# Learning Object Class Detectors from Weakly Annotated Video

Alessandro Prest[1,2], Christian Leistner[1], Javier Civera[4], Cordelia Schmid[2] and Vittorio Ferrari[3]

[1]ETH Zurich [2]INRIA Grenoble [3]University of Edinburgh [4]Universidad de Zaragoza

## Abstract

*Object detectors are typically trained on a large set of still images annotated by bounding-boxes. This paper introduces an approach for learning object detectors from real-world web videos known only to contain objects of a target class. We propose a fully automatic pipeline that localizes objects in a set of videos of the class and learns a detector for it. The approach extracts candidate spatio-temporal tubes based on motion segmentation and then selects one tube per video jointly over all videos. To compare to the state of the art, we test our detector on still images, i.e., Pascal VOC 2007. We observe that frames extracted from web videos can differ significantly in terms of quality to still images taken by a good camera. Thus, we formulate the learning from videos as a domain adaptation task. We show that training from a combination of weakly annotated videos and fully annotated still images using domain adaptation improves the performance of a detector trained from still images alone.*

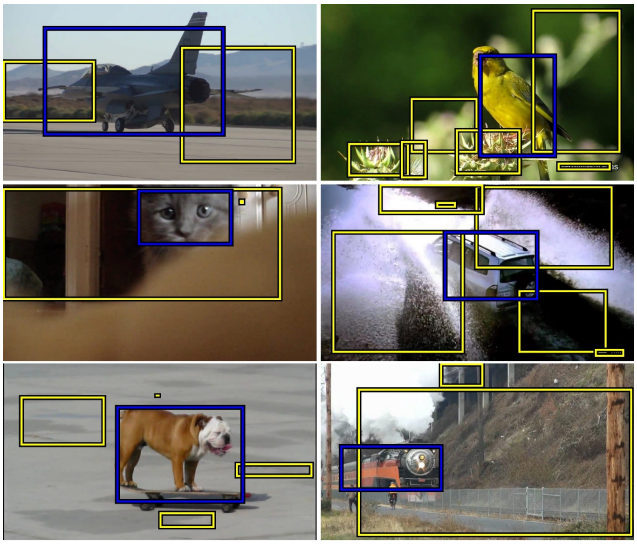


Fig. 1. *Learning from Video.* Yellow boxes represent tubes extracted by our method on the YouTube-Objects dataset. Blue boxes indicate the automatically selected tubes.

## 1. Introduction

Object class detection is a key problem in computer vision. The standard way to train state-of-the-art methods is to gather a large, diverse set of images and annotate them manually, possibly supported by crowd sourcing platforms such as Mechanical Turk [1]. The typical level of annotation needed is a bounding-box for each object instance [15, 34, 36]. In general the performance of a detector increases with the number of annotated instances [35]. Because manual annotation can be inflexible, expensive and tedious, recent work investigated methods that can learn from unlabeled or weakly annotated data [3, 6, 9, 23, 37]. However, learning a detector without location annotation is very difficult and performance is still below fully supervised methods [13, 29].

In this paper, we leave the common path of learning from images and instead exploit *video* as a source of training data. Interestingly, with a few exceptions [2, 24, 30], learning from videos has been disregarded by the vision community. Yet, video offers a rich source of data and is becoming more easily accessible through internet sources such as YouTube. The benefits of video include: (i) it is easier to automatically segment the object from the background based on motion information, (ii) each video shows significant appearances variations of an object, and (iii) a set of videos provides a large number of training images, as each video consists of many frames.

The main contribution of this paper is an approach that learns high quality object detectors from real-world weakly annotated videos. We propose a fully automatic processing pipeline that localizes objects of a target class in a set of training videos (cf. figure 1) and learns a class-specific detector. Our approach requires only one label per video, i.e., whether it contains the class or not. It does not use any other information, such as the number or location of objects. In fact, the method does not even assume that all frames in the video contain the target class. To demonstrate the

[1] www.amazon.com/mturk

technique, we collect a video dataset from YouTube, coined *YouTube-Objects*.

Although we focus on learning from videos, we want to produce a detector capable of detecting objects in images at test time, such as the PASCAL07 [14]. However, individual frames extracted from real-world videos are often of lower quality than images taken by a high-quality camera. Video frames typically suffer from compression artifacts, motion blur, low color contrast, and lower signal-to-noise ratio of the sensor. This makes video frames somewhat different to images at the signal level. Hence, we cast the learning of detectors from videos as a *domain adaptation* task, i.e., learning while shifting the domain from videos to images. As it turns out, this is crucial for learning an effective detector from a combination of images and videos simultaneously.

Experiments on our YouTube-Objects dataset demonstrate that our technique can automatically localize target objects in videos and learn object detectors for several classes. As test data we use the challenging PASCAL07 object detection data set, and show that (i) detectors trained *only* from video already yield reasonable performance; (ii) detectors trained jointly from both video and images using domain adaptation perform better than when training from only the images (i.e. the training images of PASCAL07). In practice, our augmented detector outperforms the popular detector of [15] on several classes.

In the next section, we review related work. In sec. 3 we introduce our technique for localizing objects in videos, and in sec. 4 we explain how to learn an object detector from them. In sec. 5 we state the task of improving object detectors for images by using video data as a domain adaptation problem and present an effective solution. In the experimental sec. 6 we evaluate our approach using PASCAL07 as test set.

