Decremental Generalized Discriminative Common Vectors applied to images classification

Katerine Diaz-Chito[a,*], Jesús Martínez del Rincón[b], Aura Hernández-Sabaté[a]

[a] *Centre de Visio per Computador, Universitat Autònoma de Barcelona, Spain*
[b] *Centre for Secure Information Technologies, Queen's University Belfast, UK*

Abstract

In this paper, a novel decremental subspace-based learning method called Decremental Generalized Discriminative Common Vectors method (DGDCV) is presented. The method makes use of the concept of decremental learning, which we introduce in the field of supervised feature extraction and classification. By efficiently removing unnecessary data and/or classes for a knowledge base, our methodology is able to update the model without recalculating the full projection or accessing to the previously processed training data, while retaining the previously acquired knowledge. The proposed method has been validated in 6 standard face recognition datasets, showing a considerable computational gain without compromising the accuracy of the model.

Keywords: Decremental learning, Generalized Discriminative Common Vectors, Feature extraction, Linear subspace methods, Classification

1. Introduction

Feature extraction methods are one of the most crucial steps in pattern recognition. Extracting a relevant and discriminant set of features from raw data facilitates the classification and recognition tasks to be performed by the subsequent classifier, significantly improving the overall performance of the sys-

*Corresponding author
Email addresses: kdiaz@cvc.uab.es (Katerine Diaz-Chito), j.martinez-del-rincon@qub.ac.uk (Jesús Martínez del Rincón), aura@cvc.uab.cat (Aura Hernández-Sabaté)

tem. In the last decades, vast amount of feature extraction techniques have been proposed. However, many of them are specific to a particular domain and based on a costly hand-crafted design process [1, 2]. In contrast, two more general feature extraction strategies have emerged as more general and effective methodologies: those based on deep neural networks [3] and those based on subspaces [4]. While the first one has shown state of the art performance for large datasets, subspace based methods perform better when training data is more restricted.

In this context, incremental methods [5, 6] are particularly interesting due to their properties to learn and evolve without requiring full access to the initial training information, which may be lost or under restricted access. Furthermore, updating the system with new information without having to retrain this from scratch leads to convenient trade-offs between computational cost and performance as well as space complexity. Incremental learning is specially interesting in automatic feature extraction given that some of those methodologies require days or weeks to be trained [7]. In addition, for many practical applications such as object tracking [8], image classification [9], stream processing [6], or face recognition [10], a complete set of training samples is usually not known in advance but generally provided little by little, which makes incremental learning best suited for the task.

However, incremental methods only address updating the system by adding new information, but they do not considered updating the learned model by removing wrong, misleading or obsolete information previously introduced. In this sense, we define decremental learning as an online process that allows removing samples, classes or any initial information from a previously trained model. We postulate decremental learning can be as important as incremental learning 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 by removing his/her corresponding class when a user leaves the organization.

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 subspace-based methods as part of hierarchical classification architecture, such as decision trees, will also benefit from a decremental methods where more specific subspace can be derived from a global. Finally, generalizing the decrementing of a learned model can be used for an efficient leave-m-out cross validation [11] of a classification pipeline containing a subspace-based method.

In this paper, we propose for the first time a decremental subspace-based learning method for supervised problems. Our approach is able to update the system by deleting unnecessary old data, while retaining the previously acquired knowledge, without accessing the previously processed training data. This is not only a more cost-effective approach than batch methodologies, but also allows reusing models and projections when the original training data has been lost or it is not accessible. Furthermore, it facilitates maintaining an effective and consistent subspace projection, without repeating the lengthy process of parameter tuning for every update, which takes better advantage of an initial parameter optimization process. Our novel decremental framework, called Decremental GDCV approach (DGDCV), is constructed on the basis of the Generalized Discriminative Common Vectors method (GDCV) [9] for being particularly appealing due to good performance, flexibility of implementation and capacity for dealing with the Small Sample Size (SSS) case, a common problem in applications such as computer vision and biometrics. The proposed implementation allows decrementing both full classes and individual samples, as well as any combination of them, depending on the users requirements and the given application.

The remainder of the paper is structured as follows. Section 2 presents

an overview of the related work. Section 3 describes the problem statement. Section 4 briefly introduces the batch GDCV method. Section 5 presents the Decremental GDCV, which it is the main contribution of this paper. Section 6 describes the validation and presents the results of the analysis of the proposed approach. Finally, Section 7 presents the main conclusions and some ideas about further research.

2. Related Work

Among subspace-based methods, several incremental feature extraction techniques have been proposed, such as those based on Principal Components Analysis (PCA) [6, 12, 13, 14], Linear Discriminative Analysis (LDA) [5, 10, 15, 16] and Discriminative Common Vector (DCV) [9, 17, 18, 19] methods. While incremental learning has been extensively studied in the literature, few research has been done in decremental learning or sample removal.

The initial approach to the problem was the use of instance reduction algorithms, IRA [20], and it aimed to reduce the initial training set size without affecting the overall performance through instance selection. These algorithms focus on detecting those samples in the set that are not relevant or do not contain information before training, so they are not online learning approaches. If an initially relevant sample or class is then declared obsolete, the system will need to be fully retrained without any computational benefit.

Relevant steps towards decremental systems were presented in incremental and decremental Support Vector Machines (SVM) [21] and logistic regression [22], although they only allow the removal of one sample at a time. An extended version [23] was further developed to allow the efficient removal of multiple samples. However, their decremental learning modifications are embedded into the classifier and their internal optimization processes. Since they do not involve the feature extraction process, this limits the decremental update to only a part of the recognition pipeline, obtaining overall suboptimal results. It must be noticed how these approaches have been tested in very low dimensional data

(e.g. 21 dimensions, [23]) or data that has been proved to work directly on SVM without feature extraction or preprocessing (e.g. LIBSVM datasets, [22]). These simple pipelines will struggle to solve difficult cases, such as complex and high dimensional problems and/or SSS case [24].

Regarding feature extraction and subspace learning methodologies, only two research works can be found related to decremental learning. Hall et al. [25] presented the Merging and Splitting EigenSpaces (MSES) method. This method permits simultaneous arbitrary addition and deletion operations, by transforming the eigen-value/vector decomposition (EVD) of the total scatter matrix. Jin et al. [26] introduces a incremental/decremental version of PCA. This EVD Dualdating (EVDD) method provides similar functionalities to the previous method but transforming the EVD of the total scatter matrix into a single value decomposition (SVD) updating problem. Both approaches are unsupervised and under the same limitations than PCA, i.e. the extracted subspace and the corresponding extracted features are not necessarily invariant and suited for classification purposes.

