Joint learning of object and action detectors

Vicky Kalogeiton[1,2] Philippe Weinzaepfel[3] Vittorio Ferrari[2] Cordelia Schmid[1]

Abstract

While most existing approaches for detection in videos focus on objects or *human actions separately, we aim at jointly detecting objects performing actions, such as cat eating or dog jumping. We introduce an end-to-end multitask objective that jointly learns object-action relationships. We compare it with different training objectives, validate its effectiveness for detecting objects-actions in videos, and show that both tasks of object and action detection benefit from this joint learning. Moreover, the proposed architecture can be used for zero-shot learning of actions: our multitask objective leverages the commonalities of an action performed by different objects,* e.g. *dog and cat jumping, enabling to detect actions of an object without training with these object-actions pairs. In experiments on the A2D dataset [50], we obtain state-of-the-art results on segmentation of object-action pairs. We finally apply our multitask architecture to detect visual relationships between objects in images of the VRD dataset [24].*

1. Introduction

Video understanding has received increased attention over the past decade leading to significant advances [39, 43]. However, most existing approaches focus either on object recognition [14, 34] *or* on human action recognition [29, 48] separately. For both tasks, the community has moved from small datasets [35] to large ones with thousands of videos and hundreds of classes [1, 12], from controlled environments [38] to videos in-the-wild [15]. Given the impressive success of Convolutional Neural Networks (CNNs) for object detection [23, 34], action localization has benefited as well from this improvement. In particular, Faster R-CNN [34] has been enhanced for videos by using a two-stream variant [9, 29, 48], in which both appearance and motion are used as inputs. Modern approaches first use such a detector to localize human actions in individual frames, and then either link them or track them over time to create spatio-temporal detections [9, 29, 48]. These methods focus exclusively on human action recognition.

While humans or actions alone are building blocks of video understanding, the relationship between objects and actions can yield a more complete interpretation. For instance, an autonomous car should not only be able to detect another *car* (object) or a *human walking* (action), but also a *dog running* or a *ball flying* (object-action). Other applications include content-based retrieval, video captioning [43, 52] and health-care robots, for instance helping blind people crossing streets. Therefore, to better understand videos, we need to go beyond these two independent tasks of object and human action recognition and understand the relationship between objects and actions.

In this paper, we propose to jointly detect object-action instances in uncontrolled videos, *e.g. cat eating*, *dog running* or *car rolling*, see Figure 1. We build an end-to-end two stream network architecture for joint learning of objects and actions. We cast this joint learning problem by leveraging a multitask objective. We compare our proposed end-to-end multitask architecture with alternative ones (Figure 3): (i) treating every possible combination of actions and objects as a separate class (Cartesian) and (ii) considering a hierarchy of objects-actions: the first level corresponds to objects and the second one to the valid actions for each object (hierarchical). We show that our method performs as well as these two alternatives while (a) requiring fewer parameters and (b) enabling zero-shot learning of the actions performed by a specific object, *i.e.*, when training for an ob-

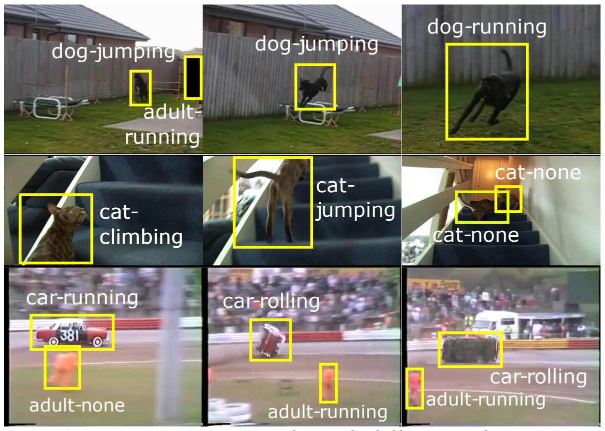

Figure 1: *Detection examples of different object-action pairs for the videos of the A2D dataset [50].*

[1]Univ. Grenoble Alpes, Inria, CNRS, Grenoble INP, LJK, 38000 Grenoble, France
[2]University of Edinburgh
[3]Naver Labs Europe

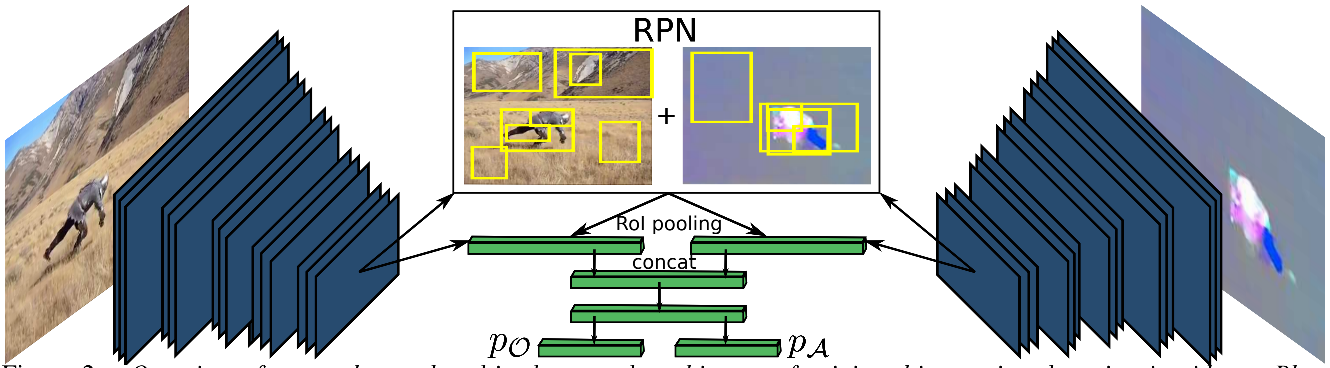

Figure 2: *Overview of our end-to-end multitask network architecture for joint object-action detection in videos. Blue color represents convolutional layers while green represents fully connected layers. The end-to-end training is done by concatenating the fully connected layers from both streams. Here, $p_{\mathcal{O}}$ and $p_{\mathcal{A}}$ are the outputs of the two branches that predict the object and action labels, resulting in the loss described in Equation 2.*

ject class alone without its actions, our multitask network is able to predict actions for that object class by leveraging actions performed by other objects.

