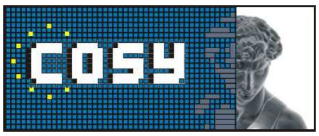


FP6-004250

CoSy

Cognitive Systems for Cognitive Assistants

Integrated Project

Information Society Technologies

DR.5.4

Object models suitable for continuous and human–assisted learning

Due date of deliverable: 30/9/2006
Actual submission date: 6/10/2006

Start date of project: September 1st, 2004 **Duration:** 48 months

Organisation name of lead contractor for this deliverable:
University of Ljubljana

Revision: Final
Dissemination Level: PU

Executive Summary

Learning is a fundamental capability of any cognitive system. To enable the efficient operation of a cognitive agent in a real-world environment, visual learning has to be a continuous process, enabling the system to adapt to changes and to improve its performance through time. It is therefore important that the representations can be learned in an incremental way.
In this deliverable we present several algorithms for incremental learning of representations. First, we present a method for incremental and robust learning of *reconstructive* subspace representations, such as PCA. We then extend this work and propose to combine reconstructive and discriminative subspace methods to enable incremental learning of *discriminative* subspace representations (such as LDA) as well.
We then also present a new algorithm for efficient clustering and matching for object class recognition. Our new scheme lends itself to online, continuous learning, since it allows us to build a hierarchical clustering tree on large number of images in general, and to incrementally compute image/object representations by matching them to the tree.
Finally, we present a continuous learning framework, which is used to evaluate different types of incremental learning mechanisms that require different levels of supervision provided by a tutor. We present a simple method for learning cross-modal associations between words (describing object visual properties) and simple visual features (extracted from images), as well as associations between words describing scenes in terms of simple spatial relations and extracted positional features.

Role of models suitable for continuous and human–assisted learning in Cosy

Continuous learning plays one of the central roles in cognitive systems and in the CoSy project as well. Another important property of a cognitive assistant, as envisioned in CoSy, is its ability to communicate with a tutor, which can provide information necessary for an efficient and effective learning process. The issues discussed in this deliverable are thus highly relevant to CoSy.

Relation to the Demonstrators

Any cognitive system should be able to learn about its body, its environment, objects, scenes, about actions and consequences of (inter)actions that it can perform and observe. It also has to be able to adapt to a changing world and to learn new concepts and extend its ontology when needed. We would like to implement most of these capabilities in the demonstrators we are developing in CoSy. Therefore, several techniques presented in this deliverable are directly applicable in demonstrators and some of them are being integrated into the PlayMate scenario (e.g., the continuous learning framework).

Object models suitable for continuous and human–assisted learning

Danijel Skočaj, Aleš Leonardis, Barry Ridge,
Bastian Leibe, Krystian Mikolajczyk, Bernt Schiele

October 6, 2006

1 Object models suitable for continuous and human–assisted learning

1.1 Incremental learning of discriminative subspace representations

Learning is a fundamental capability of any cognitive system. To enable efficient operation of a cognitive agent in a real-world environment, visual learning has to be a continuous process. The representations should not be learned once and for all in a training stage and then used in their fixed form for recognition, planning and acting. They should be continuously updated over time, adapting to the changes in the environment, new tasks, user reactions, or user preferences. It is therefore important that the representations employed allow the learning to be an incremental process.

In this work we focus on subspace methods for visual learning and recognition and analyse their suitability for incremental learning. In [1], which is based on our previous work, we develop methods for incremental and robust learning of *reconstructive* subspace representations, such as PCA. We present an incremental method, which sequentially updates the principal subspace considering weighted influence of individual images as well as individual pixels within an image. We further extend this approach to enable determination of consistencies in the input data and imputation of the inconsistent values using the previously acquired knowledge, resulting in a novel method for incremental, weighted, and robust subspace learning. We demonstrate the effectiveness of the proposed concept in several experiments on learning of object and background representations.

It turns out that a straightforward application of this approach to *discriminative* subspace methods does not produce satisfactory results. Discriminative representations (such as LDA) are more compact and task dependent, therefore they do not encompass sufficient reconstructive information, which would enable detection of outliers and reliable reconstruction of their values. Therefore, in [2] we propose to combine reconstructive and discriminative subspace methods to enable incremental learning of discriminative representations. Inspired by [3] (see DR.7.1), we achieve this by embedding LDA learning and classification into the incremental PCA framework. The combined subspace consists of a truncated PCA subspace and a few additional basis vectors that encompass the discriminative information, which would be lost by the discarded principal vectors. As such it contains both sufficient reconstructive information to enable incremental learning, and the previously extracted discriminative information to enable efficient classification as well. We demonstrate that we are able to efficiently update the current model with new instances of the already learned classes as well as being able to introduce new classes.

1.2 Efficient clustering and matching for object class recognition

Many of today's object class recognition approaches use clustering and matching of local features to build object models. Frequently used clustering strategies are k-means and agglomerative clustering. Other methods like mean-shift are also becoming more and more popular. However, their relative performances have not been compared for computer vision tasks, and no guidelines are available for judging the tradeoffs in representational capacity, accuracy, and run-time. K-means is frequently used because of its computational simplicity. However, the clustering solution is suboptimal when the number of outliers is large. In the agglomerative clustering scheme the number of clusters is automatically determined based on a more intuitive threshold, yet both its runtime and memory requirements are significantly higher than for the other methods. In [4] we show, that agglomerative clustering has several inherent properties that make it highly attractive for object class recognition: first, matching can be done efficiently using ball-tree search in high-dimensional spaces and with large numbers of clusters; second the clusters reflect the distribution of features resulting in fewer matches and lower complexity; and third, recognition performance is often better than for k-means clusters. In [4] we introduce various improvements for agglomerative clustering in the context of processing large numbers of high-dimensional features. In addition, we show how to use the clustering result to build a data structure for efficient matching. These improvements result not only in a practically feasible and efficient clustering scheme, but also in significant speed-ups for matching. In this deliverable we experimentally validate, and present several results to analyze the performance of the new method. Our clustering algorithm offers advantages for tasks such as database retrieval, image matching, and object recognition (see an example in WP7, D.7.5). Our new scheme lends itself to online, continuous learning, since it allows us to build a hierarchical clustering tree on large number of images in general, and to incrementally compute image/object representations by matching them to the tree.

1.3 On different modes of continuous learning of visual properties

The interaction between a tutor and an artificial cognitive system plays an important role in a continuous learning framework. One goal of the learning mechanism could be to find associations between words spoken by the tutor and visual features automatically extracted by the cognitive visual system, i.e. to ground the semantic meaning of the visual objects and their properties into the visual features. When implementing a continuous learning mechanism, two main issues have to be addressed. Firstly, the representation, which is used for modeling the observed world, has to allow for updates when presented with newly acquired information. This update step should be efficient and should not require access to the previously observed data while still preserving the previously acquired knowledge. Secondly, a crucial issue is the quality of the updating, which highly depends on the correctness of the interpretation of the current visual input. With this in mind, several learning strategies can be used, ranging from completely supervised learning to a completely unsupervised approach. In [5] we discuss three such learning strategies:

- **Tutor-driven approach.** The correct interpretation of the visual input is always correctly given by the tutor.
- **Tutor-supervised approach.** The system tries to interpret the visual input. If it succeeds to do this reliably, it updates the current model, otherwise asks the tutor for the correct interpretation.
- **Exploratory approach.** The system updates the model with the automatically obtained interpretation of the visual input. No intervention from the tutor is provided.

We propose a method for finding associations between *words* and simple *visual features*, such as hue or intensity values of the corresponding pixels. In particular we present a method for learning *visual attributes* (e.g., colour, shape) and their *qualitative values* (e.g., red, yellow; circular, square). Using this simple method, we then evaluate different types of incremental learning mechanisms that require different levels of supervision provided by a tutor.
The results are as anticipated. The tutor-driven learning finds the correct associations between visual features and visual properties of objects that yield close to optimal results. Tutor-supervised learning also proved to be quite successful, while exploratory learning is useful only after the system has already acquired a sufficient level of knowledge. So, as expected, there is a trade-off between the quality of the results and the the number of necessary questions.
We then applied the same method and the learning framework for learning simple *spatial relations*, such as "*A is to the left of B*, "*A is near to B*", or "*A is on the right*". The learning method had to find associations between simple positional features (the coordinates of the centers of the objects in the image, the distance between them, etc.) and these spatial relations. In a dialogue with a tutor (initially in a tutor driven manner, while later in a tutor supervised manner), the system was able to find suitable associations and to adequately model the spatial relationships. Although in the beginning it did not have any concept about spatial relations, it was gradually extending its knowledge, and after a while it was able to produce simple descriptions of a scene using *learned* object visual properties and *learned* spatial relations. The developed framework for continuous learning thus proved to be useful in two different domains.

2 Future Work

We plan to extend the framework for continuous learning of object properties in several directions. We will make it more robust, include new, more complex properties and features, and improve the feature selection process and association model. We want to develop a unified framework for continuous learning of object properties as well as spatial relations, and object affordances. The main emphasis will be on exploration of different learning strategies involving different levels of human supervision. In this respect we want to address several challenging problems and answer related questions, like what the correct sequence of learning strategies is, how the order of training images influences the results, what can be learned in the image domain only, without having a word description of the scene, what the best trade-off between the quality of the results and the number of necessary questions is, how the number of attributes to learn and the number of extracted features influence the results and the learning curve, how to introduce prior knowledge and other types of learning, etc.
One of the principle limitations for continuous and online learning is the inability to transfer knowledge between multiple learning settings. The key idea is to enable transfer of knowledge across objects by modeling not just entire object but also their respective part-structure and similarities to other object classes. We therefore aim to develop and learn a part-object hierarchy in a cross-modal fashion from language and vision. This hierarchy should enable the transfer of knowledge on the object part level as well as in the super-ordinate and even higher level object categories.

3 References

3.1 Applicable documents

- **DR.7.1** Basic models and representations and their structural organisation for various objects, object categories and places
- **DR.7.3** Mechanisms for self-supervised acquisition and consolidation of acquired models with different degrees of supervision from the outside

3.2 Reference documents

[1] D. Skočaj and A. Leonardis. Incremental and robust learning of subspace representations. *Image and Vision Computing*, 2006. (accepted for publication).

[2] D. Skočaj, M. Uray, A. Leonardis, and H. Bischof. Why to combine reconstructive and discriminative information for incremental subspace learning. In *CVWW 2006*, Telč, Czech Republic, February 2006.

[3] Sanja Fidler, Danijel Skočaj, and Aleš Leonardis. Combining reconstructive and discriminative subspace methods for robust classification and regression by subsampling. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 28(3):337–350, March 2006.

[4] B. Leibe, K. Mikolajczyk, and B. Schiele. Efficient clustering and matching for object class recognition. In *British Machine Vision Conference (BMVC'06)*, 2006.

[5] D. Skočaj, B. Ridge, and A. Leonardis. On different modes of continuous learning of visual properties. In *15th International Electrotechnical and Computer Science Conference ERK 2006*, Portorož, Slovenia, September 2006.

Annexes

In the following pages, the scientific papers [1, 2, 4, 5] are attached as a part of this deliverable.

To appear in Image and Vision Computing.

Incremental and robust learning of subspace representations

Danijel Skočaj * Aleš Leonardis

University of Ljubljana, Faculty of Computer and Information Science, Tržaška 25, SI-1001 Ljubljana, Slovenia

Abstract

Learning is a fundamental capability of any cognitive system. To enable efficient operation of a cognitive agent in a real-world environment, visual learning has to be a continuous and robust process. In this article we present a method for subspace learning, which takes these considerations into account. We present an incremental method, which sequentially updates the principal subspace considering weighted influence of individual images as well as individual pixels within an image. We further extend this approach to enable determination of consistencies in the input data and imputation of the inconsistent values using the previously acquired knowledge, resulting in a novel method for incremental, weighted, and robust subspace learning. We demonstrate the effectiveness of the proposed concept in several experiments on learning of object and background representations.

Key words: subspace learning, incremental learning, robust learning

* Corresponding author. Tel.: +386 1 4776631; Fax.: +386 1 4264647.
Email address: danijel.skocaj@fri.uni-lj.si.

1 Introduction

Cognitive vision has become an important emerging discipline over the last years caused by the growing need for visually-enabled cognitive systems. The main scientific foundations of cognitive vision are, among others, the issues of visual architecture, representations, memory, learning, and recognition [1]. There is no doubt, however, that *learning* plays a major role in developing intelligent visual and cognitive systems. Cognitive systems need to acquire the information about the external world through learning or association, as the complex interrelationships between percepts and their contextual frames could never be specified explicitly through programming [2] due to their complexity and the need for adaptability. In this paper we will focus on two important aspects of learning: incrementality and robustness.

In order to avoid evolutionary time-scales in the development of a cognitive system, it has to initially encompass a certain level of predefined functionality. It then has to be able to build new concepts upon the initial ones and keep developing throughout its lifetime. It is therefore important that the representations employed allow learning to be a continuous, open-ended, life-long process. In other words, the representations should not be learned once and for all in a training stage and then used in their fixed form for recognition, planning, and acting. They should be continuously updated over time, adapting to the changes in the environment, new tasks, user reactions, user preferences, etc. The learning process should facilitate such incremental way of building and updating of representations. So there should be no strict distinction between the activities of learning and recognising — these activities should be interleaved.

This is a non-trivial challenge. Most of the state-of-the-art algorithms for visual learning and recognition do not consider continuous learning. They follow the standard paradigm, dividing the off-line learning stage and the recognition stage [3–10]. Most of these approaches are not designed in a way that would enable efficient updating of the learned model, which is a basic prerequisite for incremental learning.

There are two issues which are important for a reliable continuous learning. Firstly, the representation, which is used for modeling the observed world, has to allow for updating with newly acquired information. This update step should be efficient and should not require access to the previously observed data while still preserving the previously acquired knowledge. And secondly, a crucial issue is the quality of updating, which highly depends on the correctness of the interpretation of the current visual input. When the correct interpretation of the current visual input is given by a tutor, the update step is risk-free in the sense that the algorithm can update the model with a high con-

fidence. On the other hand, when the information, which is to be added to the representation, is autonomously extracted by the agent in an unsupervised manner (e.g., using a recognition procedure) there is a risk of propagating an erroneous extraction from the data through the learning process. Consequently, the representation could be corrupted with false data, resulting in a poorer performance and a less reliable representation. Robust mechanisms, which prevent such propagation of errors, play an important role in the process of continuous learning.

Robustness is thus another important capability of a cognitive system. A cognitive agent is equipped with imperfect sensors and effectors, and it is supposed to operate in a partially unpredictable real-world environment. Nevertheless, in the vision literature ideal training conditions are most commonly assumed. They enable the construction of a reliable model of the environment, which can then be used for determining outliers in new images and performing *robust recognition*. But yet, a fully operational cognitive system should be able also to build the representation under non-ideal conditions. *Robust learning* is, however, a greater challenge than robust recognition. During the learning process a sufficient knowledge required for determining the relevance of visual input is still to be acquired. Therefore, robust learning should strongly be intertwined with the process of continuous learning, which could provide enough redundant information to determine statistically consistent data (this information can also be provided by a user acting as a tutor). Only the consistent data would then be used to build the representations of objects, enabling robust learning (and updating of the representations) under non-ideal real-world conditions.

Traditionally, learning is performed in a batch way. However, as discussed above, batch methods are in most cases not appropriate for cognitive vision systems; they are not biologically plausible and they are not feasible for very large sets of data. Nevertheless, they serve as a good basis for evaluation of incremental methods, since they have all original input data available and can thus extract the information, which is the most important for building a faithful representation. The incremental methods, on the other hand, have only one or a few original input images available, and only the representations of the previously seen (and learned) images. Therefore, one could expect that in general the incremental methods produce somehow inferior results than the batch methods. A very interesting and important question is, how severe these degradations of the results are? What factors influence the results; i.e., does the order of training images influence the results? These questions may be even more pronounced in the case of non-ideal training conditions: What happens if most of the training images are corrupted? How important is it that the images at the beginning of the learning sequence are of sufficient quality? When designing a representation to be used in a cognitive vision system, these questions need to be addressed.

A plethora of representations and approaches to visual learning and recognition has been proposed in the past. However, in the context of cognitive vision, it is very important that whatever the type of the representation is, it enables continuous and robust learning. In the following sections we will present implementation of these principles in the case of subspace-based visual learning and recognition. Since the PCA-based approach is originally designed as a batch method and is inherently non-robust to non-gaussian noise, we propose several extensions of the standard approach, which enable incremental and robust learning.

The paper is organized as follows. In Section 2 we present the subspace-based approach to visual learning and recognition, expose its shortcomings, review the previously proposed improvements, and outline our approach. In two following sections we elaborate our approach in detail. In Section 3, we first present the basic algorithm for incremental learning. In Section 4 we extend this algorithm into a weighted algorithm, which considers temporal and spatial weights. Next, we present a special case of the algorithm, which can handle missing data. This algorithm is then advanced into a robust incremental algorithm, which can detect and discard inconsistencies in the input images. In Section 5 we present the experimental results. Finally, we summarize the paper, expose the contributions, and outline some work in progress.

2 Subspace-based modeling of objects and scenes

Appearance of an object combines effects of its shape, reflectance properties, pose in the scene, and illumination conditions [11]. It proves to be very difficult to separate all these factors from a set of images in order to obtain a view and illumination-invariant representation. In the appearance-based approach, the separation of these physical properties is circumvented. However, in order to obtain a complete appearance-based model of an object, one has to systematically observe the training object under different viewing and illumination conditions, which may result in a rather large set of images. Consequently, they have to be efficiently represented using a compact representation. A commonly used technique for compression of training images is based on principal component analysis (PCA) [12]. In PCA-based approach [11] an object is represented with the projections of the training images into the principal subspace, thus the object recognition is reduced to the searching of the closest point in this low-dimensional space.

2.1 Problem statement

The standard PCA approach in its original form has several shortcomings with respect to the premises mentioned in Section 1. PCA-based learning is traditionally performed in a batch mode, i.e., all training images are processed simultaneously, which means that all of them have to be given in advance; the obtained representation cannot be updated with new images without starting the process from the scratch. To make updating of the previously learned representation possible, one has to take an *incremental approach* to the principal component analysis.

Besides, in the standard PCA approach all pixels of an image receive an equal treatment. Also, all the training images have equal influence on the estimation of principal axes. To enable a selective influence of individual images and pixels, PCA can be generalized into a *weighted approach*, which considers individual pixels and images diversely, depending on the corresponding weights.

PCA in its standard form is also intrinsically non-robust to non-gaussian noise. The recognition method can be extended such that non-gaussian noise in test images is detected, and the recognition is performed by considering relevant parts of the image only, providing that a consistent representation is given. However, if the training images are taken under non-ideal conditions, the non-desirable effects should be detected in the learning stage already and not included into the representation. Thus, we need a method for *robust learning*, which is able to detect inconsistencies in the training images and build the representations from consistent data only.