3. Problem Statement

Let $X = [x_j^1 \dots x_j^{m_j}] \in \mathbb{R}^{d \times M}$, $j = 1, \dots, c$ be a data matrix of $M = \sum_{j=1}^{c} m_j$ samples belonging to c classes, where each class j has m_j samples. Each of the training samples x_j^i, $i = 1, \dots, m_j$ is therefore a d-dimensional column vector. Subspace-based learning methods aim to find a transformation or projection W from the d-dimensional input space, $\mathbb{R}^d$, into another space where the relevant information is easily separable into the different classes. In order to obtain the optimal projection W to the new subspace, the bases of the subspace, U, should be first calculated. These bases are obtained by solving the eigenproblem of the within-scatter matrix, S_w^X, of the given training data X. This scatter matrix is defined as,

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

where $\overline{x}_j$ is the average of the samples in the j^{th} class. Matrix notation can be used to simplify these mathematical expressions so that $X_c = X - \overline{X}$ is the centered matrix, where $\overline{X} = [\overline{x}_1 \dots \overline{x}_c]$ is the matrix comprising all the class-averages.

The eigendecomposition 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. U_r and U_o are bases of two complementary subspaces, the range space -containing the eigenvectors with $\lambda_i > 0$- and the null space -containing the eigenvectors with $\lambda_i = 0$- of S_w^X, respectively. Notice that $\lambda_i = 0$ for all $i > r$, being r the range of matrix S_w^X. Theses subspace can be reformulated as the *restricted range subspaces*, $\mathcal{R}_r(S_w^X)$, and the *extended null subspaces*, $\mathcal{N}_e(S_w^X)$, respectively. In those particular cases where the number of input samples is limited with respect to the dimensionality, i.e. $d > M$, the eigenproblem is unsolvable, problem known as the Small Sample Size case or SSS. For those SSS cases, the smaller matrix $X_c^T X_c$ can be used instead of $X_c X_c^T$ to calculate U_r [27].

Depending of the particular subspace-based technique, the sought projection W will have a different mathematical relation with U, U_r and U_o, giving a different resulting subspace.

As described in the introduction, our aim is to investigate the role of decremental learning on an initially calculated projection W and the corresponding subspace basis U_r. In order to keep a consistent notation throughout the document, for any variable A, its updated version after deleting a class is denoted by $\widetilde{A}$. For example, the data matrix X is changed to $\widetilde{X}$ after delete an old class. In current methods, when one or several old training classes need to be deleted the eigenproblem should be recalculated. We denote by $X = [\widetilde{X} \;\; X_D] \in \mathbb{R}^{d \times M}$ the decomposition of the initial training set in the new training one, $\widetilde{X} \in \mathbb{R}^{d \times (M - m_j)}$, and the old training classes, $X_D \in \mathbb{R}^{d \times m_j}$.

This leads to issues regarding spatial complexity, since X should be accessible at any time, and computational complexity, since the EVD problem should be solved from scratch every time, even if a single sample is due to be removed. Furthermore, as the dataset becomes smaller, the SSS case will become more prominent, leading to inconsistencies in the solution. The challenge then is to obtain the subspace, $\widetilde{U_r}$, associated to $\widetilde{X}$ without explicitly having $\widetilde{X}$ and $S_w^{\widetilde{X}}$.

4. Generalized Discriminative Common Vectors

The Generalized Discriminant Common Vector (GDCV) method [9], also referred to as Rough Common Vector, RCV [28], constitutes a different way to overcome the singularity problem in LDA. It consists of finding a projection matrix, $W \in \mathbb{R}^{d\times(c-1)}$, that maximizes the projected between-class scatter, subject to the fact that the subspace generated by W belongs to the $\mathcal{N}_e(S_w^X)$.

The singularity is avoided by extending the null space U_o to include not only null directions or basis vectors, i.e. $\lambda_i = 0$, but also with almost null directions, $\lambda_i \approx 0$. This extension implies restricting the corresponding range space U_r to the highest directions, according to α parameter

$$\alpha = 1 - \frac{tr(U_\alpha^T S_w^X U_\alpha)}{tr(S_w^X)} \tag{2}$$

where U_α is the resulting restricted basis for a $\mathcal{R}_r(S_w^X)$, where some almost null directions have been removed. The parameter α takes values in the interval $[0, 1]$. When $\alpha = 0$, $U_\alpha = U_r$. The scattering added by the extension to the null space can be measured as $tr(U_\alpha^T S_w^X U_\alpha)$. This quantity is zero when no directions are removed from U_α and increases as more and more important directions disappear by Eq. 2. For different particular values of $\alpha < 1$, different projections can be obtained with different levels of preserved variability, so that U_α spans to the *restricted range* of S_w^X according to α to a new value of $r_\alpha \leq (r-1)$. Note that decreasing variability in the *restricted range* space directly results in increasing variability in the corresponding *extended null* space.

The projection basis fulfilling the above conditions for a given value of α can be obtained through the eigendecomposition of S_w^X.

Figure 1 presents the main subspaces involved in the GDCV method. The procedure to obtain a projection basis and the corresponding generalized common vectors, and the time complexity corresponding to each of its steps are presented in algorithm 1.

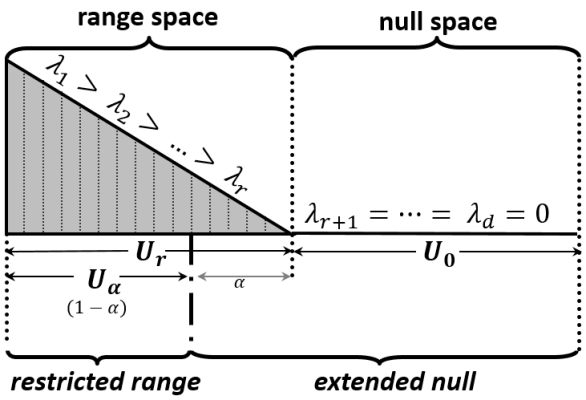


Figure 1: Main subspaces involved in the GDCV method.

Algorithm 1. *GDCV Algorithm*

Parameter: $\alpha,\ 0 \leq \alpha < 1$

Input: $X \in \mathbb{R}^{d \times M}$, $M = \sum_{j=1}^{c} m_j$

Output: $U_\alpha \in \mathbb{R}^{d \times r}$, $\Lambda_\alpha \in \mathbb{R}^{r \times r}$, $\overline{X} \in \mathbb{R}^{d \times c}$

Method:

1. Compute $S_w^X = X_c X_c^T$ // $O(d^2 M)$
   if $d > M$ use the matrix $X_c^T X_c$ // $O(dM^2)$
2. Compute U and Λ by the EVD of S_w^X // $O(d^3)$
   if $d > M$ use the EVD of the matrix $X_c^T X_c$ // $O(M^3 + dMr)$
   and extract the eigenvectors and eigenvalues in Λ according to α
3. Compute the generalized common vector as $x_{cv}^j = \overline{x}_j - U_\alpha U_\alpha^T \overline{x}_j$ // $O(drc)$
4. Define $X^{com} = [x_{cv}^1 \dots x_{cv}^c]$ and let X_c^{com} be its centered version with regard to the mean $\overline{x}_{com} = (1/c) \sum_{j=1}^{c} x_{cv}^j$ // $O(d(c-1))$
5. Compute the projection matrix such that $W = orth(X_c^{com}) \in \mathbb{R}^{d \times (c1)}$ // $O(d(c-1)^2)$
6. Obtain the discriminative common vectors as $W^T \overline{x}_j$.