Interestingly, our multitask objective not only allows to effectively detect object-action pairs but also leads to performance improvements on each individual task (*i.e.*, detection of either objects *or* actions). This is because the features learned for one task help learning the other one. We compare to the state of the art for object-action detection on the Actor-Action (A2D) dataset [50] that contains segmentation annotation for object-action pairs. For a direct comparison we transform our detections into pixelwise segmentation maps by using segmentation proposals [10, 30]. Our approach significantly outperforms the state of the art [50, 49] on this dataset. We finally apply our multitask objective to detect object-action relationships in images on the Visual Relationship Detection (VRD) dataset [24].

In summary, we make the following contributions:

• We propose an end-to-end *multitask* architecture for joint object-action detection.

• We show that this multitask objective can be leveraged for zero-shot learning of actions.

• We demonstrate the generalization of our multitask architecture by applying it to (a) object-action semantic segmentation and (b) object-action relationships in images.

2. Related Work

Most existing approaches for detection in videos focus either on object *or* on action localization. Over the past few years, the methods range from low-level features [16, 20, 26, 32, 42, 45, 46], structured models that mine mid-level elements [19, 25] to parts [7, 28, 33] and attributes [22]. However, CNNs currently constitute the dominant approach for large-scale and high-quality video detection.

Object or action detection. Recent work on object detection [8, 14, 34] has shown remarkable progress, mainly due to the use of CNNs [8, 17, 23]. R-CNN [8] tackles object detection with CNNs by casting the task as a region-proposals classification problem. Faster R-CNN [34] goes a step further and generates proposals using a Region Proposal Network (RPN), which shares convolutional features with the proposal classification branch.

These per-frame detectors are also used by state-of-the-art human action localization methods [29, 37] to obtain spatial information; then the detections are linked across time resulting in video-level localizations [9, 48]. To leverage video data, the detector operates on two streams [39]: RGB and optical flow. The two streams are trained separately and the scores are averaged at test time [9, 29, 48], *i.e.*, late fusion of scores. In contrast, our architecture is a two-stream Faster R-CNN trained end-to-end based on a fusion by a fully-connected layer that operates on concatenated features from both streams. Moreover, it is trained with a multitask objective that allows us to detect objects *and* actions jointly.

Joint modeling of objects and actions. Joint modeling of objects and actions in videos has received little attention so far. For the action localization task, some works [11, 31] propose to model the interactions of humans and objects. However, the task we tackle in this paper is significantly different as objects are not used for the actions, but they are the actors. Bojanowski *et al.* [2] have considered the case in which different entities can perform a set of actions, but these entities correspond to names of different actors, *i.e.*, to person identification. Closely related to object-action detection in videos is the work [49, 50] on segmenting object-action pairs. They use Conditional Random Fields at the supervoxel level to output a semantic segmentation at the pixel level. We show that our detections based on a multitask objective also improve the semantic segmentation performance by leveraging segmentation proposals [10, 30].

In images, however, object-action pairs have been modeled implicitly in the context of predicting sentences for images [27, 44] and more recently by visual phrases [36] and

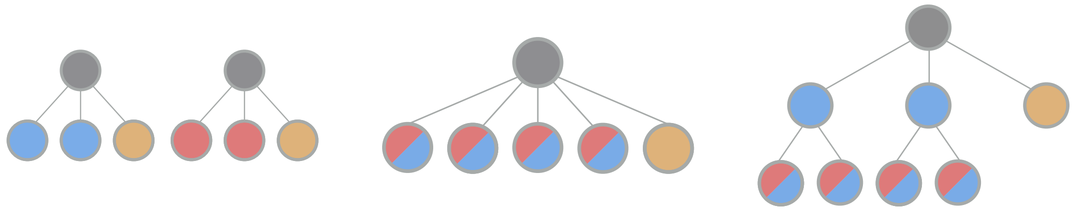
(a) Multitask (b) Cartesian (c) Hierarchical

Figure 3: *Illustration of the three different ways we consider for jointly learning objects and actions. The blue nodes represent objects and the red ones action classes, while the yellow ones represent the background class.*

relationships between objects [24]. The task consists in detecting triplets of two objects and their relationship [24, 36]. Most approaches rely on object detectors. The relationship label is predicted from the bounding box around the two objects, and sometimes from additional modalities such as languages or frequency priors in the training set. We show that our multitask objective allows to predict the relationships between objects without (a) the need to see the whole bounding box and (b) the need to include any priors. In particular, we transform each triplet into two pairs, each consisting of one of the two objects and the interaction. Then, we train our network to detect bounding boxes around objects and also predict an interaction label.

Zero-shot learning. Most existing approaches for zero-shot learning of categories rely on attributes [4, 5, 18]. Attributes have also been used for human actions [22, 51]. Liu *et al.* [22] were the first to represent actions by sets of attributes. They consider that each action class has an intra-class variability, which they try to model by searching which attributes are relevant for each class. They apply zero-shot learning by manually labeling attributes for all classes, including new ones without visual examples. In contrast, our approach does not require any attribute labels.

3. End-to-end multitask network architecture for joint learning of objects and actions

Given a video, we aim to detect the objects as well as the actions they are performing. Let $\mathcal{O}$ (resp. $\mathcal{A}$) be the set of objects (resp. actions) labels. Some combinations of actions and objects may not be valid, *e.g. car eating*. We denote by $\mathcal{V} \subset \mathcal{O} \times \mathcal{A}$ the set of valid object-action combinations.

3.1. End-to-end network architecture

We build an end-to-end two-stream multitask network that proceeds at the frame level (Figure 2). As most state-of-the-art methods for object and action detection in videos, we rely on Faster R-CNN [34] and its two-stream variant [9, 37, 39]. However, instead of training each stream separately, we propose to fuse both streams, thus enabling effective end-to-end learning. Our end-to-end network has two streams: (a) appearance, which takes as input the RGB data and (b) motion, which operates on the optical flow [3]. Following [9], the input of the motion stream is a tensor of three channels with the x and y coordinates of the flow and its magnitude, represented as a 3-channel image. A Region Proposal Network (RPN) extracts candidate bounding boxes independently for each stream. We use the set union of the two RPNs and we aggregate features for each candidate box with a Region-of-Interest (RoI) pooling layer in each stream. After one fully-connected layer, the two streams are concatenated and fed to another fully-connected layer. The remaining network layers operate on the fused stream, enabling end-to-end training. This allows us to learn the most relevant features among all possible combinations of appearance and motion. In contrast, late fusion of the softmax probabilities of the two streams [29] assumes that both appearance and motion are equally relevant for every class. As we show in Section 4.1.1, our proposed fusion significantly outperforms the late fusion.

