On-line conservative learning for person detection *

Peter M. Roth† Helmut Grabner† Danijel Skočaj‡ Horst Bischof† Aleš Leonardis‡

† Graz University of Technology
Institute for Computer Graphics and Vision
Inffeldgasse 16/II, 8010 Graz, Austria
{pmroth, hgrabner, bischof}@icg.tu-graz.ac.at

‡University of Ljubljana
Faculty of Computer and Information Science
Tržaška 25, SI-1001 Slovenia
{danijel.skocaj, alesl}@fri.uni-lj.si

Abstract

We present a novel on-line conservative learning framework for an object detection system. All algorithms operate in an on-line mode, in particular we also present a novel on-line AdaBoost method. The basic idea is to start with a very simple object detection system and to exploit a huge amount of unlabeled video data by being very conservative in selecting training examples. The key idea is to use reconstructive and discriminative classifiers in an iterative co-training fashion to arrive at increasingly better object detectors. We demonstrate the framework on a surveillance task where we learn person detectors that are tested on two surveillance video sequences. We start with a simple moving object classifier and proceed with incremental PCA (on shape and appearance) as a reconstructive classifier which in turn generates a training set for a discriminative on-line AdaBoost classifier.

1. Introduction

In recent years, the demand for automatic methods analyzing large amounts of visual data has been constantly increasing. Starting with face detection [18, 21] there has been a considerable interest in visual object detection, e.g., pedestrians [22], cars [1], bikes [14], etc. Visual surveillance has become a major research topic in computer vision. All these trends have been encouraging research in the area of automatic detection, recognition, categorization, and interpretation of objects, scenes and events.

These visual systems have to encompass a certain level of knowledge about the world they are observing and analyzing. The most convenient way of knowledge acquisition is its accumulation through learning. When a huge amount of data is required to train the system, the learning process has to be as automatic as possible, requiring a minimal hand labeling effort. If the visual system's environment is constantly changing, the system has to keep adapting to these changes. It has, therefore, to keep continuously learning and updating its knowledge.

When implementing continuous learning mechanisms, two main issues have to be addressed. First, the representation, which is used for modeling the observed world, has to allow 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. In this context, several learning strategies can be used, ranging from a completely supervised learning approach (when the correct interpretation of the current visual input is given by a tutor) to a completely unsupervised approach (when the visual system has to interpret the current input without any additional assistance). Obviously, the latter approach is preferable, especially when the amount of data to be processed is large.

Having these two issues in mind, one has to carefully select the type of representations of the objects (or subjects, or scenes) that can be used. Discriminative representations are compact, task dependent, efficient, and effective, but usually not very robust. On the other hand, reconstructive representations are usually less efficient and less effective, but more general and robust. The ultimate goal is to combine these two representations to achieve best of both worlds, which would lead to efficient and effective, while still general and robust continuous learning techniques.

In this paper we propose such a combination of reconstructive and discriminative methods combined in the *conservative learning framework*. To avoid hand labeling of input data, we want that the visual system labels data automatically. However, since during the learning process

*This work has been supported by the Austrian Joint Research Project Cognitive Vision under projects S9103-N04 and S9104-N04, by the Federal Ministry for Education, Science and Culture of Austria under the CONEX program, by the SI-A project, by the Federal Ministry of Transport, Innovation and Technology under P-Nr. I2-2-26p Vitus2, the Ministry of Higher Education, Science and Technology of Republic of Slovenia under the Research program Computer Vision P2-0214, by EU FP6-004250-IP project CoSy and by EU FP6-511051-2 project MOBVIS.

a sufficient knowledge required for reliably evaluating the visual input is still to be acquired, the process of labeling should strongly be intertwined with the process of continuous learning, which could provide enough redundant information to determine statistically consistent data. Only the sufficiently consistent data would then be used to build the representations, enabling robust learning (and updating of the representations) under non-ideal real-world conditions. We refer to this approach as conservative learning. Also importantly, such a conservative approach assures, that non-relevant (corrupted, inexact, false) data is not included into the model and that the model is not degraded.

The proposed framework is depicted in Fig. 1. The basic idea is to use a huge amount of unlabeled data that is readily available for most detection task (i.e., just mount a video camera and observe the scene) to avoid hand labeling of training data for object detection tasks.