   To test a new sample, x_{test}, project it as $W^T x_{test}$ and then the label is allocated from the minimum distance between the projected sample and the discriminative common vectors.

The computational complexity of the GDCV method is $O(d^2 M + d^3)$, when $d \leq M$. In the SSS case, $(d > M)$, the computational complexity is $O(dM^2 + M^3 + dMr)$. It is worthy to note that steps 3-6 in the algorithm 1 have a complexity of $O(drc + dc^2)$, independently of the ratio between d and M, and their impact in the total cost, which is dominated by the costs in steps 1 and 2, is almost negligible. Regarding the space complexity it is $O(min(d, M)^2)$.

5. Decremental Generalized Discriminative Common Vectors

The key idea of DGDCV algorithm is to obtain the feature extraction model, $\widetilde{U_\alpha}$, associated to $\widetilde{X}$ by accessing and processing only X_D and the current model U_α, and without explicitly having access to $\widetilde{X}$ and $S_w^{\widetilde{X}}$. Figure 2 illustrates the

subspaces involved when updating GDCV models by deleting one or several classes or samples.

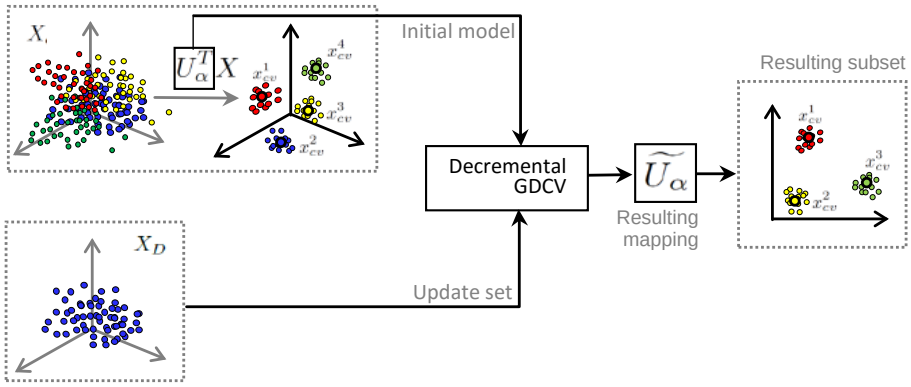


Figure 2: Main subspaces involved in the DGDCV approach. U_α spans to the restricted range of S_w^X, X_D are the data to be removed, and $\widetilde{U_\alpha}$ is the base than spans to the new restricted range.

To achieve this goal, we assume the decomposition of the within-class scatter matrix as the sum of its component following a similar reasoning and justification as in [25, 9],

$$S_w^X = S_w^{\widetilde{X}} + S_w^{X_D} \tag{3}$$

Thus, $S_w^{\widetilde{X}}$ can be estimated as:

$$S_w^{\widetilde{X}} = S_w^X - S_w^{X_D} \approx U_\alpha \Lambda_\alpha U_\alpha^T - X_{D_c} X_{D_c}^T \tag{4}$$

where $X_{D_c} = X_D - \overline{X_D}$ is the centered data matrix of X_D with respect to their own average $\overline{X_D}$.

A basis that generates the range space of the remaining data set, $S_w^{\widetilde{X}}$, can be approximated as:

$$\widetilde{U} \approx [U_\alpha \;\; V]R \tag{5}$$

where V is orthogonal to U_α, such as $V = orth(X_{D_c} - U_\alpha U_\alpha^T X_{D_c}) \in \mathbb{R}^{d \times r_D}$. $orth()$ function refers to any orthonormalization procedure - a Graham-Schmidt

Orthonormalization (GSO) is used in our case-, r_D to the range of $S_w^{X_D}$, and R is a rotation matrix that controls the dimensionality of the $\mathcal{R}_r(S_w^{\widetilde{X}})$.

By substituting 5 in the decomposition of 4

$$EVD(S_w^{\widetilde{X}}) : S_w^{\widetilde{X}} = \widetilde{U}\ \widetilde{\Lambda}\widetilde{U}^T \tag{6}$$

and projecting these scatters onto $\mathcal{R}_r(S_w^{\widetilde{X}})$ as $[U_\alpha \ \ V]^T(\cdot)[U_\alpha \ \ V]$, we obtain

$$R\ \widetilde{\Lambda}\ R^T = M_\alpha : EVD(M_\alpha) \tag{7}$$

where

$$M_\alpha = \begin{bmatrix} \Lambda_\alpha & 0 \\ 0 & 0 \end{bmatrix} - [U_\alpha \ \ V]^T X_{D_c} X_{D_c}^T [U_\alpha \ \ V] \tag{8}$$

From the eigendecomposition of M_α, we can extract the eigenvectors, R_β, as the column vectors in R corresponding to the largest eigenvalues, $\widetilde{\Lambda_\alpha}$, such that $tr(\widetilde{\Lambda_\alpha}) = \beta \cdot tr(\widetilde{\Lambda})$.

$$\beta = \left\| \frac{diag(\widetilde{\Lambda})}{diag(\Lambda)} \right\| \cdot (1 + \alpha) \tag{9}$$

Note that the factor β is defined with regard to M_α, 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). Consequently, the quality of the approximation will depend on how representative the delete class is in comparison to the whole of the training set. The final approximations for the updated extended null space projection with parameter α can be accurately written as

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

The DGDCV algorithm is presented in the algorithm 2 along with the asymptotic cost corresponding to each of its steps.

Algorithm 2. *DGDCV Algorithm*

Parameter: $\alpha,\ 0 \leq \alpha < 1$

Input: $X_D \in \mathbb{R}^{d \times m_j}$, α

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: $\widetilde{U_\alpha} \in \mathbb{R}^{d \times \widetilde{r}}$, $\widetilde{\Lambda_\alpha} \in \mathbb{R}^{\widetilde{r} \times \widetilde{r}}$, $\overline{X} \in \mathbb{R}^{d \times \widetilde{c}}$

Method:

1. Compute X_D regarding its average to obtain X_{D_c} // $O(dm_j)$
2. Compute V as $V = orth(X_{D_c} - U_\alpha U_\alpha^T X_{D_c}) \in \mathbb{R}^{d \times r_D}$ // $O(dm_j r + dm_j^2)$
3. Build M_α using Eq. 8 // $O(dm_j(r + r_D))$
4. Eigendecompose M_α in $R\widetilde{\Lambda}R^T$ // $O((r + r_D)^3)$
   and obtain the eigenvalues $\widetilde{\Lambda}_\alpha = \Lambda_\beta$ within $\widetilde{\Lambda}$ according to β Eq. 9
5. Compute the generalized common vector as
   $x_{cv}^j = \overline{x}_j - \widetilde{U_\alpha}\widetilde{U_\alpha}^T \overline{x}_j \in \mathbb{R}^{d \times \widetilde{c}}$ // $O(d\widetilde{r}\widetilde{c})$
6. Steps 4-6 of the algorithm 1.

5.1. Computational and space complexity

In this subsection, we estimate the computational complexities of DGDCV when an obsolete class is deleted from the existing training data. Table 1 shows the comparison between the DGDCV approach and the batch method.

