Detect or Track: Towards Cost-Effective Video Object Detection/Tracking

Hao Luo,[1*] **Wenxuan Xie,**[2] **Xinggang Wang,**[1] **Wenjun Zeng**[2]
[1]School of Electronic Information and Communications, Huazhong University of Science and Technology
[2]Microsoft Research Asia
{luohao, xgwang}@hust.edu.cn, {wenxie, wezeng}@microsoft.com

Abstract

State-of-the-art object detectors and trackers are developing fast. Trackers are in general more efficient than detectors but bear the risk of drifting. A question is hence raised – how to improve the accuracy of video object detection/tracking by utilizing the existing detectors and trackers within a given time budget? A baseline is frame skipping – detecting every N-th frames and tracking for the frames in between. This baseline, however, is suboptimal since the detection frequency should depend on the tracking quality. To this end, we propose a scheduler network, which determines to detect or track at a certain frame, as a generalization of Siamese trackers. Although being light-weight and simple in structure, the scheduler network is more effective than the frame skipping baselines and flow-based approaches, as validated on ImageNet VID dataset in video object detection/tracking.

Introduction

Convolutional neural network (CNN)-based methods have achieved significant progress in computer vision tasks such as object detection (Ren et al. 2015; Liu et al. 2016; Dai et al. 2016; Tang et al. 2018b) and tracking (Held, Thrun, and Savarese 2016; Bertinetto et al. 2016; Nam and Han 2016; Bhat et al. 2018). Following the tracking-by-detection paradigm, most state-of-the-art trackers can be viewed as a local detector of a specified object. Consequently, trackers are generally more efficient than detectors and can obtain precise bounding boxes in subsequent frames if the specified bounding box is accurate. However, as evaluated commonly on benchmark datasets such as OTB (Wu, Lim, and Yang 2015) and VOT (Kristan et al. 2017), trackers are encouraged to track as long as possible. It is non-trivial for trackers to be stopped once they are not confident, although heuristics, such as a threshold of the maximum response value, can be applied. Therefore, trackers bear the risk of drifting.

Besides object detection and tracking, there have been recently a series of studies on video object detection (Kang et al. 2016; 2017; Feichtenhofer, Pinz, and Zisserman 2017; Zhu et al. 2017b; 2017a; 2018; Chen et al. 2018). Beyond the baseline to detect each frame individually, state-of-the-art approaches consider the temporal consistency of the detection results via tubelet proposals (Kang et al. 2016; 2017), optical flow (Zhu et al. 2017b; 2017a; 2018) and regression-based trackers (Feichtenhofer, Pinz, and Zisserman 2017). These approaches, however, are optimized for the detection accuracy of each individual frame. They either do not associate the presence of an object in different frames as a tracklet, or associate after performing object detection on each frame, which is time-consuming.

This paper is motivated by the constraints from practical video analytics scenarios such as autonomous driving and video surveillance. We argue that algorithms applied to these scenarios should be:

- capable of **associating an object** appearing in different frames, such that the trajectory or velocity of the object can be further inferred.
- in **realtime** (e.g., over 30 fps) and as fast as possible, such that the deployment cost can be further reduced.
- with **low latency**, which means to produce results once a frame in a video stream has been processed.

Considering these constraints, we focus in this paper on the task of video object detection/tracking (Russakovsky et al. 2017). The task is to detect objects in each frame (similar to the goal of video object detection), with an additional goal of associating an object appearing in different frames.

In order to handle this task under the realtime and low latency constraint, we propose a detect or track (DorT) framework. In this framework, object detection/tracking of a video sequence is formulated as a sequential decision problem – a scheduler network makes a detection/tracking decision for every incoming frame, and then these frames are processed with the detector/tracker accordingly. The architecture is illustrated in Figure 1.

The scheduler network is the most unique part of our framework. It should be light-weight but be able to determine to detect or track. Rather than using heuristic rules (e.g., thresholds of tracking confidence values), we formulate the scheduler as a small CNN by assessing the tracking quality. It is shown to be a generalization of Siamese trackers and a special case of reinforcement learning (RL).

The contributions are summarized as follows:

- We propose the DorT framework, in which the object detection/tracking of a video sequence is formulated as a sequential decision problem, while being in realtime and

[*]This work was done when Hao Luo was an intern at Microsoft Research Asia.

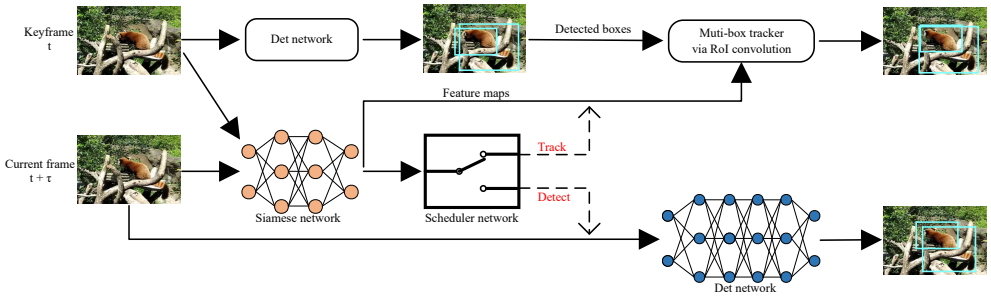


Figure 1: Detect or track (DorT) framework. The scheduler network compares the current frame $t + \tau$ with the keyframe t by evaluating the tracking quality, and determines to detect or *track* frame $t + \tau$: either frame $t + \tau$ is detected by a single-frame detector, or bounding boxes are tracked to frame $t + \tau$ from the keyframe t. If *detect* is chosen, frame $t + \tau$ is assigned as the new keyframe, and the boxes in frame $t + \tau$ and frame $t + \tau - 1$ are associated by the widely-used Hungarian algorithm (not shown in the figure for conciseness).

with low latency.