Finally, we use a multitask loss for detecting objects, actions, and regressing the bounding box coordinates according to the object classes. The total loss $\mathcal{L}$ of the network is:

$$\mathcal{L} = \mathcal{L}_{\text{RPN}_R} + \mathcal{L}_{\text{RPN}_F} + \mathcal{L}_{\text{cls}} + \mathcal{L}_{\text{reg}} \quad , \tag{1}$$

with $\mathcal{L}_{\text{RPN}_R}$ and $\mathcal{L}_{\text{RPN}_F}$ the losses of the RPN operating on the RGB and flow stream, respectively, $\mathcal{L}_{\text{cls}}$ the classification loss, *i.e.*, for recognizing objects and actions, and $\mathcal{L}_{\text{reg}}$ the bounding box regression loss.

3.2. Joint learning of objects and actions

Given the candidate boxes, the network aims at jointly predicting whether a box contains a particular object and which action this object is performing. Let o (resp. a) be the ground-truth object (resp. action) label of a region proposal in the training set. To classify the boxes, we use a multitask architecture: one component predicts the object class, and a second one predicts the action class, independently of which object is performing it. Besides our proposed multitask architecture, we consider two alternatives to jointly predict object-action pairs: Cartesian product and hierarchy of classes. We now present details for these three objectives. We illustrate them in Figure 3 and summarize their main differences in Table 1.

Multitask. Our multitask architecture relies on a multitask loss, for classifying candidate boxes with both object and action labels. The first branch predicts the object la-

	loss	# outputs	probability	# params
Multitask	$-\log\ p_{\mathcal{O}}(o) - \log\ p_{\mathcal{A}}(a)$	$\lvert\mathcal{O}\rvert + \lvert\mathcal{A}\rvert + 2$	$p_{\mathcal{O}}(o) \cdot p_{\mathcal{A}}(a)$	0.9M
Cartesian	$-\log\ p_{\mathcal{V}}(o,a)$	$\lvert\mathcal{V}\rvert + 1$	$p_{\mathcal{V}}(o,a)$	54.6M
Hierarchical	$-\log\ p_{\mathcal{O}}(o) - \log\ p_{\mathcal{A}_o}(a)$	$\lvert\mathcal{O}\rvert + \lvert\mathcal{V}\rvert + 1$	$p_{\mathcal{O}}(o) \cdot p_{\mathcal{A}_o}(a)$	55.4M

Table 1: *Comparison of different losses for object-action learning. We give the number of parameters in the classification layers from the VRD dataset [24] where* $|\mathcal{O}| = 100, |\mathcal{A}| = 140, |\mathcal{V}| = 13344$ *(Section 4.4).*

bel. It is composed of a fully-connected layer that outputs $|\mathcal{O}| + 1$ scores (one per object class and another one for background) followed by softmax. Let $p_{\mathcal{O}}$ be the output of this branch. In the same way, $p_{\mathcal{A}}$ denotes the output of the second branch that predicts the action label, *i.e.*, of dimension $|\mathcal{A}| + 1$. We use a log loss on both object and action classification:

$$\mathcal{L}_{\text{cls}}^{\text{Multitask}} = -\log\ p_{\mathcal{O}}(o) - \log\ p_{\mathcal{A}}(a)\ . \qquad (2)$$

This version uses $|\mathcal{O}| + |\mathcal{A}| + 2$ outputs (Figure 3 (a)). For $|\mathcal{O}| = 100$ and $|\mathcal{A}| = 140$ the number of parameters in the classification layers is 0.9M (VRD dataset [24] used in Section 4.4). At test time, the probability of a box to be the object-action instance (o, a) is given by $p_{\mathcal{O}}(o) \cdot p_{\mathcal{A}}(a)$.

Cartesian product. Another solution is to consider each object-action pair as a separate class, *e.g. bird flying* (Figure 3 (b)). In this case, there is only one branch for classification with $|\mathcal{V}| + 1$ outputs. We denote as $p_{\mathcal{V}}$ the output of this branch. The classification loss is:

$$\mathcal{L}_{\text{cls}}^{\text{Cartesian}} = -\log\ p_{\mathcal{V}}(o,a)\ . \qquad (3)$$

This version uses $|\mathcal{V}| + 1$ outputs, which is in the order of $|\mathcal{A}| \times |\mathcal{O}|$. For instance, for $|\mathcal{V}| = 13344$ (VRD dataset [24]) the number of parameters in the classification layer is 54.6M, *i.e.*, 50× more than in the multitask (Table 1). This makes it less scalable than our multitask objective and does not allow sharing of action labels across object classes, which is required for zero-short learning. In the multitask case, samples of an object-action pair help training the detector of this object, which in turn helps detecting it doing other actions; *e.g. adult-running* and *adult-walking* samples help improving the *adult* detector. In contrast, by using the Cartesian product, each training sample helps training only one particular object-action detector. At test time, the probability of being an object-action instance (o, a) is given by $p_{\mathcal{V}}(o, a)$.

Hierarchy of classes. We also consider the set of valid object-action classes as a hierarchy (Figure 3 (c)). The first branch $p_{\mathcal{O}}$ predicts the object. For each object o, any branch $p_{\mathcal{A}_o}$ predicts the actions among the valid ones $\mathcal{A}_o$ for o. In this case, the classification loss is:

$$\mathcal{L}_{\text{cls}}^{\text{Hierarchy}} = -\log\ p_{\mathcal{O}}(o) - \log\ p_{\mathcal{A}_o}(a)\ . \qquad (4)$$

This version uses a total of $|\mathcal{O}| + 1$ outputs for the first level and $|\mathcal{V}|$ for the second level, see Figure 3 (c). For instance, for $|\mathcal{O}| = 100$ and $|\mathcal{V}| = 13344$ the number of parameters in the classification layers is 55.4M, *i.e.*, 50× more than in the multitask (Table 1). At test time, the probability of being an object-action instance (o, a) is given by $p_{\mathcal{O}}(o) \cdot p_{\mathcal{A}_o}(a)$.