## 2. Related Work

**Learning from video.** Most existing works on object detection train from images, e.g. [15, 36, 34]. There is only a limited amount of work on learning object detectors from videos. In one of the earliest works, Ramanan et al. [30] showed how to build part-based animal models for tracking and detection without explicit supervisory information. The tracking is based on a simple hidden Markov model and the detector is a pictorial structure based on a 2D kinematic chain of rectangular segments. The model allows to detect new instances of the animal. Ommer et al. [27] learn detection models as 3D point clouds using structure-from-motion. They train from controlled, hand-recorded video and the model can detect objects in test video, but not in images. Leistner et al. [24] train a part-based random forest object detector from images and use patches extracted from videos to regularize the learning of the trees. Their approach captures the appearance variation of local patches from video and is tested on rather simple benchmarks.

Also related to our approach are works on tracking-by-detection [17, 19]. Typically, a target object is marked by hand in one frame, or initialized with a preexisting detector, then a classifier is trained on-line in order to redetect the object in each frame. These approaches continuously adapt a detector specific to one video and do not attempt to train a generic class detector. In contrast, Ali et al. [2] proposed a semi-supervised boosting variant that uses space-time coherence of video frames to determine the similarity among objects used for training a detector. The method requires a subset of fully annotated frames in each training video. Testing is performed on videos of the same scene, but at different time instances.

Our technique differs from the above ones in several respects: (i) we localize target objects in multiple training videos fully automatically and, then, use them to train an explicit object detector, e.g. [15]; (ii) our technique can train on videos alone and yet yield reasonable detectors for images; (iii) we operate on realistic video sequences downloaded from YouTube. The difficulty of applying state-of-the art vision algorithms to real-world videos from the web has been recently reported in [38].

**Weakly-supervised learning from images.** There are many works for learning object detectors from images without location annotation. These methods typically try to approximately localize object instances while learning a model of the class [3, 6, 10, 9, 13, 16, 23, 37, 29]. In particular, Deselaers et al. [13] propose a technique to select one window per training image out of a large pool of candidates, so as to maximize the appearance similarity of the selected windows. Our technique of sec. 3.3 can be seen as an extension of [13] to video.

## 3. Localizing objects in real-world videos

We start with an overview of our pipeline, see fig. 2. The input is a collection of realistic videos, all labeled as containing the target class. Each video is typically a collage of heterogeneous footage recorded at different times and places. Sec. 3.1 explains how we partition a video into shots, each corresponding to a different scene. In sec. 3.2 we extract segments of coherent motion from each shot, using the technique of Brox and Malik [8] (fig. 3 top row). We then robustly fit a spatio-temporal bounding-box to each segment, which we call a *tube* (fig. 3 bottom row). There are between 3 and 15 tubes per shot. Typically, there is one tube on the object of interest and several on other objects or the background. Given all tubes over all shots in the input training set, we jointly select one tube per shot by minimizing an energy function which measures the similarity of tubes, their visual homogeneity over time, and how likely they are to contain objects (cf. sec. 3.3). The tubes selected by these criteria are likely to contain instances of the target class (fig. 3 bottom row, blue boxes). The selected tubes are

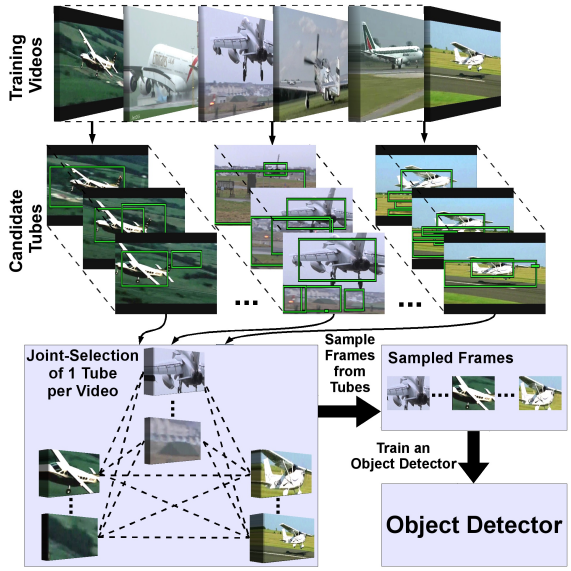



Fig. 2. Overview of our approach.

the output of our localization algorithm. We use them to train a detector for the target class, cf. sec. 4.2.

### 3.1. Temporal partitioning into shots

Consumer videos are often the concatenation of footage recorded at different times and places. This results in abrupt changes of the visual content of the video, called *shot changes*. These are very informative as they usually indicate that a new object or a new scene is filmed, thus breaking motion consistency. We detect shot changes by thresholding color histogram differences in consecutive frames [20]. We operate at a low threshold to ensure all shot changes are found. This partitions each video into multiple shots.

### 3.2. Forming candidate tubes

We extract motion segments from each shot using the recent approach of Brox and Malik [8] which is based on large-displacement optical flow (LDOF). LDOF is a variational technique that integrates discrete point matches, namely the midpoints of regions, into a continuous energy formulation. The energy is optimized by a coarse-to-fine scheme to estimate large displacements even for small scale structures. As opposed to traditional optical flow, this algorithm tracks points over multiple frames, not only over two.

The motion segments are obtained by clustering the dense point tracks based on the similarity in their motion and proximity in location. This works very well for rigid objects, where an entire object is typically put in a single segment. Moreover, in many cases this provides a consistent segmentation even if parts of an object move somewhat differently than the average motion of the whole object (e.g. the jumping cat in fig. 3). This process outputs a set of motion segments, defined by a collection of spatio-temporal point tracks (fig. 3 top row).