- We propose a light-weight but effective scheduler network, which is shown to be a generalization of Siamese trackers and a special case of RL.
- The proposed DorT framework is more effective than the frame skipping baselines and flow-based approaches, as validated on ImageNet VID dataset (Russakovsky et al. 2015) in video object detection/tracking.

Related Work

To our knowledge, we are the first to formulate video object detection/tracking as a sequential decision problem and there is no existing similar work to directly compare with. However, it is related to existing work in multiple aspects.

Video Object Detection/Tracking

Video object detection/tracking is a task in ILSVRC 2017 (Russakovsky et al. 2017), where the winning entries are optimized for accuracy rather than speed. (Deng et al. 2017) adopts flow aggregation (Zhu et al. 2017a) to improve the detection accuracy. (Wei et al. 2017) combines flow-based (Ilg et al. 2017) and object tracking-based (Nam and Han 2016) tubelet generation (Kang et al. 2017). THU-CAS (Russakovsky et al. 2017) considers flow-based tracking (Kang et al. 2016), object tracking (Held, Thrun, and Savarese 2016) and data association (Yu et al. 2016).

Nevertheless, these methods combine multiple cues (e.g., flow aggregation in detection, and flow-based and object tracking-based tubelet generation) which are complementary but time-consuming. Moreover, they apply global post-processing such as seq-NMS (Han et al. 2016) and tubelet NMS (Tang et al. 2018a) which greatly improve the accuracy but are not suitable for a realtime and low latency scenario.

Video Object Detection

Approaches to video object detection have been developed rapidly since the introduction of the ImageNet VID dataset (Russakovsky et al. 2015). (Kang et al. 2016; 2017) propose a framework that consists of per-frame proposal generation, bounding box tracking and tubelet re-scoring. (Zhu et al. 2017b) proposes to detect frames sparsely and propagates features with optical flow. (Zhu et al. 2017a) proposes to aggregate features in nearby frames along the motion path to improve the feature quality. Futhermore, (Zhu et al. 2018) proposes a high-performance approach by considering feature aggregation, partial feature updating and adaptive keyframe scheduling based on optical flow. Besides, (Feichtenhofer, Pinz, and Zisserman 2017) proposes to learn detection and tracking using a single network with a multi-task objective. (Chen et al. 2018) proposes to propagate the sparsely detected results through a space-time lattice. All the methods above focus on the accuracy of each individual frame. They either do not associate the presence of an object in different frames as a tracklet, or associate after performing object detection on each frame, which is time-consuming.

Multiple Object Tracking

Multiple object tracking (MOT) focuses on data association: finding the set of trajectories that best explains the given detections (Leal-Taixé et al. 2014). Existing approaches to MOT fall into two categories: batch and online mode. Batch mode approaches pose data association as a global optimization problem, which can be a min-cost max-flow problem (Zhang, Li, and Nevatia 2008; Pirsiavash, Ramanan, and Fowlkes 2011), a continuous energy minimization problem (Milan, Roth, and Schindler 2014) or a graph cut problem (Tang et al. 2016; 2017). Contrarily, online mode approaches are only allowed to solve the data association problem with the present and past frames. (Xiang, Alahi, and Savarese 2015) formulates data association as a Markov decision process. (Milan et al. 2017; Sadeghian, Alahi, and Savarese 2017) employs recurrent neural networks (RNNs) for feature representation and data association.

State-of-the-art MOT approaches aim to improve the data association performance given publicly-available detections since the introduction of the MOT challenge (Leal-Taixé et al. 2015). However, we focus on the sequential decision problem of detection or tracking. Although the widely-used Hungarian algorithm is adopted for simplicity and fairness in the experiments, we believe the incorporation of existing MOT approaches can further enhance the accuracy.

Keyframe Scheduler

Researchers have proposed approaches to adaptive keyframe scheduling beyond regular frame skipping in video analytics. (Zhu et al. 2018) proposes to estimate the quality of optical flow, which relies on the time-consuming flow network. (Chen et al. 2018) proposes an *easiness measure* to consider the size and motion of small objects, which is hand-crafted and more importantly, it is a detect-then-schedule paradigm but cannot determine to detect or track. (Li, Shi, and Lin 2018; Xu et al. 2018) learn to predict the discrepancy between the segmentation map of the current frame and the keyframe, which are only applicable to segmentation tasks.

All the methods above, however, solve an auxiliary task (e.g., flow quality, or discrepancy of segmentation maps) but do not answer the question directly in a classification perspective – is the current frame a keyframe or not? In contrast, we pose video object detection/tracking as a sequential decision problem, and learn directly whether the current frame is a keyframe by assessing the tracking quality. Our formulation is further shown as a generalization of Siamese trackers and a special case of RL.