Step	1	2	3	4
DGDCV	$O(dm_j)$	$O(dm_j(r + m_j))$	$O(dm_j(r + r_D))$	$O((r + r_D)^3)$
GDCV	$O(d^2M)$	–	–	$O(d^3)$
GDCV (SSS case)	$O(dM^2)$	–	–	$O(M^3 + dMr)$

Table 1: Main computational complexity for DGDCV and GDCV.

The asymptotic cost of the DGDCV is dominated by $O(dm_j^2 + (r + r_D)^3)$. In the case of the batch algorithm the complexity is dominated by $O(d^2M + d^3)$, when $d \leq M$, and $O(dM^2 + M^3)$, when $d > M$. We can seen that the DGDCV approach is more efficient than the batch algorithm in both cases since $m_j \ll M$, and $(r + r_D)^3 < min(d^3, M^3)$. Obviously, the closer the value of the number of removed samples is to the size of the initial training set, the smaller is the

computational gain by using a decremental approach since previous disparities are not fulfilled. If almost all the 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 the space complexity, the batch method presents a $O(min(d, M)^2)$ and the decremental algorithm has a $O((r + r_D)^2)$, which is also a significant improvement given that $(r + r_D)^2 < min(d^2, M^2)$.

6. Experiments and Results

6.1. Experimental setup

To demonstrate the advantages of the DGDCV approach to delete existing classes or samples from the initial training data of a classification problem, we selected six facial recognition datasets to validate our approach. The choice of face recognition as classification task has been extensively used in incremental learning approaches based on subspaces such as [29, 30, 31]. As classifier, a simple 1-Nearest Neighbors classifier using Euclidean distance between the training discriminative common vectors and the test samples projected into the discriminant subspace is employed. 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. A more complex and powerful classifier could hide or compensate the adequacy of the resulting subspace.

Figure 3 illustrates the datasets used, with a sample of 8 images per dataset on the top, and a table with their main characteristics at the bottom. All images were normalized to 40×40. For each dataset, the Training Set (TR) is composed by the 70% of the first samples of each class, and the remaining 30% is used as Test Set (TS), The α parameter was empirically optimised so that the batch GDCV algorithm provided the best accuracy result when using all samples and classes.

Name	α	c	m_j	TR	TS	c_j
AR [32]	0.02	50	14	10	4	[50–17]
BANCA [29]	0.20	52	10	7	3	[52–18]
CMU-PIE [33]	0.04	68	56	40	16	[68–23]
Altkom [29]	0.01	80	15	11	4	[80–27]
FERET [34]	0.01	200	4	3	1	[200–67]
MPEG [30]	0.18	635	5	4	1	[635–212]

Figure 3: Datasets used in validation along with their corresponding details. α is the added scatter to the null space of S_w^X. c is the number of classes. m_j is the total number of samples per class. TR and TS are the number of samples per class in the training and test set, respectively. c_j is the range of remaining classes from the training set in our experiments.

All algorithms have been implemented in Matlab and run on a computer with a Intel(R) Core(TM) i7-4790 CPU @ 3.60GHz, 3601 Mhz, and 32-GB RAM.

6.1.1. Decrement by class

In a first setup, an initial model is decremented by removing a existing class, i.e. all samples belonging to that class, at each decremental step. This is the most interesting setup since it is closer to a real application where a class may stop being relevant for a given classification problem. In all experiments under this setup, the initial model is obtained using the corresponding batch algorithm and then one class at a time is deleted, until only 1/3 of the total number of classes remains. The range of these values is represented by c_j in Fig. 3.

First, a comparison is performed to show the discriminant properties of the GDCV method in both cases, when $d \leq M$ and $d > M$. This method is compared against the well-known LDA/GSVD [35] and LDA/QR [36] methods. This will allow us to justify our choice of GDCV as a base for our decremental algorithm.

Then the proposed DGDCV algorithm is validated, both in terms of the accuracy of its approximation and the decrease in computational time regarding the batch GDCV algorithm. In this validation, two different empirical scenarios have been considered to find out if the accuracy and performance of our approximation depend not only on the number of classes removed and the number of decremental steps, but also on the size of those classes in terms of number of samples. In the first scenario all samples per class are used to create the model, i.e TR=TR (TR = 1 from now on). In the second one, classification models are created using the half and the quarter of the total number of samples in each class, i.e. TR = 0.5*TR and TR = 0.25*TR (TR = 0.5 and 0.25 from now on). The chosen training samples are randomly selected as in other incremental setups [37].

6.1.2. Decrement by sample

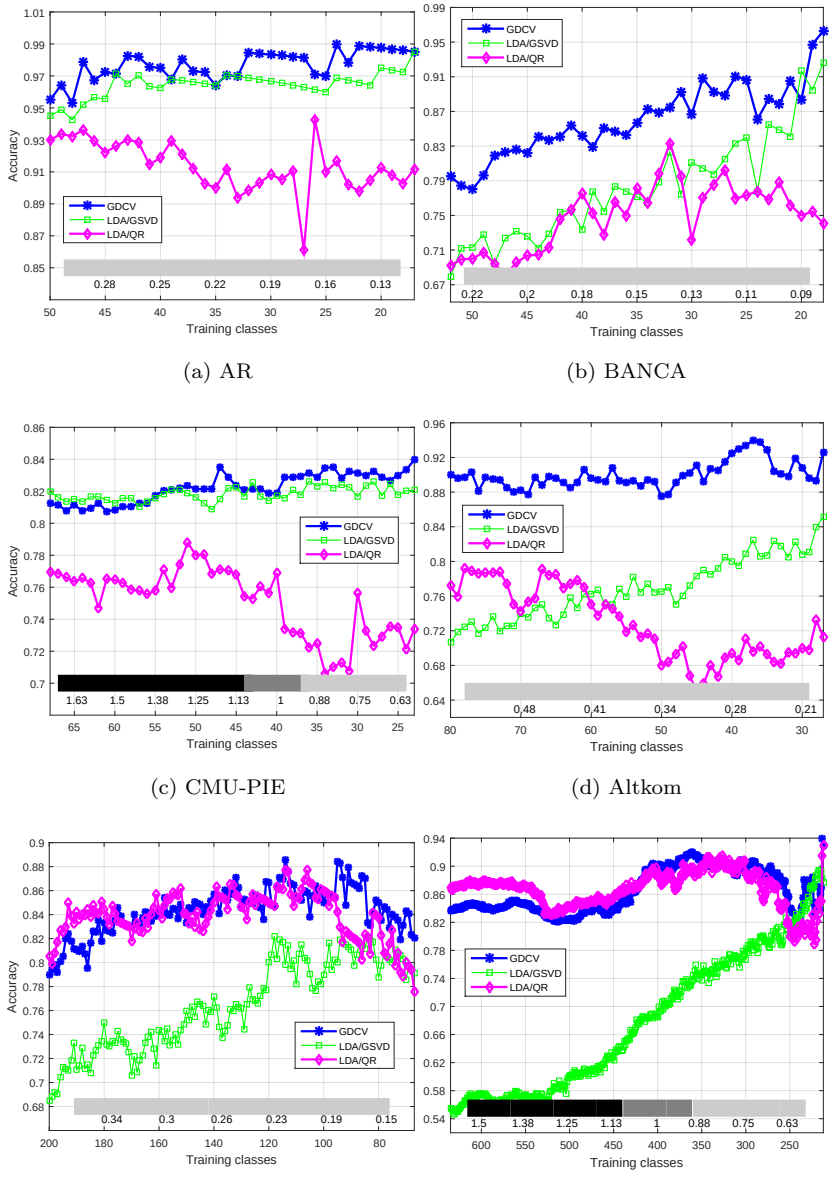
As a final experiment, we validate our approach when individual samples, rather than full classes, are decremented. In this setup, an initial model is obtained from the full training set using the corresponding batch algorithm. Then, in each iteration, a samples per class is removed until only the required minimum of two samples per class remains. Experiments are performed for the AR, BANCA, CMU-PIE and Altkom datasets. FERET and MPEG could not used due to their extremely small number of samples per class, which did not allow for even a decrement by sample iteration.