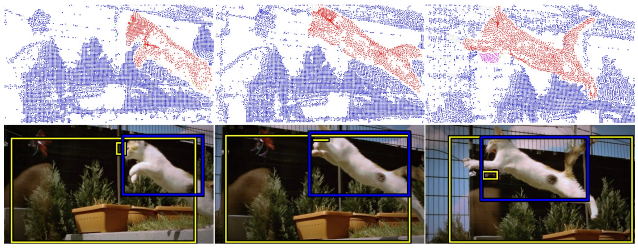


Fig. 3. *Localizing objects in videos.* (Top) Results of the motion segmentation. (Bottom) Tubes fit to the motion segments (the one selected by our approach is in blue).

The last step is to fit a spatio-temporal bounding-box to each motion segment $\mathcal{M}$. At each frame $t$, we want to derive a bounding-box $b^t$ from the set of point-tracks $\mathcal{M}^t$. Simply taking the enclosing box of $\mathcal{M}^t$ would lead to an unreliable estimate, as the outer region of a motion segment often contains spurious point-tracks. To alleviate this problem we select a subset of point tracks $\hat{\mathcal{M}} \subset \mathcal{M}$ which are required to be in the top 80-percentile of the median location of $\mathcal{M}^i$ for all $i \in \{t, \ldots, t+4\}$. The box $b^t$ is then defined as the bounding-box of $\hat{\mathcal{M}}^t$. This method is robust to outlier point tracks in $\mathcal{M}$ and leads to a temporal series of bounding-boxes which vary smoothly over time and are well-anchored on the moving object.

### 3.3. Joint selection of tubes

At this point we have a set of candidate tubes $\mathcal{T}_s = \{\mathcal{T}_s^t\}$ in each shot $s$. Typically one tube contains the object of interest and the remaining ones background and other objects (fig. 3 bottom row). We describe in the following how to select the tubes $l_s \in \{1, ..., |\mathcal{T}_s|\}$ most overlapping with the object in each shot. We model this selection problem as minimizing an energy defined jointly over all $N$ shots in all training videos. Formally, we define the energy of a configuration of tubes $L = (l_1, ..., l_N)$ to be

$$E(L|\alpha) = \sum_s \Phi_s(l_s) + \sum_{s,q} \Psi_{s,q}(l_s, l_q) \qquad (1)$$

**The pairwise potential $\Psi$.** It measures the appearance dissimilarity between tubes $l_s, l_q$ in two different shots $s, q$. It encourages selecting tubes that look similar. It is a linear combination of two dissimilarity functions $\Delta$ that compare the appearance of the tubes over several frames according to two image features (details below)

$$\Psi_{s,q}(l_s, l_q) = \alpha_1 \Delta_{s,q}^{\mathrm{BoW}}(l_s, l_q) + \alpha_2 \Delta_{s,q}^{\mathrm{Phog}}(l_s, l_q) \qquad (2)$$

The pairwise terms connect all pairs of shots. Every tube in a shot is compared to every tube in all other shots (also from other videos, fig. 4).

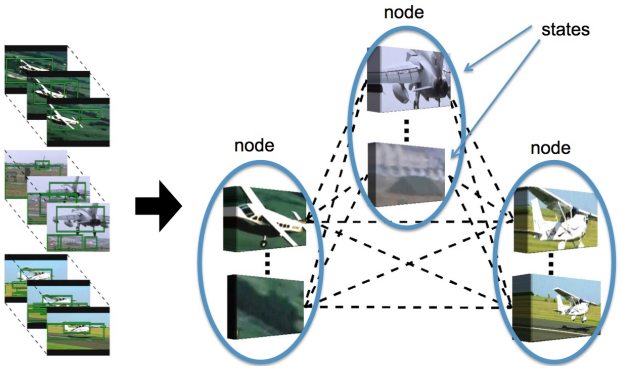



Fig. 4. *Candidate tubes* from different shots are connected through pairwise terms (dashed lines).

**The unary potential $\Phi$.** It defines the cost of selecting tube $l_s$ in shot $s$. It is a linear combination of four terms

$$\begin{aligned}\Phi_s(l_s) = \quad & \alpha_3 \Delta^{\mathrm{BoW}}_{s,s}(l_s, l_s) + \quad \alpha_4 \Delta^{\mathrm{Phog}}_{s,s}(l_s, l_s) \\ + \quad & \alpha_5 \Gamma_s(l_s) + \quad \alpha_6 \Omega_s(l_s)\end{aligned} \tag{3}$$

The first two terms prefer tubes which are visually homogeneous over time. They penalize tubes fit to incorrect motion segments, which typically start on an object but then drift to the background. For measuring homogeneity we use the same $\Delta$ functions as above, but we compare some frames of $l_s$ to other frames of $l_s$.

The $\Gamma$ term is the percentage of the bounding-box perimeter touching the border of the image, averaged over all frames in the tube. It penalizes tubes with high contact with the image border, which typically contain background (e.g., the blue segment in fig. 3 top row).

The $\Omega$ term is the objectness probability [1] of the bounding-box, averaged over all frames in the tube. It measures how likely a box is to contain an object of any class, rather than background. It distinguishes objects with a well-defined boundary and center, such as cows and telephones, from amorphous background windows, such as grass and road. For this it measures various characteristics of objects in general, such as appearing different from their surroundings and having a closed boundary (see [1] for more details).

