Improving Classifiers with Unlabeled Weakly-Related Videos*

Christian Leistner[1)], Martin Godec[2)], Samuel Schulter[2)],
Amir Saffari[3)], Manuel Werlberger[2)], Horst Bischof[2)]

[1)]Computer Vision Laboratory, ETH Zurich, Switzerland

[2)]Institute for Computer Graphics and Vision, TU Graz, Austria

[3)]Oxford Brookes University and Sony Computer Entertainment, London, UK

leistner@vision.ee.ethz.ch, {godec,schulter,werlberger,bischof}@icg.tugraz.at, amir@ymer.org

Abstract

Current state-of-the-art object classification systems are trained using large amounts of hand-labeled images. In this paper, we present an approach that shows how to use unlabeled video sequences, comprising weakly-related object categories towards the target class, to learn better classifiers for tracking and detection. The underlying idea is to exploit the space-time consistency of moving objects to learn classifiers that are robust to local transformations. In particular, we use dense optical flow to find moving objects in videos in order to train part-based random forests that are insensitive to natural transformations. Our method, which is called Video Forests*, can be used in two settings: first, labeled training data can be regularized to force the trained classifier to generalize better towards small local transformations. Second, as part of a tracking-by-detection approach, it can be used to train a general codebook solely on pair-wise data that can then be applied to tracking of instances of a priori unknown object categories. In the experimental part, we show on benchmark datasets for both tracking and detection that incorporating unlabeled videos into the learning of visual classifiers leads to improved results.*

1. Introduction

In recent years, classical computer vision tasks such as object recognition and tracking have made significant progress. This has been achieved mainly by developing and applying both improved image representations and advanced machine learning algorithms. Nevertheless, the performance of these systems is significantly lacking in comparison to humans, which is a very frustrating fact, considering the large efforts that are put into computer vision research for a task that humans seem to perform both effortlessly and accurately.

Typical approaches to object classification and detection collect lots of labeled training images, calculate one or several hand-designed features and pass them to a prominent learning algorithm of choice, *e.g.*, SVMs [9] or Boosting [36]. The algorithms try to minimize the training error by generalizing as well as possible to unseen images. It turns out that the generalization capability of such approaches improves vastly with the number of labeled training samples. As collecting large amounts of hand-labeled data can be tedious and time-intensive, there has been increased interest in semi-supervised learning methods (SSL) [43], which try to learn from only a small amount of labeled data and a large amount of unlabeled data. In practice, however, SSL is hard to apply because labeled and unlabeled data are often collected from heterogenous sources, which violates the underlying *i.i.d* assumption of most methods. Additionally, with previous methods, incorporating unlabeled samples that hardly comprise the target class can deteriorate the accuracy.

In this paper, we argue in favor of a very rich source of unlabeled data, which has often been ignored, and this is video. Video can improve a classifier with unlabeled data even when the unlabeled samples are weakly-related towards the target class, *i.e.*, the unlabeled samples come from different distributions than the labeled data and the target classes are not necessarily present in the videos. Such a setting occurs, for instance, when training a car detector

*This work has been supported by the Austrian FFG projects MobiTrick (8258408) and OUTLIER (820923) under the FIT-IT program as well as the HDVIP Bridgeproject (827544).

from both a handful of labeled car images and several unlabeled video sequences comprising arbitrary moving objects, such as aeroplanes and lions, but not a single car instance. Observing transformations of real-world objects in videos, representing different categories, can improve the car detector because they are very likely to share the same physical contraints with the labeled data when undergoing natural transformations [31]. Hence, observing these transformations allows for learning representations that are transformation invariant and yields better generalizing classifiers for the target class.

The main contribution of this paper is a part-based learning approach that uses labeled data and, simultaneously, exploits a continuous stream of unlabeled visual data from videos. As we want to exploit fully unlabeled natural sequences, our approach does not assume the class labels and locations of the objects in the videos to be known, and it is further unknown if a video comprises any objects at all. However, video delivers a very strong unsupervised cues in form of motion, which allows to determine regions that are very likely to contain objects of interest and allows to get rid of uninformative background clutter. In this work, we use dense optical flow for motion detection. Furthermore, we crop small patches in moving regions in order to train part-based classifiers, because parts are more likely to be shared among various categories. The cropped patches are tracked using flow. Due to the space-time coherence of objects appearing in videos, *i.e.*, succeeding frames are very likely to contain the same objects and content, we let the cropped patches form pairs being "same" if they correspond to succeeding frames and "different" if not. With this approach it is possible to create a large pool of pair-wise training data and – regardless of the individual categories – use a learning method that tries to keep "same" pairs together and pushes "different" pairs apart, respectively.