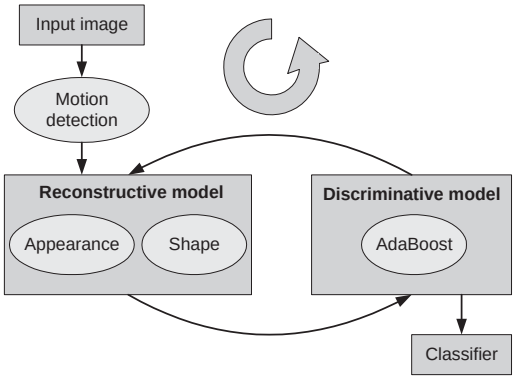

Figure 1: The proposed on-line learning framework.

We use two types of models, a reconstructive one which assures robustness and serves for verification, and a discriminative one, which actually performs the detection. To get the whole process started we use a simple motion detector to detect potential objects of interest. In fact, we miss a considerable amount of objects (which can be compensated by just using longer sequences) and we will get also a lot of miss-detections (which will be reduced in the subsequent steps). The output from the motion detector can be used to robustly build a first initial reconstructive representation (to further increase the robustness we are combining multiple cues – we use one representation on shape and the other on appearance). In particular, we use the robust incremental PCA [20] at this stage so that most of the miss-detections (background, false detections, over-segmentations, etc.) are not incorporated in the reconstructive model. This is very crucial as the discriminative classifier needs to be trained with "clean" images to produce good classification results. The discriminative model, the incremental AdaBoost, is then used to detect new objects in new images. The output of the discriminative classifier is then verified by the reconstructive model, and detected false positives are fed back into the discriminative classifier as negative examples and true positives as positive examples to further improve the discriminative model. In fact, it has been shown in the active learning community [16], that it is more effective to sample the current estimate of the decision boundary than the unknown true boundary. This is exactly achieved by our combination of reconstructive and discriminative classifiers. Exploiting the huge amount of video data, this process can be iterated to produce a stable and robust classifier. Since all the methods used operate in an incremental manner, every image can be discarded immediately after it is captured and used for updating the model.

Very recently we presented a preliminary version of the proposed approach [17]. It was based on batch methods, so it was not suitable for on-line learning. In this paper we extend this approach in several directions – most importantly, we embed the proposed approach in an incremental framework enabling on-line unsupervised learning in particular on-line boosting.

Also in the past a few attempts have been made to propose methods, which would enable automatic labeling of training data. The outlined approach is similar to the recent work of Nair and Clark [13] and Levin et al. [9]. Nair and Clark propose to use motion detection for obtaining the initial training set and then Winnow as a final classifier. Their approach does not include reconstructive classifiers, nor does it iterate the process to obtain more accurate results. In that sense our framework is more general. Levin et al. use the so called co-training framework to start with a small training set and to increase it by using a co-training of two classifiers operating on different features. We show that using a combination of reconstructive and discriminative classifiers helps to increase the performance of the discriminative one.

The rest of the paper is organized as follows. In Section 2 we detail our approach. In order to make the discussion concrete we will use person detection from videos. The experimental results in Section 3 demonstrate the approach on some challenging video sequences with groups of people and occlusions. Finally, we present some conclusions and work in progress.

2. On-line Conservative Learning

2.1. Motion

Having a stationary camera a common approach to detect moving objects is to threshold the difference image between the current frame and a background model. A simple method for computing a background model is a pixel-wise temporal median filter. To reduce computational costs and to have an on-line method McFarlane and Schofield [12] developed the approximated median. For that purpose the me-

dian is approximated by incrementing the current estimate by one if the input pixel value is larger than the estimate and by decreasing it by one if smaller. This estimate eventually converges to the real median. The obtained motion blobs can be labeled as persons if the aspect ratio of their bounding box is within the pre-specified limits.

2.2. Reconstructive model

We use a PCA-based subspace representation as a reconstructive model. This low-dimensional representation captures the essential reconstructive characteristics by exploiting the redundancy in the visual data. As such, it enables "hallucinations" and comparison of the visual input with the stored model [7]. In this way the inconsistent data can be rejected and the discriminative model can be trained from clear data only.