**Minimization.** The objective of tube selection is to find the configuration $L^*$ of tubes that minimizes $E$. We perform this minimization using the TRW-S algorithm [21], which delivers a very good approximation of the global optimum $L^* = \arg\min_{\mathrm{L}} E(L|\Theta)$ in our fully connected model. TRW-S also returns a lower bound on the energy. When this coincides with the returned solution, we know it found the global optimum. In our experiments, the lower bound is only $0.05\%$ smaller on average than the returned energy. Thus, we know that the obtained configurations are very close to the global optimum. The tubes selected by $L^*$ are the output of our localization algorithm. They are used as input to train a detector for the target class in sec. 4.

Our technique is related to [13], where an energy function was minimized to select one window per image. We extended this idea to video, redefining the problem to select one tube per shot, and introducing potentials relevant for spatio-temporal data.

**Comparing tube appearance.** To compute the $\Delta$ appearance dissimilarity function, a subset of boxes within a tube is represented by two complementary visual features. (1) $\mathrm{BoW}$, a bag-of-word of dense SURF [5] features quantized to a visual vocabulary of 500 words, learned from 200 random frames. We use a 3-level spatial pyramid to enforce spatial consistency [22]. (2) $\mathrm{Phog}$, PHOG features [7] capturing local shape as a distribution of HOG-features [11] organized in a spatial pyramid [22]. For each tube, we compute $\mathrm{BoW}$ and $\mathrm{Phog}$ for the bounding-boxes in 5 frames sampled uniformly over the temporal extent of the tube. The function $\Delta^{\mathrm{f}}_{s,q}(l_s, l_q)$ is the median of the $\chi^2$ dissimilarity of the descriptors $\mathrm{f}$ over all 25 pairs of frames (one frame from tube $l_s$ in shot $s$ and the other frame from tube $l_q$ in shot $q$).

**Weights $\alpha$.** The scalars $\alpha$ weight the terms of our energy model. We learn the optimal $\alpha$ using constraint generation [33] on a separate set of 50 held out shots of cars. We manually annotated the location of the car in one frame of each shot. The constraint generation algorithm efficiently finds the weights that maximize the localization performance wrt the ground-truth annotations. The $\alpha$ learned from this small held-out dataset is then used for all classes in our experiments.

## 4. Learning a detector from the selected tubes

The previous section selects one tube per shot likely to contain an instance of the target class. This corresponds to automatically localizing a bounding-box in each frame of the shot covered by the tube. We now describe how to train an object detector from this data. The main technical issue is sampling high quality bounding-boxes from the large pool offered by all selected tubes (sec. 4.1). The sampled bounding-boxes can then be used to train any standard object detector which requires bounding-boxes for training, e.g., [15, 18, 34] (sec. 4.2).

### 4.1. Sampling positive bounding-boxes

The tubes selected in sec. 3.3 offer a very large number of bounding-boxes that could be used as positive samples for training a detector. In our YouTube-Objects dataset, this number is about 10k-70k, depending on the class (tab. 1). This is too much data to handle for the training procedures of most modern detectors, which input about 1k positive samples [15, 18]. However, not all of these bounding-boxes contain the object of interest. Even with perfect tube selection, the best available tube might contain bounding-boxes covering other image elements. This happens when tubes mostly on the object start or end on something else, e.g.,

when the object moves out of the field of view, or when the underlying motion segment drifts to the background. Moreover, some shots might not even contain the target class as they are automatically collected from YouTube (recall we only assume annotation at the video level, not at the shot level). Using such bounding-boxes as positive samples can confuse the training of the detector.

We introduce here a sampling technique to (i) reduce the number of positive samples to a manageable quantity; (ii) select samples more likely to contain relevant objects. The first step is to quantify the quality of each bounding-box in the pool. For this we use a linear combination of its objectness probability and the percentage of perimeter touching the border of the image, exactly as the $\Omega$ and $\Gamma$ terms in eq. (3) (but applied to a single frame). The second step is to sample a fixed number $S$ of bounding-boxes according to this quality measure (treating the set of probabilities for all samples as a multinomial distribution). In all our experiments we use $S = 500$.

### 4.2. Training the object detector

The bounding-boxes sampled as described in the previous subsection can be used to train any object detector. As negative training set we randomly sample 2400 video frames from other classes. From this data we train two popular and complementary detectors:

**DPM:** the part-based model [2] of Felzenszwalb et al. [15]. It consists of a root filter and deformable part filters. The method has been demonstrated to yield highly competitive results on the PASCAL07 [14] dataset.

**SPM:** We employ SURF [5] features quantized into a 500-entries codebook learned on 500 frames randomly sampled from the training videos. A bounding-box is described by a 3-level spatial pyramid [22]. At training time we collect an initial set of negative samples by sampling 10 objectness [1] windows in every negative training image. We, then, train a preliminary SVM classifier using Intersection Kernel [25], and search exhaustively for false positives in the negative images [11]. The classifier is retrained using these additional hard negatives. At test time the detector operates on 100000 windows uniformly sampled for every test image, followed by non-maxima suppression.

## 5. Domain adaptation: from videos to images

Training a detector on videos and applying it to images corresponds to having training and test datasets from different domains. Recently, [32] highlighted the problems of visual classifiers if training and test set differ. [38] showed that this is especially valid for videos from the web that suffer from compression artifacts, low resolution, motion blur and low color contrast. Thus, it is hard to train a detector from video that yields good results on images.