Per-object regression. In all cases, we refine the proposal output by the RPN using a per-object regression of the bounding box coordinates. The RPN minimizes the geometric difference between the proposals and the ground-truth boxes. We follow [34] and make the regression target scale-invariant by normalizing it by the size of the proposal. We denote by $t_{o,a}$ the regression target for a proposal that covers an object. By using a per-object regression, we obtain the following regression loss:

$$\mathcal{L}_{\text{reg}} = \text{Smooth-L1}(u_o - t_{o,a})\ , \qquad (5)$$

with u_o the output of the regression branch u corresponding to object o, and:

$$\text{Smooth-L1}(x) = \begin{cases} 0.5x^2 & \text{if } |x| < 1, \\ |x| - 0.5 & \text{otherwise.} \end{cases} \qquad (6)$$

4. Experimental Results

In this section, we study the impact of each of our contributions separately. We first examine joint detection of objects and actions (Section 4.1) and zero-shot learning (Section 4.2). Next, we compare our proposed multitask architecture to the state of the art on semantic segmentation of object-action pairs (Section 4.3) and relationship detection in images (Section 4.4).

Implementation details. Our framework is based on Faster R-CNN [34] using the VGG-16 [40] as the underlying CNN architecture. We initialize both streams using the standard pre-training on ILSVRC 2012 [17]. This is in line with [47], which shows that pre-training on ILSVRC 2012 instead of UCF-101 [41] improves video classification accuracy.

4.1. Joint detection of objects and actions in videos

In this section, we evaluate our proposed end-to-end architecture for joint detection of object-action pairs. We start by validating the effectiveness of our end-to-end network (Section 4.1.1) and then, we examine the joint learning with the multitask objective (Section 4.1.2).

information / datasets		A2D	YTO	VID
objects		✓	✓	✓
actions		✓	-	-
# videos	training	3K	106	3,9K
	test	746	49	555
# annotations	training	16K	4K	1,7M
	test	4K	2,5K	170K

Table 2: *Overview of the video datasets we use.*

Video datasets. Table 2 shows some statistics of the datasets we use. For object-action detection we use the

input		RoI		Stream	A2D	YTO	VID
RGB	Flow	RGB	Flow	Fusion			
✓	-	✓	-	-	63.1	58.9	45.2
-	✓	-	✓	-	32.0	32.3	5.0
✓	✓	✓	✓	late	61.6	57.3	33.9
✓	✓	✓	✓	**ours**	**65.3**	**62.2**	**48.1**

Table 3: *Impact of end-to-end training: mAP for object detection of different training scenarios on the A2D, YTO and VID datasets.*

Actor-Action (A2D) dataset [50], which has sparse frame-level annotations for both objects and actions in videos. To the best of our knowledge, it is the only video dataset with bounding box and semantic segmentation annotations for object-action pairs. It contains 7 objects (*adult, baby, ball, bird, car, cat*, and *dog*) performing 8 different actions (*climb, crawl, eat, fly, jump, roll, run, walk*) or no action.

We also use two video datasets for object detection: the YouTube-Objects (YTO) dataset [13, 32] and the 'object detection in video' (VID) track of the ILSVRC [1]. YTO consists of videos collected from YouTube with 10 classes of moving objects, *e.g. aeroplane, car*. VID contains bounding boxes for 30 object classes including rigid objects, *e.g. motorcycle*, *watercraft*, and animals, *e.g. fox, monkey*.

Protocol. We measure the detection performance using the PASCAL VOC protocol [6]: a detection is correct if its intersection-over-union overlap (IoU) with a ground-truth box is greater than 0.5 and its labels (object and action) are correctly predicted. The performance for a class is the average precision (AP), and the overall performance is captured by the mean over all classes (mAP).

4.1.1 End-to-end architecture

We want to quantify the effectiveness of our proposed end-to-end architecture that consists of two streams fused (a) at the proposal (RoI) level and (b) at the feature level (Figure 2). We evaluate the impact of fusion for *object* detection alone. We perform experiments on the three video datasets (A2D, YTO and VID). Table 3 shows all the mAP results for the different cases we consider.

Impact of RGB and Flow cues. To examine the impact of the RGB and flow cues, we train each stream separately. The first two rows of Table 3 show that the RGB stream significantly outperforms the flow one. This is due to the fact that the RGB stream is able to learn information about how the objects look, which is a distinctive cue across different object classes. The flow stream performs worse than the RGB one in general, and is particularly poor on the VID dataset. This is because most objects in VID move only slightly, or their motion is not discriminative for the class.

Impact of end-to-end training. Our proposed fusion of the two streams enables end-to-end training. We examine the impact by comparing our proposed fusion of streams with late fusion of scores [9, 29]. In the latter, *i.e.*, late fusion of scores, we train the two-stream network fusing only the region-proposal layers and then average the classification scores of each stream as [9, 37]. Results in Table 3 show that, for all video datasets, using late score fusion reduces the detection performance compared to using the RGB stream alone. Interestingly, this is opposite of the findings in human action localization [9, 29], where performance increases due to the the significance of motion cues for actions. This shows that the two-stream architecture cannot be used as it is for object detection in videos and highlights a clear difference between object and human action detection. In contrast, on all object detection datasets, our proposed fusion outperforms the other cases: it leads to an increment over the late score fusion of approximately 2-3%. This shows that the network successfully learns when to leverage motion information and more importantly, how to jointly learn features coming from the two stream.

training	test on		
	objects	actions	objects + actions
objects	65.3	-	-
actions	-	56.2	-
Baseline	-	-	43.1
Cartesian	67.2	60.2	49.2
Hierarchical	67.9	59.6	49.6
Multitask	68.3	60.0	48.9

Table 4: *mAP of six different models when training with objects (first row), actions (second row), when multiplying their scores (third row) or when jointly training with objects and actions (last three rows) on A2D.*

4.1.2 Multitask learning

In this section, we evaluate our proposed multitask learning of objects and actions. We start by evaluating the performance only on *object* or on *action* detection. Therefore, we train and test our network with only object *or* only action labels (first two rows of Table 4). We also compute a baseline (third row of Table 4) for object-action detection in which we combine the object and the action detector trained separately. More precisely, for each object detection, we obtain object-action scores by multiplying the object scores with the action scores from the most overlapping action box.