In this work, we show how this learning can be accomplished with Random Forests (RFs) and thus call our approach *Video Forests*. RFs are ideal candidates for exploiting unlabeled videos because they are fast, robust to noise and allow for using parallel computing architectures. We incorporate our approach as a part of a state-of-the-art detection framework, *i.e.*, [15, 25], and use the Video Forests in two different settings: first, if some labeled data is already available, our approach additionally incorporates unlabeled video sequences in order to improve the classifiers. In the second setting, Video Forests are trained solely on unlabeled samples in order to form a general codebook that can be updated online and can be used to track arbitrary target objects.

2. Related Work

In computer vision, video is often used to improve object detectors if both training and testing can be done on video data. In such settings, having labeled objects in training videos, it is possible to design motion features that operate over pairs or sequences of frames, yielding higher accurate classifiers on test videos, *e.g.*, [37, 10]. The same principle can also be applied to action recognition [35]. Video Forests differ from these approaches in a way that, during training, we exploit video data without knowing any class labels and object locations and, during testing, the resulting classifiers can be applied to non-sequential data, such as single images. This is also related to previous approaches that try to train neural networks via exploiting the spatial continuity of moving objects [14, 38, 8].

Continuous Transformation Learning [34, 26] and Slow Feature Analysis [40] are also approaches that try to learn invariant representations by enforcing spatio-temporal continuities. Recently, Stavens and Thrun [33] showed that it is possible to tune the parameters of keypoint descriptors via observing unlabeled tracked keypoints for specific applications.

For text classification, Yang *et al.* [41] presented an approach based on SVMs that is able to improve a text classifier by additionally using weakly-related unlabeled data. However, the method is computationally demanding and the assumptions imposed for text hardly hold for images. For categorization, [28] showed that it is possible to train better representations from weakly-related, unlabeled images by basically learning natural image statistics using sparse coding. [23] improved convolutional neural networks by learning from labeled samples and unlabeled artificial objects in videos and is thus the most related to our work. In contrast to [23], we present a learning approach for random forests, which is very attractive due to its speed and simplicity and is part-based, which makes it more flexible and versatile for practical usage. Additionally, we present a processing pipeline that is able to exploit real-word videos and improve object detectors and trackers.

Video Forests are also related to approaches that try to transfer attributes among different categories in order to improve object recognition systems, *e.g.*, [18, 32]. What makes our work different is that we do not have to label attributes, *e.g.*,

"hand","head","wheel", *etc*., and we do not need to know the labels of the weakly-related objects.

Besides the above mentioned works, strong motivation for exploiting video in order to get better vision systems comes from the cognitive sciences. It has been shown that the power of human perception comes from the fact that the brain performs a rapid stage-wise feed-forward transformation of an image into a successively higher dimensional – and probably sparse – space, where classification and recognition can be done reliably with simple linear decision functions [11]. Additionally, these transformations are highly identity preserving and they allow for high intra-class variances of objects while simultaneously being discriminative between different classes. Recent findings suggest that these transformations are not entirely "hard coded", *i.e.*, they are not given natively, but are learned in the first few months of a human's life via observing motions of objects with mostly unknown categories. In other words, humans learn invariant object representations that are useful for a large amount of categories via observing a continuous stream of – mostly unlabeled – visual data [31, 38, 22, 4]. In computer vision, this corresponds to learning from unlabeled videos.

3. Learning Transformation Invariant Classifiers with Random Forests

In this section, we introduce a random forest-based approach called *Video Forests* that is able to learn from pair-wise motion cues, yielding more transformation invariant and better generalizing classifiers.

Random Forests [7] are ensembles of decision trees. We write a forest as $\mathcal{F} = \{f_1, \cdots, f_N\}$, where the n^{th} tree is $f_n(\mathbf{x}) = f(\mathbf{x}, \Phi_n) : \mathcal{X} \rightarrow \mathcal{Y}$, Φ_n is a random vector capturing the various stochastic elements of the tree, and N is the number of trees in the forest. The estimated probability for predicting class k for a sample is

$$p(k|\mathbf{x}) = \frac{1}{N} \sum_{n=1}^{N} p_n(k|\mathbf{x}), \tag{1}$$

where $\mathbf{x} \in \mathcal{X} \subseteq \mathbb{R}^d$ and $k \in \mathcal{Y} = \{1, \cdots, K\}$, whith K being the number of classes, and $p_n(k|\mathbf{x})$ is the estimated density of class labels of the leaf of the n^{th} tree. As randomized trees are inherently multi-class, the final multi-class decision function of the forest is defined as

$$C(\mathbf{x}) = \underset{k \in \mathcal{Y}}{\arg\max}\, p(k|\mathbf{x}). \tag{2}$$