In the following subsection we will first review some existing extensions of the standard PCA approach, which can cope with the problems mentioned above. Then we will outline our approach to the solution of these problems, which will be described in detail in the following sections.

2.2 Previous work

Several authors faced the problem of decomposing large covariance matrices obtained from a huge number of training images. To overcome this problem several incremental algorithms for PCA have been proposed. The first algorithm for incremental PCA in the computer vision community was proposed by Murakami and Kumar [13]. Then, Chandrasekeran et al. proposed an algorithm, which is based on SVD updating [14]. Incremental singular value decomposition was often tackled also in the past (e.g., [16,17]), and recently [15]. All these methods keep the origin of the principal subspace in the origin of the image space, assuming that the mean of the input images is always zero.

This is not true in general and this assumption may degrade the results of the classification [18]. By noting this problem, Hall et al. proposed a method for eigenspace updating, which sequentially shifts the origin of the eigenspace according to the new images, which are being added [18,19].

Several methods for weighted learning with different derivations but very similar realizations have also been proposed. Wiberg [20] has proposed a method for subspace learning when data are missing based on the weighted least squares technique. This method was later extended by Shum et al. [21]. Gabriel and Zamir [22] proposed a method for subspace learning with any choice of weights, where each data point can have a different weight determined on the basis of reliability. A similar approach was also used in the work of Sidenbladh et al. [23] and De la Torre and Black [24]. All these methods operate in a batch mode.

The only incremental methods that explicitly deal with spatial weights are the methods for incremental singular value decomposition of data with missing values introduced by Brand [15] and the method for incremental PCA very recently proposed by Li [25]. With respect to temporal weights, the latter method, as well as the incremental methods proposed by Liu and Chen [26] and Levy and Lindenbaum [27] are tailored for temporally weighted learning considering a decay parameter thus allowing newer images to have a larger influence on the estimation of the current subspace than the older ones. Therefore, their methods consider only a special case of temporal weights.

A severe limitation of the basic approach to the subspace visual modeling is its non-robustness to noise, occlusions, and cluttered background. Different approaches have been proposed to improve the robustness of the *recognition*: modular eigenspaces [28], eigenwindows [29], search-window [30], adaptive masks [31], M-estimation [32,33], hierarchial approach [34], and subsampling hypothesise-and-test-based approach [35]. However, all these methods introduce the robustness in the *recognition stage*. They assume that the images in the learning stage were ideal and that the correct visual model is available.

The *robust learning* is a much more difficult problem. Since in the learning stage the model of the object or the scene is being built, there is no reliable model at the beginning of the learning process, which could be used to estimate outliers. The authors coped with this problem in different ways. Xu and Yuille proposed an algorithm, which introduced robustness on the image level [36]. During the learning stage, they discard images, which are inconsistent with the others. However, in many practical applications this is not satisfactory. The robustness on the pixel level should be assured meaning that only single pixels should be discarded and not the entire images. Gabriel and Zamir tried to solve this problem using a weighted singular value decomposition [22]. Recently, De la Torre and Black proposed a method for robust principal component

analysis based on M-estimation [37,24]. This method, as well as the related method proposed by Skočaj et al. [38], perform well on images with sufficient temporal correlation, but are very time consuming. Very recently, Aanæs et al. [39] proposed a method for robust factorization, which is tailored for a different type of problems, however, some of its principles are relevant for robust eigenspace learning as well. These methods also operate in a batch mode, processing all training images simultaneously. Furthermore, they are executed in an iterative manner by repeating time consuming procedures on the entire set of training images. Therefore, the processing time is usually very long, and even becomes prohibitive when the number of training images is large. To overcome these problems, only very recently the incremental methods for robust estimation of the principal subspaces were proposed [40,25].

2.3 Our approach

The *incremental algorithm*, which we will present in this paper, produces the identical principal subspace as the method proposed by Hall et al. [18]. However, the subspace is obtained in a different way. A significant advantage of our method is that it is able to treat different images differently, which enables to extend it into a weighted incremental method. Furthermore, our method maintains the low-dimensional representations of the previously learned images throughout the entire learning stage, therefore enabling that each training image can be discarded immediately after the update.

Our *weighted incremental approach* considers arbitrary temporal and spatial weights, thus it is more general than the methods proposed in [26], [27], and [25]. The incremental method is also adapted for learning from incomplete data, which, in contrast to the method presented in [15], considers also the mean and updates its value at each step adequately. We also propose a method for incremental robust learning, which sequentially determines consistencies in the input images and reconstructs inconsistent pixels using the previously acquired knowledge. All the proposed methods will be described and evaluated in detail in the following sections.

3 Incremental PCA

In this section we propose a method for incremental learning. It takes the training images sequentially and computes the new eigenspace from the subspace obtained in the previous step and the current input image.

Let us suppose that we have already built an eigenspace from the first n im-

ages. In the step $n+1$ we could calculate a new eigenspace from the *reconstructions* [1] of the first n input images and a new image using the standard batch method. The computational complexity of such an algorithm would be prohibitive, since at each step we would have to perform the batch PCA on a set of high-dimensional data. However, identical results can be obtained by using low-dimensional *coefficient vectors* [2] of the first n input images instead of their high-dimensional reconstructions, since coefficient vectors and reconstructed images encompass the same visual variability, i.e., they are just represented in different coordinate frames. Since the dimension of the eigenspace is small, this algorithm is computationally very efficient.

The summarized procedure for one update of the current eigenspace is outlined in Algorithm 1 [3]. This algorithm increases the dimension of the subspace by one. After the update, we can discard the least significant principal vector to preserve the dimension of the subspace [41].

The initial values of the mean image, the eigenvectors, and the coefficients can be obtained by applying the batch PCA on a small set of images. Alternatively, one can simply set the first training image as the initial eigenspace by assigning $\boldsymbol{\mu}^{(1)} = \mathbf{x}_1$, $\mathbf{U}^{(1)} = \mathbf{0}_{M\times 1}$, and $\mathbf{A}^{(1)} = 0$. In this way, the algorithm is completely incremental, requiring only one image to be available at each time instant.

It is worth noting that this algorithm produces the identical principal subspace as the method proposed by Hall et al. [18]. However, the subspace is obtained in a different way. In contrast to our method, which between the learning steps passes coefficient vectors of all training images, the Hall's method passes only the eigenvalues. While one may consider this as an advantage, since less data is being passed from step to step and calculation of the covariance matrix is faster, this can also be disadvantageous, because the coefficients are not estimated and maintained during the learning process, thus less information is available. Our algorithm calculates the coefficients at that time instant, when the particular image is added to the model, and then maintains their values throughout the process of incremental learning. This is slightly slower, however it has two advantages. The first advantage is, that each image can be discarded immediately after it has been used for updating the subspace. This is very appropriate (and possibly required) for on-line scenarios (e.g., naviga-

[1] An image can be reconstructed by transforming its subspace coefficient vector into the high-dimensional input image space.

[2] Coefficient vectors are composed of coefficients, which are obtained by projecting an image onto the principal axes spanning the eigenspace.

[3] $\mathbf{U} \in \mathbb{R}^{M\times k}$ denotes a matrix of k M-dimensional principal axes, $\mathbf{A} \in \mathbb{R}^{k\times n}$ is a matrix of n k-dimensional coefficient vectors, $\boldsymbol{\mu} \in \mathbb{R}^{M}$ is the mean image. Superscript denotes the step which the data is related to ($\mathbf{U}^{(n)}$ denotes the values of $\mathbf{U}$ at the step n). $\mathbf{1}_{m\times n}$ denotes a $m \times n$ matrix of ones. $\|\mathbf{r}\|$ denotes the L_2 norm of the vector $\mathbf{r}$.

Algorithm 1 : Incremental PCA

Input: current mean vector $\boldsymbol{\mu}^{(n)}$, current eigenvectors $\mathbf{U}^{(n)}$, current coefficients $\mathbf{A}^{(n)}$, new input image $\mathbf{x}$.

Output: new mean vector $\boldsymbol{\mu}^{(n+1)}$, new eigenvectors $\mathbf{U}^{(n+1)}$, new coefficients $\mathbf{A}^{(n+1)}$, new eigenvalues $\boldsymbol{\lambda}^{(n+1)}$.

1: Project a new image $\mathbf{x}$ into the current eigenspace:
$\mathbf{a} = \mathbf{U}^{(n)\top}(\mathbf{x} - \boldsymbol{\mu}^{(n)})$.

2: Reconstruct the new image: $\mathbf{y} = \mathbf{U}^{(n)}\mathbf{a} + \boldsymbol{\mu}^{(n)}$.

3: Compute the residual vector: $\mathbf{r} = \mathbf{x} - \mathbf{y}$.
$\mathbf{r}$ is orthogonal to $\mathbf{U}^{(n)}$.

4: Append $\mathbf{r}$ as a new basis vector:
$\mathbf{U}' = \left[\mathbf{U}^{(n)} \;\; \frac{\mathbf{r}}{\|\mathbf{r}\|} \right]$.

5: Determine the coefficients in the new basis:
$\mathbf{A}' = \begin{bmatrix} \mathbf{A}^{(n)} & \mathbf{a} \\ \mathbf{0} & \|\mathbf{r}\| \end{bmatrix}$.

6: Perform PCA on $\mathbf{A}'$. Obtain the mean value $\boldsymbol{\mu}''$, the eigenvectors $\mathbf{U}''$, and the eigenvalues $\boldsymbol{\lambda}''$.

7: Project the coefficient vectors to the new basis: $\mathbf{A}^{(n+1)} = \mathbf{U}''^{\top}(\mathbf{A}' - \boldsymbol{\mu}''\mathbf{1}_{1\times n+1})$.

8: Rotate the subspace $\mathbf{U}'$ for $\mathbf{U}''$: $\mathbf{U}^{(n+1)} = \mathbf{U}'\mathbf{U}''$.

9: Update the mean: $\boldsymbol{\mu}^{(n+1)} = \boldsymbol{\mu}^{(n)} + \mathbf{U}'\boldsymbol{\mu}''$.

10: New eigenvalues: $\boldsymbol{\lambda}^{(n+1)} = \boldsymbol{\lambda}''$.

tion of mobile robots with limited memory resources). And finally, since more information is encompassed in the model, our method can be advanced into a method for weighted learning of eigenspaces, which can consider arbitrary temporal weights.

We will demonstrate the behavior of the proposed algorithm on a simple 2-D example. The 2-D input space contains 41 points shown as black dots in Fig. 1. The goal is to estimate 1-D principal subspace, i.e., the first principal axis. The eigenspace is being built incrementally. At each step one point (from the left to the right) is added to the representation and the eigenspace is updated accordingly. Fig. 1 illustrates how the eigenspace evolves during this process. The principal axis, obtained at every sixth step, is depicted. The points, which were appended to the model at these steps, are marked with crosses. One can observe, how the origin of the eigenspace (depicted as a square) and the orientation of the principal axis change through time, adapting to the new points, which come into the process. At the end, the estimated eigenspace, which encompasses all training points, is almost identical to the eigenspace obtained using the batch method.

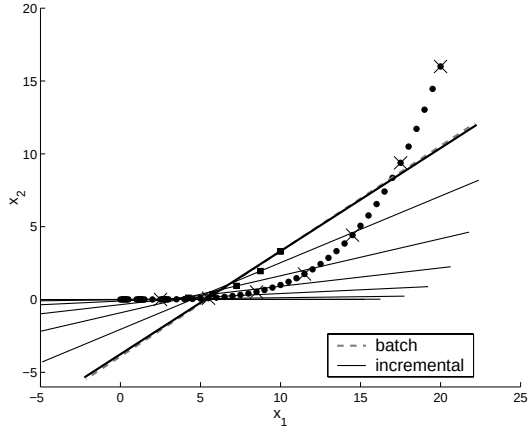


Fig. 1. Incremental learning.

4 Weighted and robust approach

In order to achieve selective influence of pixels and images, the individual pixels as well as images can be weighted with different weights. In practice, it is useful to deal with two types of weights: *temporal* weights ${}^{\mathbf{t}}\mathbf{w} \in \mathbb{R}^{1\times N}$, which put different weights on individual images and *spatial* weights ${}^{\mathbf{s}}\mathbf{w} \in \mathbb{R}^{M}$, which put different weights on individual pixels within an image [4].

4.1 Temporal weights

Temporal weights determine how important the individual images are for the estimation of principal subspace. If the temporal weight for one of the images is higher than the weights for the other images, the reconstruction error of this image should be smaller than the reconstruction errors of the other images. Similarly, the contribution of its principal components to the estimation of the variance should be larger in comparison with that of the other principal components.

From this observation we can derive an algorithm for estimation of the principal subspace considering temporal weights. The principal axes, which maximize the *weighted variance* of the projections of the input images onto the principal axes, can be obtained by eigendecomposition (or, similarly, singular value decomposition) of the *weighted covariance matrix*. If the matrix $\hat{\mathbf{X}} \in \mathbb{R}^{M\times N}$ is composed from N re-scaled input vectors centered around the weighted mean:

$$\hat{\mathbf{x}}_j = \sqrt{{}^{t}w_j}(\mathbf{x}_j - \boldsymbol{\mu}), \;\; j = 1 \ldots N \;\; , \tag{1}$$

[4] The left superscript is used to distinguish between temporal (${}^{\mathbf{t}}\mathbf{w}$) and spatial (${}^{\mathbf{s}}\mathbf{w}$) weights.

the weighted covariance matrix can be calculated as

$$\mathbf{C} = \frac{1}{\sum_{j=1}^{N} {}^{t}w_j} \hat{\mathbf{X}} \hat{\mathbf{X}}^{\top} \quad . \tag{2}$$

Using this algorithm, the estimated principal subspace does not depend on all training images equally. For instance, if a training image has the weight 2, while all the other images have the weight 1, the result of this algorithm equals the result of the standard PCA algorithm, which has two copies of the particular image in the training set.

It is quite straightforward to incorporate the temporal weights into the incremental algorithm. The core of this algorithm is still the standard batch PCA on low-dimensional data (step 6 of Algorithm 1). We can replace this standard batch PCA with the weighted algorithm, which considers temporal weights. This is feasible, because our incremental algorithm maintains low-dimensional coefficients of all input images throughout the process of the incremental learning (in contrast with the other incremental approaches). Therefore, the representation of each image can be arbitrarily weighted at each update.

To illustrate the behavior of the proposed algorithm, we put different weights on the training points from our simple 2-D example. We set temporal weights to ${}^{t}w_j = j^2$, which gives a larger influence to the recent points. Fig. 2 depicts the evolution of the eigenspace. By comparing this figure with Fig. 1 it is evident how the weights affect the learning process. At the end of the learning sequence, the weighted mean vector is closer to the points at the end of the point sequence, since the weights of these points have higher values. The principal axis is oriented in such a direction that enables superior reconstruction of these points.

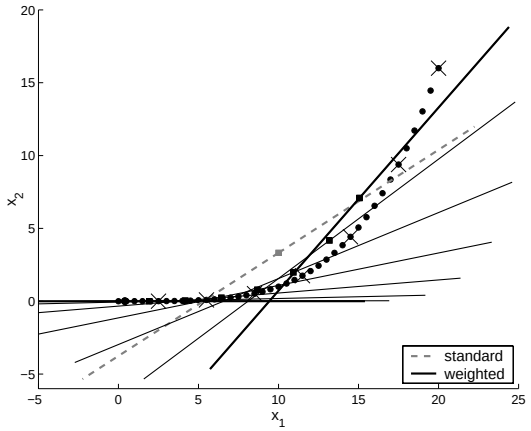


Fig. 2. Weighted incremental learning.

4.2 Spatial weights

Spatial weights control the influence of individual pixels within an image. Therefore, if a part of an image is not reliable or important for the estimation of principal components, its influence should be diminished by decreasing the weight of the corresponding pixels.

Incorporating spatial weights into the process of incremental learning is more complex. After the current eigenspace is updated with a new input image, this image is discarded and only its low-dimensional representation is preserved. Therefore, in the later stages we can not associate weights to individual pixels. This can be done only during the update.

Let us assume that the weights range from 0 to 1. If a weight is set to 1, it means that the corresponding pixel is fully reliable and should be used as is. If a weight is set to 0, it means that the value of the corresponding pixel is irrelevant or erroneous. We can recover an approximate value of this pixel by considering the knowledge acquired from the previous images. By setting the weight between 0 and 1, we can balance between the influence of the value yielded by the current model and the influence of the pixel value of the input image.

We can achieve this by adding a preprocessing step to the update algorithm. First we calculate the coefficients of the new image $\mathbf{x}$ by using the weighted method. Instead of using the standard projection, the coefficients a_j are obtained by solving an over-determined system of linear equations

$$\sqrt{{}^{s}w_i}x_i = \sqrt{{}^{s}w_i}\sum_{j=1}^{k} a_j u_{ij} \quad , \quad i = 1 \dots M \tag{3}$$

in the least squares sense. By reconstructing the coefficients we obtain the reconstructed image $\mathbf{y}$ which contains pixel values yielded by the current model. By blending images $\mathbf{x}$ and $\mathbf{y}$, considering spatial weights by using the following equation

$$x_i^{new} = {}^{s}w_i x_i + (1 - {}^{s}w_i) y_i \quad , \quad i = 1 \dots M \quad , \tag{4}$$

we obtain the image which is then used for updating the current eigenspace. In this way, a selective influence of pixels is enabled also in the incremental framework.

4.3 Missing pixels

In the real world applications, it is often the case that not all data is available. The values of some pixels are missing or they are totaly non-reliable. Such pixels are referred to as *missing pixels*. Estimation of the principal subspace in the presence of missing pixels can be regarded as a special case of spatially weighted PCA where the weights of missing pixels are set to zero.

The blending step in the algorithm for weighted incremental learning reduces to the imputation of missing pixels. Before the current eigenspace is updated with the new image, the missing pixels have to be optimally filled in. Since not all pixels of an image are known, some coordinates of the corresponding point in the image space are undefined. Thus, the position of the point is constrained to the subspace defined with the values of the known pixels. Given the current principal subspace $\mathbf{U}^{(n)}$, which models the input data seen so far, the optimal location is the point in the missing pixels subspace which is closest to the principal subspace. This point is obtained by filling-in the missing pixels with the reconstructed values, which are calculated from the coefficients estimated from the known pixels only. Since this coefficients reflect the novel information in the new image contained in the known pixels, we may assume that the prediction in the missing pixels will be fine as well. Such an improved image is the best approximation of the correct image that we can obtain from the information contained in the known pixels and in the current eigenspace.

Thus, the new image $\mathbf{x}$ if first projected into the current principal subspace $\mathbf{U}^{(n)}$ by solving a system of linear equations (3) arising from non-missing pixels. The obtained coefficient vector $\mathbf{a}$ is then reconstructed and the values in the reconstructed image are used for filling-in the missing pixels. The resulting image is then used for updating the current eigenspace.