Table 4 also reports the results of our proposed multitask architecture trained with objects and actions from the A2D dataset. The most interesting finding is that our multitask training improves the performance on each task separately (Table 4 objects, actions and multitask rows). In particular, when testing just on objects (68.3%) or just on actions (60.0%), our joint training outperforms training alone with objects (65.3%) or with actions (56.2%). The reasons are that the multitask network is (a) better able to generalize, (b) less prone to overfit to the training samples and (c) benefits from sharing examples across classes.

	climbing	crawling	eating	flying	jumping	rolling	running	walking	none	avg.
adult	7.0 (78.2)	7.8 (72.5)	19.6 (80.0)	-	11.0 (43.7)	24.3 (50.7)	6.3 (55.2)	13.6 (58.8)	33.3 (45.8)	15.3 (60.1)
baby	17.1 (63.1)	31.7 (76.4)	-	-	-	33.2 (85.4)	-	39.1 (77.9)	7.1 (31.9)	25.6 (64.9)
ball	-	-	-	0.4 (19.3)	3.5 (29.8)	10.7 (42.2)	-	-	8.0 (11.1)	5.6 (28.0)
bird	16.8 (51.8)	-	13.4 (38.0)	9.0 (66.2)	6.4 (32.3)	28.6 (60.2)	-	7.7 (55.0)	2.4 (2.3)	12.1 (43.3)
car	-	-	-	8.8 (42.2)	1.5 (90.5)	36.5 (66.8)	2.7 (63.8)	-	5.1 (17.4)	10.9 (55.9)
cat	32.3 (60.2)	-	28.9 (58.6)	-	9.6 (21.7)	43.8 (68.2)	8.0 (31.0)	19.1 (49.2)	3.1 (5.8)	20.7 (43.7)
dog	-	7.9 (58.2)	47.3 (74.2)	-	17.9 (41.6)	25.5 (38.5)	10.3 (31.4)	34.0 (67.2)	1.8 (5.3)	20.7 (42.3)

Table 5: *Evaluation of zero-shot learning for object-action pairs on A2D. For each object, we report the AP when excluding all actions of this object at training. The numbers in parenthesis indicate the AP when training with all object-action pairs.*

We also consider two alternative ways to jointly detect objects and actions (Section 3.2 and Figure 3): (a) Cartesian product of object-action labels and (b) hierarchy of object-action classes. Table 4 (Cartesian and hierarchical) reports the results when we train these two networks on the A2D dataset. We observe that they both perform similarly to our multitask network. The Cartesian and hierarchical networks have the advantage of being able to distinguish different ways objects perform each action (Table 1).

Discussion. In practice there are similarities in the way different objects perform the same action (*e.g. dog* and *cat eating*) and in the way the same object performs different actions (*e.g. dog walking* and *running*). Thus, our multitask objective allows the network to exploit the commonality among the two tasks, and hence, what is learned for each task facilitates the learning of the other. In a nutshell, our multitask architecture is a simpler model, able to reach the same performance as the alternative architectures while requiring much fewer parameters (Table 1 # params) and enabling zero-shot learning (Section 4.2). For instance, in Section 4.4 we clearly show the benefit of our multitask architecture compared to the Cartesian and hierarchical architectures for a large number of objects and actions due to its lower number of parameters.

Note that both losses (object and action) contribute equally to the overall loss (Equation 2), as they are of the same type (softmax), and the tasks they address are of the same difficulty. To validate this, we vary the weight of the action loss over 0.5, 1, 2 and observe insignificant variations ($< 0.5\%$) in the object-action mAP on A2D.

4.2. Zero-shot learning of actions

An important advantage of our end-to-end multitask architecture is its capability of predicting actions for an object without having trained for these particular object-actions combinations. To validate this intuition, we experiment on the A2D dataset (Table 2), which contains annotations for 7 objects performing 8 different actions in videos. We train the network seven times, where each time we remove for one object o' all its action labels. For instance, we remove all action labels for the object *cat*, but keep the *cat* examples for training the object detector. Equation 2 is replaced by:

$$\mathcal{L}^{\text{Multitask}}_{\text{cls zero-shot}} = -\log\ p_{\mathcal{O}}(o) - [o' \neq o] \log\ p_{\mathcal{A}}(a)\ . \quad (7)$$

Note that the object classifier is not changed, while the action classifier is learned only on the actions performed by the objects different from o'. This approach to zero-shot learning does not assume any prior knowledge such as attributes of the unseen classes [22].

We report the results of zero-shot learning in Table 5. We also report the AP when training with all object-action pairs. The results show that our network is able to infer information about actions not seen at training time for a given object. We observe that there are some object-action pair for which the AP is only slightly decreased, *e.g. cat rolling* or *dog eating*. This is because these objects share commonalities with others, *e.g. cat* and *dog eating*. In contrast, we observe poor performance for objects like *ball* which do not share similarities with other objects of the dataset. For object classes that share similarities in actions, such as *cat* and *dog*, our multitask architecture outperforms chance level classification of unknown actions by a large margin ($+15\%$), while for classes that do not share commonalities with other classes, like *adult* the gain is smaller ($+5\%$).

4.3. Object-action segmentation

A2D comes with annotations for semantic segmentation of object-action pairs. In this section, we extend our bounding box detections to pixelwise segmentation and we compare our results to the state of the art.

Metrics. Following [50], we measure class-average pixel accuracy and global pixel accuracy. Accuracy is the percentage of pixels for which the label is correctly predicted, either over all pixels (global) or first computed for each class separately and then averaged over classes (class-average). We also evaluate our segmentations using mIoU, *i.e.*, the IoU between the ground-truth segmentation and output segmentation averaged over all classes. mIoU is better suited as it is not biased towards background which is the most present class and it penalizes errors when too many pixels are set to a particular label instead of background.