During the learning process, however, a sufficient knowledge required for a reliable evaluation of the visual input is still to be acquired. Nevertheless, by considering the reconstruction error, the robust learning procedure can discard inconsistencies in the input data and train the model from consistent data only [5, 19]. Furthermore, this can be done also in an incremental way. A bunch of methods for incremental building of eigenspaces have been proposed [4, 6, 10], some of them specifically addressing also the problem of robust incremental learning [20, 11]. We use a simplified version of [20] and by checking the consistency of the input images (patches) we keep continuously accepting or rejecting potential patches as positive or negative training examples for the discriminative learner (and updating the reconstructive model).

To further increase the robustness of the reconstructive model, we build two subspace representations in parallel: appearance-based and shape-based representation. The former is created from the cropped and resized appearance patches, which are detected by the motion detector. Since the output of this detector is also a binary segmentation mask, this mask is used to calculate the shape images based on the Euclidean distance transform [3].

Having these models, each image can be checked whether it is consistent with the current models or not. When a false detection occurs, the reconstruction error is significantly larger (i.e., the original image and its reconstruction differ significantly), thus the image gets discarded. Since the main idea of conservative learning is to consider only the images (patches), which are sufficiently consistent with the current model, we accept only the images, which are close enough to both, the appearance and the shape model. We thus assure that the discriminative learner gets most of the time the clean data. And this is done in an incremental manner, feeding the learner continuously, as new data arrives.

2.3. On-line AdaBoost

We introduce a new variant of on-line AdaBoost for object detection based on the well known work of Viola and Jones [21]. The proposed on-line boosting algorithm was inspired by the ideas of Oza et al. [15].

In order to simplify the the further discussion, we will need to define some terms:

Weak classifier A weak classifier $hWeak$ performs slightly better than random guessing (i.e., for a binary decision task, the error rate must be less than 50%). The hypothesis generated by a weak classifier corresponds to a feature using a defined learning algorithm. We use two kinds of features, the classical *Haar Wavelets* [21] (using a simple threshold or a Bayesian decision criterion) and *local edge oriented histograms* [8] (using nearest neighbor for classification).

Base classifier Given a set of M weak classifiers, a base classifier $hBase$ selects exactly one weak classifier from this set: $hBase : hWeak^M \rightarrow hWeak$.

Strong classifier Given a set of N base classifiers, a strong classifier is computed as linear combination of these base classifiers: $hStrong(\mathbf{x}) = \text{sign}(\sum_{n=1}^{N} \alpha_n \cdot hBase_n(\mathbf{x}))$.

For off-line AdaBoost training a fixed set of training images (positives and negatives) and a pool of possible features are given. First, the weight distribution for these examples is initialized. Next, in each boosting iteration a new base classifier and a corresponding weight are computed (using all samples) and the weight distribution of the examples is updated. For that purpose all weak classifiers are trained separately, where the best weak classifier (i.e, the one with the lowest error with respect to the current weight distribution) is selected as base classifier. Finally, a strong classifier is computed as weighted linear combination of the base classifiers added to the system before. The principle is depicted in Fig. 2.

To obtain an on-line boosting algorithm, each of the steps described above must be on-line, where the current classifier is updated whenever a new sample arrives. On-line updating the weak classifiers and therefore updating the base classifiers and estimating the corresponding weights is not the problem. The crucial step is the computation of the weights for the samples, because we don't have a priori knowledge about the difficulty (i.e., we do not know if we have seen the sample before). Thus, the basic idea is to estimate the importance of sample while propagating it through the different base classifiers.

In particular the new on-line AdaBoost training works as follows: First, the fixed set of base classifiers is initialized with randomly chosen weak classifiers. Next, when a new

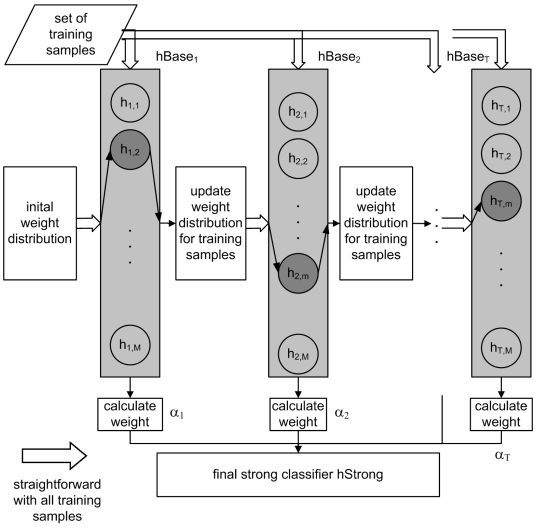

Figure 2: Off-line AdaBoost training.