6.2. Results and analysis

6.2.1. Decrement by class

Figure 4 shows the accuracy rate for the three batch methods GDCV [9], LDA/GSVD [35] and LDA/QR [36] over a decremental number of classes, where one class of the training data is deleted at each iteration. The greyscale bar in the figure represents the ratio between the number of samples in TR and the dimension of the original space, where $(M/d) > 1$ is shown in black and $(M/d) < 1$ is shown in light gray. This allows to compare the performance of

the algorithms in case of SSS (light gray) or not. Results show how GDCV gives



(a) AR (b) BANCA

(c) CMU-PIE (d) Altkom

(e) FERET (f) MPEG

Figure 4: Accuracy rate of batch methods GDCV [9], LDA/GSVD [35] and LDA/QR [36] over a decremental number of classes.

consistently the best or almost best performance and more stable discriminants properties in all datasets and cases, which justify the use of GDCV as baseline method in our decremental approach.

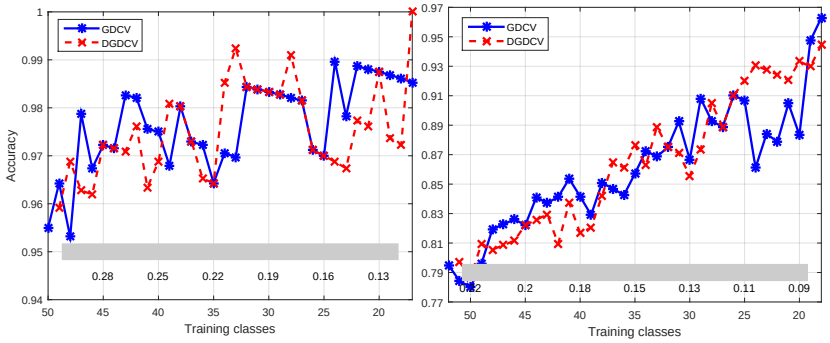
Figure. 5 shows the comparative performance between DGDCV and GDCV for the first scenario, TR = 1. We can observe how our DGDCV approach exhibits a stable performance regarding the batch method and the effect of the approximation can be considered negligible, since the difference is small (see Table 2) and no divergence is shown. It is also noticeable how the decremental method shows a more continuous and smooth performance, which seems to indicate a better resiliance against local maxima and minima and spikes in performance that may happen in the batch method, as reflected in Fig.5.b at 25-20 classes, Fig.5.d at 40-35 classes and Fig.5.f at 250-212 classes. This continuity or smoothness in performance was measured by adjusting a piecewise polynomial spline to each graph in Figure. 5 and measuring the RMSE bewteen each method's performance and its spline. The average error for DGDCV is 5.28e-4, smaller than the GDCV with 7.74e-4.

Table 2 summarizes the Root Mean Squares Error, RMSE, and the Relative Error, ER, between the DGCV and its batch method. These relative errors are computed according to:

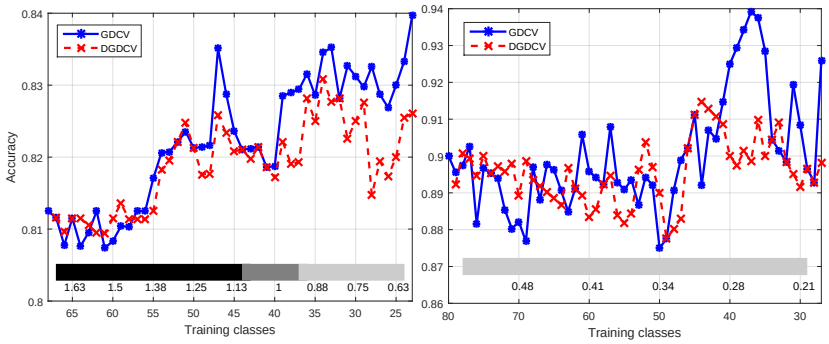
$$RMSE = \sqrt{\sum_{i=1}^{n} \frac{(Acc_{i_{update}} - Acc_{i_{batch}})^2}{n}} * 100$$

$$RE = \sum_{i=1}^{n} \frac{Acc_{i_{update}} - Acc_{i_{batch}}}{Acc_{i_{batch}}} * \frac{100}{n}$$

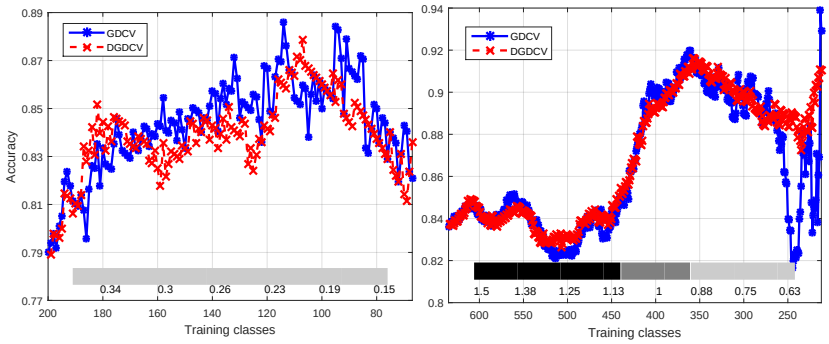
The second scenario for accuracy comparison between DGDCV and GDCV is shown in Figure 6, for different sizes of training sets and class size TR = 1, TR = 0.5 and TR = 0.25. As expected in any machine learning algorithm, the lower the number of samples, the lower performance. Similarly to the previous scenario, DGDCV shows a similar or better performance than its batch version in all cases. No differences were observed in the comparative behavior



(a) AR (b) BANCA



(c) CMU-PIE (d) Altkom



(e) FERET (f) MPEG

Figure 5: Comparison in terms of accuracy between DGDCV and GDCV.

regarding the number of samples per class, although a better resistance against degeneration is shown by our DGDCV.

