0.9540**	**0.0115**	**0.0468**	**0.0048**	267.6	1.4298
	FERET					
LDA[23]	0.7500	0.0367	0.0727	0.0038	291.0	1.3333
LDA-MS[23]	0.6664	0.0196	0.0688	0.0072	350.2	1.9322
GDCV[5]	**0.8429**	**0.0216**	0.0901	0.0035	204.5	0.8498
Dual-GDCV	**0.8421**	**0.0186**	**0.0386**	**0.0026**	190.5	1.5092

Table 9: Performance at the last update step for decremental-training at class level.



(c) Coil-20

(d) CMU-PIE

Fig. 7: Accuracy and CPU time comparison for dual training at sample level when using alexnet's features on Coil-20 (left) and CMU-PIE (right) datasets.

small improvement is mostly due to having more samples for the classifier rather than for providing new information.

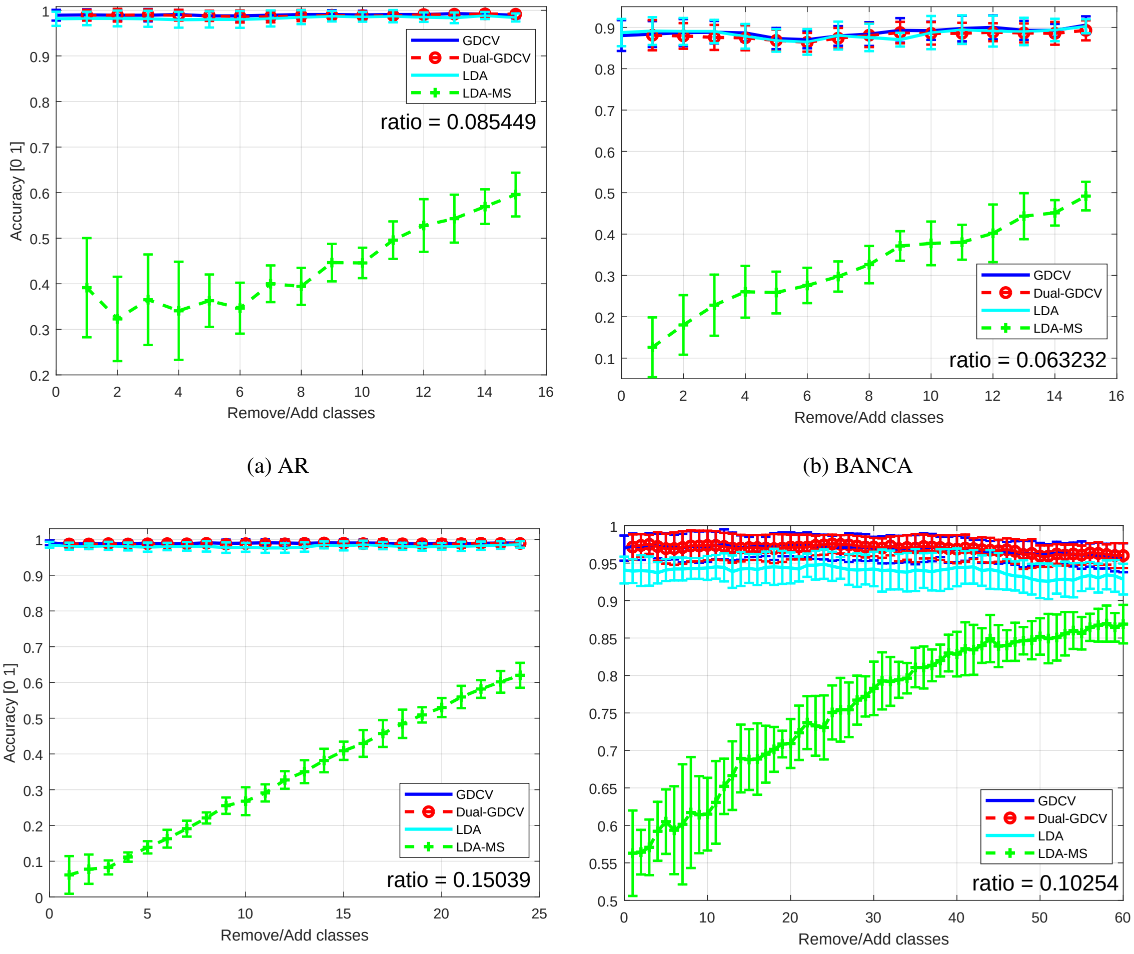
Similar conclusions can be extracted for the dual setting in Figure 11b. By increasingly adding more relevant samples and removing the least relevant ones, the performance mostly improves consistently. The exact opposite effect is achieved by reverting the process. In conclusion, this experiment proves the potential of our technique for applications where time and order are important on the updating process. It also underlines the importance for the user to provide relevant updating samples during the learning process.