A practically equivalent rule for imputation of missing pixels was proposed also by Brand in the context of incremental singular value decomposition [15]. As shown in [15], such a rule for imputation of missing pixels minimizes the distance of the vector representing a new image to the current subspace and maximizes the concentration of the variance in the top singular values. Consequently, such imputation rule minimizes the rank of the updated SVD guaranteeing parsimonious model of the data.

4.4 Robust approach

The developed method for subspace learning from incomplete data can be further extended in a method for robust learning. In the robust framework the positions of 'bad' pixels are not known in advance, however, we are aware

that images may contain outliers. We treat as outliers all pixels, which are not consistent with the information contained in other images. Since at each step we have a current model of the object or scene seen so far, we can detect outliers in the new image and treat them as missing pixels.

This is achieved by projecting the new image into the current eigenspace in a robust manner. Instead of a simple projection, a robust procedure based on subsampling and hypothesize-and-select paradigm is used [35]. Coefficients are obtained mainly from inliers, thus their reconstructions tend to the correct values in outliers as well. Consequently, the reconstruction error in outliers is large, which makes their detection easier. Therefore, to make the incremental learning robust, we first detect outliers in a new image and replace their values with reconstructed values, which are good approximations of the correct values. Such an improved outlier-free image is then used for updating the eigenspace. Providing that the outliers are detected during the learning process using the robust procedure, the obtained eigenspace is robust as well.

We can refer to this procedure as a 'hard' robust algorithm, since the pixels, which are detected as outliers, are replaced with reconstructed values, while the remaining pixels stay intact. An alternative 'soft' approach is to weight each pixel according to its reliability, which is determined with respect to the reconstruction error. The new image $\mathbf{x}$ is thus projected into the current eigenspace using the simple (and fast) standard projection and the obtained coefficients are used for reconstruction ($\mathbf{y}$). The obtained reconstruction error yields the spatial weights (e.g., ${}^{s}w_i = 1/(|x_i - y_i| + 1)$), which are then used by the weighted algorithm to update the current principal subspace.

To demonstrate the behavior of the robust incremental algorithm, we significantly changed the values of the second coordinate of five points in our 2-D example. Fig. 3 shows that when the non-robust incremental method is used, these outlying points pull the origin in a wrong direction and incorrectly orient the estimated principal axis. On the other hand, the robust method sequentially detects the outlying coordinate values, replaces these values with their reconstructions (shown as circles) and updates the eigenspace accordingly. At the end, the principal axis obtained using this approach is very close to the optimal one.

An important advantage of such *incremental* method is that it processes only one image at each step, while the iterative batch robust methods process all images at each iteration. For that reason, the incremental method is significantly faster and enables robust learning from a large number of training images. Since the model is being incrementally updated with new images, this method is very suitable for on-line applications as well.

On the other hand, it suffers (like all incremental methods) from a potential

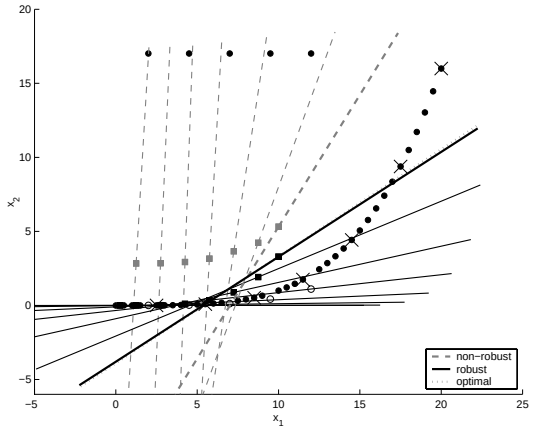


Fig. 3. Robust incremental learning.

danger of error propagation. If the initial eigenspaces, built in the early stages of the learning process, encompass only a limited number of appearances of an object or a scene, then all the pixels in the subsequent images, which significantly differ from the appearances of the first images, are considered as outliers and no novel information is added to the model. This particularly holds true for the 'hard' robust version of the updating algorithm. Therefore, the initial eigenspace, which is built in the beginning of the learning process, should be reliable and stable. It should roughly model heterogeneous appearances of an object or a scene and it should be obtained from a set of pixels containing as few outliers as possible. When the model encompasses a sufficient number of appearances it becomes more stable and this is no longer a problem [42].

5 Experimental results

5.1 Incremental PCA

Principal component analysis in its standard batch form is optimal in the sense of the squared reconstruction error. Thus, its incremental version necessarily degrades the results. But, how severe is this degradation? Are the results still usable? What additional factors influence the results? To clarify these issues and to evaluate the proposed algorithm we will explore the following questions:

- How much does the incremental method degrade the results in comparison with the batch method?
- How does discarding of training images influence the results?
- How does the order of the training images influence the results?

To answer these questions we performed several experiments. First, we built eigenspaces of various dimensions from 720 images of twenty objects from the

COIL database (Fig. 4(a)). Such number of training images can be processed using the batch method allowing a fair comparison. Fig. 4(b), depicts the mean squared reconstruction errors (MSRE) of the images reconstructed from the coefficients obtained by projecting the training images into the eigenspaces, which were built using the batch method (in the plots indicated as *batch*) and the proposed incremental method (*incXseq*). The results are very similar; MSRE obtained using the incremental method is only 3.1% worse on the average. The curve *incAseq* represents reconstruction errors of images obtained from the coefficients, which were calculated at that time instant, when the particular image was added to the model and then maintained throughout the process of incremental learning. Using this approach, an image can be discarded immediately after the model is updated. As one can observe, the squared reconstruction errors are still quite similar. In this case, the degradation of the results is 8.6% on the average.

In the first experiment the images were coming into the learning process in a sorted order, i.e., first all images of the first object, then all images of the second object, and so on. In the second experiment we changed this sequence by giving the training images to the learning process in a random order. Thus, the eigenspace in the early stage of the learning process already encompassed images of several objects. Therefore, it was a good approximation of the final eigenspace. The incoming training images in the later stages were just refining the current eigenspace. Consequently, the results have improved. MSRE produced by *incArnd* and *incXrnd* approaches, are only 3.1% and 1.3% worse than the results of the batch method, respectively.

Fig. 4(c) shows the MSRE of all 720 images for the dimension of the eigenspace 50. One can observe, that the curves representing the incremental approach follow the curve produced by the batch method very closely without large deviations over the whole sequence of images.

All the results clearly indicate that the incremental method is almost as effective as the batch. In all experiments the *squared* reconstruction error degraded for less than 10%, which means that the coefficients are still estimated well enough for most applications. It is also evident that the sequential order influenced the results. What really matters is the order of the training images in the early stages of the learning process. To obtain very good results, these images should be heterogeneous, encompassing different objects and views. This assures that the evolving eigenspace is rich and comprehensive enough in the beginning of the learning process already and that it is not specialized for representing a specific object only. In this way, the eigenspace can be adapted to the images of all objects more effectively.

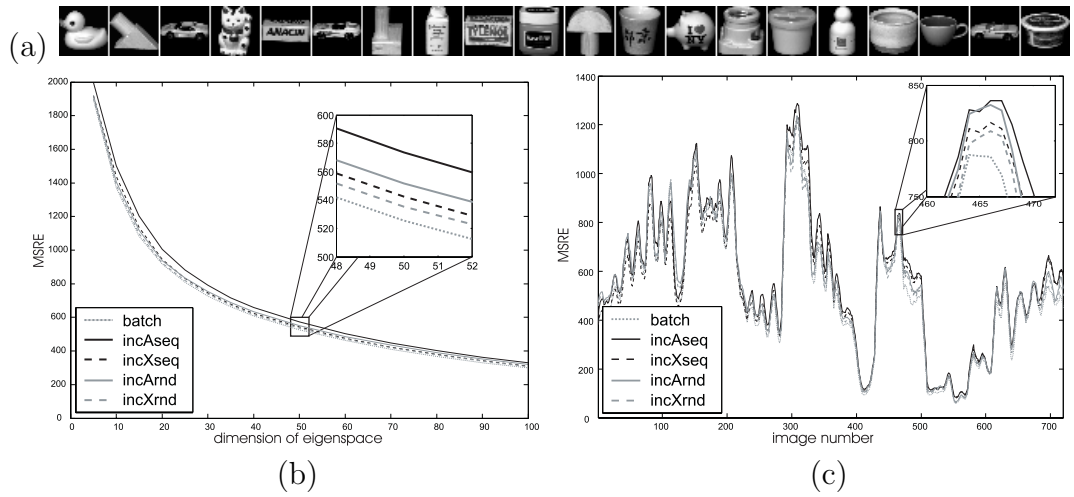


Fig. 4. (a) Training images. MSRE produced by the batch and four versions of the incremental approach: (b) for various dimensions of the eigenspace, (c) for dimension 50.

5.2 Incremental weighted method

Then, we put on each image a weight, which was proportional to the second power of the image index, giving more influence to the objects and the images at the end of the image sequence. The results of *incremental and temporally weighted* method are depicted in Fig. 5. The reconstruction errors of the incremental weighted method (*WincA*, *WincX*) do not differ significantly from the results of the batch weighted method (*Wbatch*). And certainly, the results of the batch and the incremental weighted methods are better than the results of the standard methods for images with larger weights. This is also reflected in better weighted squared reconstruction errors as presented in Table 1.

Table 1
Weighted reconstruction errors of batch and incremental methods.

batch	*incA*	*incX*	*Wbatch*	*WincA*	*WincX*
617	658	648	554	583	565

Next, we present the results of the proposed *incremental method for learning from incomplete data* to improve the results of the *visually-based localization* of a mobile robot [43]. In the learning stage, the representation of the environment (our lab in this case) is built from panoramic images taken from several locations. We can simulate the in-plane robot rotation by shifting cylindrical panoramic images and generating spinning images [44]. Three such views of two locations are depicted in Figs. 6(a,b). We thus obtain all necessary views of the environment, which are used for building the representation using PCA. Later, in the localization stage, a novel image is taken and projected into the eigenspace. The location of the robot is determined by searching for the closest

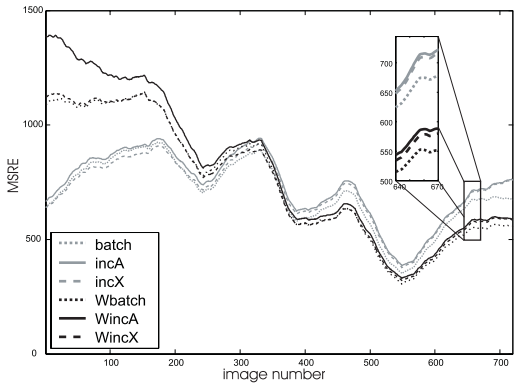


Fig. 5. Reconstruction errors of batch and incremental standard and weighted methods.

projected training image.

However, due to the construction of the panoramic sensor, not only the environment is captured in the image, but also the holder of the panoramic mirror (the dark vertical bar in Figs. 6(a,b)) and the surface of the robot (lower part of the images). If the robot is oriented differently in the localization stage, the holder appears in a different position in the image, which makes the test image less similar to the correct training image and the localization can fail.

The proposed method offers a solution to this problem. Since we know, that the holder is not a part of the environment, we can mask it out during the learning, and learn only the parts, which belong to the environment. We can achieve this by using the incremental method for learning from incomplete data considering undesirable parts as missing pixels (see Figs. 6(c)).

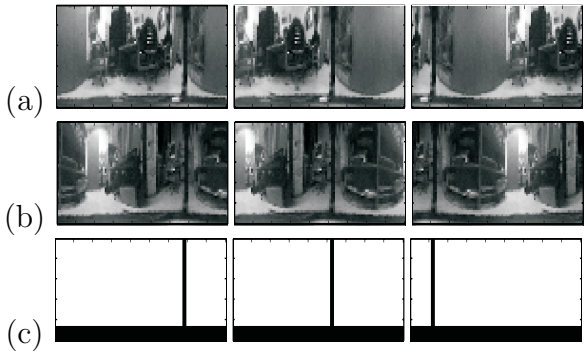
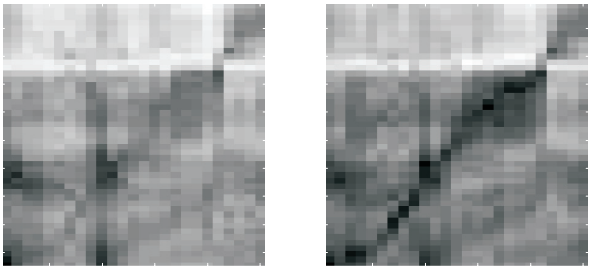


Fig. 6. (a,b) Spinning images from two locations. (c) Weights.

In the learning stage, the robot was moving from one part of the lab to the other. In the localization stage, the robot returned to the starting position following approximately the same path in the opposite direction. The results are presented in Fig. 7. The gray levels represent coefficient errors; i.e., the distances between the projections of the test images (given in the x axis) and the projections of the training image (y axis). Since the path of the robot was approximately the same as in the learning stage, we expect that the coeffi-

cient error would be minimal on the diagonal of the error matrix. Since the standard approach incorporated in the representation also the vertical holder, which was in a different image position in the localization stage, the results of the standard method are not very good (the diagonal of the error matrix in Fig. 7(a) is very indistinct). In contrast, the proposed method did not incorporate the holder into the representation of the environment. Consequently, the values around the diagonal in Fig. 7(b) are significantly smaller, which makes the localization much more accurate and reliable.



(a) (b)

Fig. 7. Coefficient errors using (a) standard and (b) proposed method.

5.3 Incremental robust method

We will demonstrate the performance of the robust incremental algorithm on the problem of the background modelling. The goal of the background modeling is to build a model of the background by detecting and discarding the objects (foreground) in a sequence of images [45,37,38]. Due to its incremental nature and simplicity the proposed incremental PCA is very well suited for solving such type of problems.

In order to obtain quantitative results of the proposed robust incremental method, we first tested its performance on the images with *known ground truth*. We synthetically applied gradual illumination changes and nonlinear illumination changes (a shadow—the vertical "cloud") to a set of 100 images. In addition, we added, as an outlier area, a square on a randomly chosen position in 80% of the images (see Fig. 8(a), the first row). The goal was to learn the representation capturing the illumination variations (linear and nonlinear) but discarding the outliers.

We tested several approaches to exhibit some properties of the proposed method. The results are given using two measures. The first measure is the mean squared reconstruction error of the reconstructed outlier-free (ground truth) images (Table 2). Besides MSRE, a precision/recall curve is given for each method in Fig. 8(b). In addition, some reconstructed training images are visualized in Fig. 8(a) (rows two to five).

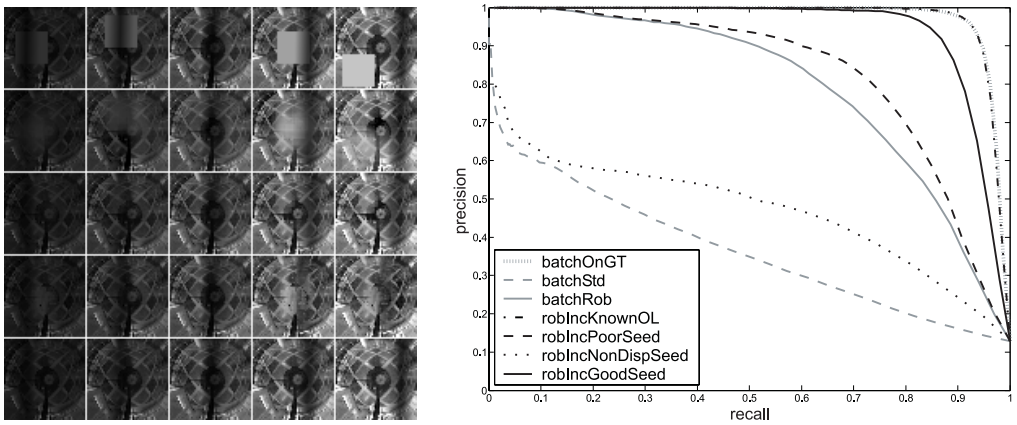
First, we applied the standard batch method on ground truth images, i.e., training images without outliers (in the table and plot indicated as *batchOnGT*), which produced optimal results. Then, we applied the standard batch method (*batchStd*) on the training images containing outliers, which generated poor results, since the standard method is sensitive to occlusions. Next, we applied the robust batch method [38] (*batchRob*), which produced better results. However, since significant occlusions were present in the training images, the results were still not satisfactory.

Then, we tested the proposed robust incremental method. First we applied this method under the assumption, that the occlusions were known and regarded as missing pixels (*robIncKnownOL*). The results are excellent; they are very close to the optimal ones. This means that the algorithm for updating the eigenspace works fine even if some data in the input images are missing and that the efficiency of the robust incremental algorithm mainly depends on the ability to detect outliers. It turns out that this ability significantly depends on the initial stage of the learning process. If the seed (the initial eigenspace, which is used for the initialization of the incremental algorithm) is not reliable and is affected by occlusions (*robIncPoorSeed*), the results of the proposed 'hard' robust incremental method are not very good. If the seed is too small and is built from the training images, which are not dispersed over the whole image sequence (*robIncNonDispSeed*), the results are very poor. To demonstrate this, we built a seed using a few images from the first half of the image sequence. Consequently, the first half of the images were reconstructed well, however the images from the end of the sequence were reconstructed poorly. Since not even a rough appearance of these images was encompassed in the initial eigenspace, all the changes in these images were considered to be outliers and were not added to the representation. For this reason, the vertical cloud was not modelled correctly as can be observed in the fourth row of Fig. 8(a). At last, we built the seed from the images with the lowest reconstruction errors (images without outliers), which were evenly dispersed over the whole image sequence (*robIncGoodSeed*). This approach produced excellent results, which are rather close to the optimal ones. This indicates that when the eigenspace, which is being updated, is stable enough, i.e., roughly encompassing different views of objects or scenes, the outliers in the training images are successfully detected and correctly reconstructed.

Table 2
MSRE obtained using different learning methods and seeds.

batch			*robInc*			
OnGT	*Std*	*Rob*	*KnownOL*	*PoorSeed*	*NonDispSeed*	*GoodSeed*
1.7	61.1	29.8	2.0	21.2	166.0	2.9

Then we performed the experiments on the real-wold PETS'2001 training



(a) (b)

Fig. 8. (a) From top to bottom: Training images, reconstructions using *batchStd*, *batchRob*, *robIncNonDispSeed*, and *robIncGoodSeed* approaches, respectively. (b) Precision/recall curves.

sequences [5] . Six images from one sequence are depicted in Fig. 9(a). The goal was to detect pedestrians, cars, and bikers, which are crossing the scene and to adapt the background model accordingly. We built the eigenbackground model consisting of eight eigenvectors. The backgrounds estimated at six time steps of the modelling process (i.e., the reconstructed training images, which were processed in those moments) obtained using three different approaches are presented in Figs. 9(b-d).

First we applied the proposed non-robust incremental method. One could expect that the outliers (pedestrians and cars), which significantly differ from the background, are considered as noise and are modeled with the eigenvectors corresponding to small eigenvalues and as such are not included in the principal subspace representations. However, this is not true in general; one can observe that the cars are still included in the background model in the third and fourth image in Fig. 9(b).