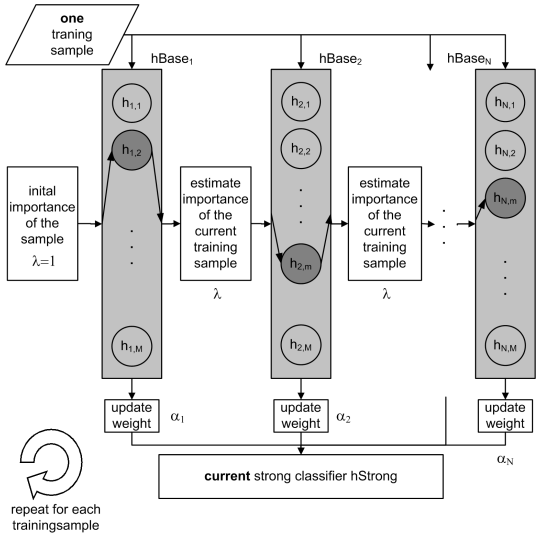

Figure 3: Proposed new on-line AdaBoost training.

training sample arrives the set of classifiers is updated. For updating the weak classifiers any on-line learning algorithm may be used, but we employ a standard Kalman filtering technique to estimate the distribution of positive and negative samples and generate a hypothesis similar as we do in the off-line case. The number of required updates is estimated with respect to the importance of the current sample. The best weak classifier (having the lowest error) is selected as base classifier, where the error for every weak classifier is calculated from the weights of correctly and wrongly classified examples seen so far. Finally, the corresponding weight and the importance weight of the sample are updated and the importance weight is passed to the next base classifier. In order to increase the diversity of the classifier pool and to allow for changes in the environment the worst classifier is replaced with a new one randomly chosen from the feature pool. This procedure is repeated for all base classifiers.

When all base classifiers have been updated a strong classifier is build from the base classifiers (linear combination using the estimated weights). In contrast to the off-line version a classifier is available at any time. The overall principle is depicted in Fig. 3 and is summarized more detailed in Algorithm 2.1.

As a drawback compared to off-line algorithms the discriminative complexity of the classifier is limited, because the number of base classifiers is fixed. But since the proposed system runs on-line, only a small number of weak classifiers is sufficient for updating the base classifiers. By replacing the worst weak classifier within a base classifier in every update step, most of the possible features would be processed, if the process is running for a long time.

3. Experimental Results

3.1. Test Data

For testing our framework we used two different surveillance video sequences. For the first one (*CoffeeCam*), showing a corridor in a public building near to a coffee dispenser, we have recorded images over several days. A simple motion detector triggers the camera and then each second one image is recorded. In total we have recorded over 35000 images. To train the classifiers a sequence containing 1200 frames has been used. For evaluation purposes we have generated a challenging independent test set of 300 frames (containing groups of persons, persons partially occluding each other and persons walking in different directions) and a corresponding ground truth.

The second sequence (*Caviar*), showing a corridor in a shopping center, was taken by the CAVIAR project and is publicly available[1]. There is a great number of short sequences that have been joined to a single one. To avoid redundancy the frame rate was reduced to approx. 1 fps (only every 25th frame was stored). For evaluation purposes an independent test set of 144 frames was created. Note that CAVIAR provides a ground truth in XML format which we used for our evaluation. This ground truth annotates also persons which are only partially visible (e.g., only a hand or head). These test sequences were already used by Wang et al. [23] in a tracking task, but there are no results reported that can be compared to our evaluations. But as can be seen from their videos only a few persons are tracked.