4.6 On the stability of the method

In order to validate the stability of our methodology over thousands of updating steps, two experiments are presented in this section.

First, a model is generated using the full CMU-PIE dataset as initial training set. On this model, two samples per class are first removed and then added in two consecutive decremental and incremental steps. This process is repeated 10000 times. Results in Figure 12 shows that our approximation is stable over large amounts of steps.

To evaluate the influence of relevant/irrelevant samples on the stability of our method, a second test is performed using a larger dataset than the ones in previous sections, the CASPEAL-2[9] dataset with $\alpha = 0.82$. This dataset is composed of facial images with yaw angle between -45 to 45 with steps of 15 degrees to has 7 classes, where each class has 2816 images of 939 subjects with frontal view and pitch angle of -30, 0 and 30 degrees. This allows us to perform more than 3500 updating iterations on different samples randomly selected. Figure 13 depicts the results of this experiment where a diverge and drop in accuracy is observed after



(a) AR

(b) BANCA

(c) Altkom

(d) FERET

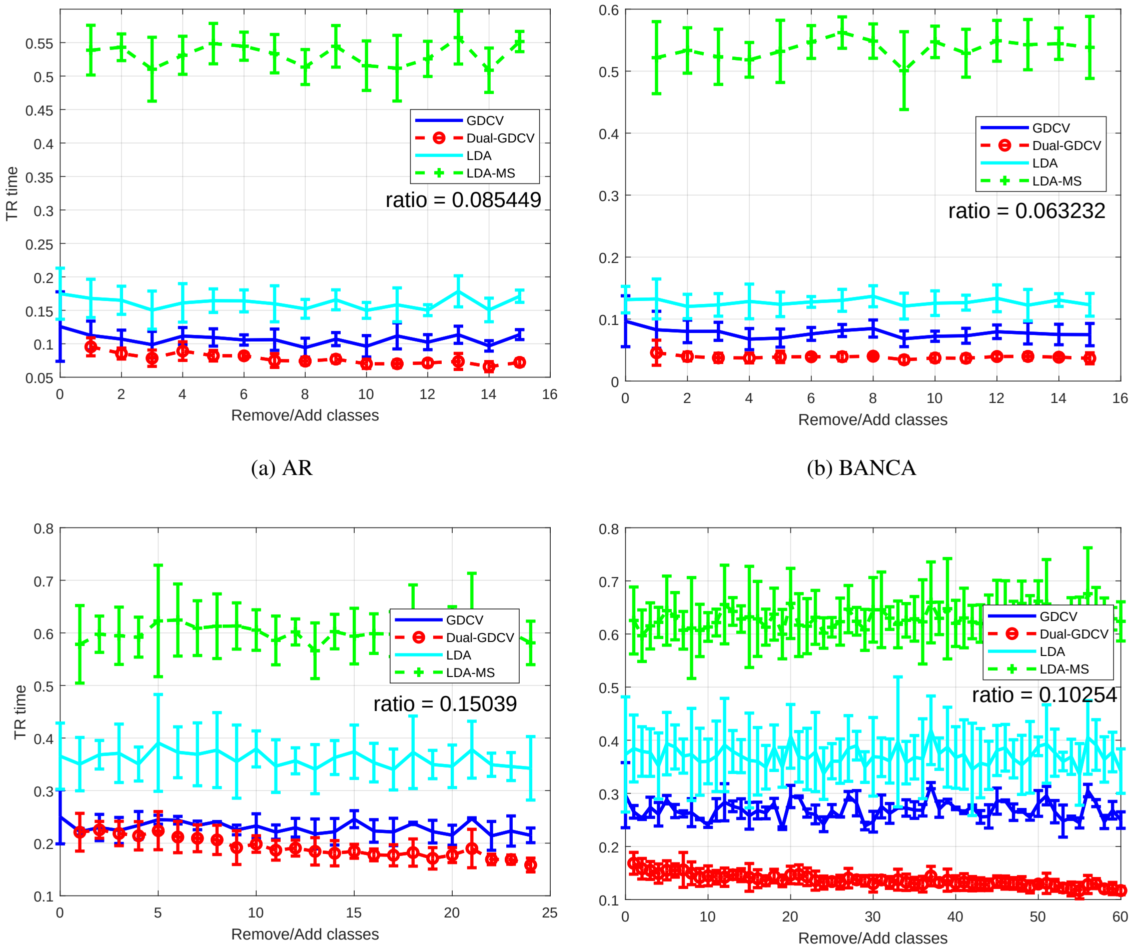
Fig. 8: Accuracy comparison for dual training at class level by using alexnet's features on AR, BANCA, Altkom and FERET datasets