Setup. Our multitask model predicts bounding boxes for each object-action pair. We extend our detections to pixelwise segmentations of object-action pairs by using segmentation proposals from either (a) the recently proposed SharpMask [30] or (b) the hierarchical video segmentation method GBH [10], which is the one used by the state-of-the-

methods	object			action			object + action		
	ave	glo	mIoU	ave	glo	mIoU	ave	glo	mIoU
Trilayer [50]	45.7	74.6	-	47.0	74.6	-	25.4	76.2	-
GPM (TSP) [49]	58.3	85.2	33.4	60.5	85.3	32.0	43.3	84.2	19.9
GPM (GBH) [49]	59.4	84.8	33.3	**61.2**	84.9	31.9	43.9	83.8	19.9
Ours (GBH)	72.9	85.8	42.7	**61.4**	84.6	35.5	**48.0**	83.9	24.9
Ours (SharpMask)	**73.7**	**90.6**	**49.5**	60.5	**89.3**	**42.2**	47.5	**88.7**	**29.7**

Table 6: *Comparison to the state of the art for object, action and object-action segmentation on A2D using class-average pixel accuracy (ave), global pixel accuracy (glo) and mean Intersection over Union (mIoU) metrics.*

art GPM method [49]. For each frame, we first apply non-maximum suppression on the detections that have a score greater than 0.5. Then, for each detection, we select the segmentation proposal that overlaps the most with it (according to IoU). If there is no such proposal, we directly use the rectangular detection itself as a segmentation mask. While our setup is simple, it serves as a baseline to evaluate our detections for semantic segmentation.

Results. The first three rows of Figure 4 show correctly labeled and segmented object-action pairs. We observe that our segmentation results are accurate, even in difficult cases, such as small objects (*e.g. birds*) or cluttered scenes (*e.g. adults running*). The two last rows show typical failure cases. In the fourth row, the action label of one *adult* is incorrect and there are some detections considered as wrong due to missing annotations. In the last row we miss the *adult* for which only one arm is visible.

Table 6 provides a quantitative comparison between our results and the state of the art [49, 50] on A2D. When using SharpMask, we outperform the previous state of the art for all metrics and all tasks, except for average accuracy on action segmentation, where we match [49]. Our improvements are particularly significant for object segmentation (+14% class-average accuracy, +16% mIoU) and joint object and action segmentation (more than +5% on all metrics). Note that we do not use any training segmentation from the A2D dataset (SharpMask is pre-trained on MS COCO [21]). Furthermore, we observe that even when using the same underlying method (GBH [10]), we perform on par or better than [49, 50] in all metric-task combinations.

4.4. Relationship detection of objects and actions

In this section we use only images, and therefore we use only the RGB stream as there is no flow for images. We apply our model to visual relationship detection, where we detect relationships between objects, defined as triples: object1 - interaction - object2. To do so, we transform each triplet into two pairs, each consisting of an object and an interaction and use them to train our multitask architecture.

Dataset and protocol. We employ the Visual Relationship Detection (VRD) dataset [24] that examines object relationships. It contains 4k training and 1k test images with 38k relationships between objects, such as *person kick ball, person wear shirt, motorcycle has wheel*. There are 100 different objects and 70 interaction types.

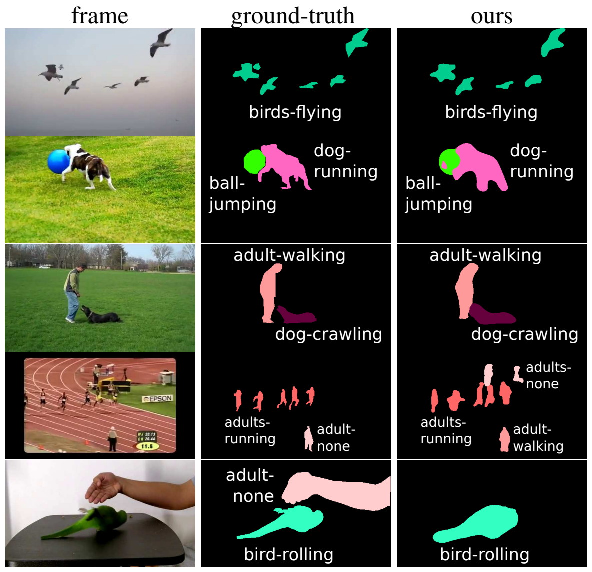

Figure 4: *Examples of semantic segmentation with (from left to right): the frame, the ground-truth and the segmentation output obtained when combining our approach with proposals from SharpMask [30]. The colors of the segmentations represent an object-action pair. Note that we do not use any object-action segmentation at training time.*

We consider here visual phrase detection [36], where the goal is to output a triplet object1 - interaction - object2 and localize it with one box having an IoU over 0.5 with the ground-truth box. We also evaluate relationship detection: the task consists in detecting a triplet object1 - interaction - object2 with two bounding boxes on object1 and object2, both having an IoU over 0.5 with their ground-truth boxes.

For evaluation, the metric used is recall @100 and recall @50 (denoted as R@N) and not mAP, as not all possible interactions are annotated in the test images. In each image, the top N detections are kept and recall is measured.

Model. To detect relationships using our multitask architecture, we transform each object1-interaction-object2 triplet into two pairs, each consisting of an object and an interaction label. More precisely, we double the set of all possible interactions, by including their passive forms. For example, the triplet *human kicks ball* becomes two pairs: (i) one with object *human* and action *kick*, and (ii) another pair with ob-

Modality	Method	Phrase detection		Relationship detection	
		R@100	R@50	R@100	R@50
V	VP [36]	0.07	0.04	-	-
	Joint CNN [39]	0.09	0.07	0.09	0.07
	VRD [24]	2.6	2.2	1.9	1.6
	Baseline	11.9	7.7	7.1	4.5
	Ours Multitask	18.3	14.5	11.3	8.6
V+L+F	VRD [24]	17.0	16.2	14.7	13.9

Table 7: *Comparison to different architectures and to the state-of-the-art visual relationships on the VRD dataset for phrases and relationship detection. We report R@100 and R@50 for methods using only visual cue (V) or also language and frequency priors (V+L+F).*

ject *ball* and action $\widetilde{kick}$ = *being kicked*. In that way, our training set consists of 100 object classes performing 140 different actions. Note here that the possible number of outputs is $100 + 140 + 2$ for our multitask objective.

At test time, we keep all detection with score over 0.5 and apply non-maximum suppression. For each pair of object detections, we score each possible interaction using the multiplication of the object scores and the interaction score. The interaction score is defined as the combination of the score of an interaction from the first object and its passive form from the second object, *i.e.*, the interaction score of *kick* in *human kicks ball* includes both scores of *kick* for the *human* and *being kicked* for the *ball*.