Then we applied the 'hard' robust method. This method successfully detected the pedestrians and cars, reconstructed their values and excluded them from the representation (Fig. 9(c)). In the subsequence of images around the images presented in the third and the fourth columns in Fig. 9 one car leaves the scene and another car parks in the spare lot. This changes are detected as 'foreground' and do not affect the background model. Using the 'hard' robust procedure, the background adapts only to smooth changes, which are not detected as outliers.

If a more flexible model is required, we can use the temporally weighted 'soft' robust method. In this case, the outliers are only down-weighted and are not

[5] The images are publicly available on http://www.visualsurveillance.org/PETS2001.

completely replaced. As a consequence, they are not included in the model, if they appear only for a short period of time, however, if they appear for a longer period, they are gradually incorporated in the model of the background. This is evident from the last two images in Fig. 9(d). The car, which has left the scene is not a part of the background any more, while the new car, which has parked in the spare lot, has been integrated into the current background. In this way, the eigenbackground model can be more adaptive, accommodating to the current appearance of the scene.

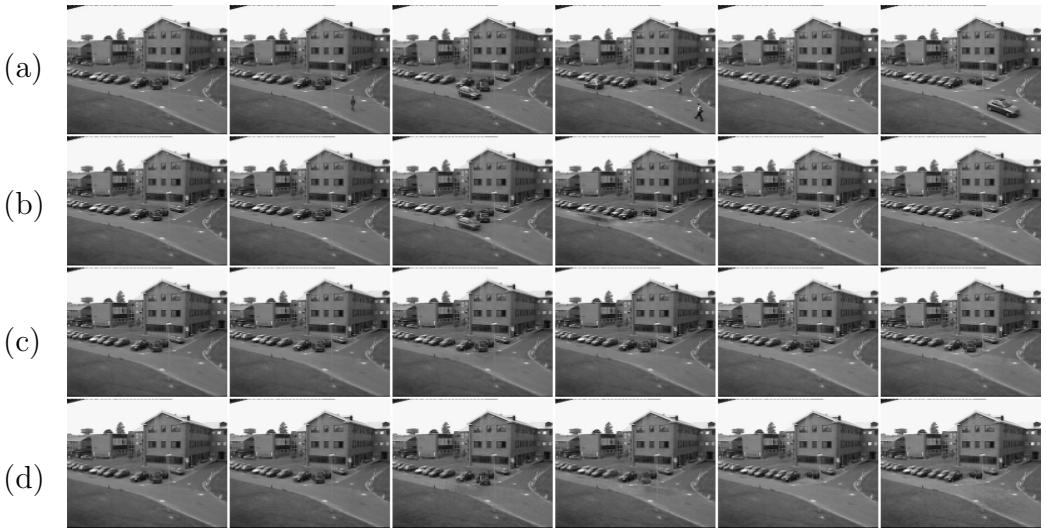


Fig. 9. (a) Six input images from PETS'2001 training sequence. Background extracted by (b) non-robust IPCA, (c) 'hard' robust IPCA, and (d) temporally weighted 'soft' robust IPCA.

6 Conclusion

Learning is a fundamental capability of any cognitive vision system. In order to enable efficient operation of a cognitive agent in a real-world environment, visual learning has to be a continuous and robust process. Learning should be an incremental, open-ended, life-long process, which keeps continuously updating the representations by adapting them to the changes in the changing world. At the same time, this process should also be robust; it should be able to filter out undesirable input signals and to update the representations using relevant data only. It is important that regardless of the type of representation employed, a cognitive vision system should allow incremental and robust learning. After discussing these requirements in general in the beginning of this article, we then focused on the subspace-based representations, which in their original form do not allow continuous nor robust learning. To overcome these shortcomings, we extended the standard PCA approach.

We proposed a novel subspace method for weighted and robust incremental learning. The proposed incremental algorithm for PCA has the same general advantages over the batch method as other previously reported incremental approaches: it is significantly faster when the number of training images is high, and it enables updating the current eigenspace to allow for on-line learning. In addition, there are two advantageous features that make our method fundamentally distinct. Firstly, our method maintains the coefficients throughout the process of learning, thus the original images can be discarded immediately after the update. For some applications with a limited amount of memory resources (e.g., mobile platforms, wearable computing) this may be the only option. Using other methods, the images have to be kept in the memory until the end of the learning process, if we want to obtain their representations in the final eigenspace. And secondly, since our method maintains the coefficients of all images, it can be advanced into a weighted method, which considers an arbitrary temporal weight at each image at every step. Furthermore, the proposed weighted method also handles spatial weights, which can be set for each pixel in every image separately. Finally, by adding the robust preprocessing step, the method is suited for visual learning in non-ideal training conditions as well. Due to its incremental nature, this method for robust learning of eigenspaces is significantly faster than previously proposed batch methods.

The method is suitable for continuous on-line learning, where the model adapts to input images as they arrive. The algorithm is flexible, since it is able to treat each pixel and each image differently. Therefore, more recent (or more reliable, or more informative, or more noticeable) images can have a stronger influence on the model then others. The principles of short-term and long-term memory, forgetting, and re-learning can be implemented and investigated. These topics are the subject of our ongoing research along with applying these principles to other types of representations.

Acknowldegment

This research has been supported in part by the following funds: Research program Computer Vision P2-0214 (RS), EU FP6-004250-IP project CoSy, EU FP6-511051-2 project MOBVIS, CONEX project, and SI-A project.

References

[1] D. Vernon et al. A research roadmap of cognitive vision. Technical report, ECVision: European research network for cognitive computer vision systems,

http://www.ecvision.org/research_planning/Research_Roadmap.htm, February 2005.

[2] G. Granlund. Cognitive vision background and research issues. Technical report, Linköping University, Computer Vision Laboratory, http://www.ecvision.org/research_planning/CVResearchIssues.pdf, November 2002.

[3] M. Weber, M. Welling, and P. Perona. Unsupervised learning of models for visual object class recognition. In *ECCV 2000*, pages 18–32, 2000.

[4] B. Schiele and J. L. Crowley. Recognition without correspondence using multidimensional receptive field histograms. *IJCV*, 36(1):31–50, January 2000.

[5] D.G. Lowe. Local feature view clustering for 3D object recognition. In *CVPR 2001*, pages I:682–688, 2001.

[6] S. Agarwal and D. Roth. Learning a sparse representation for object detection. In *ECCV 2002*, pages 113–130, 2002.

[7] R. Fergus, P. Perona, and A. Zisserman. Object class recognition by unsupervised scale-invariant learning. In *CVPR 2003*, pages II: 264–271, 2003.

[8] B. Leibe and B. Schiele. Scale invariant object categorization using a scale-adaptive mean-shift search. In *DAGM 2004*, pages 145–153, Aug. 2004.

[9] V. Ferrari, T. Tuytelaars, and L. Van Gool. Simultaneous object recognition and segmentation by image exploration. In *ECCV 2004*, volume I, pages 40–54, May 2004.

[10] A. Opelt, M. Fussenegger, A. Pinz, and P. Auer. Weak hypotheses and boosting for generic object detection and recognition. In *ECCV 2004*, volume II, pages 71–84, May 2004.

[11] S. K. Nayar, H. Murase, and S. A. Nene. Parametric appearance representation. *Early Visual Learning*, pages 131–160, 1996.

[12] H. Hotelling. Analysis of a complex of statistical variables into principal components. *Journal of Educational Psychology*, 24:417–441, 1933.

[13] H. Murakami and V. Kumar. Efficient calculation of primary images from a set of images. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 4(5):511–515, September 1982.

[14] S. Chandrasekaran, B. S. Manjunath, Y. F. Wang, J. Winkeler, and H. Zhang. An eigenspace update algorithm for image analysis. *Graphical Models and Image Processing*, 59(5):321–332, September 1997.

[15] M. Brand. Incremental singular value decomposition of uncertain data with missing values. In *ECCV 2002*, volume I, pages 707–720, May 2002.

[16] J. R. Bunch and C. P. Nielsen. Updating the singular value decomposition. *Numerische Mathematik*, 31:111–129, 1978.

[17] M. Gu and S. T. Eisenstat. A stable and fast algorithm for updating the singular value decomposition. Tech. report YALEU/DCS/RR-966, Department of Computer Science, Yale University, New Haven, 1993.

[18] P. Hall, D. Marshall, and R. Martin. Incremental eigenanalysis for classification. In *British Machine Vision Conference*, volume 1, pages 286–295, September 1998.

[19] P. Hall, D. Marshall, and R. Martin. Merging and spliting eigenspace models. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(9):1042–1048, 2000.

[20] T. Wiberg. Computation of principal components when data are missing. In *Proc. Second Symp. Computational Statistics*, pages 229–236, 1976.

[21] H. Shum, K. Ikeuchi, and R. Reddy. Principal component analysis with missing data and its application to polyhedral object modeling. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 17(9):854–867, 1995.

[22] K. Gabriel and S. Zamir. Lower rank approximation of matrices by least squares with any choice of weights. *Technometrics*, 21(21):489–498, 1979.

[23] H. Sidenbladh, F. de la Torre, and M. J. Black. A framework for modeling the appearance of 3D articulated figures. In *AFGR00*, pages 368–375, 2000.

[24] F. De la Torre and M. J. Black. A framework for robust subspace learning. *IJCV*, 54(1):117–142, September 2003.

[25] Y. Li. On incremental and robust subspace learning. *Pattern recognition*, 37:1509–1518, 2004.

[26] X. Liu and T. Chen. Shot boundary detection using temporal statistics modeling. In *ICASSP 2002*, Orlando, FL, USA, May 2002.

[27] A. Levy and M. Lindenbaum. Sequential karhunen-loeve basis extraction and its application to images. *IEEE Trans. on Image Processing*, 9:1371–1374, June 2000.

[28] A. P. Pentland, B. Moghaddam, and T. Starner. View-based and modular eigenspaces for face recognition. In *CVPR 1994*, pages 84–91, July 1994.

[29] K. Ohba and K. Ikeuchi. Detectability, uniqueness, and reliability of eigen windows for stable verification of partially occluded objects. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 19(9):1043–1048, September 1997.

[30] H. Murase and S. K. Nayar. Image spotting of 3D objects using parametric eigenspace representation. In *SCIA95*, pages 325–332, 1995.

[31] J. L. Edwards and H. Murase. Coarse-to-fine adaptive masks for appearance matching of occluded scenes. *MVA*, 10(5-6):232–242, April 1998.

[32] M. J. Black and A. D. Jepson. Eigentracking: Robust matching and tracking of articulated objects using a view-based representation. *IJCV*, 26(1):63–84, January 1998.

[33] R. Dayhot, P. Charbonnier, and F. Heitz. Robust visual recognition of color images. *CVPR 2000*, pages 685–690, June 2000.

[34] R. P. N. Rao. Dynamic appearance-based recognition. In *CVPR 1997*, pages 540–546, 1997.

[35] A. Leonardis and H. Bischof. Robust recognition using eigenimages. *Computer Vision and Image Understanding*, 78:99–118, 2000.

[36] L. Xu and A. Yuille. Robust principal component analysis by self-organizing rules based on statistical physics approach. *IEEE Trans. Neural Networks*, 6(1):131–143, 1995.

[37] F. De la Torre and M. J. Black. Robust principal component analysis for computer vision. In *ICCV 2001*, pages I: 362–369, 2001.

[38] D. Skočaj, H. Bischof, and A. Leonardis. A robust PCA algorithm for building representations from panoramic images. In *ECCV 2002*, volume IV, pages 761–775, May 2002.

[39] H. Aanæs, R. Fisker, K. Åström, and J. M. Carstensen. Robust factorization. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 24(9):1215–1225, September 2002.

[40] D. Skočaj and A. Leonardis. Weighted and robust incremental method for subspace learning. In *ICCV 2003*, II:1494–1501, October 2003.

[41] M. Artač, M. Jogan, and A. Leonardis. Incremental PCA for on-line visual learning and recognition. In *ICPR 2002*, volume 3, pages 781–784, August 2002.

[42] D. Skočaj. *Robust subspace approaches to visual learning and recognition*. PhD thesis, University of Ljubljana, Faculty of Computer and Information Science, Ljubljana, Slovenia, February 2003.

[43] M. Artač, M. Jogan, and A. Leonardis. Mobile robot localization using an incremental eigenspace model. In *ICRA 2002*, pages 1025–1030, May 2002.

[44] M. Jogan and A. Leonardis. Robust localization using the eigenspace of spinning-images. In *IEEE Workshop on Omnidirectional Vision*, pages 37–44, 2000.

[45] N. M. Oliver, B. Rosario, and A. P. Pentland. A bayesian computer vision system for modeling human interactions. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 22(8):831–843, August 2000.

Computer Vision Winter Workshop 2006, Ondřej Chum, Vojtěch Franc (eds.)
Telč, Czech Republic, February 6–8
Czech Pattern Recognition Society

Why to Combine Reconstructive and Discriminative Information for Incremental Subspace Learning

Danijel Skočaj[1], Martina Uray[2], Aleš Leonardis[1], and Horst Bischof[2]

[1] University of Ljubljana, Faculty of Computer and Information Science, Tržaška 25, SI-1001 Ljubljana, Slovenia
danijel.skocaj@fri.uni-lj.si, ales.leonardis@fri.uni-lj.si

[2] Graz University of Technology, Institute for Computer Graphics and Vision, Inffeldgasse 16/II, 8010 Graz, Austria
uray@icg.tu-graz.ac.at, bischof@icg.tu-graz.ac.at

Abstract *In the paper we propose a novel method for incremental visual learning by combining reconstructive and discriminative subspace methods. This is achieved by embedding LDA learning and classification into the incremental PCA framework. The combined subspace consists of a truncated PCA subspace and a few additional basis vectors that encompass the discriminative information, which would be lost by the discarded principal vectors. As such it contains both sufficient reconstructive information to enable incremental learning, and the previously extracted discriminative information to enable efficient classification as well. We demonstrate that we are able to efficiently update the current model with new instances of the already learned classes as well as to introduce new classes.*

1 Introduction

Visual learning and recognition/categorization has become an important and popular research topic in the computer vision community. Several different methods have been proposed in recent years. Based on the type of object representations they use, most of them can be classified in one of two main categories: reconstructive or discriminative methods. The *reconstructive* representations strive to be as informative as possible in terms of well approximating the original data. Their goal is predominantly to encompass the variability of the training data and are as such not task-dependent. On the other hand, *discriminative* methods usually do not provide good reconstruction of the data, they are task-dependent, but spatially and computationally much more efficient and often give superior classification results compared to the reconstructive methods.

We will study the properties of these two types of methods from the perspective of *incremental learning*. Incremental learning is very often a desirable or even essential property of an artificial cognitive system. In contrast to the batch approaches, which process all training images simultaneously, incremental methods process one image after another. Thus, only one image (or maybe a few of them) is processed at each step, and only the *representations* of the previously encountered images are available; the original training images are discarded immediately after being processed. The advantages of an incremental over the batch method are that not all training images have to be given in advance (enabling online learning), that less calculation time is needed (to update a model is less expensive than to build a new model from scratch) and that less storage is required (since only representations of the images are being kept). In the case of *reconstructive* methods, these representations can be used as a good approximations of the discarded training images but in general, this does not hold true for *discriminative* methods due to the lack of the information that would enable a good reconstruction. Thus, in order to enable incremental updating of discriminative representations as well, we have to combine them with the reconstructive methods.

This is the problem that we want to address in this paper. We will create a representation, which combines the reconstructive models and discriminative classifiers. The reconstructive property of such a representation will bring sufficient redundancy in the data to enable updating of the representations, while the discriminative property of the representation would still keep the representation efficient and effective.

In this paper we focus on the Principal Component Analysis (PCA) [11] and the Linear Discriminant Analysis (LDA) [4]. PCA is a well known reconstructive method, which encompasses the reconstructive task-independent information that can approximate the training data well. LDA, on the other hand, is a discriminative method, which keeps only the discriminative task-dependent information about input images. While the LDA is recognized to be superior over the PCA in recognition tasks, it is less suitable for incremental learning due to the reasons elaborated above. Therefore we propose to combine both methods to achieve the best of both worlds.

We thus embed the LDA learning and classification into the PCA framework facilitating incremental updating of the already learned representations. The combined subspace consists of a truncated PCA subspace and a few additional basis vectors that encompass the discriminative information, which would be lost by the discarded principal vectors. As such it contains both sufficient reconstructive information to enable incremental learning, and the previously extracted discriminative information to enable efficient classification as well.

The proposed method allows for two types of updating the current representation, thus coping with different aspects of incremental learning. It is possible to add **new instances** of known classes such that the representations of the classes improve and adapt to the new appearances of the known objects/subjects improving the classification results. Additionally it is possible to add **new classes** such that previously not observed classes can be introduced and their representations can be created and then maintained through the process of incremental learning. Both types of learning that fully exploit the reconstructive and discriminative nature of the proposed method are presented and experimentally evaluated demonstrating the advantages of the proposed approach.

The paper is organized as follows. First we discuss related work in Section 2. In Section 3 we introduce the notation and define the problem, while we describe the proposed method in Section 4. To verify our claims we present the experimental results in Section 5. Finally, we summarize the paper, expose the contributions, and outline some possible extensions.

2 Related work

Several research topics are directly or indirectly related to the method presented in this paper: combining the reconstructive and discriminative methods, incremental learning, and combining different subspace methods. In this section we will discuss some of them and expose the differences with respect to the method we are proposing.

Combination of generative and discriminative methods as well as integration of representative and discriminant models have gained a lot of attention, which resulted also in a plethora of methods published very recently [6, 10, 14]. Most of these methods aim at improving the classification results or increasing the robustness. They do not, however, consider incremental learning.

Incremental learning have been better studied in the domain of subspace methods, especially the reconstructive ones. Many methods for incremental building of principal subspaces have been proposed [3, 8] and different extensions have also been introduced, such as weighted [12] and robust incremental learning [13, 22]. These methods are unsupervised and do not take into account the prior information about object labels, thus they do not exploit all information, which is available for classification.

Several methods for incremental LDA have also been already proposed [9, 15, 20, 21, 24]. Most of these methods focus on the updating of the between class scatter matrix and the within class scatter matrix, thus keeping the discriminative information only. In contrast, our method keeps updating the current representation of the images encompassing the discriminative and reconstructive information as well. The richer representation allows for updating of the acquired knowledge in a more powerful way.

PCA and LDA have often been combined in the past. Even some of the methods for incremental LDA mentioned above involve the estimation of PCA subspaces. Also in many other discriminative approaches PCA is first used as a preprocessing step for dimensionality reduction or to avoid singularity problems [1, 2, 23, 25]. In addition, many approaches aim at improving the classification power of discriminative methods by incorporating the PCA information in different ways [16, 17]. We rather focus on incremental aspects of the learning process; this is the main reason for combining LDA with PCA. In our approach the methods are tightly coupled in a principled way; the LDA-relevant information is being considered during the creation of the PCA subspace as well, resulting in a combined representation, which is the main novelty of the proposed approach.