15000 steps. This effect is due to the accumulation of multiple updating steps where the random allocation of samples produce small reduction of the subspace rank in the decremental functionality. As conclusion, we will recommend a every thousand steps for large-scale updating operations in order to mitigate the problem.

5 Conclusions

A dual online subspace-based learning method called Dual-Generalized Discriminative Common Vectors method (Dual-GDCV) is presented in this paper. By making use of both incremental and decremental learning, Dual-GDCV allows efficiently and simultaneously adding new data and/or removing unnecessary data to a knowledge base. Our methodology is able to update a feature-space without recalculating the full projection or accessing the previously processed training data, while retaining the previously acquired knowledge. Moreover, the presented approach does not only allow to add new data to existing classes but also to add new classes into the classification problem.

The proposed method has been validated in six standard datasets, in two scenarios -adding/removing samples and adding/removing classes- and six experiments - incremental, decremental and dual -. Dual-GDCV shows the same discriminant properties than its batch version but exhibiting a significant reduction on computational cost in all experiments and datasets. Dual-GDCV also outperforms almost all other incremental-only and dual methodologies in the state of the art, being only slightly slower than a incremental-only QR-ILDA, but with the important advantage of being able to remove samples and/or classes. This improvement



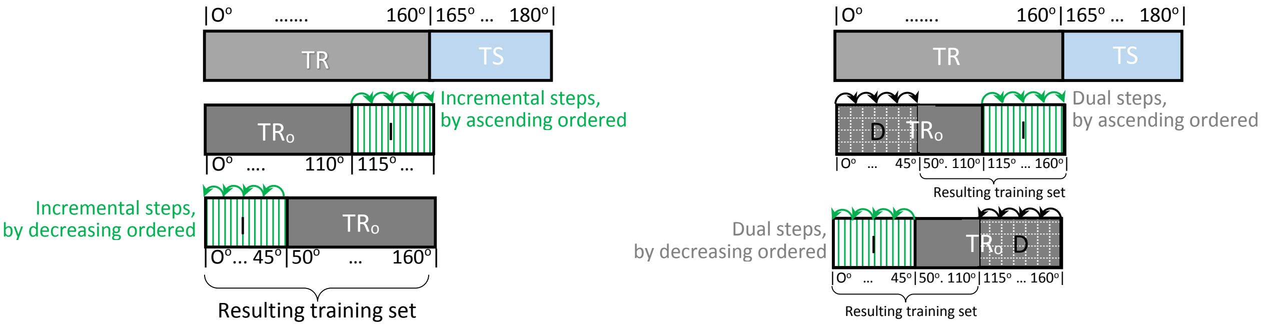
(a) AR

(b) BANCA

(c) Altkom

(d) FERET

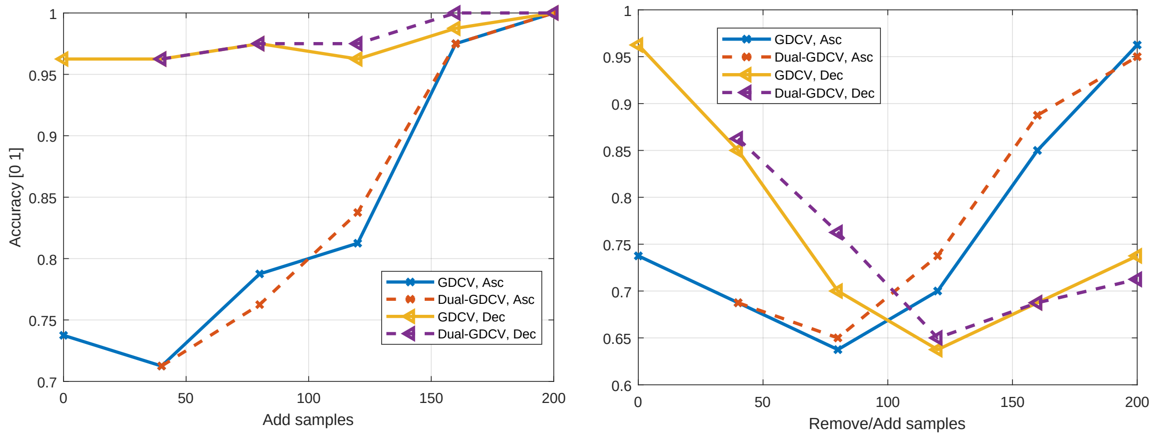