The DorT Framework

Video object detection/tracking is formulated as follows. Given a sequence of video frames $F = \{f_1, f_2, \ldots, f_N\}$, the aim is to obtain bounding boxes $B = \{b_1, b_2, \ldots, b_M\}$, where $b_i = \{rect_i, fid_i, score_i, id_i\}$, $rect_i$ denotes the 4-dim bounding box coordinates and fid_i, $score_i$ and id_i are scalars denoting respectively the frame ID, the confidence score and the object ID.

Considering the realtime and low latency constraint, we formulate video object detection/tracking as a sequential decision problem, which consists of four modules: single-frame detector, multi-box tracker, scheduler network and data association. An algorithm summary follows the introduction of the four modules.

Single-Frame Detector

We adopt R-FCN (Dai et al. 2016) as the detector following deep feature flow (DFF) (Zhu et al. 2017b). Our framework, however, is compatible with all single-frame detectors.

Efficient Multi-Box Tracker via RoI Convolution

The SiamFC tracker (Bertinetto et al. 2016) is adopted in our framework. It learns a deep feature extractor during training such that an object is similar to its deformations but different from the background. During testing, the nearby patch with the highest confidence is selected as the tracking result. The tracker is reported to run at 86 fps in the original paper.

Despite its efficiency, there are usually 30 to 50 detected boxes in a frame outputted by R-FCN. It is a natural idea to track only the high-confidence ones and ignore the rest. Such an approach, however, results in a drastic decrease in mAP since R-FCN detection is not perfect and many true positives with low confidence scores are discarded. We therefore need to track all the detected boxes.

It is time-consuming to track 50 boxes without optimization (about 3 fps). In order to speed up the tracking process, we propose to share the feature extraction network of multiple boxes and propose an RoI convolution layer in place of the original cross-correlation layer in SiamFC. Figure 2 is an illustration. Through cropping and convolving on the feature maps, the proposed tracker is over 10x faster than the time-consuming baseline while obtaining comparable accuracy.

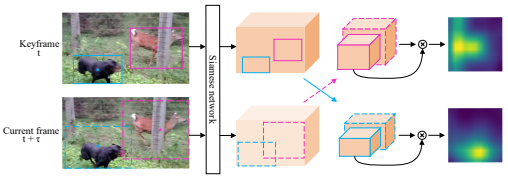


Figure 2: RoI convolution. Given targets in keyframe t and search regions in frame $t + \tau$, the corresponding RoIs are cropped from the feature maps and convolved to obtain the response maps. Solid boxes denote detected objects in keyframe t and dashed boxes denote the corresponding search region in frame $t + \tau$. A star $\star$ denotes the center of its corresponding bounding box. The center of a dashed box is copied from the tracking result in frame $t + \tau - 1$.

Notably, there is no learnable parameter in the RoI convolution layer, and thus we can train the SiamFC tracker following the original settings in (Bertinetto et al. 2016).

Scheduler Network

The scheduler network is the core of DorT, as our task is formulated as a sequential decision problem. It takes as input the current frame $f_{t+\tau}$ and its keyframe f_t, and determines to detect or track, denoted as $Scheduler(f_t, f_{t+\tau})$. We will elaborate this module in the next section.

Data Association

Once the scheduler network determines to detect the current frame, there is a need to associate the previous tracked boxes and the current detected boxes. Hence, a data association algorithm is required. For simplicity and fairness in the paper, the widely-used Hungarian algorithm is adopted. Although it is possible to improve the accuracy by incorporating more advanced data association techniques (Xiang, Alahi, and Savarese 2015; Sadeghian, Alahi, and Savarese 2017), it is not the focus in the paper. The overall architecture of the DorT framework is shown in Figure 1. More details are summarized in Algorithm 1.

The Scheduler Network in DorT

The scheduler network in DorT aims to determine to detect or track given a new frame by estimating the quality of the tracked boxes. It should be efficient itself. Rather than training a network from scratch, we propose to reuse part of the tracking network. Firstly, the l-th layer convolutional feature map of frame t and frame $t + \tau$, denoted respectively as x_l^t and $x_l^{t+\tau}$, are fed into a correlation layer which performs point-wise feature comparison

$$x_{corr}^{t,t+\tau}(i,j,p,q) = \left\langle x_l^t(i,j), x_l^{t+\tau}(i+p, j+q) \right\rangle \quad (1)$$

Algorithm 1 The Detect or Track (DorT) Framework

Input: A sequence of video frames $F = \{f_1, f_2, \ldots, f_N\}$.
Output: Bounding boxes $B = \{b_1, b_2, \ldots, b_M\}$ with ID, where $b_i = \{rect_i, fid_i, score_i, id_i\}$.