Each tree in the forest is built and tested independently from other trees, which inherently allows for using parallel computing architectures. While growing a randomized tree each node selects the best split according to some quality measurement which scores the potential information gain

$$\Delta H = -\frac{|I_l|}{|I_l| + |I_r|} H(I_l) - \frac{|I_r|}{|I_l| + |I_r|} H(I_r), \tag{3}$$

where I_l and I_r are the left and right subsets of the training data, respectively, and $H(I)$ is the node score, usually measured using the entropy. The term "random" comes from the fact that the node splits are selected randomly, which increases the training speed and reduces the correlation among the trees.

3.1. Learning from Pair-Wise Data

A standard random forest as reviewed above is trained given training data in the form of $\{(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_n, y_n)\}$, where $\mathbf{x}_i$ is an instance and $y_i \in \{-1, +1\}$, in the binary case, the corresponding label. As our goal is to train from unlabeled video, where the individual frames comprise unlabeled samples in the form of $\{(\mathbf{x}_{n+1}), \ldots, (\mathbf{x}_{n+U})\}$, one can in principle take a semi-supervised extension of RFs, *e.g.*, as proposed in [21]. However, we assume that the unlabeled data come from different distributions than the labeled samples, wheras standard semi-supervised methods assume both labeled and unlabeled samples to be drawn *i.i.d* and, hence, do not bring any benefit in our setting [43]. However, as stated above, unlabeled data coming from video has an underlying structure that makes it informative due to the space-time consistency of objects appearing in consecutive frames. In the following, we show how to modify a RF so that it is able to exploit the underlying structure from unlabeled videos.

In order to create training samples out of succeeding video frames (see also Section 4), we extract training pairs in form of $\{(\mathbf{x}^t, \mathbf{x}^{t+1}, y)\}$, where y is a "pseudo" label denoted as "same" if $(\mathbf{x}^t, \mathbf{x}^{t+1})$ are taken from two succeeding frames and y is denoted as "different" if they do not correspond. The goal now is to train trees that try to keep "same" samples together and push "different" samples apart, respectively. However, during the evaluation phase, the trees should be able to deliver class predictions for single instances in form $p(y|\mathbf{x})$. Note that this setting is different to standard approaches in distance function learning, *e.g.*, [17, 24], where both training and evaluation is performed over pairs.

As we want to learn a part-based classifier, we assume pairs being small patches $\{(\mathbf{p}_i^t, \mathbf{p}_i^{t+1})\}_{i=1}^K$, which are K pairs of patches extracted from corresponding spatial locations at each pair of frames $\{(\mathbf{x}^t, \mathbf{x}^{t+1}, y)\}$. The appearance of a patch can be written as $\mathcal{I}_i = (I_i^1, I_i^2, \ldots, I_i^C)$, where C is the number of channels. Thus, each frame is able to provide a large number of patches extracted from moving objects.

In our tree, each node uses randomly selected binary tests that are evaluated on single patches in the form $\tau(\mathbf{p}_i) \rightarrow \{0, 1\}$, where $\mathbf{p}_i$ is the i^{th} patch that falls into the node. In particular, a binary test is defined as

$$\tau(\mathbf{p}; x, y) = \begin{cases} 1 & if \quad \mathbf{p}(x) < \mathbf{p}(y) + \theta \\ 0 & else \end{cases} ,$$

where θ is a threshold and $\mathbf{p}(x)$ and $\mathbf{p}(y)$ are feature responses of patch $\mathbf{p}$ at location x and y, respectively. During training, a node is split according to a random test that best minimizes the within pair splitting error

$$\begin{aligned} \varepsilon_{pair} = & \frac{1}{I_{|same|}} \sum_{I_{same}} \mathbb{I}(\tau(\mathbf{p}_i^t; x, y) \neq \tau(\mathbf{p}_i^{t+1}; x, y)) \\ & + \frac{1}{I_{|diff|}} \sum_{I_{diff}} \mathbb{I}(\tau(\mathbf{p}_i^t; x, y) = \tau(\mathbf{p}_i^{t+1}; x, y)), \end{aligned} \tag{4}$$

where $\mathbb{I}(\cdot)$ is an indicator function. Considering a subset of the training pairs $I_{|same|}$ and $I_{|diff|}$ that fall into the current node, a node is penalized for a split test τ and a threshold θ that splits patches within a "same" pair into different child nodes and keeps patches within a "different" pair together, respectively. As a side effect, due to the hard splitting rule of the nodes, the trees discard patches from both "same" and "different" pairs in case they are split apart by the chosen random test. To sum up, although the trees are trained from pair-wise data, the node splits are designed to be applied on single instances, which allows the final random forest to be evaluated on non-pairwise data.