Fig. 9: CPU training time comparison for dual training at class level by using alexnet's features on AR, BANCA, Altkom and FERET datasets



(a) Incremental procedure

(b) Dual procedure

Fig. 10: Initial training (TR_o), incremental training (I), decremental training (D) and test (TS) set splits used for the ordered experiment for incremental learning in (a) and dual learning in (b) under the Coil-20 dataset.



(a) (b)

Fig. 11: Accuracy of the dual-learning on Coil-20 dataset with ordered samples on the training set with an ascend and decrease ordered.

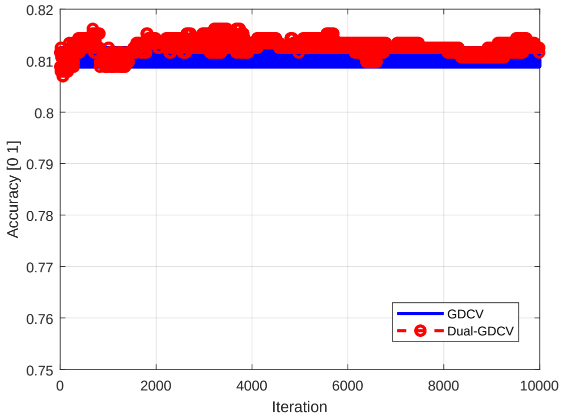


Fig. 12: Accuracy of the CMU-PIE dataset on the dual-learning buy adding and removing the same sample from the training set, to 10000 iterations.

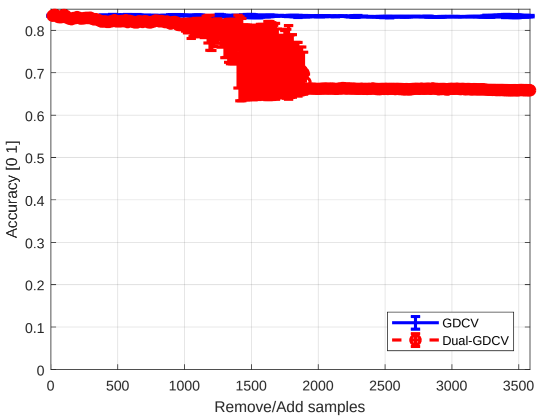


Fig. 13: Accuracy of DGDCV on the dual-learning setup for the CASPEAL-2 dataset.

regarding the state of the art is even more obvious when new classes are added and/or removed from the classification space. Since our DGDCV formulation includes IGDCV as a particular case, it provides the same good performance than IGDCV in the incremental case only, but with the added ability to remove classes and samples at the same time.

Acknowledgements This work was supported by the project TIN2014-52072-P of the Spanish Ministry of Economy, Industry and Competitiveness with FEDER funds and the CERCA Programme / Generalitat de Catalunya.

Appendix

A Decomposition of $S_w^{\widetilde{X}}$

The within-class scatter matrix of each training set is defined as

$$S_w^D = \sum_{j=1}^{c}\sum_{i=1}^{m_{D_j}} (x_j^i - \bar{u}_{D_j})(x_j^i - \bar{u}_{D_j})^T = D_c D_c^T,$$

$$S_w^I = \sum_{j=1}^{c}\sum_{i=1}^{m_{I_j}} (x_j^i - \bar{u}_{I_j})(x_j^i - \bar{u}_{I_j})^T = I_c I_c^T,$$