$B \leftarrow \{\}$
$t \leftarrow 1$ ▷ t is the index of keyframe
Detect f_1 with the single-frame detector.
Assign new ID to the detected boxes.
Add the detected boxes in f_1 to B.
for $i \leftarrow 2$ to N **do**
  $d \leftarrow Scheduler(f_t, f_i)$ ▷ decision of scheduler
  if $d = detect$ **then**
    Detect f_i with single-frame detector.
    Match boxes in f_i and f_{i-1} using Hungarian.
    Assign new ID to unmatched boxes in f_i.
    Assign corresponding ID to matched boxes in f_i.
    $t \leftarrow i$ ▷ update keyframe
  else ▷ the decision is to $track$
    Track boxes from f_t to f_i.
    Assign corresponding ID to tracked boxes in f_i.
    Assign corresponding detection score to tracked boxes in f_i.
  end if
  Add the bounding boxes in f_i to B.
end for

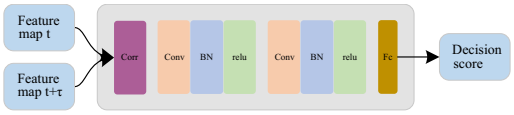


Figure 3: Scheduler network. The output feature map of the correlation layer is followed by two convolutional layers and a fully-connected layer with a 2-way softmax. As discussed later, this structure is a generalization of the SiamFC tracker.

where $-d \leq p \leq d$ and $-d \leq q \leq d$ are offsets to compare features in a neighbourhood around the locations (i, j) in the feature map, defined by the maximum displacement d. Hence, the output of the correlation layer is a feature map of size $x_{corr} \in \mathbb{R}^{H_l \times W_l \times (2d+1)^2}$, where H_l and W_l denote respectively the height and width of the l-th layer feature map. The correlation feature map x_{corr} is then passed through two convolutional layers and a fully-connected layer with a 2-way softmax. The final output of the network is a classification score indicating the probability to detect the current frame. Figure 3 is an illustration of the scheduler network.

Training Data Preparation

Existing groundtruth in the ImageNet VID dataset (Russakovsky et al. 2015) does not contain an indicator of the tracking quality. In this paper, we simulate the tracking process between two sampled frames and label it as *detect* (0) or *track* (1) in a strict protocol.

As we have sampled frame t and frame $t+\tau$ from the same sequence, we track all the groundtruth bounding boxes using SiamFC from frame t to frame $t + \tau$. If all the groundtruth boxes in frame $t + \tau$ are matched with the tracked boxes (e.g., IOU over 0.8), the frame is labeled as *track*; otherwise, it is labeled as *detect*. Any emerging or disappearing objects indicates a *detect*. Several examples are shown in Figure 4.

We have also tried to learn a scheduler for each tracker, but found it difficult to handle high-confidence false detections and non-trivial to merge the decisions of all the trackers. In contrast, the proposed approach to learning a single scheduler is an elegant solution which directly learns the decision rather than an auxiliary target such as the fraction of pixels at which the semantic segmentation labels differ (Li, Shi, and Lin 2018), or the fraction of low-quality flow estimation (Zhu et al. 2018).

Relation to the SiamFC Tracker

The proposed scheduler network can be seen as a generalization of the original SiamFC (Bertinetto et al. 2016). In the correlation layer of SiamFC, the target feature ($6\times6\times128$) is convolved with the search region feature ($22\times22\times128$) and obtains the response map ($17 \times 17 \times 1$, which can be equivalently written as $1 \times 1 \times 17^2$). Similarly, we can view the correlation layer of the proposed scheduler network (see Eq. 1) as convolutions between multiple target features in the keyframe and their respective nearby search regions in the current frame. The size of a target equals the receptive field of the input feature map of our scheduler. Figure 5 shows several examples of targets. Actually, however, targets include all possible patches in a sliding window manner.

In this sense, the output feature map of the correlation layer $x_{corr} \in \mathbb{R}^{H_l \times W_l \times (2d+1)^2}$ can be regarded as a set of $H_l \times W_l$ SiamFC tracking tasks, where the response map of each is $1 \times 1 \times (2d+1)^2$. The correlation feature map is then fed into a small CNN consisting of two convolutional layers and a fully-connected layer.

In summary, the generalization of the proposed scheduler network over SiamFC lies in two fold:

- SiamFC correlates a target feature with its nearby search region, while our scheduler extends the number of tasks from one to many.
- SiamFC directly picks the highest value in the correlation feature map as the result, whereas the proposed scheduler fuses the multiple response maps with a CNN.

The validity of the proposed scheduler network is hence clear – it first convolves patches in frame t (examples shown in Figure 5) with their respective nearby regions in frame $t+\tau$, and then fuses the response maps with a CNN, in order to measure the difference between the two frames, and more importantly, to assess the tracking quality. The scheduler is also resistant to small perturbations by inheriting SiamFC's robustness to object deformation.

Relation to Reinforcement Learning

The sequential decision problem can also be formulated in a RL framework, where the action, state, state transition function and reward need to be defined.

Action. The action space $\mathcal{A}$ contains two types of actions: $\{detect, track\}$. If the decision is *detect*, object detector is applied to the current frame; otherwise, boxes tracked from the keyframe are taken as the results.

(a) Positive examples (b) Negative examples

Figure 4: Examples of labeled data for training the scheduler network. Red and green boxes denote groundtruth and tracked results, respectively. (a) Positive examples, where the IOU of each groundtruth box and its corresponding tracked box is over a threshold; (b) Negative examples, where at least one such IOU is below a threshold or there are emerging/disappearing objects.

Figure 5: Examples of targets on keyframes. The size of a target equals the receptive field of the input feature map of the scheduler. As shown, a target patch might be an object, a part of an object, or totally background. The "tracking" results of these targets will be fused later. It should be noted that targets include all possible patches in a sliding window manner, but not just the three boxes shown above.

State. The state $s_{t,\tau}$ is defined as a tuple $(x_l^t, x_l^{t+\tau})$, where x_l^t and $x_l^{t+\tau}$ denote the l-th layer convolutional feature map of frame t and frame $t+\tau$, respectively. Frame t is the keyframe on which object detector is applied, and frame $t+\tau$ is the current frame on which actions are to be determined.