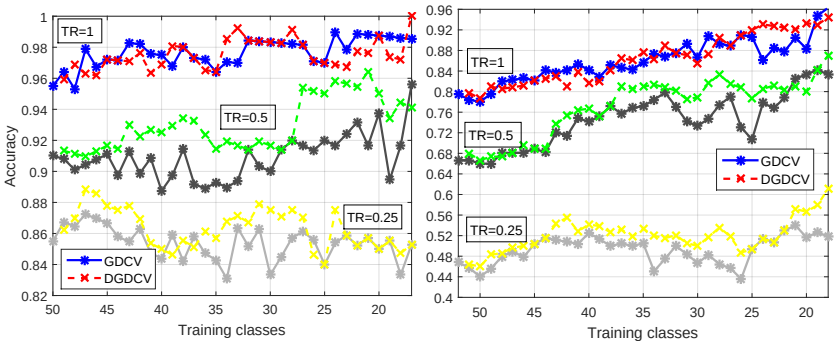
Dataset	RMSE %	ER %
AR [32]	1.0 ± 0.7	-0.2 ± 1.0
BANCA [29]	2.4 ± 1.5	0.3 ± 2.7
CMU-PIE [33]	0.6 ± 0.4	-0.4 ± 0.6
Altkom [29]	1.5 ± 1.0	-0.4 ± 1.6
FERET [34]	1.4 ± 0.9	-0.5 ± 1.6
MPEG v1 [30]	1.5 ± 1.3	0.5 ± 1.7

Table 2: Relative RMSE and ER between DGCV and GDCV, for each dataset and TR = 1.

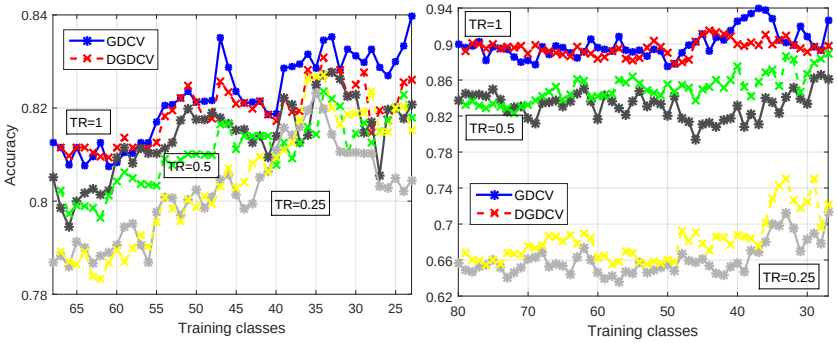
The computational cost for both decremental and batch methods is depicted in Figure 7. GDCV shows an almost quadratic behavior decreasing with the number of samples and classes with inflexion points to a linear model where the SSS case stops being relevant. On the contrary, DGDCV show a big computational gain regarding its batch version and it exhibits a much milder linear tendency, almost constant in the majority of the cases. The explanation for this linear behavior is directly due to the fact that the cost term dm_j^2, in $(dm_j^2 + (r + r_D)^3)$, behaves as dm_j due to the sublinear decreasing of the ranks since $(r + r_D)^3 \ll dm_j^2$. Please note that the initial cost to generate the initial model to be decremented is not considered in either method.

6.2.2. Decrement by sample

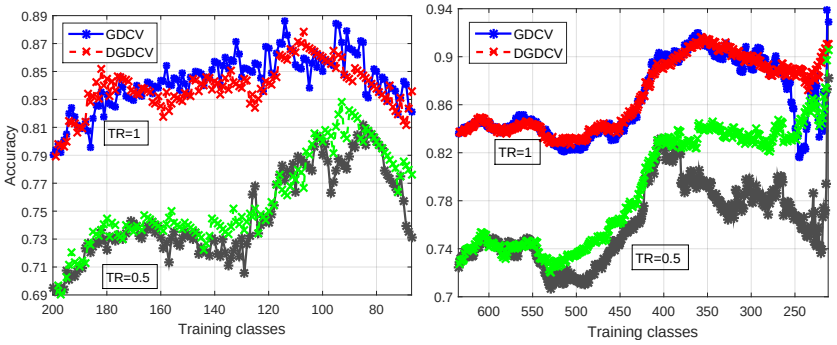
Finally, Figures 8 and 9 show the accuracy rates and the CPU time of the methods when individual samples per class are deleted in each iteration. As expected, we can observe how having less training samples per class will reduce the recognition rate in both batch and decremental version. However, DGDCV reduces the module degeneration and poor generalisation when few sample are available per class, exhibited in the batch version. Our decremental DGDCV seems to keep relevant information about the class after each decremental iteration in spite of removing the sample. Thus, the model generated using DGDCV provides the same or better discriminative properties than the batch model. Regarding the computational cost, DGDCV also shows a significative gain regarding the batch approach as in all previous experiments.



(a) AR (b) BANCA

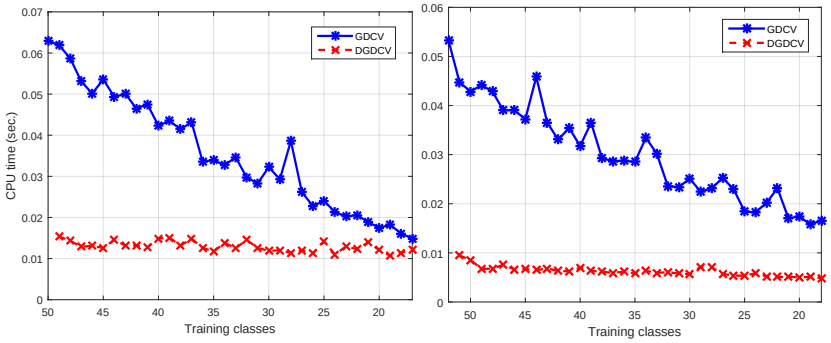


(c) CMU-PIE (d) Altkom

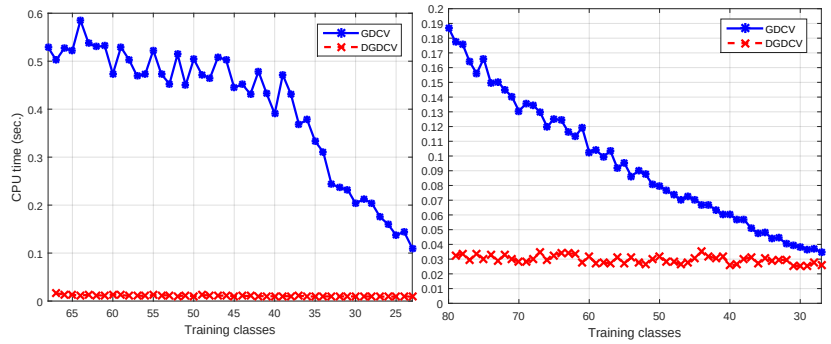


(e) FERET (f) MPEG

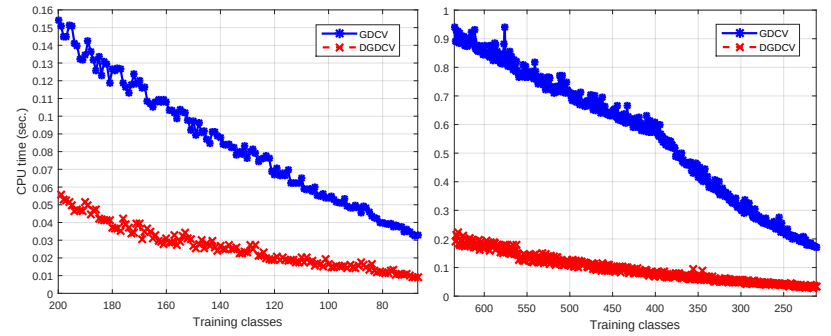
Figure 6: Accuracy of DGDCV (dotted line) and GDCV (continuous line) for TR = 1, TR = 0.5 and TR = 0.25. In graphs (e) and (f), TR=0.25 was not calculated due to the small size of each classes.