[2] www.cs.brown.edu/ pff/latent/

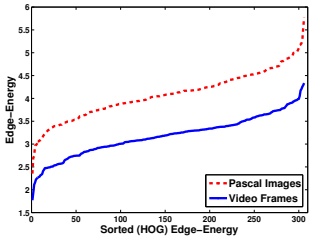



Fig. 5. **Images vs videos**. Left: the sum of the magnitude of the HOG features in an object bounding-box, normalized by its size (computed using the implementation of [15]). The 300 samples are sorted by gradient energy along the x-axis.

To illustrate that video and still images can be seen as two different domains, we conducted two experiments. The first is illustrated in fig. 5 and compares the HOG representation for 300 aeroplanes on frames from our YouTube-Objects dataset to 300 similar aeroplane images from PASCAL07 [14]. We can observe that the gradient energy of images is significantly larger, i.e., around one third, than that of video frames. In the second experiment, we follow the *Name that Dataset* protocol of [32]. We trained an SVM on GIST features [26] to distinguish video from images. This achieves a classification accuracy of 83%, confirming that images and videos are indeed different domains. Another difference between images and videos is the different distribution of viewpoints in which an object typically appears.

### 5.1. Domain adaptation

Domain Adaptation (DA) approaches try to improve classification accuracy in scenarios where training and test distributions differ. Let $\mathcal{X} = \mathcal{R}^F$ be the input space with $F$ being the number of feature dimensions, and let $\mathcal{Y} = \{-1, 1\}$ be the output space (e.g. aeroplane or not). In DA, we have access to $N$ samples from the source domain $\mathcal{X}^s$ (e.g. video) together with their labels $\mathcal{Y}^s$, and $M$ samples from the target domain $\mathcal{X}^t$ (e.g. images) along with their labels $\mathcal{Y}^t$. Usually, $N \gg M$. The task of domain adaptation is to train a function $f : \mathcal{X} \rightarrow \mathcal{Y}$ that performs well on the target domain. Note that the case where the performance decreases for the target domain is referred to as *negative transfer* [28]. There exists a number of approaches for domain adaptation [28]. For our application we explore a few simple methods that have been reported to perform surprisingly well [12]. We summarize below the three approaches we experiment with, following the nomenclature and notation of [12].

**All.** The simplest possible approach is to ignore the domain shift and directly train a single classifier using the union of all available training data $\{(\mathbf{x}_i^s, y_i^s)\}_{i=1}^N$ and $\{(\mathbf{x}_j^t, y_j^t)\}_{j=1}^M$. This is a simple baseline that should be beaten by any real domain adaptation technique.

**Pred.** A popular approach is to use the output of the source classifier as an additional feature for training the target classifier. More precisely, we first train a classifier $f_s(\mathbf{x})$ on the source data $\{(\mathbf{x}_i^s, y_i^s)\}_{i=1}^N$. We, then, expand the feature vector of each target sample $\mathbf{x}^t$ to $[\mathbf{x}^t \;\; f_s(\mathbf{x}^t)]$. Finally, we

train the target classifier using the expanded target training data $\{\left(\left[\mathbf{x}_j^t \;\; f_s(\mathbf{x}_j^t)\right], y_j^t\right)\}_{j=1}^M$.

**Prior.** One of the most popular DA methods in Computer Vision is *Prior*, where the parameters of the source classifier are used as a prior when learning the target classifier [31, 4]. For instance, an SVM can be regularized in form of $||\mathbf{w}_s - \mathbf{w}_t||^2$, where $\mathbf{w}$ is a learned weight vector.

**LinInt.** Another technique is to first train two separate classifiers $f_s(\mathbf{x}), f_t(\mathbf{x})$ from the source and the target data, and then linearly interpolate their predictions on new target data at test time. Thus, *LinInt* forms a new classifier $f_{st}(\mathbf{x}) = \lambda f_s(\mathbf{x}) + (1-\lambda) f_t(\mathbf{x})$, where $\lambda$ is the interpolation weight (it can be set so as to minimize the expected loss on the target domain).

Beside these four approaches, there exist many others. One advantage of Pred and LinInt over other techniques is that they can combine heterogeneous classifiers, where the dimensionality or even the kind of features differ between the source and target domains.

### 5.2. LinInt for object detection

In the context of sliding-window object detection, as opposed to classification, *LinInt* cannot directly be applied. Different detectors might not only operate on different kinds of features, but even on different sets of windows in an image. For example, most detectors [15, 29] learn an optimal aspect-ratio from the training data and only score windows of that aspect-ratio on a test image. In our case, the aspect-ratio learned from video data might differ from that learned from images. Other differences might include the sliding-window step and the sampling of the scale-space.