[1] http://homepages.inf.ed.ac.uk/rbf/CAVIARDATA1/

Algorithm 2.1 On-line AdaBoost for feature selection

Require: training example $\langle \mathbf{x}, y \rangle$
Require: strong classifier $hStrong$ (initialized randomly)
Require: weights $\lambda_{n,m}^{corr}$, $\lambda_{n,m}^{wrong}$ (initialized with 1)

initialize $\lambda = 1$
for $n = 1, 2, .., N$ **do**
  // update base classifier $hBase_n$
  set k according to $Poisson(\lambda)$
  for $m = 1, 2, .., M$ **do**
    do k **times**
      update classifier $hWeak_{n,m}$ using $\langle \mathbf{x}, y \rangle$
    end do

    // estimate errors
    if $hWeak_{n,m}(\mathbf{x}) = y$ **then**
      $\lambda_{n,m}^{corr} = \lambda_{n,m}^{corr} + \lambda;\ e_{n,m} = \frac{\lambda_{n,m}^{wrong}}{\lambda_{n,m}^{corr} + \lambda_{n,m}^{wrong}}$
    else
      $\lambda_{n,m}^{wrong} = \lambda_{n,m}^{wrong} + \lambda;\ e_{n,m} = \frac{\lambda_{n,m}^{wrong}}{\lambda_{n,m}^{corr} + \lambda_{n,m}^{wrong}}$
    end if
  end for

  // choose weak classifier with the lowest error
  $m^+ = \arg\min_m(e_{n,m})$
  $e_n = e_{n,m^+};\ hBase_n = hWeak_{n,m^+}$
  if $e_n = 0$ or $e_n > \frac{1}{2}$ **then**
    exit
  end if

  // calculate weighting factor
  $\alpha_n = \frac{1}{2} \cdot \ln\left(\frac{1-e_n}{e_n}\right)$

  // update importance weight
  if $hBase_n(\mathbf{x}) = y$ **then**
    $\lambda = \lambda \cdot \frac{1}{2 \cdot (1-e_n)}$
  else
    $\lambda = \lambda \cdot \frac{1}{2 \cdot e_n}$
  end if

  // replace worst weak classifier with a new one
  $m^- = \arg\max_m(e_{n,m})$
  $\lambda_{n,m^-}^{corr} = 1;\ \lambda_{n,m^-}^{wrong} = 1;$
  get new $hWeak_{n,m^-}$
end for

// update strong classifier
$hStrong(\mathbf{x}) = \text{sign}\Big(\sum_{n=1}^{N} \alpha_n \cdot hBase_n(\mathbf{x})\Big)$

3.2. Description of Experiments

In the conservative learning framework we perform updates only if we are very confident, in particular we used following update rule: The current classifier is applied to a training image; all patches that were labeled as object are verified by motion and PCA (appearance and shape). If the reconstruction error for both, appearance and shape, is very low there is a positive update of the classifier; if the reconstruction error is big and some motion restrictions are fulfilled there is a negative update. Both, positive and negative updates are required.

To monitor the progress during on-line training after several training images the classifier obtained up to now is evaluated on the test sequence.

The only input parameters we need are a crude estimation of the ground plane and the aspect ratio of a person. In all experiments we have used the following parameters: number of eigenvectors for appearance and shape PCA: 10; number of weak classifiers in each base classifier: 120; number of base classifiers: 40.

The experiments are split into two main parts: First, we trained and evaluated classifiers on the *CoffeeCam* sequence. Second, to demonstrate the on-line adaptation capability, a classifier trained on the *CoffeeCam* sequence was applied to the *Caviar* test set.

3.3. CoffeeCam

First, we need an initialization phase to collect positive and negative samples applying a motion based classifier. All patches where motion detection has detected an object are selected as positive examples. The negative examples are obtained by randomly sampling regions where no motion was detected (*AdaBoost1*). Since the motion detector returns approx. 10% false positives a robust reconstructive representation (PCA on appearance and shape) is computed from the output of the motion detector. Thus, the false positives can be filtered out and may be used as negative examples (*AdaBoost2*). Next, we can use the thus obtained data sets to train an initial AdaBoost classifier and PCA models for appearance and shape and start the on-line training. The PCA models were updated using the incremental PCA later on.

Let us have a look at the results obtained by on-line learning. As an evaluation criterion we used similar to [1], precision, recall and the F-measure that can be considered as trade-off between recall and precision. Fig. 4 depicts the performance curves if we start on-line training from *AdaBoost2*. One can see a clear improvement (especially in the first 100 steps) where a lot of false positives can be eliminated. The sudden decrease in performance around frame no. 900 is caused by a single background patch that is detected as a false positive over the whole test sequence; after the next update the curves get back to the previous level.