such that

$$
\begin{aligned}
S_w^{\widetilde{X}} &= \underbrace{\sum_{j=1}^{c}\sum_{i=1}^{m_j}(x_j^i-\bar{u}_{\widetilde{X}_j})(x_j^i-\bar{u}_{\widetilde{X}_j})^T}_{\textcircled{1}} \\
&\quad -\underbrace{\sum_{j=1}^{c}\sum_{i=1}^{m_{D_j}}(x_j^i-\bar{u}_{\widetilde{X}_j})(x_j^i-\bar{u}_{\widetilde{X}_j})^T}_{\textcircled{2}} \\
&\quad +\underbrace{\sum_{j=1}^{c}\sum_{i=1}^{m_{I_j}}(x_j^i-\bar{u}_{\widetilde{X}_j})(x_j^i-\bar{u}_{\widetilde{X}_j})^T}_{\textcircled{3}}
\end{aligned}
$$

where

$$
\begin{aligned}
\textcircled{1} &= \sum_{j=1}^{c}\sum_{i=1}^{m_j}(x_j^i-\bar{x}_j+\bar{x}_j-\bar{u}_{\widetilde{X}_j})(x_j^i-\bar{x}_j+\bar{x}_j-\bar{u}_{\widetilde{X}_j})^T \\
&= \sum_{j=1}^{c}\sum_{i=1}^{m_j}[\underbrace{(x_j^i-\bar{x}_j)(x_j^i-\bar{x}_j)^T}_{} \\
&\quad +\underbrace{(x_j^i-\bar{x}_j)(\bar{x}_j-\bar{u}_{\widetilde{X}_j})^T+(\bar{x}_j-\bar{u}_{\widetilde{X}_j})(x_j^i-\bar{x}_j)^T}_{} \\
&\quad +\underbrace{(\bar{x}_j-\bar{u}_{\widetilde{X}_j})(\bar{x}_j-\bar{u}_{\widetilde{X}_j})^T}_{}] \\
&= S_w^X+0+\sum_{j=1}^{c}m_j(\bar{x}_j-\bar{u}_{\widetilde{X}_j})(\bar{x}_j-\bar{u}_{\widetilde{X}_j})^T
\end{aligned}
$$

$$
\begin{aligned}
\textcircled{2} &= \sum_{j=1}^{c}\sum_{i=1}^{m_{D_j}}(x_j^i-\bar{u}_{D_j}+\bar{u}_{D_j}-\bar{u}_{\widetilde{X}_j})(x_j^i-\bar{u}_{D_j}+\bar{u}_{D_j}-\bar{u}_{\widetilde{X}_j})^T \\
&= \sum_{j=1}^{c}\sum_{i=1}^{m_{D_j}}[\underbrace{(x_j^i-\bar{u}_{D_j})(x_j^i-\bar{u}_{D_j})^T}_{} \\
&\quad +\underbrace{(x_j^i-\bar{u}_{D_j})(\bar{u}_{D_j}-\bar{u}_{\widetilde{X}_j})^T+(\bar{u}_{D_j}-\bar{u}_{\widetilde{X}_j})(x_j^i-\bar{u}_{D_j})^T}_{} \\
&\quad +\underbrace{(\bar{u}_{D_j}-\bar{u}_{\widetilde{X}_j})(\bar{u}_{D_j}-\bar{u}_{\widetilde{X}_j})^T}_{}] \\
&= S_w^D+0+\sum_{j=1}^{c}m_{D_j}(\bar{u}_{D_j}-\bar{u}_{\widetilde{X}_j})(\bar{u}_{D_j}-\bar{u}_{\widetilde{X}_j})^T
\end{aligned}
$$

$$
\begin{aligned}
\textcircled{3} &= \sum_{j=1}^{c}\sum_{i=1}^{m_{I_j}}(x_j^i-\bar{u}_{I_j}+\bar{u}_{I_j}-\bar{u}_{\widetilde{X}_j})(x_j^i-\bar{u}_{I_j}+\bar{u}_{I_j}-\bar{u}_{\widetilde{X}_j})^T \\
&= \sum_{j=1}^{c}\sum_{i=1}^{m_{I_j}}[\underbrace{(x_j^i-\bar{u}_{I_j})(x_j^i-\bar{u}_{I_j})^T}_{} \\
&\quad +\underbrace{(x_j^i-\bar{u}_{I_j})(\bar{u}_{I_j}-\bar{u}_{\widetilde{X}_j})^T+(\bar{u}_{I_j}-\bar{u}_{\widetilde{X}_j})(x_j^i-\bar{u}_{I_j})^T}_{} \\
&\quad +\underbrace{(\bar{u}_{I_j}-\bar{u}_{\widetilde{X}_j})(\bar{u}_{I_j}-\bar{u}_{\widetilde{X}_j})^T}_{}] \\
&= S_w^I+0+\sum_{j=1}^{c}m_{I_j}(\bar{u}_{I_j}-\bar{u}_{\widetilde{X}_j})(\bar{u}_{I_j}-\bar{u}_{\widetilde{X}_j})^T
\end{aligned}
$$