We introduce here a technique for combining arbitrary heterogeneous sliding-window detectors based on *LinInt*. We first let each detector score its own set of windows $\mathcal{D}$ on a test image $I$. Each window is represented as a 5-D vector $\mathbf{d} = \{x_1, y_1, x_2, y_2, s\}$ composed of the coordinates of the window in the image and its score. The two detectors, each trained separately, produce two separate sets of windows $\mathcal{D}_1$ and $\mathcal{D}_2$ for the same image $I$. We combine them into a set of windows $\mathcal{D}_c$ with the following algorithm. First, we initialize $\mathcal{D}_c = \emptyset$. Then, for each window $\mathbf{d_1} \in \mathcal{D}_1$ we do

1. Find the most overlapping window in $\mathcal{D}_2$: $\mathbf{d_2} = \arg\max_{\mathbf{d_2} \in \mathcal{D}_2} \mathrm{IoU}(\mathbf{d_1}, \mathbf{d_2})$ with $\mathrm{IoU}(\mathbf{d_1}, \mathbf{d_2}) = \frac{|d_1 \cap d_2|}{|d_1 \cup d_2|}$ the spatial overlap of two windows.
2. Combine the scores of $\mathbf{d_1}, \mathbf{d_2}$ with a modified *LinInt*: $s = \lambda \cdot d_1^s + (1-\lambda) \cdot \mathrm{IoU}(\mathbf{d_1}, \mathbf{d_2}) \cdot d_2^s$ where $\lambda \in [0, 1]$ weights the two detectors.
3. Add a new window to $\mathcal{D}_c$ with the coordinates of $\mathbf{d_1}$ but with score $s$.

This procedure is flexible as it can combine detectors defined on arbitrary sets of windows. It matches windows between the two sets based on their overlap and combines scores of matched pairs of windows. In practice two matched windows often have very high overlap and are almost identical. However, in the case a window from $\mathcal{D}_1$ has no good match in $\mathcal{D}_2$, our technique automatically reduces the combination weight.

| class | videos | shots | frames | class | videos | shots | frames |
|---|---|---|---|---|---|---|---|
| aero | 13 | 1097 | 71327 | cow | 11 | 212 | 29642 |
| bird | 16 | 205 | 27532 | dog | 24 | 982 | 82432 |
| boat | 17 | 606 | 74501 | horse | 15 | 432 | 70247 |
| car | 9 | 208 | 14129 | mbike | 14 | 511 | 40604 |
| cat | 21 | 220 | 42785 | train | 15 | 1034 | 117890 |

Table 1. Statistics of our YouTube-Objects dataset. In total the dataset is composed of 155 videos, divided into 5507 shots and amounting to 571089 frames.

## 6. Experiments

### 6.1. Dataset

Our YouTube-Objects dataset is composed of videos collected from YouTube. For each of 10 object classes, we collected between 9 and 24 videos, whose duration varies between 30 seconds and 3 minutes (tab. 1). The videos are weakly annotated, i.e. we only ensure that at least one object of the relevant class is present in each video. For evaluating our automatic localization technique, we annotated bounding-boxes on a few frames containing the object of interest. For each class we annotated one frame per shot on $100-290$ different shots. Importantly, these annotations are used exclusively to evaluate our technique (sec. 6.2). They are not input at any point in our fully automatic pipeline.

### 6.2. Localizing objects in the training videos

We evaluate here the quality of object localization in the training videos. We adopt the CorLoc performance measure used in [13, 29], i.e., the percentage of ground-truth bounding boxes which are correctly localized up to the PASCAL criterion (intersection-over-union $>= 0.50$). We evaluate separately the quality of (i) the tube extraction process (sec. 3.2) and of (ii) the tube selection process (sec. 3.3). To evaluate the tube extraction process, we select the tube with the maximum overlap with the ground-truth bounding box. The CorLoc of this best available tube can be seen as an upper-bound on the CorLoc that can be achieved by the automatic tube selection process.

The results are shown in tab. 2. Our automatic tube selection technique is very effective. In 7 classes out of 10 it performs close to the upper-bound, indicating it selects the best available tube most of the time. Cat, dog and train present higher intra-class variability which makes it harder to leverage on recurrent appearance patterns to select tubes covering the target objects. The performance of tube extraction varies substantially from class to class and is in general higher for rigid objects. The overall performance is satisfactory, but could be improved as it is currently missing many objects. The main reason for this is the difficulty of motion

| | aero | bird | boat | car | cat | cow | dog | horse | mbike | train | AVG |
|---|---|---|---|---|---|---|---|---|---|---|---|
| Tube extraction | 53.9 | 19.6 | 38.2 | 37.8 | 32.2 | 21.8 | 27.0 | 34.7 | 45.4 | 37.5 | 34.8 |
| Tube selection | 51.7 | 17.5 | 34.4 | 34.7 | 22.3 | 17.9 | 13.5 | 26.7 | 41.2 | 25.0 | 28.5 |

Table 2. Evaluation of the object localization performance of our method, as the percentage of correctly localized objects for (i) the best available extracted tube (first row) and (ii) the automatically selected tubes (second row).

| | aero | boat | horse | mbike | train | mAP |
|---|---|---|---|---|---|---|
| DPM-VID | 17.4 | 9.2 | 16.2 | 27.3 | 15.0 | 17.0 |
| [29] | 11.5 | 3.0 | 20.3 | 9.1 | 13.2 | 11.4 |

Table 4. Comparison of our approach to weakly supervised learning from images [29].

segmentation in low quality, uncontrolled YouTube videos.

### 6.3. Training from video

Here we test on the 4952 images from the PASCAL07 test set using the official protocol [14]. This is a very challenging dataset, with large variations in pose, appearance and scale. We report results on a subset of 10 classes, corresponding to objects that move and for which enough video data is available online. Tab. 3 presents results for three different training regimes:

**VOC:** models trained on manual bounding-box annotations from the PASCAL07 Train + Val image set.

**VMA:** models trained on the manually annotated frames from YouTube-Objects (sec. 6.1).

**VID:** models trained from YouTube-Objects on the output of our fully automatic pipeline (sec. 3 and 4).

Tab. 3 reports performance for the SPM as well as DPM detectors. The VOC training regime serves as a high quality reference. First we observe that the VID model performs close to the manually annotated VMA, confirming the quality of our automatic learning approach. Moreover, compared to the VOC models, the VID models offer reasonable performance, especially considering they take no manual intervention to train.