3 Problem definition

Let n be the number of images in the training set, each of them containing m pixels, aligned in the columns of the matrix $\mathbf{X} \in \mathbb{R}^{m \times n}$, let $\mu \in \mathbb{R}^m$ be the mean image, and c the number of classes the images belong to. The goal of subspace methods is to find subspaces that transform the input data (images) in a way that enables efficient classification of novel images. Reconstructive and discriminative methods offer different solutions to this problem.

Reconstructive methods are designed to find a linear representation that best describes the input data, i.e.[1],

$$\mathbf{X} \approx \mathbf{U}_{\underline{k}} \mathbf{A}_{\underline{k}} + \mu \mathbf{1}_{1 \times n} \tag{1}$$

where k vectors in the columns of $\mathbf{U}_{\underline{k}} = [\mathbf{u}_1, \ldots, \mathbf{u}_k] \in \mathbb{R}^{m \times k}$ form the reconstructive basis and n vectors in the rows of $\mathbf{A}_{\underline{k}} = [\mathbf{a}_1^T, \ldots, \mathbf{a}_n^T]^T \in \mathbb{R}^{k \times n}$ are referred to as coefficient vectors, i.e., the k-dimensional representations of the training images.

PCA looks for a low-dimensional representation of the data which minimizes the squared reconstruction error [11]. Therefore it guaranties the best possible representation of the input images in a linear subspace of a given dimension k. PCA is thus an unsupervised method, which does not look for differences between the images belonging to different classes, but rather tries to model each image as well as possible. Hence it keeps as much information about the training images as possible, and stores it in k-dimensional representations (usually $c \ll k \ll n \ll m$). Since the model is not built for a specific task, it is general and task-independent.

Discriminative methods are designed in a different way and are particulary suited for classification tasks. They assume that prior knowledge about the classes of the training data is available, which is integrated in the supervised learning process to produce a small number of hyperplanes that are capable of separating the training data. To be more specific, the objective of discriminative methods is to find a linear function

$$g(\mathbf{x}) = \mathbf{W}^T(\mathbf{x} - \mu) \,, \tag{2}$$

where $\mathbf{W} = [\mathbf{w}_1, \ldots, \mathbf{w}_{(c-1)}] \in \mathbb{R}^{m \times (c-1)}$ is used for transforming the data into a lower-dimensional classification space upon which it is decided to which class a given sample $\mathbf{x}$ belongs.

LDA finds the projection directions on which the intra-class scatter is minimized whilst the inter-class scatter is

[1] $\mathbf{1}_{m \times n}$ denotes a $m \times n$ matrix of ones.

maximized [4]. That is, it finds $c-1$ vectors that can be used for efficiently separating the images belonging to different classes. The model is thus very compact (c is usually very small) and efficient, but it is task-dependent. And since the projections of the images in the LDA space are very low-dimensional ($(c-1)$-dimensional) and $\mathbf{W}$ is not particulary designed to encompass the reconstructive information, they can not be used for reconstruction of the training images.

The comparison between discriminative and reconstructive methods for classification tasks has been a subject of extensive research and testing [1, 18]. The general conclusion was that discriminative methods outperform the reconstructive methods. The explanation for this is rather obvious: the discriminative methods focus more on specific prior knowledge, which can thus be more efficiently integrated into the learning process. However, the latter observation can also be disadvantageous, when we want to put learning in an incremental framework. If the discriminative methods are too focused to specific discriminative features sufficient for classifying the given data, then many features, which may be useful for discriminating in the feature, get discarded, and the representations can not be adapted to new information. Since the images get discarded in the training process, and their representations are rather poor, there is not sufficient information, which would enable reconsidering the discarded training images. A new model, which would consider the discarded images and the new ones, can not be created. To overcome these deficiencies, the model should encompass also a certain level of reconstructive information.

4 Our approach

In this section we describe the algorithm for incremental updating of LDA representations (Algorithm 1). It takes the training images sequentially and computes the new representation from the current representation and the new input image.

Let n be the number of training images observed so far. The idea is to represent these n images in a way that includes both reconstructive *and* discriminative properties. Reconstructive representation is based on PCA [22]. As most of the visual variability of the images is contained in the first k principal vectors (where $k \ll n$), only the k-dimensional principal subspace is retained. However although the first k coefficients contain most of the reconstructive information, there is no guarantee that most of the discriminative information is present in them as well. In order not to lose discriminative information, we propose to augment the truncated principal subspace with $c-1$ additional basis vectors, which keep all information relevant for LDA.

Now, let us suppose that we have already built an augmented PCA subspace (APCA subspace) from the first n images. The current augmented reconstructive model therefore consists of basis vectors[2] $\widehat{\mathbf{U}}^{(n)} \in \mathbb{R}^{m\times(k+c-1)}$, mean vector $\mu^{(n)} \in \mathbb{R}^m$, and coefficient vectors $\widehat{\mathbf{A}}^{(n)} \in \mathbb{R}^{(k+c-1)\times n}$. In step $n+1$ we can calculate a new APCA subspace from the *representations* (coefficient vectors) of the first n input images and a new image as proposed in [22]. Since the dimension of the APCA subspace is small, this update is computationally very efficient. The procedure for one update of the current APCA subspace is outlined in the first eight steps of Algorithm 1.

Once we updated the current representations of the images observed so far, we can perform LDA on these updated low-dimensional coefficient vectors[3] aligned in $\mathbf{A} \in \mathbb{R}^{(k+c)\times(n+1)}$. LDA yields the discriminative representation in the form of LDA vectors aligned in $\mathbf{V} \in \mathbb{R}^{(k+c)\times(c-1)}$.

Until now, no reconstructive nor discriminative information has been lost, since all the information contained in the novel image has been incorporated into the model. However, as a consequence, the model has grown; the dimension of the APCA space has increased by one. To keep the size of the model, we have to truncate the obtained matrix $\mathbf{U} \in \mathbb{R}^{m\times(k+c)}$ (and consequently $\mathbf{A}$ and $\mathbf{V}$) by one. We propose to truncate $\mathbf{U}$ in a way, which preserves the discriminative information, similarly to [5].

Note that the classification using the combination of the reconstructive and discriminative representations is performed as a two step procedure: first a novel image is projected into the augmented PCA basis and the obtained coefficient vector is then projected onto the low-dimensional LDA vectors. The classification function is thus $g(\mathbf{x}) = \mathbf{V}^T\mathbf{U}^T(\mathbf{x}-\mu^{(n+1)})$. Now we will show how to truncate $\mathbf{U}$ and $\mathbf{V}$ by one dimension and still keep the classification function unchanged.

Let us first divide the matrices $\mathbf{U}$, $\mathbf{A}$, and $\mathbf{V}$ on submatrices containing the first k dimensions we want to keep and the last c dimensions we want to truncate by one[4] (line 10 in Algorithm 1). Then let us orthonormalize $\mathbf{V}_{\underline{c}}$ and update the APCA basis, the coefficients and the LDA vectors (lines 11 to 15 in Algorithm 1). We will show that the obtained updated representation, which is of the same size as at the beginning of the update step ($\widehat{\mathbf{U}}^{(n+1)} \in \mathbb{R}^{m\times(k+c-1)}$, $\widehat{\mathbf{A}}^{(n+1)} \in \mathbb{R}^{(k+c-1)\times(n+1)}$, and $\widehat{\mathbf{V}}^{(n+1)} \in \mathbb{R}^{(k+c-1)\times(c-1)}$) preserves the discriminative information. To verify this, let us rewrite the new classification function $\widehat{g}(\mathbf{x}) := \widehat{\mathbf{V}}^{(n+1)T}\widehat{\mathbf{U}}^{(n+1)T}(\mathbf{x}-\mu^{(n+1)})$ as

$$\begin{aligned}
\widehat{g}(\mathbf{x}) &= (\widehat{\mathbf{U}}^{(n+1)}\widehat{\mathbf{V}}^{(n+1)})^T(\mathbf{x}-\mu^{(n+1)}) = \\
&= \left(\left[\mathbf{U}_{\underline{k}}, \mathbf{U}_{\underline{c}}\widetilde{\mathbf{V}}_{\underline{c}}\right]\begin{bmatrix}\mathbf{V}_{\underline{k}} \\ (\mathbf{V}_{\underline{c}}^T\mathbf{V}_{\underline{c}})^{1/2}\end{bmatrix}\right)^T(\mathbf{x}-\mu^{(n+1)}) = \\
&= \left[\mathbf{U}_{\underline{k}}\mathbf{V}_{\underline{k}} + \mathbf{U}_{\underline{c}}\mathbf{V}_{\underline{c}}\right]^T(\mathbf{x}-\mu^{(n+1)}) = \\
&= \left(\left[\mathbf{U}_{\underline{k}}, \mathbf{U}_{\underline{c}}\right]\begin{bmatrix}\mathbf{V}_{\underline{k}} \\ \mathbf{V}_{\underline{c}}\end{bmatrix}\right)^T(\mathbf{x}-\mu^{(n+1)}) = \\
&= (\mathbf{U}\mathbf{V})^T(\mathbf{x}-\mu^{(n+1)}) = \\
&= g(\mathbf{x})\,. \qquad (3)
\end{aligned}$$

[2] A superscript denotes the step which the data is related to ($\widehat{\mathbf{U}}^{(n)}$ denotes the values of $\widehat{\mathbf{U}}$ at the step n).

[3] Note that also in the standard LDA (fisher space) approaches, LDA is performed on the vectors of the PCA coefficients and not on the original images to avoid the singularity problems LDA encounters when dealing with high-dimensional data such as images. However, the complete vectors of principal coefficients are used in these cases.

[4] We denote the first k *columns* of $\mathbf{U}$, the first k *rows* of $\mathbf{A}$, and the first k *rows* of $\mathbf{V}$ with $\mathbf{U}_{\underline{k}}$, $\mathbf{A}_{\underline{k}}$, and $\mathbf{V}_{\underline{k}}$, respectively, and the last c *columns* of $\mathbf{U}$ and *rows* of $\mathbf{A}$ and $\mathbf{V}$ with $\mathbf{U}_{\underline{c}}$, $\mathbf{A}_{\underline{c}}$, and $\mathbf{V}_{\underline{c}}$, respectively.

It is therefore equivalent to the original classification function $g(\mathbf{x})$, thus all the discriminative information has been preserved.

Algorithm 1 : ILDA – incremental LDA

Require: current augmented principal subspace (mean vector $\mu^{(n)}$, APCA vectors $\widehat{\mathbf{U}}^{(n)}$, APCA coefficients $\widehat{\mathbf{A}}^{(n)}$) and new input image $\mathbf{x}^{(n+1)}$.

Ensure: new augmented principal subspace (mean vector $\mu^{(n+1)}$, APCA vectors $\widehat{\mathbf{U}}^{(n+1)}$, coefficients $\widehat{\mathbf{A}}^{(n+1)}$), new low-dimensional LDA vectors $\widehat{\mathbf{V}}^{(n+1)}$ and class centers $\boldsymbol{\nu}^{(n+1)}$.

1: Project a new image $\mathbf{x}^{(n+1)}$ into the current eigenspace: $\mathbf{a} = \widehat{\mathbf{U}}^{(n)\top}(\mathbf{x}^{(n+1)} - \mu^{(n)})$.

2: Reconstruct the new image: $\mathbf{y} = \widehat{\mathbf{U}}^{(n)}\mathbf{a} + \mu^{(n)}$.

3: Compute the residual vector: $\mathbf{r} = \mathbf{x}^{(n+1)} - \mathbf{y}$.

4: Append $\mathbf{r}$ as a new basis vector: $\mathbf{U}' = \left[\begin{array}{cc} \widehat{\mathbf{U}}^{(n)} & \frac{\mathbf{r}}{\|\mathbf{r}\|} \end{array} \right]$.

5: Determine the coefficients in the new basis: $\mathbf{A}' = \left[\begin{array}{cc} \widehat{\mathbf{A}}^{(n)} & \mathbf{a} \\ \mathbf{0} & \|\mathbf{r}\| \end{array} \right]$.

6: Perform PCA on $\mathbf{A}'$. Obtain the mean value μ'' and the eigenvectors $\mathbf{U}''$.

7: Project the coefficient vectors to the new basis: $\mathbf{A} = \mathbf{U}''^{\top}(\mathbf{A}' - \mu''\mathbf{1}_{1\times(n+1)})$.

8: Rotate the subspace $\mathbf{U}'$ for $\mathbf{U}''$: $\mathbf{U} = \mathbf{U}'\mathbf{U}''$.

9: Perform LDA on $\mathbf{A}$. Obtain low-dimensional LDA vectors $\mathbf{V}$ and class centers $\boldsymbol{\nu}$.

10: Divide $\mathbf{U}$, $\mathbf{A}$, and $\mathbf{V}$ on submatrices: $\mathbf{U} = \left[\begin{array}{cc} \mathbf{U}_{\underline{k}} & \mathbf{U}_{\underline{c}} \end{array} \right], \mathbf{A} = \left[\begin{array}{c} \mathbf{A}_{\underline{k}} \\ \mathbf{A}_{\underline{c}} \end{array} \right], \mathbf{V} = \left[\begin{array}{c} \mathbf{V}_{\underline{k}} \\ \mathbf{V}_{\underline{c}} \end{array} \right]$.

11: Orthonormalize $\mathbf{V}_{\underline{c}}$: $\widetilde{\mathbf{V}}_{\underline{c}} = \mathbf{V}_{\underline{c}}(\mathbf{V}_{\underline{c}}^T\mathbf{V}_{\underline{c}})^{-1/2}$.

12: Update the mean: $\mu^{(n+1)} = \mu^{(n)} + \mathbf{U}'\mu''$.

13: Update the APCA basis: $\widehat{\mathbf{U}}^{(n+1)} = \left[\begin{array}{cc} \mathbf{U}_{\underline{k}} & \mathbf{U}_{\underline{c}}\widetilde{\mathbf{V}}_{\underline{c}} \end{array} \right]$.

14: Update the coefficients: $\widehat{\mathbf{A}}^{(n+1)} = \left[\begin{array}{c} \mathbf{A}_{\underline{k}} \\ \widetilde{\mathbf{V}}_{\underline{c}}^T\mathbf{A}_{\underline{c}} \end{array} \right]$.

15: New LDA vectors: $\widehat{\mathbf{V}}^{(n+1)} = \left[\begin{array}{c} \mathbf{V}_{\underline{k}} \\ (\mathbf{V}_{\underline{c}}^T\mathbf{V}_{\underline{c}})^{1/2} \end{array} \right]$.

16: New class centers: $\boldsymbol{\nu}^{(n+1)} = \boldsymbol{\nu}$.

Using Algorithm 1 we can thus update the current reconstructive and discriminative representations without losing a valuable discriminative information and without enlarging the reconstructive basis. Since at each step LDA is recalculated using low-dimensional APCA representations of the training images, the update step is fast, while still enabling various types of updating.

The model can be updated with an image of a known class making the representation of this class more reliable. Additionally an image of a novel class can be introduced. In this case, a new class is initialized, c is incremented by one, and consequently all the matrices keeping the representations are enlarged by one dimension. The performance of the proposed method for these approaches is evaluated in the next section.

5 Experimental results

In the experiments we focus on comparison of the proposed method, which combines both reconstructive and discriminative information, with the methods that exploit only one type of information. We also always show the performance of the standard batch LDA method (denoted as *batchLDA*) giving the best results because it processes all training images simultaneously, therefore it can find the hyperplane which is optimally suitable for the given data. Thus our aim is to achieve similar results with the incremental training.

The idea of incremental learning is to start with a given model (denoted as starting model) and update it when new information is available. In the following we compare three different approaches differing in the usage of discriminative and reconstructive information.

ILDAonK is the incremental LDA based on a truncated PCA basis keeping only k PCA-eigenvectors. It thus predominantly contains the reconstructive information, while some important discriminative information may be discarded. On the other hand, *ILDAonL* does not keep any additional reconstructive information. The training images are represented only by $(c-1)$-dimensional LDA coefficient vectors, which are propagated in the updating steps. Finally, *ILDAaPCA*, the proposed method, combines both types of representations keeping the reconstructive as well as the discriminative information.

In all experiments the dimension k of the truncated PCA space is fixed such that the starting model contains 80% of the energy (a fraction of the total variance). Since in the case of *ILDAaPCA* method the truncated principal subspace is augmented by $(c-1)$ basis vectors, we do not truncate the principal subspace in *ILDAonK* approach at k but rather at $k + (c-1)$ to enable a fair comparison. In this way both approaches produce representations of the same size.

In the following we will show that *ILDAaPCA* is actually capable of facing two challenges. It is possible to add images of already known categories and to add new categories.

We will test the above described methods on the precropped Sheffield Face Database [7]. It consists of 20 persons with at least 19 images of each individual and the images cover poses from profile to frontal views. We took 9 images (every second one) of each person for training (e.g., see Figure 1) and 10 images for testing.

Figure 1: Training images for one person in the Sheffield Face Database.

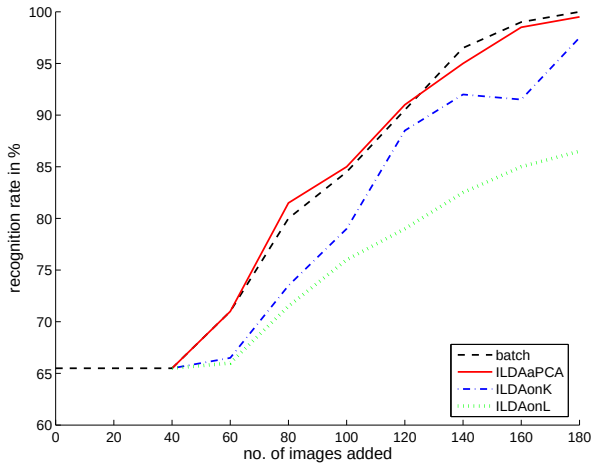


Figure 2: Comparison of the recognition rate on Sheffield Face Database of *batchLDA*, *ILDAaPCA*, *ILDAonK* and *ILDAonK*

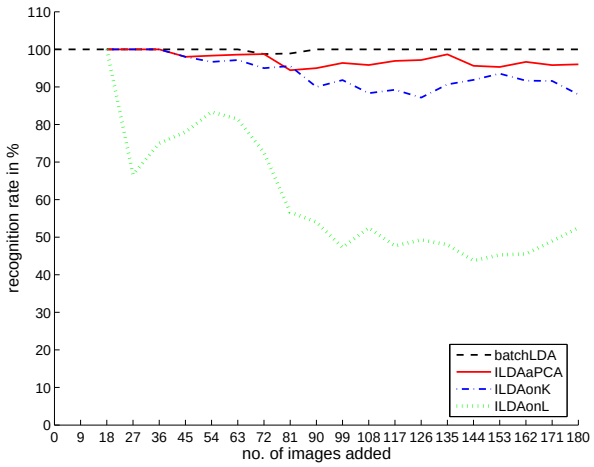


Figure 3: Recognition rate for test images of already trained classes on Sheffield Face Database

Adding new instances: To build the starting model for the first task we took two images of each class, having 40 images at the beginning and added 140 images, 7 of each class, in the sequential updating steps.