Fig. 5 depicts the performance curves if we start on-line training from *AdaBoost1*. Since this initial classifier is worse there is a greater number of false positives in the

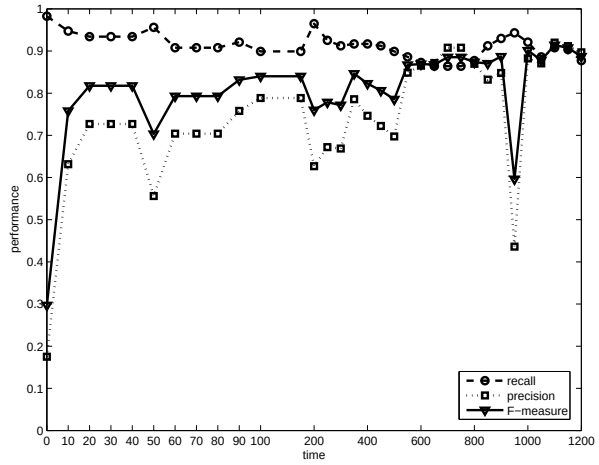

Figure 4: On-line learning started from *AdaBoost2*.

beginning. Therefore more than 300 frames are necessary to obtain comparable results to the previous classifier. This example demonstrates that it is beneficial (1) first to perform a few steps of off-line learning at the beginning and then switch to an on-line version and (2) to use clean data for training.

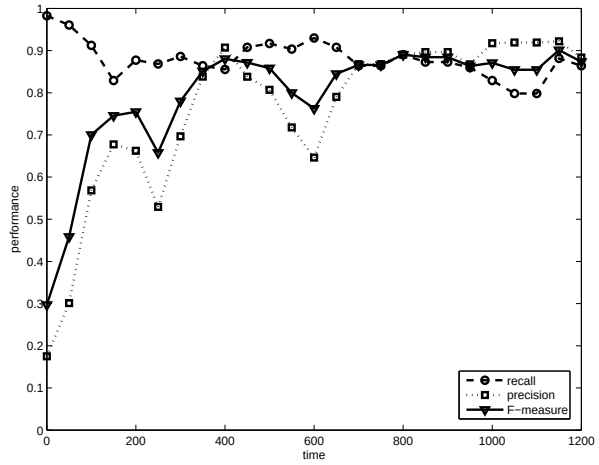

Figure 5: On-line learning started from *AdaBoost1*.

Fig. 6 shows the detections by three different on-line classifiers (initial, after 300 and after 1200 training frames) on the test sequence. There are many false positives in (a) that can be completely removed by on-line training, as can be seen in (b) and (c). Fig. 7 depicts some more examples of persons correctly detected by the final classifier that was trained with 1200 frames.

Finally, we want to show that the on-line algorithms are comparable to the off-line versions of the methods. Therefore we have trained classifiers of different stages using the

(a) (b) (c)

Figure 6: Improvement of on-line classifier: (a) initial classifier, (b) after 300 frames and (c) after 1200 frames.

(a) (b) (c)

Figure 7: Detections by the final classifier trained with 1200 frames.

off-line framework we have proposed in [17] and evaluated them on the test sequence. The initialization phase (collect patches and build an initial classifier) is the same as described above (*Off-line AdaBoost1* and *Off-line AdaBoost2*). To train a new classifier the current classifier is evaluated on another sequence. Thus, new positive and negative examples can be added to the current training set; a new classifier is trained (*Off-line AdaBoost3*). Table 1 depicts these increasingly better results compared to the final classifiers obtained by the on-line framework.

method	recall	prec.	F-m.
Off-line AdaBoost1	97.4 %	27.5 %	42.9 %
Off-line AdaBoost2	91.9 %	57.4 %	70.7 %
Off-line AdaBoost3	93.6 %	94.8 %	94.2 %
AdaBoost1	86.4 %	88.3 %	87.4 %
AdaBoost2	87.7 %	89.7 %	88.7 %

Table 1: Experimental results of the off-line and of the on-line framework.