Interestingly, the results confirm the domain shift discussed in sec. 5. There is a significant performance gap between VOC and VMA, although they are trained with a similar number of samples and level of supervision. In sec. 6.5 we will improve over these results by training jointly from both images and video with domain adaptation.

### 6.4. Comparison to WS learning from images [29]

We compare our approach to a recent weakly supervised method for learning from images [29]. It learns each model from all PASCAL07 training images corresponding to a class, given also its ground-truth average aspect-ratio. This information is required as their approach iteratively refines the DPM detector and does not converge if initialized with a square window. This is a little more supervision than used normally in weakly supervised learning, where only class labels for the image/video are given (as is our case).

Results [29] are presented for a subset of 14 classes of PASCAL07. We obtained the learned DPM detectors [15] from the authors and evaluated them on the test set. Tab. 4 presents results on the 5 classes that have been evaluated in both our work and theirs. Our models learned on videos outperform their models learned from images. We believe this result validates our approach as an effective alternative to weakly supervised learning from images.

### 6.5. Training from video and images

Here we train detectors from a combination of weakly supervised video data *and* PASCAL07 fully supervised images using domain adaptation. Tab. 5 presents results for various adaptation methods (sec. 5). We report differential results wrt learning only on the target domain (still images, VOC rows in tab. 3). We train the $\lambda$ parameter of LinInt by maximizing performance on the Val set of PASCAL07.

The results show that the All method, which combines training data at the earliest stage, degrades performance (negative transfer [28]). The problem is only partially alleviated when the combination happens at the feature level (Pred) or the parameter level (Prior). The LinInt method instead is immune to negative transfer and prevents a weaker model to harm the combined performance (by automatically setting $\lambda$ to 0). Moreover, for DPM in 5 out of 10 classes LinInt improves over VOC, demonstrating that knowledge can be transferred from the video to the image domain, leading to a *better detector* than one trained from images alone ($+2.0\%$ AP on average on the 5 classes where $\lambda > 0$).

## 7. Conclusions

We introduced a technique for learning object detectors from real-world web videos. Furthermore, we formulated the problem of learning from both videos and still images jointly as a domain adaptation task. On PASCAL07 this resulted in increased performance compared to training from images alone.

The YouTube-Objects dataset is available for download on our project page [3] together with additional videos showing intermediate processing stages of our approach.

**Acknowledgements.** This work was partially funded by the QUAERO project supported by OSEO, French State agency for innovation, the European integrated projects AXES and RoboEarth, DPI2009-07130, SNSF IZK0Z2-136096, CAI-DGA IT 26/10 and a Google Research Award.

[3] http://groups.inf.ed.ac.uk/calvin/learnfromvideo

| | | aero | bird | boat | car | cat | cow | dog | horse | mbike | train | mAP |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| SPM | VOC | 22.5 | 10.0 | 10.4 | 27.4 | 22.1 | 10.8 | 13.4 | 21.1 | 25.8 | 27.3 | 19.1 |
| | VMA | 16.8 | 3.9 | 2.6 | 18.7 | 16.5 | 10.1 | 6.0 | 8.5 | 15.9 | 12.0 | 11.1 |
| | VID | 14.0 | 9.4 | 0.6 | 15.0 | 14.9 | 9.4 | 7.5 | 11.9 | 14.6 | 5.6 | 10.3 |
| DPM | VOC | 29.6 | 10.1 | 17.1 | 55.0 | 18.4 | 24.7 | 11.3 | 57.7 | 47.8 | 44.5 | 31.6 |
| | VMA | 23.6 | 9.9 | 8.4 | 40.4 | 2.0 | 18.6 | 9.7 | 28.3 | 29.5 | 15.7 | 18.6 |
| | VID | 17.4 | 9.3 | 9.2 | 35.7 | 9.4 | 9.7 | 3.3 | 16.2 | 27.3 | 15.0 | 15.2 |

Table 3. Evaluation of different detectors and different training regimes on the PASCAL07 test set, in Average Precision computed as the area under the precision-recall curve. The mAP column reports the mean AP over all classes.

| | | aero | bird | boat | car | cat | cow | dog | horse | mbike | train |
|---|---|---|---|---|---|---|---|---|---|---|---|
| SPM | All(VOC,VID) | 1.7 | −4.7 | −8.4 | −1.7 | 1.5 | −2.2 | 0.2 | −2.1 | −2.2 | −4.2 |
| | Pred(VOC,VID) | 1.3 | −4.6 | 0.3 | −4.3 | 2.9 | −0.3 | −5.2 | −1.6 | −0.3 | −0.1 |
| | Prior(VOC,VID) | −1.1 | 0.5 | −1.1 | 0.6 | −2.2 | −5.8 | −6.8 | 0.7 | −1.3 | −0.3 |
| | LinInt(VOC,VID) | 2.9 | 0.0 | 0.2 | 0.6 | 0.0 | 0.0 | 0.0 | 0.0 | 1.9 | 0.0 |
| DPM | All(VOC,VID) | 1.1 | −0.4 | −7.5 | −2.7 | −1.0 | −9.2 | −0.7 | −9.0 | −2.6 | −12.2 |
| | LinInt(VOC,VID) | 4.5 | 1.4 | 2.4 | 0.0 | 1.5 | 0.4 | 0.0 | 0.0 | 0.0 | 0.0 |

Table 5. We show relative improvement in Average-Precision wrt VOC models.