State transition function. After the decision of action $a_{t,\tau}$ in state $s_{t,\tau}$. The next state is obtained upon the action:

- *detect*. The next state is $s_{t+\tau,1} = (x_l^{t+\tau}, x_l^{t+\tau+1})$. Frame $t+\tau$ is fed to the object detector and is set as the new keyframe.
- *track*. The next state is $s_{t,\tau+1} = (x_l^t, x_l^{t+\tau+1})$. Bounding boxes tracked from the keyframe are taken as the results in frame $t+\tau$. The keyframe t remains unchanged.

As shown above, no matter whether the keyframe is t or $t+\tau$, the task in the next state is to determine the action in frame $t+\tau+1$.

Reward. The reward function is defined as $r(s,a)$ since it is determined by both the state s and the action a. As illustrated in Figure 4, a labeling mechanism is proposed to obtain the groundtruth label of the tracking quality between two frames (i.e., a certain state s). We denote the groundtruth label as $GT(s)$, which is either *detect* or *track*. Hence, the reward function can be defined as follows:

$$r(s,a) = \begin{cases} 1, & GT(s) = a \\ 0, & GT(s) \neq a \end{cases} \quad (2)$$

which is based on the consistency between the groundtruth label and the action taken.

After defining all the above, the RL problem can be solved via a deep Q network (DQN) (Mnih et al. 2015) with a discount factor γ, penalizing the reward from future time steps. However, training stability is always an issue in RL algorithms (Anschel, Baram, and Shimkin 2017). In this paper, we set $\gamma = 0$ such that the agent only cares about the reward from the next time step. Therefore, the DQN becomes a regression network – pushing the predicted action to be the same as the *GT* action, and the scheduler network is a special case of RL. We empirically observe that the training procedure becomes easier and more stable by setting $\gamma = 0$.

Experiments

The DorT framework is evaluated on the ImageNet VID dataset (Russakovsky et al. 2015) in the task of video object detection/tracking. For completeness, we also report results in video object detection.

Experimental Setup

Dataset description. All experiments are conducted on the ImageNet VID dataset (Russakovsky et al. 2015). ImageNet VID is split into a training set of 3862 videos and a test set of 555 videos. There are per-frame bounding box annotations for each video. Furthermore, the presences of a certain target across different frames in a video are assigned with the same ID.

Evaluation metric. The evaluation metric for video object detection is the extensively used mean average precision (mAP), which is based on a sorted list of bounding boxes in descending order of their scores. A predicted bounding box is considered correct if its IOU with a groundtruth box is over a threshold (e.g., 0.5).

In contrast to the standard mAP which is based on bounding boxes, the mAP for video object detection/tracking is based on a sorted list of tracklets (Russakovsky et al. 2017). A tracklet is a set of bounding boxes with the same ID. Similarly, a tracklet is considered correct if its IOU with a groundtruth tracklet is over a threshold. Typical choices of

IOU thresholds for tracklet matching and per-frame bounding box matching are both 0.5. The score of a tracklet is the average score of all its bounding boxes.

Implementation details. Following the settings in (Zhu et al. 2017b), R-FCN (Dai et al. 2016) is trained with a ResNet-101 backbone (He et al. 2016) on the training set.

SiamFC is trained following the original paper (Bertinetto et al. 2016). Instead of training from scratch, however, we initialize the first four convolutional layers with the pre-trained parameters from AlexNet (Krizhevsky, Sutskever, and Hinton 2012) and change Conv5 from 3×3 to 1×1 with the Xavier initializer. Parameters of the first four convolutional layers are fixed during training (He et al. 2018). We only search for one scale and discard the upsampling step in the original SiamFC for efficiency. All images being fed into SiamFC are resized to 300×500. Moreover, the confidence score of a tracked box (for evaluation) is equal to its corresponding detected box in the keyframe.

The scheduler network takes as input the Conv5 feature of our trained SiamFC. The SGD optimizer is adopted with a learning rate 1e-2, momentum 0.9 and weight decay 5e-4. The batch size is set to 32. During testing, we raise the decision threshold of *track* to $\delta = 0.97$ (i.e., the scheduler outputs *track* if the predicted confidence of *track* is over δ) to ensure conservativeness of the scheduler. Furthermore, since nearby frames look similar, the scheduler is applied every σ frames (where σ is a tunable parameter) to reduce unnecessary computation.

All experiments are conducted on a workstation with an Intel Core i7-4790k CPU and a Titan X GPU. We empirically observe that the detection network and the tracking/scheduler network run at 8.33 fps and 100fps, respectively. This is because the ResNet-101 backbone is much heavier than AlexNet. Moreover, the speed of the Hungarian algorithm is as high as 667 fps.

Video Object Detection/Tracking

To our knowledge, the most closely related work to ours is (Lan et al. 2016), which handles cost-effective face detection/tracking. Since face is much easier to track and is with less deformation, the paper achieves success by utilizing non-deep learning-based detectors and trackers. However, we aim at general object detection/tracking in video, which is much more challenging. We demonstrate the effectiveness of the proposed DorT framework against several strong baselines.