Results. Table 7 reports the R@100 and R@50 for the two tasks we examine, *i.e.*, phrase and relationship detection. We outperform all previous state-of-the-art results on both tasks and at both operating points, when comparing to methods based purely on the images ([24, 36, 39]). Moreover, our results are only a little worse than those of [24], where they enhance their visual model with some frequency prior as well as language priors by leveraging the semantic similarities of relationships in term of words. In particular, we perform on par on phrase detection (+1% at R@100 and −2% at R@50). Note how our method features a clear increment from R@50 to R@100, which shows its potential to correctly detect interactions that may be lower in the recall list. Hence, including some language or spatial priors could significantly increase our performance. Figure 5 shows some qualitative results.

Benefits of the multitask training. We compare our multitask architecture with a baseline approach where we multiply the scores of two separate networks, one trained on objects and another one trained on interactions. Table 7 shows that our multitask architecture outperforms this alternative ('Baseline' row). This comparison highlights the benefit of joint training compared to training for each task separately. We have also evaluated the Cartesian and hierarchical combination of objects and actions (Section 3.2) and found that they perform poorly (for both R@100 is around 0%). This can be explained by lack of training data necessary to determine the large number of parameters (55M in Table 1).

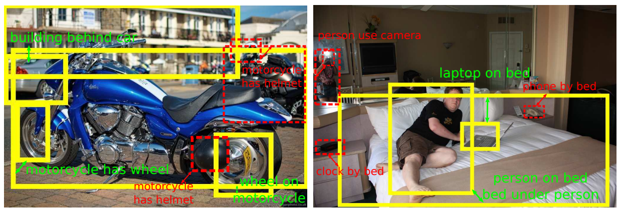

Figure 5: *Qualitative object-action relationships on the VRD dataset: (yellow): our correct boxes with their green label, (red): missed interactions for 100 retrieved boxes.*

Modality	Method	Phrase detection		Relationship detection	
		R@100	R@50	R@100	R@50
V	VRD [24]	1.1	0.8	0.8	0.7
	Baseline	4.3	2.3	2.4	1.3
	Ours Multitask	5.5	3.4	2.9	1.9
V+L+F	VRD [24]	3.8	3.4	3.5	3.1

Table 8: *Comparison to the state-of-the-art zero-shot detection of visual relationships on the VRD dataset. We report R@100 and R@50 for methods using only visual cue (V) or also language and frequency priors (V+L+F).*

Zero-shot learning. The test set of the VRD dataset contains 1.9k triplets that never occur in the training set. Our architecture allows zero-shot learning and we report the results on these triplets in Table 8. Our method outperforms the state-of-the-art method [24] when using only the visual modality (no language or frequency prior). Additionally, for phrase detection we detect unseen-at-training interactions better than [24], even when they also use language and frequency priors. Finally, our multitask architecture outperforms the baseline by a significant margin, highlighting the benefit of joint training compared to separate one.

5. Conclusions

Most state-of-the-art works for video detection aim at localizing either objects *or* actions. Instead, we jointly detect objects and actions in uncontrolled video scenes. To this end, we propose an end-to-end network built upon Faster R-CNN [34]. The key point is that our network operates with a multitask objective. We show that this joint training: (a) outperforms training alone with objects or with actions, as the network can generalize better, is less prone to overfit and benefits from sharing statistical strength between classes, (b) performs as well as other variants while requiring fewer parameters and (c) allows zero-shot learning of actions performed by an object, for which no action labels are present at training time. Our network can also be applied to different tasks including semantic segmentation and visual relationship detection.

Acknowledgments. This work was supported in part by the ERC grants ALLEGRO and VisCul, the MSR-Inria joint project, a Google research award, a Facebook gift, an Intel gift and an Amazon research award. We gratefully acknowledge the support of NVIDIA with the donation of GPUs used for this research.

References

[1] Imagenet large scale visual recognition challenge (ilsvrc). http://www.image-net.org/challenges/LSVRC/2015, 2015. 1, 5

[2] P. Bojanowski, F. Bach, I. Laptev, J. Ponce, C. Schmid, and J. Sivic. Finding actors and actions in movies. In *ICCV*, 2013. 2

[3] T. Brox, A. Bruhn, N. Papenberg, and J. Weickert. High accuracy optical flow estimation based on a theory for warping. In *ECCV*, 2004. 3

[4] M. Bucher, S. Herbin, and F. Jurie. Improving semantic embedding consistency by metric learning for zero-shot classiffication. In *ECCV*, 2016. 3

[5] V. Escorcia, J. C. Niebles, and B. Ghanem. On the relationship between visual attributes and convolutional networks. In *CVPR*, 2015. 3

[6] M. Everingham, L. Van Gool, C. Williams, J. Winn, and A. Zisserman. The PASCAL Visual Object Classes Challenge 2007 Results, 2007. 5

[7] P. Felzenszwalb, R. Girshick, D. McAllester, and D. Ramanan. Object detection with discriminatively trained part based models. *IEEE Trans. on PAMI*, 2010. 2

[8] R. Girshick, J. Donahue, T. Darrell, and J. Malik. Rich feature hierarchies for accurate object detection and semantic segmentation. In *CVPR*, 2014. 2

[9] G. Gkioxari and J. Malik. Finding action tubes. In *CVPR*, 2015. 1, 2, 3, 5

[10] M. Grundmann, V. Kwatra, M. Han, and I. Essa. Efficient hierarchical graph-based video segmentation. In *CVPR*, 2010. 2, 6, 7

[11] A. Gupta, A. Kembhavi, and L. S. Davis. Observing human-object interactions: Using spatial and functional compatibility for recognition. *IEEE Trans. on PAMI*, 2009. 2

[12] F. C. Heilbron, V. Escorcia, B. Ghanem, and J. C. Niebles. Activitynet: A large-scale video benchmark for human activity understanding. In *CVPR*, 2015. 1

[13] V. Kalogeiton, C. Schmid, and V. Ferrari. Analysing domain shift factors between videos and images for object detection. In *IEEE Trans. on PAMI*, 2016. 5

[14] K. Kang, W. Ouyang, H. Li, and X. Wang. Object detection from video tubelets with convolutional neural networks. In *CVPR*, 2016. 1, 2

[15] A. Karpathy, G. Toderici, S. Shetty, T. Leung, R. Sukthankar, and L. Fei-Fei. Large-scale video classification with convolutional neural networks. In *CVPR*, 2014. 1