# References


[1] B. Alexe, T. Deselaers, and V. Ferrari. What is an object? In *CVPR*, 2010.

[2] K. Ali, D. Hasler, and F. Fleuret. Flowboost - appearance learning from sparsly labeled video. In *CVPR*, 2011.

[3] H. Arora, N. Loeff, D. Forsyth, and N. Ahuja. Unsupervised segmentation of objects using efficient learning. In *CVPR*, 2007.

[4] Y. Aytar and A. Zisserman. Tabula rasa: Model transfer for object category detection. In *ICCV*, 2011.

[5] H. Bay, A. Ess, T. Tuytelaars, and L. van Gool. SURF: Speeded up robust features. *CVIU*, 110(3):346–359, 2008.

[6] M. Blaschko, A. Vedaldi, and A. Zisserman. Simulatenous object detection and ranking with weak supervision. In *NIPS*, 2010.

[7] A. Bosch, A. Zisserman, and X. Munoz. Representing shape with a spatial pyramid kernel. In *CIVR*, 2007.

[8] T. Brox and J. Malik. Object segmentation by long term analysis of point trajectories. In *ECCV*, 2010.

[9] O. Chum and A. Zisserman. An exemplar model for learning object classes. In *CVPR*, 2007.

[10] D. J. Crandall and D. Huttenlocher. Weakly supervised learning of part-based spatial models for visual object recognition. In *ECCV*, 2006.

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

[12] H. Daume III. Frustratingly easy domain adaption. In *ICML*, 2007.

[13] T. Deselaers, B. Alexe, and V. Ferrari. Localizing objects while learning their appearance. In *ECCV*, 2010.

[14] M. Everingham, L. Van Gool, C. Williams, J. Winn, and A. Zisserman. The PASCAL Visual Object Classes Challenge 2007 Results, 2007.

[15] P. F. Felzenszwalb, R. B. Girshick, D. McAllester, and D. Ramanan. Object detection with discriminatively trained part based models. *PAMI*, 2009.

[16] R. Fergus, P. Perona, and A. Zisserman. Object class recognition by unsupervised scale-invariant learning. In *CVPR*, 2003.

[17] H. Grabner and H. Bischof. On-line boosting and vision. In *CVPR*, 2006.

[18] H. Harzallah, F. Jurie, and C. Schmid. Combining efficient object localization and image classification. In *ICCV*, 2009.

[19] Y. Kalal, J. Matas, and K. Mikolajcyzk. P-N learning: Bootstrapping binary classifiers from unlabeled data by structural constraints. In *CVPR*, 2010.

[20] W.-H. Kim and J.-N. Kim. An adaptive shot change detection algorithm using an average of absolute difference histogram within extension sliding window. In *ISCE*, 2009.

[21] V. Kolmogorov. Convergent tree-reweighted message passing for energy minimization. *PAMI*, 28(10):1568–1583, 2006.

[22] S. Lazebnik, C. Schmid, and J. Ponce. Beyond bags of features: Spatial pyramid matching for recognizing natural scene categories. In *CVPR*, 2006.

[23] Y. J. Lee and K. Grauman. Learning the easy things first: Self-paced visual category discovery. In *CVPR*, 2011.

[24] C. Leistner, M. Godec, S. Schulter, A. Saffari, and H. Bischof. Improving classifiers with weakly-related videos. In *CVPR*, 2011.

[25] S. Maji, A. Berg, and J. Malik. Classification using intersection kernel support vector machines is efficient. In *CVPR*, 2008.

[26] A. Oliva and A. Torralba. Modeling the shape of the scene: a holistic representation of the spatial envelope. *IJCV*, 42(3):145–175, 2001.

[27] B. Ommer, T. Mader, and J. M. Buhmann. Seeing the objects behind the dots: Recognition in videos from a moving camera. *IJCV*, 83(1):57–71, 2009.

[28] S. J. Pan and Q. Yang. A survey on transfer learning. *IEEE Trans. on Knowledge and Data Engineering*, 2010.

[29] M. Pandey and S. Lazebnik. Scene recognition and weakly-supervised object localization with deformable part-based models. In *ICCV*, 2011.

[30] D. Ramanan, D. A. Forsyth, and K. Barnard. Building models of animals from video. *PAMI*, 2006.

[31] T. Tommasi, F. Orabona, and B. Caputo. Safety in numbers: Learning categories from few examples with multi model knowledge transfer. In *CVPR*, 2010.

[32] A. Torralba and A. Efros. An unbiased look on dataset bias. In *CVPR*, 2011.

[33] I. Tsochantaridis, T. Joachims, T. Hofmann, and Y. Altun. Large margin methods for structured and interdependent output variables. *JMLR*, 6:1453–1484, 2005.

[34] A. Vedaldi, V. Gulshan, M. Varma, and A. Zisserman. Multiple kernels for object detection. In *ICCV*, 2009.

[35] S. Vijayanarasimhan and K. Grauman. Large-scale live active learning: Training object detectors with crawled data and crowds. In *CVPR*, 2011.

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

[37] J. Winn and N. Jojic. LOCUS: learning object classes with unsupervised segmentation. In *ICCV*, 2005.

[38] S. Zanetti, L. Zelnik-Manor, and P. Perona. A walk through the web's video clips. In *CVPRW*, 2008.