3.2. Regularizing Hough Forests with Unlabeled Data

There exist several works that have shown that it is possible to train highly accurate object detectors based on the Generalized Hough Transform [5, 20]. Recently, Gall and Lempitsky [15] as well as Okada [25] proposed an approach based on random forests that yields state-of-the-art detection results via learning a part-based classifier and simultaneously a discriminative Hough codebook. The key idea is to use not only classification nodes, *e.g.*, see Eq. (3), but also use regression nodes. These nodes minimize the offset error of patches with respect to the center of the target object as $U(I) = \sum_{i=1}^{|I|} (\mathbf{d}_i - \mathbf{d}_I)^2$, where $\mathbf{d}_i$ is the offset of the patch and $\mathbf{d}_I$ is the mean offset vector over all patches in the node. Training a tree with interleaving classification and regression nodes leads to leafs with low variations in both class labels and offsets. During evaluation, such trees cast probabilistic votes for each test patch into a 2D Hough image $V(\mathbf{x})$ for each pixel location. See [15, 25] for more details.

In order to benefit from unlabeled video sequences, we extend Hough forests by regularizing classification nodes in a way which aims at minimizing the error on pair-wise samples. Thus, such a regularized entropy measure can be written as

$$\Delta H^* = \Delta H - \lambda \cdot \varepsilon_{pair},$$

where $\lambda \in [0, 1]$ steers the influence of the pair-wise data. It is easy to see that such a test forces a node to find splits that both maximize the information gain on labeled data and minimize the error on pair-wise samples.

3.3. Learning Codebooks for Tracking

Up to now, we have shown how a random forest can be trained on motion-constrained pair-wise data and how this can be used to regularize a Hough forest, with the goal to better generalize to natural appearance changes of local patches. The same principle can also be applied to the task of visual object tracking. In particular, we follow the "tracking-by-detection" principle [2], where typically at the first frame a classifier is trained by using a marked target object as the positive sample and the negative samples are extracted from the surrounding background, respectively. During tracking, the classifiers update themselves online at new object locations in order to adapt to illumation and appearance changes. Although these methods yield highly effective trackers, it would be beneficial to learn valid transformations from previously observed unlabeled objects, which share the same physical contraints, and transfer this knowldege to a specific tracking target.

To this end, we can use Video Forests to learn an invariant codebook from pair-wise unlabeled data

extracted from videos. Once having created such a codebook from unlabeled data only, we can use it for tracking as follows: as soon as a specific target object is marked, we extract labeled foreground and background patches and pass them through our codebook. While the split nodes remain unchanged, the leaf nodes gather statistics for the labeled patches, which can be used to track the object. During tracking, the codebook ensures that patches do not end up in different leaf nodes from frames t to $t+1$ due to small transformations, while the statistics are further updated online in order to yield an adaptive tracker.

Again, we encode this principle inside a Hough framework and follow [16], where it has been shown that traditional Hough forests can also be applied to tracking, but with the limitation that the targeted object category has to be known before. In particular, [16] show that a pre-trained class-specific Hough forest can be adapted online in order to track instances of that class by adding instance-specific leaf probabilies $p(P_i \in I | c = 1, L(y))$ to the pre-calculated class-specific leaf probabilities $p(c = 1 | L(y))$, with patch y ending up at leaf $L(y)$ and I being the specific instance of class c.

Although the method yields high tracking accuracy for categories such as persons and faces, it has the main disadvantage that it cannot be applied to arbitrary objects. However, when the class-specific codebook is replaced by our general codebook, which is trained on pair-wise unlabeled samples, it is possible to track arbitrary objects by modeling target-specific leaf statistics purely online as $p(P_i \in I | L(y))$ and the offset vectors $\mathbf{d}_i$ during runtime when we already know our target object.

4. System Approach

In the previous section, we introduced a generic learning approach based on random forests that is able to learn on pair-wise data extracted from videos. Moreover, we use this data to regularize a Hough forest in order to select binary tests that are less sensitive to small local appearance changes. In this part of the paper, we explain the further steps of our method that are necessary to build a real-world system that is able to exploit unlabeled videos.