For *ILDAaPCA*, *ILDAonK* and *ILDAonK* the eigenspace was updated after each image was presented, while for *batchLDA* the LDA space was always built from scratch using the current number of training images. In both cases the recognition rate was calculated after adding 20 images, one of each class, and repeated 7 times.

As can be seen in Figure 2, the recognition rate keeps growing with increasing number of training images. This demonstrates that new images bring additional knowledge in the model and improve the current representations resulting in a better performance of the classifier. It is also evident that *ILDAaPCA* clearly outperforms *ILDAonK* and *ILDAonK* being nearly as good as *batchLDA*. As one could expect, *ILDAonK* yields the worst results, since the very low-dimensional discriminative representations do not suffice for updating the model. We can also conclude that *ILDAonK* approach discards some discriminative information and produce results inferior to *ILDAaPCA* method, which preserves this discriminative information.

Adding new classes: Here we started with a basis created from all training images of two subjects (18 images altogether) and then added new faces one by one. The model was updated with all the training images of the new class before adding the next one. We classified only those test images for which the model of the corresponding class was already built.

The results are displayed in Fig. 3. As expected, the recognition rate drops a little bit in all approaches by increasing the number of classes, since it is more difficult to discriminate between 20 classes then between only a few of them. However, the results again clearly demonstrate that the proposed *ILDAaPCA* method outperforms *ILDAonK* and *ILDAonK* approaches.

To demonstrate that our method works for different tasks too, we present the results of an object recognition task. The experiment was performed on the Columbia image database COIL20 [19]. It consists of 20 objects with 72 gray scale images of views from 0 to 360 degrees in 5 degree steps. For our tests we took 14 images for training and the remaining 58 for testing.

We started again with a basis created from two images of each class, and then added the remaining images in $12 * 20$ update steps. As can be seen in Figure 4 the behavior of the curves is similar to the experiment on faces, again showing that *ILDAaPCA* achieves the highest recognition rate.

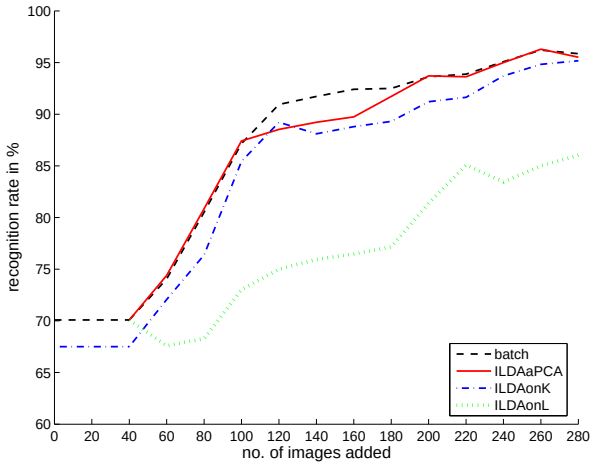


Figure 4: Comparison of the recognition rate on COIL20 database

6 Conclusion

In this paper we proposed a method that combines reconstructive models and discriminative classifiers to enable updating of the already learned representations. To achieve that, we enrich the discriminative LDA representations with reconstructive information. This is realized by embedding the LDA learning and classification into an augmented

PCA subspace enabling incremental updating of the already learned representations without discarding significant discriminative information.

The augmented PCA subspace thus contains sufficient reconstructive information, which enables incremental learning, and the previously extracted discriminative information, which enables efficient classification. In this way we are able to efficiently update the current model with new instances of the already learned classes and to introduce new classes. In addition, this method could in principle also enable updating the current model to new tasks. Moreover, the reconstructive representation would also enable detection of outliers [5, 22], thus the proposed method could be further extended in a robust approach for incremental learning of LDA representations. This technique could also be applied to other reconstructive and discriminative linear subspace methods (Independent Component Analysis, Non-negative Matrix Factorization; Canonical Correlation Analysis, Support Vector Machines). The combination of reconstructive and discriminative methods thus offers a promise to achieve best of both worlds; to enable successful discrimination using efficient task-dependent discriminative representations, while at the same time enabling robustness, and adaptation to new images using the reconstructive representations.

Acknowledgement

This research has been supported in part by the following funds: Research program Computer Vision P2-0214 (Slovenian Ministry of Higher Education, Science and Technology), EU FP6-004250-IP project CoSy, EU FP6-511051 project MOBVIS, EU FP6-507752 NoE MUSCLE IST and CONEX project.

References

[1] P. N. Belhumeur, J. P. Hespanha, and D. J. Kriegman. Eigenspaces vs. fisherfaces: Recognition using class specific linear projection. *IEEE PAMI*, 19(7):711–720, 1997.

[2] M. Borga and H. Knutsson. Canonical correlation analysis in early vision processing. In *Proc. of the 9th European Symposium on Artificial Neural Networks*, 2001.

[3] S. Chandrasekaran, B. S. Manjunath, Y. F. Wang, J. Winkeler, and H. Zhang. An eigenspace update algorithm for image analysis. *Graphical Models and Image Processing*, 59(5):321–332, 1997.

[4] R. O. Duda, P. E. Hart, and D. G. Stork. *Pattern classification*. Wiley, 2001.

[5] S. Fidler, D. Skočaj, and A. Leonardis. Combining reconstructive and discriminative subspace methods for robust classification and regression by subsampling. *IEEE PAMI*, 28(3):337-350, 2006.

[6] M. Fritz, B. Leibe, B. Caputo, and B. Schiele. Integrating representative and discriminative models for object category detection. In *ICCV05*, II:1363–1370, 2005.

[7] D. B. Graham and N. M. Allinson. In *Face recognition: From theory to applications*, *NATO ASI Series F, Computer and Systems Sciences*, 163:446 – 456. 1998.

[8] P. Hall, D. Marshall, and R. Martin. Merging and splitting eigenspace models. *IEEE PAMI*, 22(9):1042–1048, 2000.

[9] K. Hiraoka, S. Yoshizawa, K. Hidai, M. Hamahira, H. Mizoguchi and T. Mishima. Convergence analysis of online linear discriminant analysis. In *Proc. of the IEEE–INNS–ENNS International Joint Conference on Neural Networks (IJCNN 2000)*, 3:387 – 391, 2000.

[10] A. Holub, M. Welling, and P. Perona. Combining generative models and fisher kernels for object recognition. In *ICCV05*, I:136–143, 2005.

[11] H. Hotelling. Analysis of a complex of statistical variables into principal components. *Journal of Educational Psychology*, 24:417–441, 1933.

[12] A. Levy and M. Lindenbaum. Sequential karhunen-loeve basis extraction and its application to images. *IEEE Trans. on Image Processing*, 9(8):1371–1374, 2000.

[13] Y. Li. On incremental and robust subspace learning. *Pattern recognition*, 37(7):1509–1518, 2004.

[14] Y. Li, L. Shapiro, and J. Bilmes. A generative/discriminative learning algorithm for image classification. In *ICCV05*, II:1605–1612, 2005.

[15] R. Lin, M. Yang and S. E. Levinson. Object tracking using incremental fisher discriminant analysis. In *ICPR04*, 2:757 – 760, 2004.

[16] X. Lu, Y. Wang, and A.K. Jain. Combining classifiers for face recognition. In *ICME 2003*, III:13 – 16, 2003.

[17] G. L. Marcialis and F. Roli. Fusion of PCA and LDA for face verification. In *Proc. of Post-ECCV Workshop on Biometric Authentication (BIOMET)*, pp. 30 – 37, 2002.

[18] A. M. Martinez and A. C. Kak. PCA versus LDA. *IEEE PAMI*, pages 23(2):228 – 233, 2001.

[19] S. A. Nene, S. K. Nayar and H. Murase. Columbia object image library (COIL-20). Technical Report CUCS-005-96, Columbia University, 1996.

[20] S. Pang, S. Ozawa and N. Kasabov. Chunk incremental LDA computing on data streams. In *Advances in Neural Networks ISNN 2005*, 3497:51 – 56, 2005.

[21] S. Pang, S. Ozawa and N. Kasabov. Incremental linear discriminant analysis for classification of data streams. *IEEE Transactions on Systems, Man and Cybernetics, Part B*, 35(5):905 – 914, 2005.

[22] D. Skočaj and A. Leonardis. Weighted and robust incremental method for subspace learning. *ICCV03*, II:1494–1501, 2003.

[23] J. Yang and J.-Y. Yang. Why can LDA be performed in PCA transformed space?. *Pattern recognition*, 36:685–690, 2003.

[24] J. Ye, Q. Li, H. Xiong, H. Park, R. Janardan and V. Kumar. IDR/QR: An incremental dimension reduction algorithm via QR decomposition. In *Proc. of the 10th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD 2004)*, pp. 364 – 373, 2004.

[25] W. Zhao, A. Krishnaswamy, R. Chellappa, D. Swets, and J. Weng. Discriminant analysis of principal components for face recognition. *Face Recognition: From Theory to Applications*, pp. 73 – 85, 1998.

Efficient clustering and matching for object class recognition

Bastian Leibe
ETH Zurich
Zurich, Switzerland

Krystian Mikolajczyk
University of Surrey
Guildford, UK

Bernt Schiele
TU Darmstadt
Darmstadt, Germany

Abstract

In this paper we address the problem of building object class representations based on local features and fast matching in a large database. We propose an efficient algorithm for hierarchical agglomerative clustering. We examine different agglomerative and partitional clustering strategies and compare the quality of obtained clusters. Our combination of partitional-agglomerative clustering gives significant improvement in terms of efficiency while maintaining the same quality of clusters. We also propose a method for building data structures for fast matching in high dimensional feature spaces. These improvements allow to deal with large sets of training data typically used in recognition of multiple object classes.

1 Introduction

Many of todays models and approaches for object class recognition are based on local features. Local features are typically extracted from images and subsequently grouped into appearance clusters [1, 12, 23]. Besides reducing the size of the feature space, appearance clusters allow to capture a larger variability of local image structure than individual features, as well as to focus on parts which re-occur on many instances of the object class and consequently generalize over new instances. While appearance clusters seem to be an essential component of several successful approaches, they have been applied only to a relatively small number of object classes using small training and test sets. A typical feature detector might extract 10s-100s of features per image. Learning models for 100s of object classes using 10s or even 100s of images per class would imply that the approach has to deal with 100,000s to 1,000,000s of features during training. This number is but a conservative estimate, since we might want to scale to many more object classes or use far more images in the context of unsupervised learning or topic discovery [24]. It is however unclear if and how approaches based on appearance clusters can deal with such massive amounts of high-dimensional data.

In general, clustering is a powerful tool for finding structure in large data sets [10]. However, the question what is a good clustering method cannot be answered without the context of a task. Our main interest is the use of clustering for object class recognition using local-feature based approaches. When using appearance clusters for building object models we can differentiate three aspects: 1) *clustering* to obtain the appearance clusters, 2) *matching* during training and recognition, and 3) the *recognition* method. In this paper we focus on the first two aspects, namely *clustering* and *matching*.

In computer vision, frequently used clustering strategies are k-means [23, 26] and agglomerative clustering [1, 12, 17]. Other methods like Mean Shift [6] also become more and more popular. However, their performance has not been compared for computer vision tasks, and no guidelines are available for judging the tradeoffs in representational

capacity, accuracy, and run-time. K-means is frequently used because of its computational simplicity. However, the clustering solution is suboptimal when the number of outliers in the point distribution is large. Moreover, the solution depends on an arbitrarily set number of clusters and random initialization, which makes it less attractive for object categorization. In the agglomerative clustering scheme the number of clusters is automatically determined. However, both the runtime and memory requirements are often significantly higher for agglomerative methods. Given the large amounts of data that need to be processed, an efficient implementation of the clustering algorithm is therefore crucial for its applicability.

The second important aspect is to efficiently match features to appearance clusters. Numerous methods have been proposed for efficient search [3, 19]. In high-dimensional spaces, however, these methods are no longer effective. Many methods therefore use approximate nearest-neighbor search techniques [2, 9, 14]. In object recognition and categorization, however, we are interested in matches within a similarity distance to a feature point. This type of volume search is much harder to do efficiently. In contrast to k-means clustering, the result of agglomerative clustering can be used directly to obtain a data structure for efficient volume search, namely a ball-tree [20]. In experiments we observe speedup factors of 20- 200 for matching through the use of this technique.

In this paper we introduce several improvements to agglomerative clustering (Sec. 3), making it tractable for large data sets while preserving cluster quality. We use the clustering results to build a data structure for efficient volume search in high-dimensional spaces (Sec. 4). In the experiments we show significant speedup factors for matching and we also compare the performance of k-means and the agglomerative scheme, both in terms of computational cost and recognition performance (Sec. 5).

2 Recognition approach

We briefly describe two main stages of a recognition algorithm which are similar in many state-of-the art approaches [1, 12, 17]. While there exist differences between the individual approaches, the following describes an object class representation and a matching procedure which may be seen as a common basis of many approaches.

Object representation. In our approach clustering is used to build an appearance model in which each cluster is represented by its center. For each cluster, an occurrence distribution is computed, specifying where and at which scales the local appearance occurs on the objects. The location distribution significantly increases the discriminative power of the representation and it allows to localize the object within the image.

Matching. The next stage of many recognition methods is matching. Given a query image, features are detected and matched to the object model represented by the appearance clusters. Typically, this stage involves a distance measure, a similarity threshold, and a search technique. The distance measure and similarity threshold depend on the feature descriptor at hand. A fast search method is necessary if the object representation contains a large number of clusters. The clusters that match to query features cast votes for possible object identities, locations, and scales based on the learned location distribution. Finally, local maxima are searched in multi-dimensional voting spaces. Additional stages can be applied to refine the hypotheses and improve detection precision [12].

3 Clustering methods

In this section, we present an efficient method for clustering large numbers of features. We discuss two main clustering techniques, namely partitional K-means and agglomerative method. We propose an efficient algorithm for the latter and introduce a multi-stage procedure combining the benefits of both techniques.

K-means. The k-means algorithm [15] is one of the simplest and most popular clustering methods. It is initialized randomly by k seed points for the clusters. In all following iterations, each data point is assigned to the closest cluster center, where the centers are computed as the means of associated data points. In practice, this process converges to a local optimum within a few iterations. Many approaches employ k-means because of its computational simplicity, which is convenient for large data sets [23, 26]. Its time complexity is $O(Nk\ell d)$ when clustering N data points of d dimensions with k centers and ℓ iterations. However, the complexity is high when k is comparable with N. It can be improved by using kd-trees [22] or triangular inequality [8]. K-means is often initialized randomly, which may result in different clustering solution from run to run. Several methods [21] were proposed to overcome this problem but they add computational overhead to k-means. Finally, there is no guarantee that the obtained clusters are visually compact. Because of the fixed value of k, some cluster centers may lie in-between several real clusters, so that the centers are not representative.

Agglomerative clustering. Agglomerative clustering builds the solution by initially assigning each point to its own cluster and then repeatedly selecting and merging pairs of clusters. Thus, it builds a hierarchical merging tree from the bottom (leaves) towards the top (root). The key parameter here is the criterion used for selecting clusters to be merged. We focus on the Group Average criterion (UPGMA in [11]), which measures the similarity of two candidate clusters as the average pairwise similarity between their members. Thus, the average-link criterion allows to specify the size or compactness of the resulting clusters. This property is very useful in building compact appearance clusters and makes the algorithm robust to outliers. A similarity threshold that produces visually compact clusters only depends on the employed feature descriptors, thus can be estimated experimentally and used on different data sets. Another advantage of agglomerative methods is that given the clustering trace from a full hierarchical clustering, i.e. the indices of clusters merged in every step and the similarities between them, we can rebuild the clusters for a different similarity threshold at almost no computational cost.

The main drawback of the standard average-link algorithm is its $O(N^2 \log N)$ run-time and $O(N^2)$ space complexity. This comes from the requirement that clusters should be merged in decreasing order of similarity and that the distances must be recomputed after each agglomeration. In order to make agglomerative clustering applicable to large data sets, both complexities have to be reduced. The improvement proposed here is based on the insight from [4] that for some criteria the same clustering solution can be achieved with different merging order. Furthermore, the similarities between clusters can be efficiently recomputed based only on the centers and variances.

RNN algorithm. The improved clustering method is based on the construction of *reciprocal nearest neighbor* pairs (RNN pairs), that is of pairs of points a and b, such that a is b's nearest neighbor and vice versa [4]. RNN is applicable to clustering criteria that fulfill *reducibility property* [5] :

$$d(c_i,c_j) \leq \inf(d(c_i,c_k),d(c_j,c_k)) \Rightarrow \inf(d(c_i,c_k),d(c_j,c_k)) \leq d(c_i \cup c_j,c_k)$$

Algorithm 1 Average-Link algorithm with RNNs for R points.

$last \leftarrow -1$
while $R \neq \emptyset$ **do**
  if $last < 0$ **then** *// Initialize a new chain with a random point $v \in R$.*
    $last \leftarrow 0$; $Chain[last] \leftarrow v \in R$; $R \leftarrow R \backslash \{v\}$; $Sim[last] \leftarrow 0$; (1)
  $s \leftarrow$ findNearestNeighbor($Chain[last]$, R); $sm \leftarrow sim(Chain[last], s)$ (2)
  if $sm > Sim[last]$ **then** *// No RNNs, add s to the chain.*
    $last \leftarrow last + 1$; $Chain[last] \leftarrow s$; $R \leftarrow R \backslash \{s\}$; $Sim[last] \leftarrow sm$; (3)
  else *// Found RNNs → agglomerate the last two points in the chain*
    if $Sim[last] > SimThreshold$ **then**
      $s \leftarrow$ agglomerate($Chain[last], Chain[last-1]$); $R \leftarrow R \cup \{s\}$; $last \leftarrow last - 2$; (4)
    else $last \leftarrow -1$ *// Discard the current chain.*

where c_i, c_j and c_k are clusters and $d(c_j, c_k)$ is a distance measure. This property effectively states that the agglomeration of a RNN pair does not alter the nearest-neighbor relations of other clusters. It is fulfilled for the average-link criterion regardless of the employed similarity measure. The key to an efficient implementation is therefore to ensure that RNNs can be found with as little computation as possible. This can be achieved by building a *nearest-neighbor chain* [4]. An NN-chain consists of an arbitrary point, followed by its NN, which is again followed by its NN from among the remaining points, and so on. Thus, each NN-chain ends with an RNN pair. The strategy of the algorithm is thus to start with an arbitrary point (Alg. 1, step (1)) and build up an NN-chain (2,3). As soon as an RNN pair is found, the corresponding clusters can be agglomerated (4). The reducibility property guarantees that after the last two clusters from the chain are merged, the NN assignments stay valid for the remaining chain members, which can then be used in the next iteration. Whenever the current chain is empty, a new chain is started with another random point (1). When a new cluster is created by merging an RNN pair, its new distance to other clusters has to be recomputed. Instead of expensively computing the average of all distances between cluster members, we use the following equivalence:

$$sim_{Euclid}(c_x, c_y) = \frac{1}{NM} \sum_{i=1}^{N} \sum_{j=1}^{M} (x^{(i)} - y^{(j)})^2 = \sigma_x^2 + \sigma_y^2 + (\mu_x - \mu_y)^2$$

where x and y are the cluster members, μ_x and μ_y are the centroids, σ_x^2 and σ_y^2 are the variances. Both the mean and variance of the new cluster can then be computed incrementally:

$$\mu_{new} = \frac{N\mu_x + M\mu_y}{N+M}, \quad \sigma_{new}^2 = \frac{1}{N+M}\left(N\sigma_x^2 + M\sigma_y^2 + \frac{NM}{N+M}(\mu_x - \mu_y)^2\right)$$

An amortized analysis shows that this algorithm has a computational complexity of $O(N^2 d)$ with only linear space requirements. This is an important improvement compared to the standard algorithm, since it makes it possible to cluster 100,000s of data points, which was not feasible before. However, the time complexity is still high when N is large. In the following, we present a strategy to further improve also the run-time efficiency.