3.4. Switch to Caviar

After we have shown that the on-line learning framework is working on the *CoffeeCam* data we want to demonstrate two interesting aspects: First, we show that the classifiers trained on the *CoffeeCam* data describe a generalized per-

son model. Therefore Fig. 8 depicts the performance of the classifiers while training them on *CoffeeCam* training data and evaluating them on the *Caviar* test sequence. One can clearly see that the precision (and therefore the F-measure) is improved by on-line training while the recall is roughly constant. This shows that a person model is learned that generalizes over specific setups.

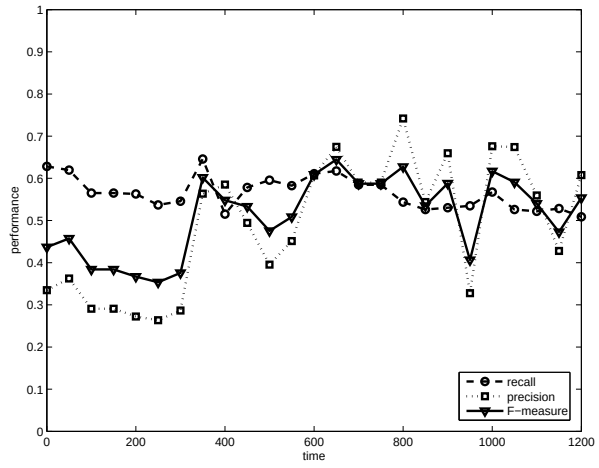

Figure 8: Performance of *CoffeeCam* classifier evaluated on *Caviar* test set.

Second, we demonstrate that the on-line learning framework is able to adapt to a completely different scene. Therefore we perform on-line training on the *Caviar* data set starting with a classifier obtained by the *CoffeeCam* training. Due to the compression noise a new model for appearance is required. Furthermore motion detection can not be applied to the *Caviar* data set, because the quality of the motion blobs is too bad for classifying based on size and aspect ratio restrictions only. Since the shape model is more robust (holes etc.), the shape model estimated from the *CoffeeCam* data can be used to collect patches for generating an appearance based PCA model. The PCA models (appearance and shape) were updated using the incremental PCA later on.

Fig. 9 depicts the evaluation results of this experiment. The main improvement is achieved within the first 100 frames (false positives in the background). The precision (and therefore the F-measure) can be clearly increased. Please note that the obtained performance of approx. 60% detection rate might look quite low, but this is actually quite good considering the given ground truth. If we exclude all persons that are not at least 50% visible we get a detection rate of approx. 79%.

The noisy behavior of the curves can be explained by the nature of on-line learning and the way we perform the evaluation. We have a fixed test set, therefore it may happen that particular cases occurring in the test set are not occurring in the training sequence for some time. Thus, the classifier

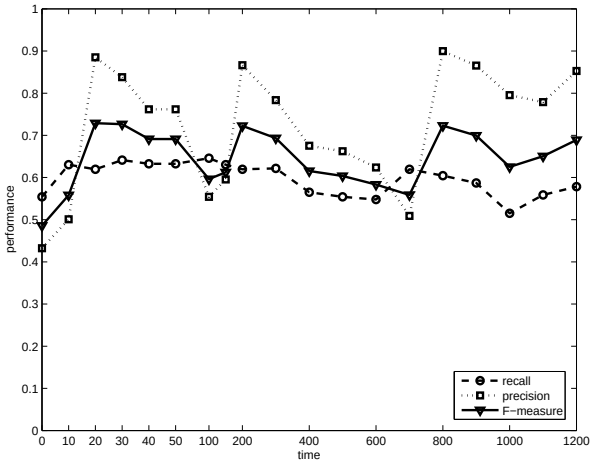

Figure 9: Performance of Caviar *on-line* learning.

will not perform well on this particular data, i.e., if this happens to parts of the background that are visible most of the time.

Finally, we show in Fig. 10 some detections when the final classifier that was trained with 1200 frames is applied on the test sequence.