[16] A. Klaser, M. Marszalek, and C. Schmid. A Spatio-Temporal Descriptor Based on 3D-Gradients. In *BMVC*, 2008. 2

[17] A. Krizhevsky, I. Sutskever, and G. E. Hinton. Imagenet classification with deep convolutional neural networks. In *NIPS*, 2012. 2, 4

[18] C. H. Lampert, H. Nickisch, and S. Harmeling. Attribute-based classification for zero-shot visual object categorization. *IEEE Trans. on PAMI*, 2014. 3

[19] T. Lan, Y. Zhu, A. Roshan Zamir, and S. Savarese. Action recognition by hierarchical mid-level action elements. In *ICCV*, 2015. 2

[20] I. Laptev. On space-time interest points. *IJCV*, 2005. 2

[21] T.-Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollár, and C. L. Zitnick. Microsoft coco: Common objects in context. In *ECCV*, 2014. 7

[22] J. Liu, B. Kuipers, and S. Savarese. Recognizing human actions by attributes. In *CVPR*, 2011. 2, 3, 6

[23] W. Liu, D. Anguelov, D. Erhan, C. Szegedy, S. Reed, C.-Y. Fu, and A. C. Berg. SSD: Single shot multibox detector. In *ECCV*, 2016. 1, 2

[24] C. Lu, R. Krishna, M. Bernstein, and L. Fei-Fei. Visual relationship detection with language priors. In *ECCV*, 2016. 1, 2, 3, 4, 7, 8

[25] S. Ma, J. Zhang, N. Ikizler-Cinbis, and S. Sclaroff. Action recognition and localization by hierarchical space-time segments. In *ICCV*, 2013. 2

[26] T. Malisiewicz, A. Gupta, and A. Efros. Ensemble of exemplar-svms for object detection and beyond. In *ICCV*, 2011. 2

[27] J. Mao, W. Xu, Y. Yang, J. Wang, Z. Huang, and A. Yuille. Deep captioning with multimodal recurrent neural networks (m-rnn). In *ICLR*, 2015. 2

[28] M. Pandey and S. Lazebnik. Scene recognition and weakly supervised object localization with deformable part-based models. In *ICCV*, 2011. 2

[29] X. Peng and C. Schmid. Multi-region two-stream r-cnn for action detection. In *ECCV*, 2016. 1, 2, 3, 5

[30] P. O. Pinheiro, T.-Y. Lin, R. Collobert, and P. Dollàr. Learning to refine object segments. In *ECCV*, 2016. 2, 6, 7

[31] A. Prest, V. Ferrari, and C. Schmid. Explicit modeling of human-object interactions in realistic videos. *IEEE Trans. on PAMI*, 2013. 2

[32] A. Prest, C. Leistner, J. Civera, C. Schmid, and V. Ferrari. Learning object class detectors from weakly annotated video. In *CVPR*, 2012. 2, 5

[33] M. Raptis, I. Kokkinos, and S. Soatto. Discovering discriminative action parts from mid-level video representations. In *CVPR*, 2012. 2

[34] S. Ren, K. He, R. Girshick, and J. Sun. Faster R-CNN: Towards real-time object detection with region proposal networks. In *NIPS*, 2015. 1, 2, 3, 4, 8

[35] M. D. Rodriguez, J. Ahmed, and M. Shah. Action mach: a spatio-temporal maximum average correlation height filter for action recognition. In *CVPR*, 2008. 1

[36] M. A. Sadeghi and A. Farhadi. Recognition using visual phrases. In *CVPR*, 2011. 2, 3, 7, 8

[37] S. Saha, G. Singh, M. Sapienza, P. H. Torr, and F. Cuzzolin. Deep learning for detecting multiple space-time action tubes in videos. In *BMVC*, 2016. 2, 3, 5

[38] C. Schüldt, I. Laptev, and B. Caputo. Recognizing human actions: a local svm approach. In *Proc. ICPR*, 2004. 1

[39] K. Simonyan and A. Zisserman. Two-stream convolutional networks for action recognition in videos. In *NIPS*, 2014. 1, 2, 3, 8

[40] K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. In *ICLR*, 2015. 4

[41] K. Soomro, A. R. Zamir, and M. Shah. UCF101: A Dataset of 101 Human Actions Classes From Videos in The Wild. In *CRCV-TR-12-01*, 2012. 4

[42] J. R. Uijlings, K. E. van de Sande, T. Gevers, and A. W. Smeulders. Selective search for object recognition. *IJCV*, 2013. 2

[43] S. Venugopalan, M. Rohrbach, J. Donahue, R. Mooney, T. Darrell, and K. Saenko. Sequence to sequence-video to text. In *ICCV*, 2015. 1

[44] O. Vinyals, A. Toshev, S. Bengio, and D. Erhan. Show and tell: A neural image caption generator. In *CVPR*, 2015. 2

[45] P. Viola and M. Jones. Rapid object detection using a boosted cascade of simple features. In *CVPR*, 2001. 2

[46] H. Wang, D. Oneata, J. Verbeek, and C. Schmid. A robust and efficient video representation for action recognition. *IJCV*, 2015. 2

[47] L. Wang, Y. Xiong, Z. Wang, Y. Qiao, D. Lin, X. Tang, and L. Van Gool. Temporal segment networks: towards good practices for deep action recognition. In *ECCV*, 2016. 4

[48] P. Weinzaepfel, Z. Harchaoui, and C. Schmid. Learning to track for spatio-temporal action localization. In *ICCV*, 2015. 1, 2

[49] C. Xu and J. J. Corso. Actor-action semantic segmentation with grouping-process models. In *CVPR*, 2016. 2, 7

[50] C. Xu, S.-H. Hsieh, C. Xiong, and J. J. Corso. Can humans fly? Action understanding with multiple classes of actors. In *CVPR*, 2015. 1, 2, 5, 6, 7

[51] B. Yao, X. Jiang, A. Khosla, A. L. Lin, L. Guibas, and L. Fei-Fei. Human action recognition by learning bases of action attributes and parts. In *ICCV*, 2011. 3

[52] L. Yao, A. Torabi, K. Cho, N. Ballas, C. Pal, H. Larochelle, and A. Courville. Describing videos by exploiting temporal structure. In *ICCV*, 2015. 1