Effectiveness of scheduler. The scheduler network is a core component of our DorT framework. Since SiamFC tracking is more efficient than R-FCN detection, the scheduler should predict *track* when it is safe for the trackers and be conservative enough to predict *detect* when there is sufficient change to avoid track drift.

We compare our DorT framework with a frame skipping baseline, namely a "fixed scheduler" – R-FCN is performed every σ frames and SiamFC is adopted to track for the frames in between. As aforementioned, our scheduler can also be applied every σ frames to improve efficiency. Moreover, there could be an oracle scheduler – predicting the groundtruth label (*detect* or *track*) as shown in Figure 4 during testing. The oracle scheduler is a 100% accurate scheduler in our setting. The results are shown in Figure 6.

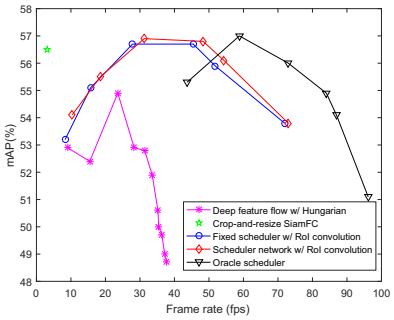
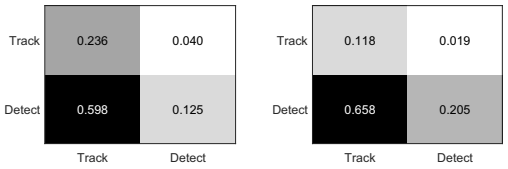


Figure 6: Comparison between different methods in video object detection/tracking in terms of mAP. The detector (for deep feature flow and fixed scheduler) or the scheduler (for scheduler network and oracle scheduler) can be applied every σ frames to obtain different results.



(a) $\sigma = 1$ (b) $\sigma = 10$

Figure 7: Confusion matrix of the scheduler network. The horizontal axis is the groundtruth and the vertical axis is the predicted label. The scheduler is applied every σ frames.

We can observe that the frame rate and mAP vary as σ changes. Interestingly, the curves are not monotonic – as the frame rate decreases, the accuracy in mAP is not necessarily higher. In particular, detectors are applied frequently when $\sigma = 1$ (the leftmost point of each curve). Associating boxes using the Hungarian algorithm is generally less reliable (given missed detections and false detections) than tracking boxes between two frames. It is also a benefit of the scheduler network – applying tracking only when confident, and thus most boxes are reliably associated. Hence, the curve of the scheduler network is on the upper-right side of that of the fixed scheduler as shown in Figure 6.

However, it can be also observed that there is certain distance between the curve of the scheduler network and that of the oracle scheduler. Given that the oracle scheduler is a 100% accurate classifier, we analyze the classification accuracy of the scheduler network in Figure 7. Let us take the $\sigma = 10$ case as an example. Although the classification accuracy is only 32.3%, the false positive rate (i.e., misclassifying a *detect* case as *track*) is as low as 1.9%. Because we empirically find that the mAP drops drastically if the scheduler mistakenly predicts *track*, our scheduler network is made conservative – *track* only when confident and *detect* if unsure. Figure 8 shows some qualitative results.

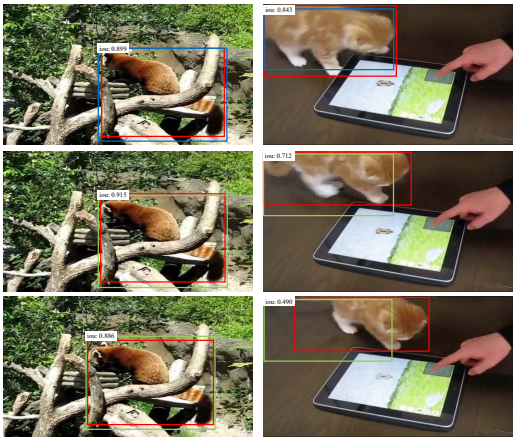


Figure 8: Qualitative results of the scheduler network. Red, blue and green boxes denote groundtruth, detected boxes and tracked boxes, respectively. The first row: R-FCN is applied in the keyframe. The second row: the scheduler determines to *track* since it is confident. The third row: the scheduler predicts to *track* in the first image although the red panda moves; however, the scheduler determines to *detect* in the second image as the cat moves significantly and is unable to be tracked.

Effectiveness of RoI convolution. Trackers are optimized for the crop-and-resize case (Bertinetto et al. 2016) – the target and search region are cropped and resized to a fixed size before matching. It is a nice choice since the tracking algorithm is not affected by the original size of the target. It is, however, slow in multi-box case and we propose RoI convolution as an efficient approximation. As shown in Figure 6, crop-and-resize SiamFC is even slower than detection – the overall running time is 3 fps. Notably, its mAP is 56.5%, which is roughly the same as that of our DorT framework empowered with RoI convolution. Our DorT framework, however, runs at 54 fps when $\sigma = 10$. RoI convolution obtains over 10x speed boost while retaining mAP.

Comparison with existing methods. Deep feature flow (Zhu et al. 2017b) focuses on video object detection without tracking. We can, however, associate its predicted bounding boxes with per frame data association using the Hungarian algorithm. The results are shown in Figure 6. It can be observed that our framework performs significantly better than deep feature flow in video object detection/tracking.