(a) AR (b) BANCA

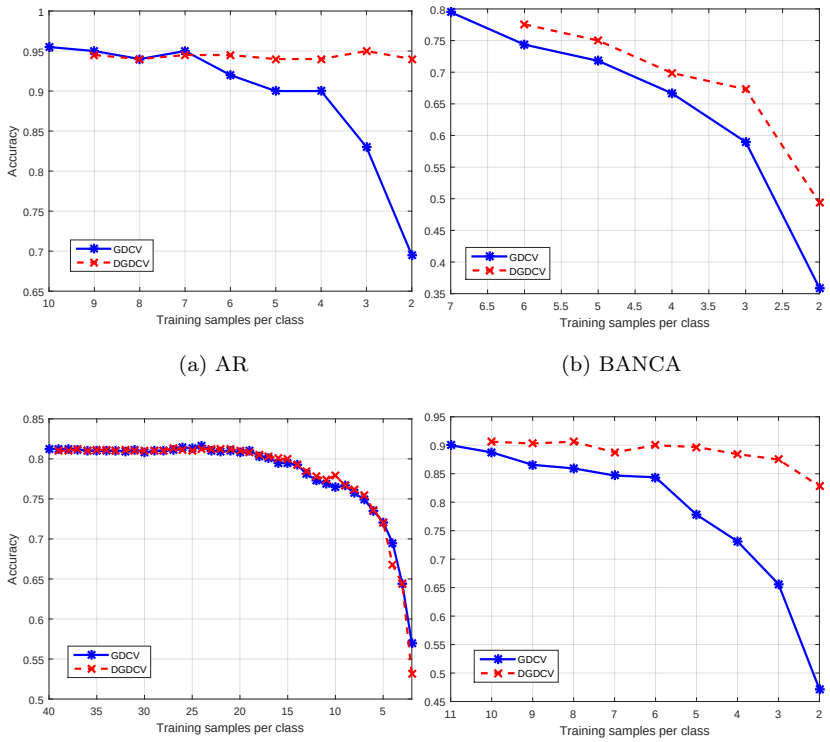


(c) CMU-PIE (d) Altkom



(e) FERET (f) MPEG

Figure 7: CPU time in seconds vs training classes in DGDCV and GDCV methods.



(a) AR (b) BANCA

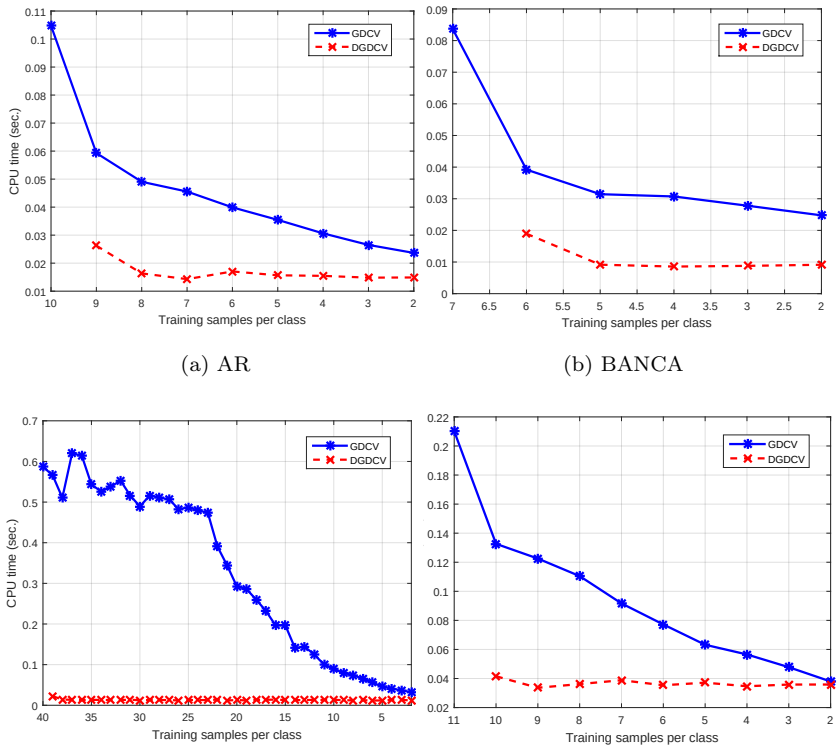
(c) CMU-PIE (d) Altkom

Figure 8: Accuracy of DGDCV (dashed line) and GDCV (solid line) when samples are deleted

7. Conclusions

This paper presents, for the first time, a novel decremental subspace-based learning method DGDCV, capable of updating a feature space model by deleting unnecessary samples/classes, while retaining the previously acquired knowledge, without accessing to the previously processed training data. The new method shows a significant computational gain in computational cost and memory. Both corrupted samples classes and/or obsolete full classes can be removed in our implementation.

The proposed method has been evaluated in 6 standard datasets for face recognition with different characteristics. Our methodology has shown to be consistent in all experiments, a similar or better performance to its batch equiv-



(a) AR (b) BANCA

(c) CMU-PIE (d) Altkom

Figure 9: CPU time in seconds vs training samples in DGDCV and GDCV methods.

alents. This validates the approximations required to perform the decrement of its initial model without recomputing the projection and all other calculations from scratch. As a main advantage, the computational cost when performing each decremental iteration is significantly smaller than recomputing the full model and follows a small linear or constant trend. All these conclusions are also true in the SSS case. Moreover, DGDCV only needs to know the class samples to be removed, which reduce the amount of memory and memory accesses in our method, as well as the required permission and availability of the initial training samples.

Although the method has no limitations regarding the relation of the number of training samples or their dimensionality, the closer the value of the number

of removed samples is to the size of the initial training set, the smaller the computational gain by using our decremental approach results. This is since $m_j << M$ and $(r + r_D)^3 < min(d^3, M^3)$ conditions are not fulfilled. If almost every sample/class of the initial training set has to be deleted, it is simpler to train the system from scratch.

Another limitation of our current method is that this decremental approach does not include incremental learning. Therefore, if new information needs to be added or incremental and decremental steps need to be alternate, the incremental algorithm [9] needs to be added as a separate process. As future work, we aim to extend this method to integrate dual updates allowing both adding and removing samples/classes at a time.

References

[1] D. G. Lowe, Object recognition from local scale-invariant features, in: International Conference on Computer Vision, 1999, pp. 1150–1157.