Combined partitional-agglomerative algorithm (CPA). The idea of this improved algorithm is to first partition the set of features and perform agglomerative clustering within each partition independently [27]. However, there are several issues with this method. The first is how to set the partitions so that they contain features that cluster well. A possible solution is to use a natural partition of the data points, stemming from properties of the employed interest point detector. The scale invariant interest points are

detected at local maxima and minima of the Laplacian [13, 16]. If e.g. SIFT descriptors are used, which make a clear distinction between bright and dark structures, these extrema form two distinct groups which do not intersect. For other descriptors, this property has to be verified. Anothers suitable partitioning method is k-means. The number of initial partitions has to be small, otherwise k-means is not efficient. A problem occurs if a real cluster is split over several partitions, since the agglomeration is initially done between points which are NNs within one partition only. This can alter the cluster centers and variances and thus produce a different clustering tree. To reduce the impact of this effect on the final clustering solution, we agglomerate clusters within each partition only up to a certain similarity threshold. Next, given the cluster centers and variances obtained from all partitions, we continue the agglomeration up to the root of the tree. If the similarity threshold for initial clustering is smaller than for the final appearance clusters, then the initial agglomeration provides small building blocks used by the next level. However, the initial threshold should produce a number of clusters which is significantly lower than N, otherwise the complexity reduction would be limited.

To summarize the approach, we first partition features on two sets of Laplacian maxima and minima. Then we apply k-means to each set to further partition the features. Agglomerative clustering is applied within each partition. Finally, the agglomerative method is applied once more on all the cluster centers computed in the previous step. This combined partitional-agglomerative method leads to an approximate clustering solution, but as the experimental results show, the difference from the exact solution is negligible.

4 Fast matching

In this section, we propose a data structure for fast search in high dimensional feature spaces. Many fast NN search methods are based on hypercube or hyperrectangle approximations [2, 19]. They partition each feature dimension independently and trim the candidates for NN dimension by dimension. However, in this approach the efficiency depends critically on the size of the hypercube. It also relies on the fact that a single NN is searched in the whole space, for which the L_{inf} (hypercube) norm can be used. However, in object class recognition we are often interested in finding all features which are similar to our query point, for which the L_2 (hypersphere) norm is needed. Although L_2 is bounded by L_{inf}, in high-dimensional spaces the corners of the hypercube contain far more volume (data points) than the inscribed hypersphere. A solution to this problem is a data structure based on the L_2 norm. We describe a fast data structure and an algorithm for range search based only on a triangle-inequality-obeying distance metric.

4.1 Ball tree search

Ball tree structure. A ball tree (or metric tree) is a hierarchical structure for representing a set of points with the only assumption that the distance function between points is a metric [25]. Each node $(a...r)$ of the tree is represented by two parameters: center and radius (Fig. 1(a)). The node center is a mean vector of all the children nodes, and the radius is determined by the point farthest from the center. The radius can also be smaller if we are ready to accept a subset of the points similar to the query in return for a possible speedup. We propose to set the radius as a quantile of ordered feature distances from the node center.

Building ball trees. The problem of building an optimal ball tree structure is inherently similar to that of agglomerative clustering [18, 20]. In the agglomerative tree each node

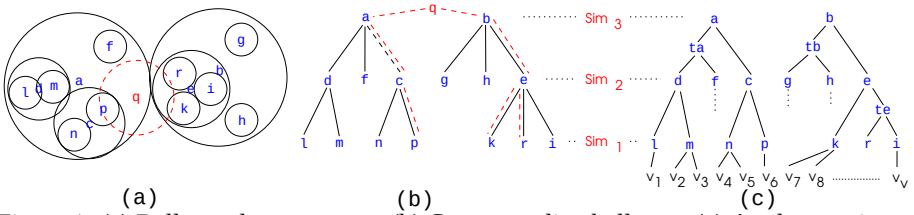


Figure 1: (a) Ball tree data structure. (b) Corresponding ball-tree. (c) Agglomerative tree.

contains two child nodes, since the algorithm merges two clusters at a time. However, given the clustering trace which contains the indices of merged clusters and their similarities, we can easily reconstruct a tree in which the number of children of a node is determined as a function of their similarity. This is illustrated in Fig. 1(b,c). Intermediate nodes ta, tb and te are merged with corresponding a, b and e. The size of the nodes is increasing from the leaves to the root of the tree. Thus we obtain a ball-tree structure from the agglomerative clustering trace with minimal additional cost.

Ball-tree search. A range search is a simple recursive procedure, which is illustrated in Figs. 1(a) and 1(b). We start by computing the similarity of a query point q to the top nodes a and b and use the triangle inequality property. The search is continued if the distance to the node center minus the nodes radius is less than the query radius, i.e. if the query ball intersects with the node ball. The search is continued further to all children nodes that intersect with the query ball. Exhaustive search is applied within each node. The speed of the search thus depends on the number of tree levels, the node radii, and the query radius. The number of levels and the node radii can be chosen experimentally at a low cost using the precomputed clustering trace. If we are only interested in the NNs, the search can be made more efficient, since the search radius can be progressively reduced with each new NN candidate that is found.

5 Experiments

In this section we present and discuss the evaluation results.

5.1 Test Data

Our test data consists of 1,000,000 scale invariant features provided by Harris-Laplace and Hessian-Laplace detectors [16] with SIFT descriptor [13]. Features are detected in 5,000 images from the Caltech database and the PASCAL set[1], containing pedestrians, cars, motorbikes, faces, and cows. To validate the results, we also compare the recognition performance of the baseline approach using the proposed clustering and matching methods and the UIUC multi-scale car set [1]. Additional experiments on more object classes can be found in [17].

5.2 Clustering

Similarity measure. As described in Section 3, the agglomerative clustering method is driven by the similarity measure and a threshold. To produce meaningful clusters we determine a reasonable range for the similarity distance using the evaluation protocol from [16], originally developed for matching pairs of images. It computes precision (i.e. the ratio of correct to false matches) and recall of matches with respect to the similarity threshold. Precision is high up to a given similarity threshold and decreases for larger

[1] http://www.pascal-network.org/challenges/VOC/

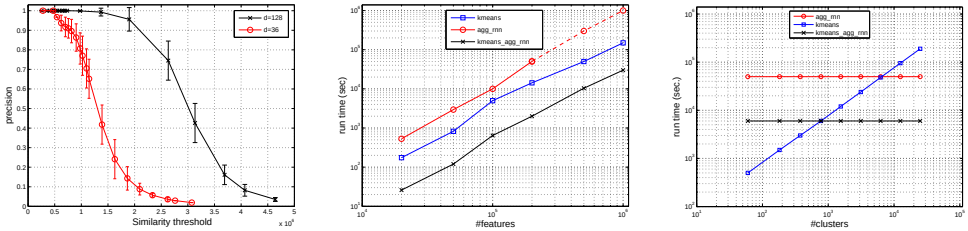


Figure 2: (a) Matching precision vs. similarity distance. (b) Run-time vs. number of features. (c) Run time vs. number of clusters.

thresholds (cf. Fig. 2(a)). The useful thresholds are in the steep part of the curve. For small thresholds, only very similar features match, resulting in a poor generalization of the model to new object instances. For large thresholds, false matches dominate, thus the recognition performance is low and the complexity increases. The above method provides a reliable and computationally inexpensive insight on the similarity thresholds that can be used for agglomerative clustering. In contrast, the size and distribution of k-means clusters depend on the k parameter, which is difficult to optimize if the real distribution of features is unknown.

Run-time. Given a set of features, we first run the RNN method with a fixed similarity threshold obtained from Fig. 2(a), which results in a number of clusters. We then run standard k-means for the same number of clusters with the maximum number of iterations set to 25. Finally, we run the CPA method with initial number of k-means partitions set to $\#features/20000$, and the initial agglomerative threshold set to half the one obtained from Fig. 2(a). Thus the methods are compared for the same number of features and the same number of clusters. Fig. 2(b) shows the run-time with respect to the number of features in the database. The run-time of CPA is an order of magnitude lower than for k-means and 2 orders of magnitude lower than the RNN algorithm. For example, clustering of 1M features takes 555h for RNN[2], 41h for k-means, and 5h for CPA.

Fig. 2(c) shows the run-time with respect to the number of resulting clusters, using 200k features. For k-means, the run time increases linearly with k. This is to be expected since the complexity is directly related to the number of clusters if the exhaustive search is used during clustering. However, it is important to note that this is the upper bound, since the run time can be shorter if convergence is obtained in less iterations, $k \approx N$, fast NN search techniques [22], or other speedups [8] are used. The run-time of the RNN is high but almost independent of the number of clusters, since most of the computation is spent at the bottom of the clustering tree, when the number of clusters is still large. For a large number of clusters, the run-time for k-means exceeds the one for RNN. From our experience, the compression ratio $\#features/\#clusters$ which gives the best recognition performance is in the range, where the proposed CPA method outperforms k-means.

Cluster quality. Fig 3(a) displays the average intra-class variance of clusters obtained with the three methods. The results are reported with respect to the number of features, and using the same number of clusters as in Fig. 2(b). Single-member clusters were discarded from this experiment in order not to bias the results. The diagram shows that the variance of clusters obtained for both agglomerative methods is lower than for k-means

[2]The run-times for RNN agglomerative clustering in the range of 500,000-1,000,000 points are estimated since we were not able to run the clustering due to time constraints.

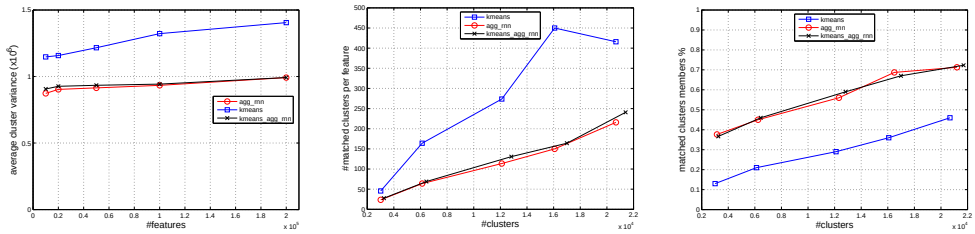


Figure 3: (a) Average variance of clusters. (b) Average number of matched clusters per feature. (c) Average percentage of matched features per matched cluster.

clusters. The variances for RNN and CPA are nearly the same. To compare the distribution of cluster centers and the compactness of the clusters, we carried out an additional experiment. We count the number of cluster centers which are within a given similarity radius of a query point (cf. Fig. 3(b)). For k-means, the number of matched clusters per query feature is significantly larger than for RNN and CPA. In Fig. 3(c) we measure how many cluster members do indeed match to the query feature. The percentage of matched cluster members is higher for agglomerated clusters. Together, these results show that the k-means clusters are less compact and therefore match to more features compared to agglomerated clusters, and that k-means cluster centers are less representative for the cluster members.

5.3 Matching

In this section we compare the efficiency of the ball tree algorithm to exhaustive search. We report the speedup factor as the ratio of run-times for 1,000 random queries. The efficiency depends on several parameters: the number of features in the dataset, the number of tree levels, the node radii, and the query radius. We have chosen experimentally 10 levels between the size of the appearance clusters (bottom nodes) and the size of the top node. The impact of the other parameters on the speedup is investigated in the following experiments. We use 200k of 128 dimensional descriptors and 200k of 36 dimensional descriptors obtained with PCA. To show the results for different numbers of features, we also use a set of 50k points with 128-dim. descriptors.

Fig. 4(a) shows the speedup factor with respect to the fraction of lost matches. We vary the radius of the nodes and compare the efficiency and the returned matches with exhaustive search. If we are looking for the exact matches, the ball tree is nearly 80 times faster than exhaustive search (for 200k features of 128 dim). This factor significantly increases up to 200 with 20% of lost matches[3]. The gain is smaller for the dataset of 50k points and for low dimensional features, which indicates that we can expect further improvement with increasing number of features and dimensions. Fig. 4(b) shows the results for different query radii (as a fraction of the top node size). The efficiency significantly drops as the size of the query increases, since many more nodes have to be examined. In most of our recognition experiments, the root node radius was 10 times larger than the size of the appearance clusters. Thus, the useful query radius is in the range of 0.1-0.2.

5.4 Recognition performance.

Finally, we compare the recognition performance of object class representations obtained with the different clustering methods. We use the UIUC multi-scale car database and

[3]While it is difficult to make a general claim how many lost matches are acceptable we experimentally observed that we can accept 10% and more lost matches without any loss in recognition performance

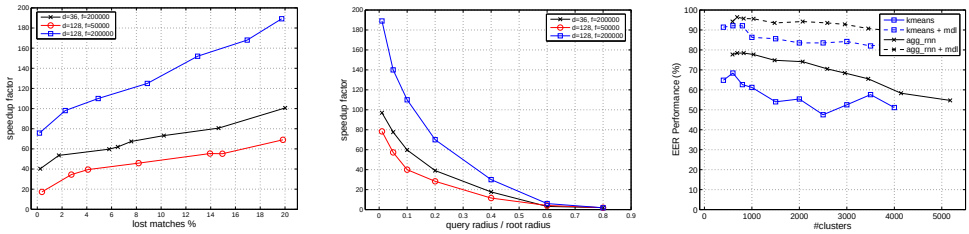


Figure 4: (a) Ball-tree speedup factor vs. number of lost matches. (b) Ball-tree speedup factor vs. query radius. (c) Recognition performance.

the evaluation criteria from [1]. We learn object representations on a training set of 50 car images from the PASCAL collection (cf. Sec. 5.1), from which we extract a total of 10,351 features with 36 dimensions. We use the evaluation criteria from [1] based on the overlap of ground truth and detected bounding boxes.

Fig. 4(c) shows precision-recall performance at the equal error rate (EER) as a function of the the number of clusters for both k-means and the agglomerative method. The solid curves depict the performance when the simple recognition approach is used (cf. Sec. 2); this performance can then still be improved by applying the MDL verification [12], as shown by the dashed curves. We make three observations. First, the recognition score is higher for agglomerated clusters (EER: 78.4%) than for k-means (EER: 68%). The methods reach different performance levels initially, but can both be taken to approximately the same performance (EERs: 96.4% and 92.1%) by the verification stage. Second, for both clustering schemes the performances degrade gracefully for different number of clusters, which is a result of our soft matching within a search hypersphere. Third, since the cost of the soft matching increases with the number of clusters that fall inside the search radius and k-means does in fact produces many more such matches for the same number of clusters (see Fig. 3(b)) we conclude that agglomerative clustering is preferable to k-means in terms of recognition costs.

Conclusions

Many of todays object class recognition approaches use clustering and matching of local features to build object models. While k-means is the most popular method, this paper shows that agglomerative clustering has several inherent properties that make it highly attractive for object class recognition: first, matching can be done efficiently using ball-tree search in high-dimensional spaces and with large numbers of clusters; second the clusters reflect the distribution of features resulting in fewer matches and lower complexity; and third, recognition performance is often better than for k-means clusters.

This paper introduces various improvements of agglomerative clustering in the context of processing large numbers of high-dimensional features. In addition, it shows how to use the clustering result to build a data structure for efficient matching. These improvements result not only in a practically feasible and efficient clustering scheme (we report clustering results up to 1,000,000 features), but also in significant speedups for matching (up to 200 times faster). Last but not least, the proposed algorithms and the expected improvements are experimentally validated.

Acknowledgments. This work has been funded, in part, by the EU project CoSy (IST-2002-004250).

References

[1] S. Agarwal, A. Awan, and D. Roth, Learning to detect objects in images via a sparse, part-based representation. *PAMI*, 26(11):1475–1490, 2004.

[2] J. Beis and D. Lowe, Shape Indexing Using Approximate Nearest-Neighbour Search in High-Dimensional Spaces. In *CVPR*, pages 1000–1006, 1997.

[3] J.L. Bentley, Multidimensional binary search trees used for associative searching. In *Communications of the ACM*, 18(9):509–517, 1975.

[4] J.P. Benzécri, Construction d'une Classification Ascendante Hiérarchique par la Recherche en Chaîne des Voisins Réciproques. *CAD*, 7(2):209–218, 1982.

[5] M. Bruynooghe, Méthodes Nouvelles en Classification Automatique des Données Taxinomiques Nombreuses. *Statistique et Analyse des Données*, 3:24–42, 1977.

[6] D. Comaniciu and P Meer, Mean Shift: A Robust Approach toward Feature Space Analysis *PAMI*, 24(5):603–619, 2002.

[7] W.H.E. Day and H. Edelsbrunner, Efficient Algorithms For Agglomerative Hierarchical Clustering Methods. *Journal of Classification*, 1:7–24, 1984.

[8] C. Elkan, Using the Triangle Inequality to Accelerate KMeans *In ICML*, pages 147-153, 2003.

[9] P. Indyk, R. Motwani, Approximate Nearest Neighbors: Towards Removing the Curse of Dimensionality. In *STOC*, pages 604–613, 1998.

[10] A.K. Jain and R.C. Dubes, Algorithms for Clustering Data, Prentice-Hall, 1988

[11] G.N. Lance and W.T. Williams, A General Theory of Classificatory Sorting Strategies: II. Clustering Systems. *Computer Journal*,10:271–277, 1967.

[12] B. Leibe, E. Seemann, and B. Schiele, Pedestrian detection in crowded scenes. In *CVPR*, pages 878–885, 2005.

[13] D. Lowe, Distinctive image features from scale-invariant keypoints. *IJCV*, 2(60):91–110, 2004.

[14] T. Liu, A. Moore, A. Gray, and K. Yang, An Investigation of Practical Approximate Nearest Neighbor Algorithms. In *NIPS*, pages 825–832, 2004.