We assume having a set of video sequences $\{\mathcal{V}_j\}$ consisting of J sequences, where each $\mathcal{V}_j$ has T_j frames and comprises various natural objects. First, since we do not assume any a priori information, such as class-labels and locations of specific objects, we calculate a dense optical flow for each pair of succeeding frames (Fig. 1). Optical flow is computed according to [39] as the implementation is publicly available. The approach is robust and fast enough to process a large amount of data in reasonable time. As cameras are often moving, we subtract the background motion of the flow. In sequences where moving objects are pictured, the camera movement often already provides a good framing of the dominant object (*c.f.* [42]). In contrast to [42], we do not restrict the camera motion only to translational movement but allow for any transformation to happen. For this purpose we take a small part of the optical flow's border region to compute possible hypotheses and a RANSAC approach then gives a robust estimate of the camera movement even when parts of the border regions are cluttered. Next, similar to [13], the resultant camera-movement compensated optical flow is segmented into binary regions using a Potts model approach [27]. At frame t we then extract patches at random locations inside the largest connected region and use flow to track the movement of each patch to frame $t+1$. Finally, for each patch in t we crop its corresponding patch at the new location inside frame $t+1$, which allows for forming a large number of positive patch pairs. Negative pairs are simply created by cropping patches from moving objects and corresponding patches from random negative images.

5. Experiments

We tested Video Forests on object detection and tracking. To collect sequences of naturally moving objects, we downloaded short public available animal videos from Youtube [1], which do not contain any instance of the target classes. Using the approach presented in Section 4, we extracted 10000 unlabeled random pairs of size 16x16 pixels. For both tracking and detection, we used the public available Hough forest implementation of [15], which has shown to yield state-of-the-art results, and modified it so that it is able to both learn from pair-wise data using our proposed node-splits and can be updated online, where the latter one is necessary for tracking.

5.1. Object Detection

As [15], we evaluate the Video Hough forests (VHF) on the TUD pedestrian dataset [1] and Weizmann horses [6], where we modified the latter one in a way that we randomly mirrored some of the horse

[1] Code and data can be downloaded from www.vision.ee.ethz.ch/∼cleistne/code.html

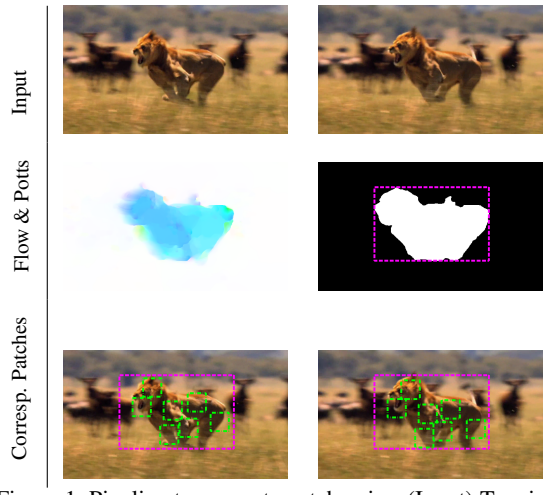


Figure 1. Pipeline to generate patch pairs: (Input) Two input frames taken from a video $\mathcal{V}_j$ used as input to compute optical flow with color encoded directions (Flow & Potts). After an additional camera motion compensation and a labeling stage we cluster dominant movements together and, alhough not illustrated, resize the segments to a unit height. Then, we extract the "same" patches at random locations in frame t and the corresponding new locations in frame $t+1$ (Corresp. Patches). The small green rectangles indicate corresponding patches.

images, so that they are not always looking to the left side, making the task more challenging. For the car detection task, we used the Leuven multi-view car dataset [19]. We set the influence of the video-regularization to $\lambda = 0.5$, a value determined empirically. We used 10 trees with a maximum depth of 15. For performance measure, we used the PASCAL overlap criteria [12]. The first row of Figure 2 compares our detector with the baseline Hough Forest [15] algorithm on these datasets. As can be seen on all these datasets, training with pairs of unlabeled patches improves the baseline significantly. We also incorporated the state-of-the-art SSL approach for random forests [21] into the training of the Hough Forest. In contrast to the Video Forests, using [21] in order to learn from the unlabeled videos always led to a decrease in performance [2]. One explanation for this is that, in contrast to [21], Video Forests are "aware" that the unlabeled samples are drawn from different distributions than the labeled ones and, hence, do not try to estimate the real labels. Additionally, pairs that are not similar in their appearance to the labeled positive samples fall into negative sub-branches of a tree. Also, "noisy" same pairs, *i.e.*, pairs that are hard to keep together, are inherently discarded and do not have influence on the regularization further down the tree. We believe that this is very important for the robustness of the approach.