From the above expressions,

$$S_w^{\widetilde{X}} = S_w^X + S_w^I + A_X A_X^T + A_I A_I^T - S_w^D - A_D A_D^T$$

with

$$
\begin{aligned}
A_X &= [a_{X_1}\dots a_{X_c}] \quad & a_{X_j} &= \sqrt{m_{X_j}}(\bar{u}_{X_j}-\bar{u}_{\widetilde{X}_j}) \\
A_I &= [a_{I_1}\dots a_{I_c}] \quad & a_{I_j} &= \sqrt{m_{I_j}}(\bar{u}_{I_j}-\bar{u}_{\widetilde{X}_j}) \\
A_D &= [a_{D_1}\dots a_{D_c}] \quad & a_{D_j} &= \sqrt{m_{D_j}}(\bar{u}_{D_j}-\bar{u}_{\widetilde{X}_j})
\end{aligned}
$$

If the classes in I are different from the classes in X,

$$S_w^{\widetilde{X}} = S_w^X + S_w^I + A_X A_X^T - S_w^D - A_D A_D^T$$

References

1. Chandra, B., Sharma, R.K.: Fast learning in deep neural networks. Neurocomputing **171**, 1205 – 1215 (2016). DOI https://doi.org/10.1016/j.neucom.2015.07.093
2. Chu, D., Liao, L., Ng, M., Wang, X.: Incremental linear discriminant analysis: A fast algorithm and comparisons. Neural Networks and Learning Systems, IEEE Transactions on **26**(11), 2716–2735 (2015)
3. Deng, J., Dong, W., Socher, R., L.-J-Li, Li, K., Fei-Fei, L.: Imagenet: A large-scale hierarchical image database. In: Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on, pp. 248–255. IEEE (2009)
4. Diaz-Chito, K., Díaz-Villanueva, F.F.W.: Null space based image recognition using incremental eigendecomposition. In: Pattern Recognition and Image Analysis, vol. 6669, pp. 313–320 (2011)
5. Diaz-Chito, K., Ferri, F., Diaz-Villanueva, W.: Incremental generalized discriminative common vectors for image classification. Neural Networks and Learning Systems, IEEE Transactions on **26**(8), 1761–1775 (2015)
6. Duan, G., Chen, Y.: Batch-incremental principal component analysis with exact mean update. In: Image Processing (ICIP), 2011 18th IEEE International Conference on, pp. 1397–1400 (2011)
7. Ferri, F., Diaz, K., Díaz, W.: Efficient dimensionality reduction on undersampled problems through incremental discriminative common vectors. In: The 10th IEEE International Conference on Data Mining Workshops, ICDM Workshops, pp. 1159–1166 (2010)
8. Ferri, F., Diaz-Chito, K., Diaz-Villanueva, W.: Fast approximated discriminative common vectors using rank-one svd updates. In: Neural Information Processing, vol. 8228, pp. 368–375 (2013)
9. Gao, W., Cao, B., Shan, S., Chen, X., Zhou, D., Zhang, X., Zhao, D.: The cas-peal large-scale chinese face database and baseline evaluations. IEEE Transactions on Systems, Man, and Cybernetics - Part A: Systems and Humans **38**(1), 149–161 (2008)
10. Hall, P., Marshall, D., Martin, R.: Incremental eigenanalysis for classification. In: in British Machine Vision Conference, pp. 286–295 (1998)
11. Hall, P., Marshall, D., Martin, R.: Merging and splitting eigenspace models. IEEE Trans on Pattern Analysis and Machine Intelligence **22**(9), 1042–1049 (2000)
12. Jin, B., Jing, Z., Zhao, H.: Evd dualdating based online subspace learning. Mathematical Problems in Engineering **429451**, 21 (2014)
13. Kim, T., Kenneth, K., Stenger, B., Kittler, J., Cipolla, R.: Incremental linear discriminant analysis using sufficient spanning set approximations. In: Computer Vision and Pattern Recognition, 2007. CVPR '07. IEEE Conference on, pp. 1–8 (2007)
14. Kim, T., Stenger, B., Kittler, J., Cipolla, R.: Incremental linear discriminant analysis using sufficient spanning sets and its applications. International Journal of Computer Vision **91**(2), 216–232 (2011)
15. Krizhevsky, A., Sutskever, I., Hinton, G.: Imagenet classification with deep convolutional neural networks. In: Advances in Neural Information Processing Systems 25, pp. 1097–1105. Curran Associates, Inc. (2012)
16. Li, Y.: On incremental and robust subspace learning. Pattern Recognition **37**, 1509–1518 (2004)
17. Liu, L., Jiang, Y., Zhou, Z.: Least square incremental linear discriminant analysis. In: Data Mining, 2009. ICDM '09. Ninth IEEE International Conference on, pp. 298–306 (2009)
18. Lu, G., Zou, J., Wang, Y.: Incremental complete lda for face recognition. Pattern Recognition **45**(7), 2510–2521 (2012)
19. Lu, G., Zou, J., Wang, Y.: Incremental learning of discriminant common vectors for feature extraction. Applied Mathematics and Computation **218**(22), 11,269–11,278 (2012)
20. Martinez, A., Benavente, R.: The ar face database. Technical Report 24, Computer Vision Center CVC (1998)