Concurrent works that deal with video object detection/tracking are the submitted entries in ILSVRC 2017 (Deng et al. 2017; Wei et al. 2017; Russakovsky et al. 2017). As discussed in the Related Work section, these methods aim only to improve the mAP by adopting complicated methods and post processing, leading to inefficient solutions without guaranteeing low latency. Their reported results on the test set ranges from 51% to 65% mAP. Our proposed DorT, notably, achieves 57% mAP on the validation set, which is comparable to the existing methods in magnitude, but is much more principled and efficient.

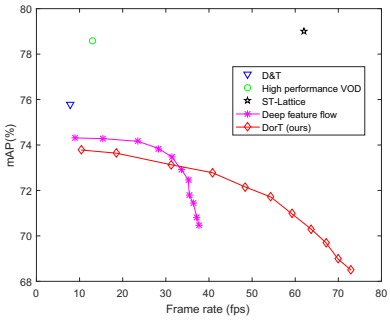


Figure 9: Comparison between different methods in video object detection in terms of mAP. Results of D&T, High performance VOD and ST-Lattice are copied from the original papers. The detector (for deep feature flow) or the scheduler (for scheduler network) can be applied every σ frames to obtain different results.

Video Object Detection

We also evaluate our DorT framework in video object detection for completeness, by removing the predicted object ID. Our DorT framework is compared against deep feature flow (Zhu et al. 2017b), D&T (Feichtenhofer, Pinz, and Zisserman 2017), high performance video object detection (VOD) (Zhu et al. 2018) and ST-Lattice (Chen et al. 2018). The results are shown in Figure 9. It can be observed that D&T and high performance VOD manage to achieve a speed-accuracy balance. They obtain higher results but cannot fit into realtime (over 30 fps) scenarios. ST-Lattice, although being fast and accurate, adopts detection results in future frames and is thus not suitable in a low latency scenario. As compared with deep feature flow, our DorT framework performs significantly faster with comparable performance (no more than 1% mAP loss). Although our aim is not the video object detection task, the results in Figure 9 demonstrate the effectiveness of our approach.

Conclusion and Future Work

We propose a DorT framework for cost-effective video object detection/tracking, which is in realtime and with low latency. Object detection/tracking of a video sequence is formulated as a sequential decision problem in the framework. Notably, a light-weight but effective scheduler network is proposed, which is shown to be a generalization of Siamese trackers and a special case of RL. The DorT framework turns out to be effective and strikes a good balance between speed and accuracy.

The framework can still be improved in several aspects. The SiamFC tracker can search for multiple scales to improve performance as in the original paper. More advanced data association methods can be applied by resorting to the state-of-the-art MOT algorithms. Furthermore, there is room to improve the training of the scheduler network to approach the oracle scheduler. These are left as future work.

Acknowledgment. This work was partly supported by NSFC (No. 61876212 & 61733007). The authors would like to thank Chong Luo and Anfeng He for fruitful discussions.

References

Anschel, O.; Baram, N.; and Shimkin, N. 2017. Averaged-dqn: Variance reduction and stabilization for deep reinforcement learning. In *ICML*.

Bertinetto, L.; Valmadre, J.; Henriques, J. F.; Vedaldi, A.; and Torr, P. H. 2016. Fully-convolutional siamese networks for object tracking. In *ECCVw*.

Bhat, G.; Johnander, J.; Danelljan, M.; Khan, F. S.; and Felsber, M. 2018. Unveiling the power of deep tracking. In *ECCV*.

Chen, K.; Wang, J.; Yang, S.; Zhang, X.; Xiong, Y.; Loy, C. C.; and Lin, D. 2018. Optimizing video object detection via a scale-time lattice.

Dai, J.; Li, Y.; He, K.; and Sun, J. 2016. R-fcn: Object detection via region-based fully convolutional networks. In *NIPS*.

Deng, J.; Zhou, Y.; Yu, B.; Chen, Z.; Zafeiriou, S.; and Tao, D. 2017. Speed/accuracy trade-offs for object detection from video. http://image-net.org/challenges/talks_2017/Imagenet%202017%20VID.pdf.

Feichtenhofer, C.; Pinz, A.; and Zisserman, A. 2017. Detect to track and track to detect. In *ICCV*.

Han, W.; Khorrami, P.; Paine, T. L.; Ramachandran, P.; Babaeizadeh, M.; Shi, H.; Li, J.; Yan, S.; and Huang, T. S. 2016. Seq-nms for video object detection. *arXiv*.

He, K.; Zhang, X.; Ren, S.; and Sun, J. 2016. Deep residual learning for image recognition. In *CVPR*.

He, A.; Luo, C.; Tian, X.; and Zeng, W. 2018. A twofold siamese network for real-time object tracking. In *CVPR*.

Held, D.; Thrun, S.; and Savarese, S. 2016. Learning to track at 100 fps with deep regression networks. In *ECCV*.

Ilg, E.; Mayer, N.; Saikia, T.; Keuper, M.; Dosovitskiy, A.; and Brox, T. 2017. Flownet 2.0: Evolution of optical flow estimation with deep networks. In *CVPR*.

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

Kang, K.; Li, H.; Xiao, T.; Ouyang, W.; Yan, J.; Liu, X.; and Wang, X. 2017. Object detection in videos with tubelet proposal networks. In *CVPR*.