[15] J. MacQueen, Some Methods for Classification and Analysis of Multivariate Observations. In *Symp. on Math. Statistics and Probability*, pages 281–297, 1967.

[16] K. Mikolajczyk and C. Schmid, A performance evaluation of local descriptors. *PAMI*, 27(10):1615–1630, 2005.

[17] K. Mikolajczyk, B. Leibe and B. Schiele, Multiple Object Class Detection with a Generative Model. *CVPR*, 2006.

[18] A.W. Moore, The Anchors Hierarchy: Using the Triangle Inequality to Survive High Dimensional Data. In *UAI*, AAAI Press, pages 397–405, 2000.

[19] S. Nene, S. Nayar, A Simple Algorithm for Nearest Neighbor Search in High Dimensions. *PAMI*, 19(9):989–1003, 1997.

[20] S.M. Omohundro, Five balltree construction algorithms. Technical Report TR-89-063, 1989.

[21] K. Popat and R.W. Picard, Cluster-Based Probability Model and Its Application to Image and Texture Processing. *TIP*, 1997.

[22] V. Ramasubramanian and K.K. Paliwal, A Generalized optimization of the k-d tree for fast nearest neighbour search. *TENCON*, pages 565-568, 1989.

[23] J. Sivic and A. Zisserman, Video google: A text retrieval approach to object matching in videos. In *ICCV*. pages 1470–1478, 2003.

[24] J. Sivic, B. Russell, A. Efros, A. Zisserman, and W. Freeman, Discovering object categories in image collections. In *ICCV*, 2005.

[25] J.K. Uhlmann, Satisfying general proximity/similarity queries with metric trees. In *Information Processing Letters, 40*, pages 175–179, 1991.

[26] M. Weber, M. Welling, and P. Perona, Unsupervised Learning of Models for Recognition. In *ECCV*, pages 628-0641, 2000.

[27] Y. Zhao and G. Karypis, Evaluation of hierarchical clustering algorithms for document datasets. In *CIKM*, pages 515–524, 2002.

Published in the Proceedings of ERK 2006.

On different modes of continuous learning of visual properties *

Danijel Skočaj, Barry Ridge, and Aleš Leonardis
University of Ljubljana, Faculty of Computer and Information Science
Tržaška 25, SI-1001 Ljubljana, Slovenia
{*danijel.skocaj, barry.ridge, ales.leonardis*}*@fri.uni-lj.si*

O različnih načinih inkrementalnega učenja vizualnih lastnosti

Za vsak spoznavni sistem, tudi umetni, je zelo pomembno, da se je sposoben učiti in pridobljeno znanje nadgrajevati. V tem članku obravnavamo različne načine inkrementalnega učenja, ki to omogoča. Predstavimo učenje, pri katerem uporabnik oz. učitelj zagotovi umetnemu sistemu vse potrebne informacije, ki jih potrebuje, nato učenje, pri katerem sistem zahteva od uporabnika informacije glede na stopnjo nedoločenosti, ter učenje, pri katerem sistem nadgrajuje svoje znanje popolnoma brez pomoči uporabnika. V članku tudi predstavimo metodo, ki omogoča inkrementalno učenje vizualnih lastnosti predmetov na vse tri načine. Z eksperimentalnimi rezultati vse tri pristope tudi ovrednotimo.

1 Introduction

In a real world environment, a cognitive system should possess the ability to learn and adapt in a continuous, open-ended, life-long fashion from the variable input that such an environment would present. As an example of such a learning framework, we need look no further than at the successful application of *continuous learning* in human beings. For example, a child will learn to recognise what a cat is by seeing a few examples of one. Later, as the child encounters more cats that are different to the original examples, he/she will not only recognise the new cats as being cats, but will also update his/her representation of what a cat is, based on the salient properties of the new examples and without having visual access to the previous examples.

While the primary focus of this idea is on the incremental nature of the knowledge update, another key aspect is the scrutinisation of various visual features and the determination of which features are useful for representing the visual attributes of the object in question. Since a continuous learning framework would not retain complete data from previously learned samples, it would not have the luxury of being able to reference specific details across multiple samples in order to learn. Given this restriction, continuous learning perhaps lends itself to an abstract multi-modal system involving interaction with a user.

In this paper we discuss such cross-modal learning, namely association between *words* and simple *visual features*, such as hue or intensity values of the corresponding pixels. In particular we will present a method for learning *visual attributes* (e.g., colour, shape) and their *qualitative values* (e.g., red, yellow; circular, square). The problem of coupling words and images involves computer vision and linguistic methods, therefore it has been tackled by the researchers from both communities (see e.g., [1, 3]). In their work the emphasis is on association mechanisms, which are mainly based on batch approaches. In this paper we instead focus on an incremental learning paradigm and different types of incremental learning mechanisms that require different levels of supervision provided by a tutor.

The paper is organised as follows. In the next section we propose a general framework for continuous learning. In Section 3 we present a specific method for incremental learning and embed it in the proposed framework. We then present the experimental results in Section 4. Finally, we summarise the paper and outline some work in progress.

2 Continuous learning framework

The interaction between a tutor and an artificial cognitive system plays an important role in a continuous learning framework. One goal of the learning mechanism could be to find associations between words spoken by the tutor and visual features automatically extracted by the cognitive visual system, i.e. to ground the semantic meaning of the visual objects and their properties into the visual features [2].

When implementing a continuous learning mechanism, two main issues have to be addressed. Firstly, the representation, which is used for modeling the observed world, has to allow for updates when pre-

* This research has been supported in part by the following funds: Research program Computer Vision P2-0214 (RS), EU project CoSy, and EU project VISIONTRAIN.

sented with newly acquired information. This update step should be efficient and should not require access to the previously observed data while still preserving the previously acquired knowledge. Secondly, a crucial issue is the quality of the updating, which highly depends on the correctness of the interpretation of the current visual input. With this in mind, several learning strategies can be used, ranging from completely supervised to completely unsupervised learning. In this paper we discuss three such strategies:

- **Tutor-driven approach (*TD*).** The correct interpretation of the visual input is always correctly given by the tutor.
- **Tutor-supervised approach (*TS*).** The system tries to interpret the visual input. If it succeeds to do this reliably, it updates the current model, otherwise asks the tutor for the correct interpretation.
- **Exploratory approach (*EX*).** The system updates the model with the automatically obtained interpretation of the visual input. No intervention from the tutor is provided.

We further divide *tutor-supervised learning* into two sub-approaches:

- **Conservative approach (*TSc*).** The system asks the tutor for the correct interpretation of the visual input whenever it is not completely sure that its interpretation is correct.
- **Liberal approach (*TSl*).** The system relies on its recognition capabilities and asks the tutor only when its recognition is very unreliable.

Similarly, we also allow for *conservative* and *liberal exploratory* sub-approaches (*EXc*, *EXl*).

To formalise the description of these approaches, let us assume that the recognition algorithm always gives one of the following five answers when asked to confirm the interpretation of the current visual scene (e.g., the question may be: "Is this red?"): *'yes' (YES), 'probably yes' (PY), 'probably no' (PN), 'no' (NO),* and *'don't know' (DK)*. Table 1 presents actions that are undertaken after an answer from the recognition process is obtained. The system can either *ask* the tutor for the correct interpretation of the scene (or the tutor provides it without being asked), *update* the model with its interpretation, or do nothing. As can be seen from Table 1, the system can communicate with the tutor all of the time (*TD* learning), often (*TSc*), occasionally (*TSl*) or even never (*EX* learning).

To speed up the initial phase of the learning process and to enable development of consistent basic concepts, one could start with mainly tutor-driven learning with many user interactions. These concepts would then be used to detect new concepts with limited help from the user. Later on in the process, when the ontology is sufficiently large, many new concepts could be acquired without user interaction.

Table 1: Update table.

	YES	PY	PN	NO	DK
TD	ask	ask	ask	ask	ask
TSc	upd	ask	ask	/	ask
TSl	upd	upd	/	/	ask
EXc	upd	/	/	/	/
EXl	upd	upd	/	/	/

3 Our method

The main task of the learning algorithm is to assign associations between visual features and attribute values. It has to consider two main issues: *consistency* and *specificity*. It must determine the visual features that are *consistent* over all images sharing a particular visual attribute and that are, at the same time, *specific* for that visual attribute only.

With these requirements in mind, we have designed algorithms for incremental learning and recognition of visual properties based on a generative representation of features associated with visual attributes. Each visual attribute is associated with a visual feature that best models the corresponding images according to the consistency and specificity criteria mentioned above. The learning algorithm thus selects from N_F one-dimensional features (e.g., median hue value, area of segmented region, etc.), the feature whose values are most consistent over all images sharing the same visual attribute (i.e. the variance is small and the feature values are concentrated around the mean value). At the same time it also ensures that the same does not hold true for some other visual attribute, thus satisfying the specificity criterion. A visual attribute value is therefore represented with the mean and variance of the best feature.

The main idea is described in an algorithmic form in Algorithm 1. In the basic batch form, the algorithm requires all training images to be given in advance, together with a list of attribute values (e.g., red, large, square) for each image. Since the mean and variance of a set of feature values can be calculated in an incremental way without losing any information, this algorithm can be incrementalised. Algorithm 2 shows the pseudo-code of one update step. Using this algorithm, the model can be sequentially updated by considering only one image at a time.

Once the models of visual attributes have been acquired, the system is able to recognise visual properties of a novel object using Algorithm 3 (e.g., answering the question "Is this red?"). If the value of a feature associated with a particular attribute value is quite close to the values observed during learning (i.e. it is very close to the mean of previously observed values), then the system answers 'yes'. Based on the distance from the typical value of the feature

(considering variance as well), the system may also answer "probably yes", "probably not", or "no". By changing the thresholds $Tyes$, Tpy, and Tno, we can achieve more conservative or more liberal behaviour of the recognition algorithm. Combining this recognition algorithm with the incremental learning algorithm and by considering the update table presented in Table 1, we arrive at the incremental learning framework described in the previous section.

Algorithm 1 : Batch learning

Input: Set of training images $\mathcal{X}$, list of attribute values $\mathcal{AV}_i$ for every training image X_i

Output: Models of attribute values $mAV_i, i = 1 \dots N_{AV}$

1: Extract all visual features $F_j, j = 1 \dots N_F$ from every training image in $\mathcal{X}$.
2: **for** each $AV_i, i = 1 \dots N_{AV}$ **do**
3: Find a set of images $\mathcal{X}_i$ containing all images labeled with AV_i.
4: Calculate *means* and *variances* of the values of every feature F_j over all images in $\mathcal{X}_i$.
5: **end for**
6: Calculate *min* and *max* of all the values of every feature F_j.
7: Normalise all the variances with the obtained intervals of feature values, i.e., $nvar_{ij} = var_{ij}/(maxVar_j - minVar_j)^2$, $i = 1 \dots N_{AV}, j = 1 \dots N_F$.
8: **for** each $AV_i, i = 1 \dots N_{AV}$ **do**
9: Select the feature F_j with the smallest normalised variance $nvar_{ij}$.
10: Store *mean* and *variance* of the selected F_j to form mAV_i, a model of AV_i.
11: **end for**

Algorithm 2 : Update step

Input: Models of attribute values $mAV_i, i = 1 \dots N_{AV}$, feature statistics FS, new image X and corresponding attribute value AV

Output: Updated $mAV_i, i = 1 \dots N_{AV}$ and FS

1: If the model for AV has not been learned yet, initialise it.
2: Update feature *means* and *variances* related to AV (stored in FS).
3: Update total feature *mins* and *maxs*.
4: Proceed with the steps 7-11 of Algorithm 1.

4 Experimental results

We tested the algorithms by running a number of experiments on both artificial and real data. Basic shapes of various different colours and sizes were selected as test objects. Some of them are depicted in

Algorithm 3 : Recognition

Input: Image X, question "`Is this AV?`"

Output: Answer.

1: If the model for AV has not been learned yet, answer '`Don't know.`'
2: Determine which feature F_j the attribute value AV is associated with in the model mAV.
3: Extract the value of this feature F_j from the image X.
4: Calculate $d = (F_j - mAV.mean)/\sqrt{mAV.var}$.
5: If $d \in [0, Tyes]$, answer '`Yes.`'
6: If $d \in (Tyes, Tpy]$, answer '`Probably yes.`'
7: If $d \in (Tpy, Tno]$, answer '`Probably not.`'
8: If $d \in (Tno, \infty)$, answer '`No.`'

Fig. 1. We considered three visual attributes (colour, size and shape), and ten values of these visual attributes altogether (red, green, blue, yellow; small, large; square, circular, triangular, and rectangular).

The objects were first perspective-rectified and segmented from the background. Then the visual features were extracted. We used six simple one-dimensional features; three colour features (median of hue, saturation and intensity over all pixels in the segmented region) and three simple shape descriptors (area, perimeter and compactness of the region). The main goal was to find associations between ten given attribute values and six extracted features.

We put half of the images in the training set and other half in the test set (64 per half in the case of synthetically generated images and 70 per half in the case of real images). We embedded the proposed learning method in the learning framework and kept incrementally updating the representations with the training images using different learning strategies. At each step, we evaluated the current knowledge by recognising the visual properties of all test images. The evaluation measure we used is *recognition score*, which rewards successful recognition (true positives and true negatives) and penalises incorrectly recognised visual properties (false positives and false negatives).

Results (the curves of the evolution of the recognition score through time) of the experiment on the synthetic images (averaged over 40 trials on different sets of generated images with added noise) are presented in Fig. 2(a). All different learning strategies

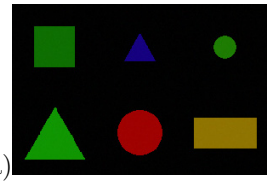
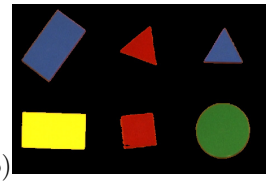
(a)  (b) 

Figure 1: (a) Synthetic images. (b) Perspective-rectified and segmented real images.

presented in Section 2 were tested. First, we applied the incremental learning process from the very beginning, starting with one training image (denoted as *TSc1*, *TSl1*, etc.). Then we repeated the experiment by first applying the batch algorithm on the first 10 images, and then continuing by incrementally adding the rest of the images (*TSc10*, *TSl10*, etc.). Fig. 2(a) shows the plots of recognition scores, while Fig. 2(b) plots the number of questions the system asked the tutor at each step (i.e., how much data were given to the system by the tutor).

In the experiment on the synthetic images, the tutor-driven learning successfully associates the colours of the input objects with the *hue* feature, their sizes with the *area* feature and their shapes with the *compactness* feature. Recognition of visual attributes is very successful; it almost gets the maximal score (640 in this case). Tutor-supervised learning proved to be quite successful as well. In this case conservative strategy yields better results, since it asks the tutor for reliable information more often. This is also evident from Fig. 2(b). In the beginning the system does not have a lot of knowledge, so the tutor is asked for help more frequently. After the knowledge is acquired, the number of questions decreases. The explorative approach, which does not involve interaction with the tutor from the very beginning, does not significantly improve the model. So, as expected, there is a trade-off between the quality of results and the wish to decrease the need for user interaction. Similar conclusions can also be drawn from the results of the experiment on real data shown in Fig 2(c).

5 Conclusion

In this paper we presented a framework for continuous learning, which enables three modes of learning requiring different levels of tutor supervision. We proposed a method for incremental learning of visual properties by building associations between words describing objects' visual properties and visual features extracted from images. By embedding this method into the proposed learning framework, we were able to experimentally evaluate three learning strategies. The main conclusion is that the learning process should start with tutor-driven learning to enable development of consistent basic concepts. Once these concepts are acquired, the system can take the initiative and keep upgrading the knowledge in a tutor-supervised way, and when the knowledge is stable enough, even in an exploratory way.

Beyond this initial work, we aim to improve the learning method as well as to further analyse the proposed framework and evaluate different learning strategies under various conditions and in various applications.

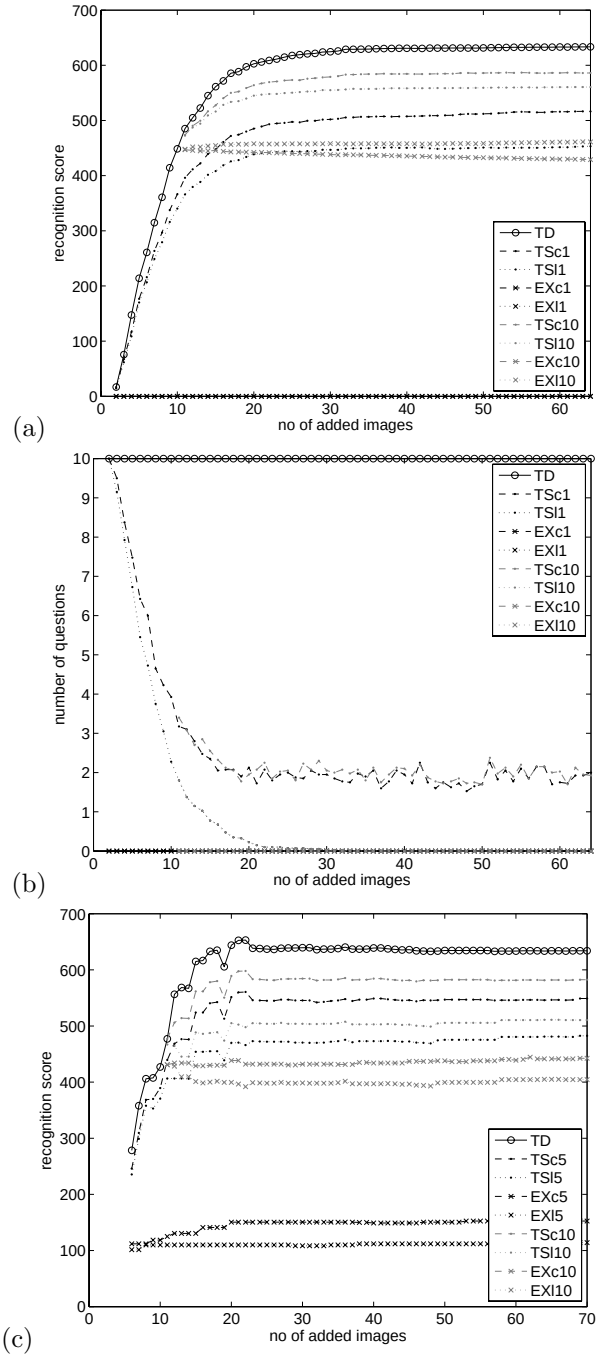


Figure 2: (a) Rec. score and (b) number of questions on synthetic images, (c) rec. score on real images.

References

[1] K. Barnard, P. Duygulu, D. Forsyth, N. de Freitas, D. Blei, M. Jordan. Matching words and pictures. *J. Mach. Learn. Res.*, 3:1107–1135, 2003.

[2] S. Harnad. The symbol grounding problem. *Physica*, D(43):335–346, 1990.

[3] D. Roy. Learning words and syntax for a visual description task. *Computer Speech and Language*, 16(3):353–385, 2002.