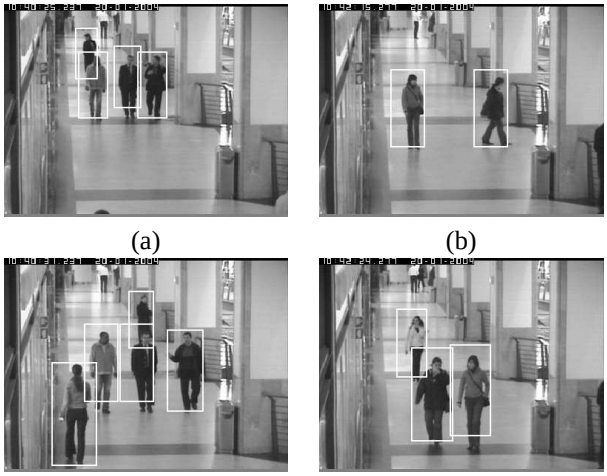

Figure 10: Detections by the final classifier trained with 1200 frames.

4. Summary and Conclusions

In this paper we have presented an on-line conservative learning framework that avoids hand labeling of training data. This framework has been used on two challenging person detection tasks. We have demonstrated that on-line learning obtains comparable results to off-line learning. We have also shown that we can adapt (i.e, reducing the num-

ber of false positives) an already trained person detector to a quite different set-up. While the *CoffeeCam* sequence was obtained with a high quality camera looking down a corridor, the *Caviar* sequence was obtained with a consumer video camera mounted almost horizontally. Moreover, the sequences contain a considerable amount of compression noise. Nevertheless the on-line framework was able to adapt to this quite different scenario. Furthermore, the on-line AdaBoost algorithm can by applied for different applications (e.g., a tracking task similar to [2]).

The proposed framework is quite general (i.e, it can be used to learn completely different objects (e.g., cars)) and can be extended in several ways. More modules (generative and discriminative classifiers) operating on different modalities will further increase the robustness and generality of the system. In particular we will add a tracking algorithm to obtain a wider variety of positive samples (which in turn should increase the detection rate).

References

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

[2] S. Avidan. Ensemble tracking. In *Proc. CVPR 2005*, volume II, pages 494–501, 2005.

[3] H. Breu, J. Gil, D. Kirkpatrick, and M. Werman. Linear time euclidean distance transform algorithms. *IEEE Trans. PAMI*, 17(5):529–533, 1995.

[4] 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.

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

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

[7] A. Leonardis and H. Bischof. Robust recognition using eigenimages. *CVIU*, 78:99–118, 2000.

[8] K. Levi and Y. Weiss. Learning object detection from a small number of examples: The importance of good features. In *Proc. CVPR 2004*, volume II, pages 53–60, 2004.

[9] A. Levin, P. Viola, and Y. Freund. Unsupervised improvement of visual detectors using co-training. In *Proc. ICCV 2003*, volume I, pages 626–633, 2003.

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

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

[12] N. McFarlane and C. Schofield. Segmentation and tracking of piglets. *Machine Vision and Applications*, 8(3):187–193, 1995.

[13] V. Nair and J. J. Clark. An unsupervised, online learning framework for moving object detection. In *Proc. CVPR 2004*, volume II, pages 317–324, 2004.

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

[15] N. Oza and S. Russell. Online bagging and boosting. In *Artificial Intelligence and Statistics 2001*, pages 105–112, 2001.

[16] J.-H. Park and Y.-K. Choi. On-line learning for active pattern recognition. *IEEE Signal Processing Letters*, 3(11):301–303, 1996.

[17] P. Roth, H. Grabner, D. Skočaj, H. Bischof, and A. Leonardis. Conservative visual learning for object detection with minimal hand labeling effort. In *Proc. DAGM 2005*, volume 3663 of *LNCS*, pages 293–300. Springer, 2005.

[18] H. Rowley, S. Baluja, and T. Kanade. Neural network-based face detection. *IEEE Trans. PAMI*, 20(1):23–38, 1998.

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

[20] D. Skočaj and A. Leonardis. Weighted and robust incremental method for subspace learning. In *Proc. ICCV 2003*, volume II, pages 1494–1501, 2003.

[21] P. Viola and M. Jones. Rapid object detection using a boosted cascade of simple features. In *Proc. CVPR 2001*, volume I, pages 511–518, 2001.

[22] P. Viola, M. Jones, and D. Snow. Detecting pedestrians using patterns of motion and appearance. In *Proc. ICCV 2003*, volume II, pages 734–741, 2003.

[23] J. Wang, X. Chen, and W. Gao. Online selecting discriminative tracking features using particle filter. In *Proc. CVPR 2005*, volume II, pages 1036–1041, 2005.