Kristan, M.; Leonardis, A.; Matas, J.; Felsberg, M.; Pflugfelder, R.; Cehovin Zajc, L.; Vojir, T.; Hager, G.; Lukezic, A.; Eldesokey, A.; and Fernandez, G. 2017. The visual object tracking vot2017 challenge results. In *ICCVw*.

Krizhevsky, A.; Sutskever, I.; and Hinton, G. E. 2012. Imagenet classification with deep convolutional neural networks. In *NIPS*.

Lan, X.; Xiong, Z.; Zhang, W.; Li, S.; Chang, H.; and Zeng, W. 2016. A super-fast online face tracking system for video surveillance. In *ISCAS*.

Leal-Taixé, L.; Fenzi, M.; Kuznetsova, A.; Rosenhahn, B.; and Savarese, S. 2014. Learning an image-based motion context for multiple people tracking. In *CVPR*.

Leal-Taixé, L.; Milan, A.; Reid, I.; Roth, S.; and Schindler, K. 2015. Motchallenge 2015: Towards a benchmark for multi-target tracking. *arXiv*.

Li, Y.; Shi, J.; and Lin, D. 2018. Low-latency video semantic segmentation. In *CVPR*.

Liu, W.; Anguelov, D.; Erhan, D.; Szegedy, C.; Reed, S.; Fu, C.-Y.; and Berg, A. C. 2016. Ssd: Single shot multibox detector. In *ECCV*.

Milan, A.; Rezatofighi, S. H.; Dick, A. R.; Reid, I. D.; and Schindler, K. 2017. Online multi-target tracking using recurrent neural networks. In *AAAI*.

Milan, A.; Roth, S.; and Schindler, K. 2014. Continuous energy minimization for multitarget tracking. *TPAMI*.

Mnih, V.; Kavukcuoglu, K.; Silver, D.; Rusu, A. A.; Veness, J.; Bellemare, M. G.; Graves, A.; Riedmiller, M.; Fidjeland, A. K.; Ostrovski, G.; et al. 2015. Human-level control through deep reinforcement learning. *Nature*.

Nam, H., and Han, B. 2016. Learning multi-domain convolutional neural networks for visual tracking. In *CVPR*.

Pirsiavash, H.; Ramanan, D.; and Fowlkes, C. C. 2011. Globally-optimal greedy algorithms for tracking a variable number of objects. In *CVPR*.

Ren, S.; He, K.; Girshick, R.; and Sun, J. 2015. Faster r-cnn: Towards real-time object detection with region proposal networks. In *NIPS*.

Russakovsky, O.; Deng, J.; Su, H.; Krause, J.; Satheesh, S.; Ma, S.; Huang, Z.; Karpathy, A.; Khosla, A.; Bernstein, M.; et al. 2015. Imagenet large scale visual recognition challenge. *IJCV*.

Russakovsky, O.; Park, E.; Liu, W.; Deng, J.; Li, F.-F.; and Berg, A. 2017. Beyond imagenet large scale visual recognition challenge. http://image-net.org/challenges/beyond_ilsvrc.

Sadeghian, A.; Alahi, A.; and Savarese, S. 2017. Tracking the untrackable: Learning to track multiple cues with long-term dependencies. In *ICCV*.

Tang, S.; Andres, B.; Andriluka, M.; and Schiele, B. 2016. Multi-person tracking by multicut and deep matching. In *ECCV*.

Tang, S.; Andriluka, M.; Andres, B.; and Schiele, B. 2017. Multiple people tracking by lifted multicut and person reidentification. In *CVPR*.

Tang, P.; Wang, C.; Wang, X.; Liu, W.; Zeng, W.; and Wang, J. 2018a. Object detection in videos by high quality object linking. *arXiv*.

Tang, P.; Wang, X.; Bai, S.; Shen, W.; Bai, X.; Liu, W.; and Yuille, A. L. 2018b. Pcl: Proposal cluster learning for weakly supervised object detection. *TPAMI*.

Wei, Y.; Zhang, M.; Li, J.; Chen, Y.; Feng, J.; Dong, J.; Yan, S.; and Shi, H. 2017. Improving context modeling for video object detection and tracking. http://image-net.org/challenges/talks_2017/ilsvrc2017_short(poster).pdf.

Wu, Y.; Lim, J.; and Yang, M.-H. 2015. Object tracking benchmark. *TPAMI*.

Xiang, Y.; Alahi, A.; and Savarese, S. 2015. Learning to track: Online multi-object tracking by decision making. In *ICCV*.

Xu, Y.-S.; Fu, T.-J.; Yang, H.-K.; and Lee, C.-Y. 2018. Dynamic video segmentation network. In *CVPR*.

Yu, F.; Li, W.; Li, Q.; Liu, Y.; Shi, X.; and Yan, J. 2016. Poi: Multiple object tracking with high performance detection and appearance feature. In *ECCVw*.

Zhang, L.; Li, Y.; and Nevatia, R. 2008. Global data association for multi-object tracking using network flows. In *CVPR*.

Zhu, X.; Wang, Y.; Dai, J.; Yuan, L.; and Wei, Y. 2017a. Flow-guided feature aggregation for video object detection. In *ICCV*.

Zhu, X.; Xiong, Y.; Dai, J.; Yuan, L.; and Wei, Y. 2017b. Deep feature flow for video recognition. In *CVPR*.

Zhu, X.; Dai, J.; Yuan, L.; and Wei, Y. 2018. Towards high performance video object detection. In *CVPR*.