In the last row of Figure 2, we provide a detailed analysis of the algorithm by varying the value of λ, the number of labeled samples, and the number of pairs used for training with Weizmann horses. We conclude from the experiments that our approach is pretty robust over the choice of λ and also helps improving the performance when we increase the number of labeled samples. In Figure 2f, we see that the accuracy improves with various numbers of unlabeled patches but the improvement becomes smaller when the number of unlabeled samples becomes comparably large.

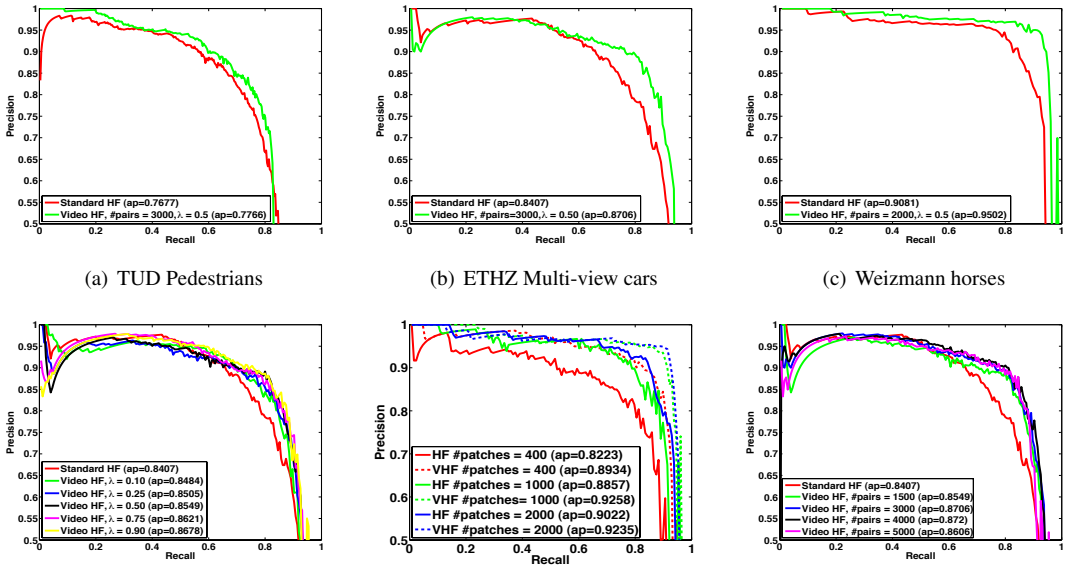
5.2. Tracking with a Generic Codebook

As described in Section 3.3, we perform pair-wise training on the animal sequences to generate a general codebook with empty leaf statistics suitable for tracking of a priori unknown objects. When the object is marked at the first frame we start gathering leaf statistics, which are further self-updated during ongoing tracking. For the codebook, we use a forest with 8 trees and a maximum depth of 8, trained with 10000 pair-wise samples. At the first frame, we use 5 perspective transformations of our initial input image cropping 100 positive and 500 negative samples to increase the robustness of the leaf statistics. During updating, we perform 10 positive and 50 negative updates per frame and limit the number of center votings per leaf to 30. For measurement, we report the number of frames tracked correctly (Pascal VOC overlap criterion > 0.5) over 5 independent runs.

Pedestrian Tracking. As a first tracking experiment, we compare to the recent approach of Gall *et al.* (oaHF) [16][3]. Since we do not handle scaling, we rescaled the center of the ground truth bounding boxes to the size of the tracking box. Due to this, the tracking result has to be more centered on the object in order to get high scores, *i.e.*, to get a good overlap, which makes it harder for our approach compared to [16]. In Table 1, sequences two and three, it can be seen that oaHF delivers better results than Video Forests when the tracking sequence is very hard. This comes from the fact that the person-specific Hough forest is trained on the tracking category and can easily recover from temporal failure. In contrast, for sequences one and four, Video Forests

[2]Note that we skip the detailed numbers since we were never able to achieve increased accuracy with SSL.

[3]Thanks to the authors of [16] for providing their dataset.