21. Nene, S., Nayar, S., Murase, H.: Columbia object image library (coil-20). Tech. rep. (1996)
22. Ozawa, S., Pang, S., Kasabov, N.: Incremental learning of chunk data for online pattern classification systems. Neural Networks, IEEE Transactions on **19**(6), 1061–1074 (2008)
23. Pang, S., Ban, T., Kadobayashi, Y., Kasabov, N.K.: Lda merging and splitting with applications to multiagent cooperative learning and system alteration. IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics) **42**(2), 552–564 (2012)
24. Pang, S., Ozawa, S., Kasabov, N.: Incremental linear discriminant analysis for classification of data streams. IEEE Transactions on Systems, Man, and Cybernetics, Part B **35**(5), 905–914 (2005)
25. Peng, Y., Pang, S., Chen, G., Sarrafzadeh, A., Ban, T., Inoue, D.: Chunk incremental idr/qr lda learning. In: Neural Networks (IJCNN), The 2013 International Joint Conference on, pp. 1–8 (2013)
26. Phillips, J., Wechsler, H., Huang, J., Rauss, P.: The feret database and evaluation procedure for face-recognition algorithms. Image and Vision Computing **16**(5), 295–306 (1998)
27. Ren, C., Dai, D.: Incremental learning of bidirectional principal components for face recognition. Pattern Recognition **43**(1), 318 – 330 (2010)
28. Ross, D., Lim, J., Lin, R., Yang, M.: Incremental learning for robust visual tracking. International Journal of Computer Vision **77**(1-3), 125–141 (2008)
29. Sim, T., Baker, S., Bsat, M.: The CMU pose, illumination, and expression (PIE) database. In: Proceedings of the 5th International Conference on Automatic Face and Gesture Recognition (2002)
30. Uray, M., Skocaj, D., Roth, P.M., Bischof, H., Leonardis, A.: Incremental lda learning by combining reconstructive and discriminative approaches. In: Proceedings of the British Machine Vision Conference, pp. 44.1–44.10 (2007)
31. Zeng, X., Li, G.: Covariance free incremental principal component analysis with exact mean update. Journal of Computational Information Systems **5**(16), 181–192 (2013)
32. Zhao, H., Yuen, P.: Incremental linear discriminant analysis for face recognition. Systems, Man, and Cybernetics, Part B, IEEE Transactions on **38**(1), 210–221 (2008)
33. Zhao, H., Yuen, P., Kwok, J.T.: A novel incremental principal component analysis and its application for face recognition. IEEE Transactions on Systems, Man, and Cybernetics (Part B) **36**, 873–886 (2006)
34. Zheng, W., Tang, X.: Fast algorithm for updating the discriminant vectors of dual-space lda. IEEE Transactions on Information Forensics and Security **4**(3), 418–427 (2009)