[2] N. Dalal, B. Triggs, Histograms of oriented gradients for human detection, in: IEEE Conference on Computer Vision and Pattern Recognition, 2005, pp. 886–893.

[3] H. Lee, C. Ekanadham, A. Y. Ng, Sparse deep belief net model for visual area v2, in: Advances in Neural Information Processing Systems, 2008, pp. 873–880.

[4] I. Guyon, A. Elisseeff, An introduction to variable and feature selection, Journal of Machine Learning Research 3 (2003) 1157–1182.

[5] D. Chu, L. Liao, M. Ng, X. Wang, Incremental linear discriminant analysis: A fast algorithm and comparisons, IEEE Trans. on Neural Networks and Learning Systems 26 (11) (2015) 2716–2735.

[6] X. Zeng, G. Li, Covariance free incremental principal component analysis with exact mean update, Journal of Computational Information Systems 5 (16) (2013) 181–192.

[7] K. He, J. Sun, Convolutional neural networks at constrained time cost, in: IEEE Conference on Computer Vision and Pattern Recognition, 2015, pp. 5353–5360.

[8] D. Ross, J. Lim, R. Lin, M. Yang, Incremental learning for robust visual tracking, International Journal of Computer Vision 77 (1-3) (2008) 125–141.

[9] K. Diaz-Chito, F. Ferri, W. Diaz-Villanueva, Incremental generalized discriminative common vectors for image classification, IEEE Trans. on Neural Networks and Learning Systems 26 (8) (2015) 1761–1775.

[10] Y. Peng, S. Pang, G. Chen, A. Sarrafzadeh, T. Ban, D. Inoue, Chunk incremental idr/qr lda learning, in: International Joint Conference on Neural Networks, 2013, pp. 1–8.

[11] M. Karasuyama, I. Takeuchi, R. Nakano, Efficient leave-m-out cross-validation of support vector regression by generalizing decremental algorithm, New Generation Computing 27 (4) (2009) 307–318.

[12] H. Zhao, P. Yuen, J. T. Kwok, A novel incremental principal component analysis and its application for face recognition, IEEE Trans. on Systems, Man, and Cybernetics (Part B) 36 (2006) 873–886.

[13] S. Ozawa, S. Pang, N. Kasabov, Incremental learning of chunk data for online pattern classification systems, IEEE Trans. on Neural Networks 19 (6) (2008) 1061–1074.

[14] G. Duan, Y. Chen, Batch-incremental principal component analysis with exact mean update, in: IEEE International Conference on Image Processing, 2011, pp. 1397–1400.

[15] T. Kim, K. Kenneth, B. Stenger, J. Kittler, R. Cipolla, Incremental linear discriminant analysis using sufficient spanning set approximations, in: IEEE Conference on Computer Vision and Pattern Recognition, 2007, pp. 1–8.

[16] G. Lu, J. Zou, Y. Wang, Incremental complete lda for face recognition, Pattern Recognition 45 (7) (2012) 2510–2521.

[17] K. Diaz-Chito, F. Ferri, W. Díaz-Villanueva, Null space based image recognition using incremental eigendecomposition, in: Pattern Recognition and Image Analysis: 5th Iberian Conference, 2011, pp. 313–320.

[18] G. Lu, J. Zou, Y. Wang, Incremental learning of discriminant common vectors for feature extraction, Applied Mathematics and Computation 218 (22) (2012) 11269–11278.

[19] F. Ferri, K. Diaz-Chito, W. Diaz-Villanueva, Fast approximated discriminative common vectors using rank-one svd updates, in: International Conference on Neural Information Processing, 2013, pp. 368–375.

[20] I. Czarnowski, P. Jdrzejowicz, Family of instance reduction algorithms versus other approaches, in: Intelligent Information Processing and Web Mining. Advances in Soft Computing, Vol. 21, 2005, pp. 23–30.

[21] G. Cauwenberghs, T. Poggio, Incremental and decremental support vector machine learning, in: International Conference on Neural Information Processing Systems, 2000, pp. 388–394.

[22] C. Tsai, C. Lin, C. Lin, Incremental and decremental training for linear classification, in: International Conference on Knowledge Discovery and Data Mining, 2014, pp. 343–352.

[23] M. Karasuyama, I. Takeuchi, Multiple incremental decremental learning of support vector machines, IEEE Trans. on Neural Networks 21 (7) (2010) 1048–59.

[24] L.-F. Chen, H.-Y. Liao, M.-T. Ko, J.-C. Lin, G.-J. Yu, A new lda-based face recognition system which can solve the small sample size problem, Pattern Recognition 33 (10) (2000) 1713–1726.

[25] P. Hall, D. Marshall, R. Martin, Merging and splitting eigenspace models, IEEE Trans. on Pattern Analysis and Machine Intelligence 22 (9) (2000) 1042–1049.

[26] B. Jin, Z. Jing, H. Zhao, Evd dualdating based online subspace learning, Mathematical Problems in Engineering (2014) 1–21.

[27] H. Murakami, B. Kumar, Efficient calculation of primary images from a set of images, IEEE Trans. on Pattern Analysis and Machine Intelligence 4 (5) (1982) 511–515.

[28] A. Tamura, Q. Zhao, Rough common vector: A new approach to face recognition, in: IEEE International Conference on Systems, Man and Cybernetics, 2007, pp. 2366–2371.

[29] T. Kim, B. Stenger, J. Kittler, R. Cipolla, Incremental linear discriminant analysis using sufficient spanning sets and its applications, International Journal of Computer Vision 91 (2) (2011) 216–232.

[30] T. Kim, W. Hwang, J. Kittler, Component-based lda face description for image retrieval and mpeg-7 standardisation, Image and Vision Computing 23 (7) (2005) 631–642.

[31] K. Diaz-Chito, F. Ferri, W. Díaz-Villanueva, Image recognition through incremental discriminative common vectors, in: International Conference on Advanced Concepts for Intelligent Vision Systems, 2010, pp. 304–311.

[32] A. Martinez, R. Benavente, The ar face database, Technical Report 24, Computer Vision Center CVC (1998).

[33] T. Sim, S. Baker, M. Bsat, The CMU pose, illumination, and expression (PIE) database, in: IEEE International Conference on Automatic Face and Gesture Recognition, 2002, pp. 1–6.

[34] J. Phillips, H. Wechsler, J. Huang, P. Rauss, The feret database and evaluation procedure for face-recognition algorithms, Image and Vision Computing 16 (5) (1998) 295–306.

[35] J. Ye, R. Janardan, C. Park, H. Park, An optimization criterion for generalized discriminant analysis on undersampled problems, IEEE Trans. on Pattern Analysis and Machine Intelligence 26 (8) (2004) 982–994.

[36] J. Ye, Q. Li, Lda/qr: an efficient and effective dimension reduction algorithm and its theoretical foundation, Pattern Recognition 37 (4) (2004) 851 – 854.

[37] G.-F. Lu, Z. Jian, Y. Wang, Incremental learning from chunk data for idr/qr, Image and Vision Computing 36 (2015) 1 – 8.