(a) TUD Pedestrians (b) ETHZ Multi-view cars (c) Weizmann horses

(d) Influence of λ (e) Influence of #labeled samples (f) Influence of #pairs

Figure 2. The first row shows the improvement when using pair-wise regularization from animal videos (green) compared to the baseline method (red). For cars, we also showed the improvement, when the videos are more related, *i.e.*, trucks (blue). In the second row, for Weizmann horses, we depict the influence on the performance depending on λ, the number of labeled samples and the number of pairs. (Best viewed in color)

Sequence	VHF	oaHF [16]
i-Lids easy	**74 (94)**	69
i-Lids medium	44(50)	**65**
i-Lids hard	27(28)	**66**
PETS09	**70 (87)**	60

Table 1. Comparison on Pedestrian Tracking (Average Pascal VOC score (percentage of correctly tracked frames)). Best performing method marked bold-face.

outperform oaHF, which might be explained by the fact that oaHF are too inertial for instances that are not well covered in the offline training set.

Tracking of arbitrary objects. Finally, we demonstrate the performance of our approach on standard tracking sequences and compared to state-of-the-art methods within this field. We compare to online Random Forests (ORF) [29], PROST [30] and online multiple instance Boosting (MILB) [3], which is a prominent approach under several boosting-based methods. Our testset consists of six sequences[4] taken from [3, 30]. These sequences capture very different objects and include variations in illumination, pose, scale, rotation and appearance, and partial occlusions. All compared approaches do not use any prior information about the target object. Table 2 shows that Video Forests are able to track arbitrary objects and yield state-of-the-art results on the test sequences. On average, we exceed the state-of-the-art by far. Remarkably, we are even better than PROST, which uses optical flow in addition.

[4]We did not use sequences with objects smaller than 70×70 pixels because they are not suitable for Hough voting in our part-based setup.

Sequence	VHF	ORF [29]	MILB [3]	PROST [30]
david	**95**	72	60	<u>80</u>
faceocc2	<u>93</u>	65	**96**	82
lemming	<u>79</u>	17	**83**	71
girl	**96**	**96**	57	89
liquor	<u>80</u>	54	21	**84**
Average	**89**	61	63	<u>81</u>

Table 2. Accuracy comparison using the Pascal VOC overlap criterion (percentage of correctly tracked frames). Best performing method marked bold-face. Second best method marked underlined.

Summarizing, the experiments show that training a generic codebook on unlabeled videos yields a highly efficient tracker, suitable for the tracking of arbitrary objects. However, the results also confirm that it is beneficial to train a class-specific codebook for tracking, in case the object category is a priori known and the used labeled data well represent the instances that are going to be tracked.

6. Conclusions

We have presented a learning approach denoted as *Video Forests* that is able to exploit unlabeled real-world video sequences in order to improve the robustness of classifiers. The main idea is to exploit the space-time coherence of moving objects in order to train a random forest, which results in better detection performance on several benchmark datasets and can also be used to learn a general codebook for a tracking-by-detection framework to track object instances of a priori unknown classes. While it is clear that the content and quality of the videos influence the performance of a specific task, we showed for both tracking and detection that visual classifiers can benefit from unlabeled videos, even if the content is highly un-related to the specific task, where the latter one cannot be achieved with standard SSL methods.

References

[1] M. Andriluka, S. Roth, and B. Schiele. People-traacking-by-detection and people-detection-by-tracking. In *CVPR*, 2008.

[2] S. Avidan. Ensemble tracking. In *CVPR*, 2005.

[3] B. Babenko, M. H. Yang, and S. Belongie. Visual tracking with online multiple instance learning. In *CVPR*, 2009.

[4] B. Balas and P. Sinha. Observing object motion induces increased generalization and sensitivity. *Perception*, 37(8):1160–1174, 2008.

[5] D. Ballard. Generalizing the hough transform to detect arbitrary shapes. *Pattern Recognition*, 1981.

[6] E. Borenstein and S. Ullmann. Class-specific, top-down segmentation. In *ECCV*, 2002.

[7] L. Breiman. Random forests. *Machine Learning*, 2001.

[8] C. Cadieu and B. Olshausen. Learning transformational invariants from natural movies. In *NIPS*, 2009.

[9] N. Dalal and B. Triggs. Histograms of oriented gradients for human detection. In *CVPR*, 2005.

[10] N. Dalal, B. Triggs, and C. Schmid. Human detection using oriented histograms of flow and appearance. In *ECCV*, 2006.

[11] J. DiCarlo and D. Cox. Untangling invariant object recognition. *Trends in Cognitive Sciences*, 2007.

[12] M. Everingham, L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman. The pascal visual object classes (voc) challenge. 2010.

[13] A. J. D. Fleet and M. Black. A layered motion representation with occlusion and compact spatial support. In *ECCV*, 2002.

[14] P. Fldiak. Learning invariance from transformation sequences. In *Neural Comput*, 1991.

[15] J. Gall and V. Lempitsky. Class-specific hough forests for object detection. In *CVPR*, 2009.

[16] J. Gall, N. Razavi, and L. V. Gool. On-line adaption of class-specific codebooks for instance tracking. In *BMVC*, 2010.

[17] T. Hertz, A. Bar-Hillel, and D. Weinshall. Learning distance functions for image retrieval. In *CVPR*, 2004.

[18] C. Lampert, H. Nickisch, and S. Harmeling. Learning to detect object classes by between-class attribute transfer. In *CVPR*, 2009.

[19] B. Leibe, N. Cornelis, K. Cornelis, and L. V. Gool. Dynamic 3d scene analysis from a moving vehicle. In *CVPR*, 2007.

[20] B. Leibe and B. Schiele. Robust object detection with interleaved categorization and segmentation. *IJCV*, 2008.

[21] C. Leistner, A. Saffari, J. Santner, and H. Bischof. Semi-supervised random forests. In *IEEE International Conference on Computer Vision (ICCV)*, 2009.

[22] N. Li and J. DiCarlo. Unsupervised natural experience rapidly alters invariant object representation in visual cortex. *Science*, pages 1502–1507, 2008.

[23] H. Mobahi, R. Collobert, and J. Weston. Deep learning from temporal coherence in video. In *ICML*, 2009.

[24] E. Nowak and F. Jurie. Learning visual similarity measures for comparing never seen objects. In *CVPR*, 2007.

[25] R. Okoda. Discriminative generalized hough transform for object detection. In *ICCV*, 2009.

[26] G. Perry, E. Rolls, and S. Stringer. Continuous transformation learning of translation invariant representations. *Experimental Brain Research*, 204:255–270, 2010.

[27] R. Potts. Some generalized order-disorder transformations. *Mathematical Proceedings of the Cambridge Philosophical Society*, Jan 1952.

[28] R. Raina, A. Battle, H. Lee, B. Packer, and A. Y. Ng. Self-taught learning: Transfer learning from unlabeled data. In *ICML*, 2007.

[29] A. Saffari, C. Leistner, J. Santner, M. Godec, and H. Bischof. On-line random forests. In *3rd IEEE ICCV Workshop on On-line Computer Vision*, 2009.

[30] J. Santner, C. Leistner, A. Saffari, T. Pock, and H. Bischof. PROST Parallel Robust Online Simple Tracking. In *CVPR*, San Francisco, CA, USA, 2010.

[31] E. Spelke. Principles of object perception. *Cognitive Science*, 1990.

[32] M. Stark, M. Goesele, and B. Schiele. A shape-based object class model for knowledge transfer. In *ICCV*, 2009.

[33] D. Stavens and S. Thrun. Unsupervised learning of invariant features using video. In *CVPR*, 2010.

[34] M. Stringer, G. Perry, E. Rolls, and J. Proske. Learning invariant object recognition in the visual system with continuous transformations. In *Biol Cybern*, 2006.

[35] G. W. Taylor, R. Fergus, Y. LeCun, and C. Bregler. Conolutional learning of spatio-temporal features. In *ECCV*, 1997.

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

[37] P. Viola, M. Jones, and D. Snow. Detecting pedestrians using patterns of motion and appearance. *IJCV*, 2005.

[38] G. Wallis and H. Bülthoff. Effects of temporal association on recognition memory. In *NAS*, 2001.

[39] M. Werlberger, W. Trobin, T. Pock, A. Wedel, D. Cremers, and H. Bischof. Anisotropic Huber-L1 optical flow. In *Proceedings of the British Machine Vision Conference (BMVC)*, London, UK, September 2009.

[40] L. Wiskott and T. Sejnowski. Slow feature analysis: Unsupervised learning of invariances. In *Neural Comput*, 2001.

[41] L. Yang, R. Jin, and R. Sukthankar. Semi-supervised learning with weakly-related unlabeled data: Towards better categorization. 2008.

[42] A. Yao, D. Uebersax, J. Gall, and L. V. Gool. Tracking people in broadcast sports. In *32nd Annual Symposium of the German Association for Pattern Recognition*, Jan 2010.

[43] X. Zhu and A. B. Goldberg. *Introduction to Semi-Supervised Learning*. Synthesis Lectures on Artificial Intelligence and Machine Learning. Morgan & Claypool, 2009.